# Autonomous Robots

## Modeling, Path Planning, and Control

Farbod Fahimi

# Autonomous Robots

## Modeling, Path Planning, and Control

Springer

Farbod Fahimi
Mechanical Engineering Department
University of Alberta
Edmonton, Alberta
Canada
ffahimi@ualberta.ca

ISBN: 978-0-387-09537-0 e-ISBN: 978-0-387-09538-7
DOI 10.1007/978-0-387-09538-7

Library of Congress Control Number: 2008931982



Printed on acid-free paper

springer.com

*Dedicated to my supportive father, who sacrificed so much for his children.*

# Preface

### Autonomous Versus Conventional Robots

It is at least two decades since the conventional robotic manipulators have become a common manufacturing tool for different industries, from automotive to pharmaceutical. The proven benefits of utilizing robotic manipulators for manufacturing in different industries motivated scientists and researchers to try to extend the applications of robots to many other areas. To extend the application of robotics, scientists had to invent several new types of robots other than conventional manipulators. The new types of robots can be categorized in two groups: redundant (and hyper-redundant) manipulators and mobile (ground, marine, and aerial) robots. These two groups of robots have more freedom for their mobility, which allows them to do tasks that the conventional manipulators cannot do.

Engineers have taken advantage of the extra mobility of the new robots to make them work in constrained environments. The constraints can range from limited joint motions for redundant (or hyper-redundant) manipulators to obstacles in the way of mobile (ground, marine, and aerial) robots. Since these constraints usually depend on the work environment, they are variable. Engineers have had to invent methods to allow the robots deal with a variety of constraints automatically. A robot that is equipped with those methods that make it able to automatically deal with a variety of environmental constraints while performing a desired task is called an autonomous robot.

### Purpose of the Book

There are many books that discuss different aspects of Robotics. However, they mostly focus on conventional robotic manipulators and at best, add a brief section to address mobile robots. Recently, the application of autonomous robots (redundant and hyper-redundant manipulators, and ground, marine, and aerial robots) is finding its way into industries and even into people's everyday life. One can mention several examples such as robotic helicopters for surveillance, aerial photography, or farm spraying, high-end cars that park themselves, robotic vacuum cleaners, etc. It is becoming more important that our students learn about autonomous robots and our

engineers have information resources for designing, analyzing, and controlling these robots.

Since most of the robotic books only deal with conventional robots, nowadays, students do not have a chance to learn about autonomous robots and engineers who design autonomous robots have to resort to extracting information from research literature to design them, which is tedious for them. The present book provides the theories and methods that are useful for understanding and designing autonomous robots to students and engineers in a form that is detailed and easy to follow.

The purpose of this book is to familiarize the Mechanical and Electrical Engineering students and engineers with the methods of modeling/analysis/control that have been proven efficient through research.

### Scope of the Book

Similar to the conventional robotic manipulators, the autonomous robots are multidisciplinary machines and can be studied from different points of view, i.e., industrial, electrical, mechanical, and controls points of view. Autonomous robots can also be studied from the Artificial Intelligence point of view. Covering all these aspects of autonomous robots in one book is almost impossible and each of these aspects has their own audience. For these reasons, the scope of the present book is the mechanics and controls of autonomous robots. The book covers the kinematic and dynamic modeling/analysis of autonomous robots as well as the methods suitable for their control.

### Level of the Book

This book is useful for last-year undergraduate and first-year graduate students as well as engineers. The readers should have passed a second year course in Dynamics and a third year course in Automatic Control (or similar) to be able to fully take advantage of this book. The mentioned prerequisites are not an obstacle for Mechanical or Electrical Engineering students and engineers, since these courses are offered in ABET (or CEAB for Canadaian higher education) accredited engineering programs.

### Features of the Book

The key feature of the present book is its contents, which have never been gathered within one book and have never been presented in a form useful to students and engineers.

- The present book contains the theoretical tools necessary for analyzing the dynamics and control of autonomous robots in one place. The topics that are practical and are of interest to autonomous robot designers have been picked from advanced robotics research literature. These topics are sorted appropriately and will form the contents of the book.

- This book presents the theoretical tools for analyzing the dynamics of and controlling autonomous robots in a form that is comprehensible for students and engineers. The advanced robotics research literature have usually been authored with the research community in mind. The mathematical notation and the presentation method of these publications are not easy to understand. These publications normally lack the necessary details and intermediate steps. The current book uses a uniform notation, provides the mathematical background of the theories presentred, expands the details, and includes the intermediate steps and comprehensive examples to ease and accelerate the reader's comprehension.
- The current book has problems at the end of each chapter. The problems allow the reader to practice the theories presented in each chapter. The solution to most of the problems need some computer aided analysis. Some of the longer problems are more suitable for term projects.

The author hopes that the present book becomes an asset for learning the application of dynamics and controls in the field of autonomous robots.

Edmonton, Alberta, Canada *Farbod Fahimi*
July 2008

# Acknowledgments

I would like to thank my dear friend, Dr. Reza N. Jazar, for his support, encouragement, and comments during the time this book was being authored.

# Contents

# Acronyms

| | |
|---|---|
| 2D | Two-Dimensional |
| 3D | Three-Dimensional |
| ASV | Autonomous Surface Vessel |
| AUSVI | Association for Unmanned Vehicle Systems Internatioanl |
| CLSFM | Constraint Least Square Fitting Method |
| DOFs | Degrees of Freedom |
| GMRES | Generalized Minimum RESidual |
| GPS | Global Positioning System |
| HVAC | Heating, Ventilating, and Air Conditioning |
| IMU | Inertial Measurement Unit |
| JLA | Joint Limit Avoidance |
| P | Prismatic |
| PD | Proportional-Derivative |
| PID | Proportional-Integral-Derivative |
| R | Revolute |
| RFM | Recursive Fitting Method |
| SOI | Surface of Influence |
| SPDM | Special Purpose Dexterous Manipulator |
| SVD | Single Value Decomposition |
| UAV | Unmanned Aerial Vehicle |
| UGV | Unmanned Ground Vehicle |
| USV | Unmanned Surface Vessel |
| WITAS | Wallenberg laboratory for research on Information Technology and Autonomous Systems |

# Chapter 1
# Introduction

## 1.1 Redundant Manipulators

### *1.1.1 Kinematics*

A kinematically redundant manipulator is a serial robotic arm that has more independently driven joints than are necessary to define the desired pose (position and orientation) of its end-effector. With this definition, any planar manipulator (a manipulator whose end-effector motion is restrained in a plane) with more than three joints is a redundant manipulator. Also, a manipulator whose end-effector can accept a spatial pose is a redundant manipulator if it has more than six independently driven joints. For example, the manipulator shown in Fig. 1.1 has two 7-DOF arms mounted on a torso with three degrees of freedom (DOFs). This provides 10 DOFs for each arm. Since the end-effector of each arm can have a spatial motion with six DOFs, the arms are redundant. The degree of redundancy of each arm is four, which is the difference between the joint number and the end-effector's DOF for the arm.

It should be noted that kinematic redundancy as defined above should not be confused with actuator or sensor redundancy. Actuator or sensor redundancy is present if a manipulator has two actuators or sensors on one joint that serve the same purpose. Actuator or sensor redundancy is introduced in a manipulator design to increase the fault tolerance and reliability of the design. The kinematics of a manipulator with redundant actuators or sensors can be treated similar to that of a conventional manipulator, whereas the kinematics of a kinematically redundant manipulator must be studied differently.

### *1.1.2 Redundancy Resolution*

For a conventional manipulator, for which the end-effector DOF is equal to the number of joints, the position/velocity of the joints can be found easily if the position/velocity of their end-effector is specified. This process is called the "inverse kinematics" solution. This can be done because the number of equations written for the pose of the end-effector is exactly equal to the number of unknowns, i.e.,

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_1

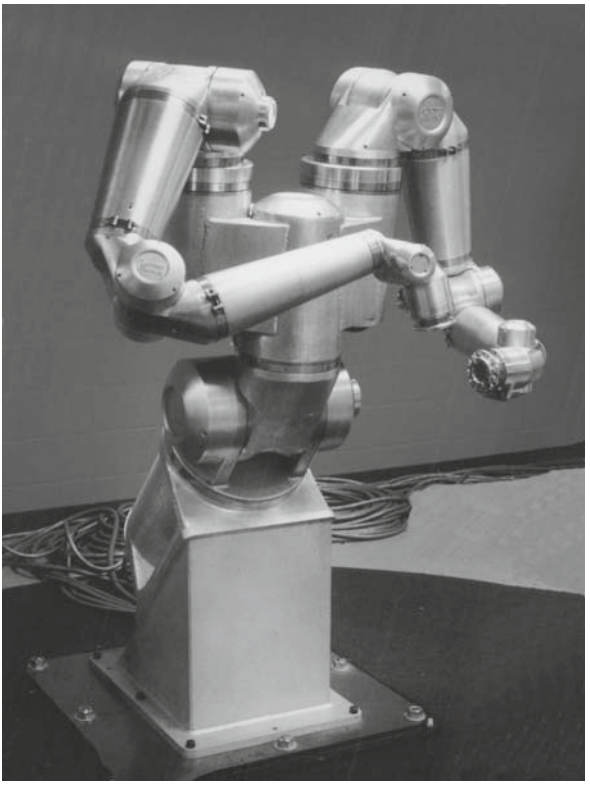

**Fig. 1.1** A 17-DOF redundant manipulator with two serial arms (Courtesy of Robotics Research Corporation, Cincinnati, Ohio, USA)

joints' positions/velocities. For a redundant manipulator, however, there are more unknowns (joints' positions/velocities) than there are equations (DOF of the end-effector). Therefore, the inverse kinematics mathematical problem does not have a unique solution. There is a need for approaches that can address this mathematical problem with multiple solutions. These approaches that solve the inverse kinematics problem for a redundant manipulator are called the "redundancy resolution" methods.

### *1.1.3 Use for Redundancy*

These multiple solutions allow for a higher task flexibility for a redundant manipulator compared to a conventional manipulator. The resources (structural strength, force/torque output abilities, extra DOFs, joint accuracies) of a redundant manipulator can be used optimally according to the task at hand. Since there are several joint configurations for which the end-effector of a redundant manipulator can reach

a certain desired pose, the manipulator can pick the one that serves an extra purpose better through optimization. Examples of extra purposes are numerous. The manipulator can choose the solution that best transmits force/torque to the end-effector. Or it can maximize joint range availability by choosing the solutions for which the joints' positions are closest to their center positions. The manipulator can pick the solution that requires the least amount of motion to minimize the joint velocities or the consumed energy. It can maximize dexterity by selecting the solution that avoids singularities. The manipulator can choose the solution that maximizes the structural stiffness to reduce the deflection errors. The extra solutions can be used by the manipulator to avoid obstacles that would otherwise prevent the end-effector from reaching its desired pose. The redundant manipulator can reallocate resources to compensate for the loss of a mechanical degree of freedom.

### *1.1.4 Mathematical Solution Methods*

There are several mathematical solution approaches that allow a redundant manipulator to automatically take advantage of its redundancy to satisfy extra tasks previously discussed. These mathematical methods can provide the manipulator with some degree of "autonomy," such that the manipulator is able to decide on the best kinematic solution according to the different environmental constraints defined for it. These methods are discussed throughly in Chapter 2.

The methods introduced in Chapter 2 provide the time history of the joint motions with which a desired task can be accomplished. However, since these methods are only kinematic methods, they cannot provide the means for driving the joint such that the desired task is physically performed by the robot. There are needs for control methods that can guarantee the physical performance of a manipulator. These control methods are presented in Chapter 5.

## 1.2 Hyper-Redundant Manipulators

Hyper-redundant manipulators are kinematically redundant manipulators that have a very large degree of redundancy. These manipulators have a morphology and operation analogous to that of snakes, elephant trunks, and tentacles. There are a number of very important applications where such robots would be advantageous. Working in cluttered environments and in tight and tunnel-like spaces are the most important features of hyper-redundant manipulators. Due to having their numerous DOFs and small link lengths, hyper-redundant manipulators are able to reach inside pipelines and ducts for repair or inspection. Their key applications are inspection, repair, and maintenance of mechanical systems related to nuclear reactors [9]. Two snake-like hyper-redundant manipulators are shown in Fig. 1.2.

Another unique feature of hyper-redundant manipulators is their grasping ability. Because of numerous DOFs, the flexibility of the hyper-redundant manipulators

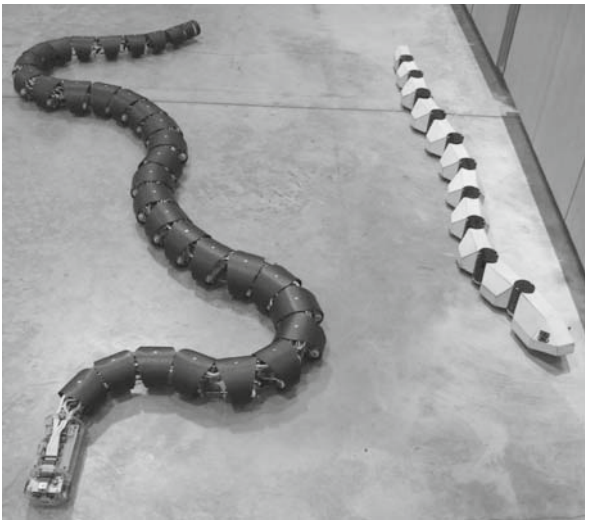

**Fig. 1.2** Two snake-like hyper-redundant robots
Source: http://en.wikipedia.org/wiki/Image:Robosnakes.jpg

allows them to grasp objects with various sizes and shapes easily. A typical hyper-redundant manipulator can grasp a large range of sizes and shapes. If conventional grippers found on nonredundant manipulators are used to cover the same range of sizes and shapes of objects, several different grippers will be required.

Hyper-redundant manipulators have been the focus of investigation of many researchers for nearly 20 years. However, they are mostly still in the laboratory research phase. There might be a number of reasons for this. The previous kinematic modeling techniques, used extensively for redundant manipulators and to be discussed in Chapter 2, are not suitable to or efficient enough for the needs of hyper-redundant robot task modeling. This is because the computational cost of the methods suitable for redundant manipulators is related to the degree of redundancy of the manipulator, and for a hyper-redundant manipulator with a large degree of redundancy, the computation cost of these methods become prohibitive. Also, the complexity of the mechanical design and implementation of hyper-redundant robots might have prevented their commercialization.

The material of Chapter 3 is meant to present efficient and straight forward redundancy resolution methods specific to hyper-redundant manipulators, to reduce the burdon of the kinematic computations to an acceptable level for real implementations. In Chapter 3, the history of joint motions that are required for a given task to be successfully performed is found. In Section 4.10, a spatial path planning method is presented that generates three-dimensional (3D) paths among obstacles. These 3D paths can be used as backbone curves for the hyper-redundant manipulators for obstacle avoidance in a spatial environment. In Chapter 5, controllers are introduced that use the desired trajectory of joint motions to calculate and apply driving forces or torque at the joints such that the physical motion can take place.

## 1.3 Mobile Robots

### *1.3.1 Common Types*

There are two common types of mobile robots that have well-designed kinematic drive chains, which reduce the chances of slip at the wheels of the robot. The two mobile robot types are Hilare-type and car-like mobile robots. Hilare-type robots have two independently driven wheels as the drive mechanism and are usually balanced by a passive caster wheel. They have good maneuvering abilities, e.g., a zero minimum turn radius, and are easier to control. They are also easier to build due to their simple drive mechanism. The Hilare-type mobile robots have different sizes and shapes depending on their application. Figure 1.3a shows a Hilare mobile robot with applications in industries, research, hospitals, and offices. Figure 1.3b shows another Hilare mobile robot used as an automated waiter. Figure 1.4 shows a robotic vacuum cleaner of Hilare type.

The Hilare-type mobile robots can have different sizes and shapes; however, since they share the same drive mechanism, their mathematical models are similar in structure. For example, the models, introduced in Chapter 6, can simply be adjusted to be useful for any Hilare-type mobile robot by using the physical parameters in the model corresponding to the robot, as long as the modeling assumptions are still valid.

Car-like mobile robots, as their name implies, have a drive mechanism similar to that of cars. They are driven by a single motor that powers a differential, which in turn distributes the motor's torque to the rear wheels. They have a steering mechanism at the front wheel(s), which is driven by a motor to generate steering angles

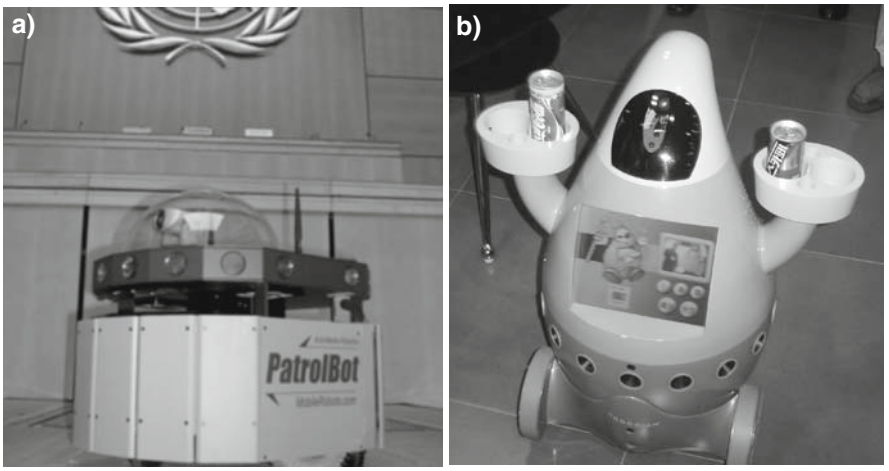


**Fig. 1.3** Hilare type-robots; (**a**) surveillance robot, (**b**) automated waiter
Source: (a) http://en.wikipedia.org/wiki/Image:PatrolBot.jpg;(b) http://en.wikipedia.org/wiki/Image:Seoul-Ubiquitous_Dream_11.jpg


**Fig. 1.4** A robotic vacuum cleaner
Source: http://en.wikipedia.org/wiki/Image:Roomba_Discovery.jpg

to steer the robot. An indoor small car-like mobile robot can be seen in Fig. 1.5. An outdoor mobile robot, developed for the 2007 DARPA Urban Challenge, a competition in which robotic ground vehicles had to drive autonomously in an urban situation, is shown in Fig. 1.6.

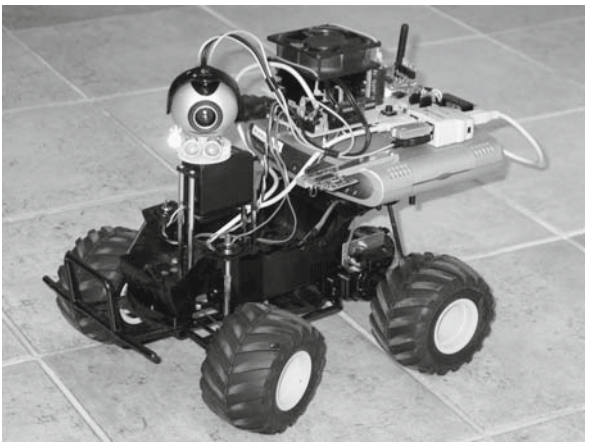

**Fig. 1.5** A car-like robot (Courtesy of Neurotechnologija, Vilnius, Lithuania)

**Fig. 1.6** A car-like robot developed by Team ENSCO for 2007 DARPA Urban Challenge
Source: http://en.wikipedia.org/wiki/Image:ElementBlack2.jpg

### *1.3.2 Applications of Mobile Robots*

Mobile robots have several applications in industries and factories. One can name transporting parts between gantries, conveyors, air tubes and other processes, transportation among non sequential processes, long-distance deliveries along winding and trafficked paths, and individualized item positioning at designated stations as the most common applications of mobile robots in industrial settings.

Many mobile robots are employed for environmental monitoring and inspection. Application examples are monitoring Heating, Ventilating, and Air Conditioning (HVAC) effectiveness, watching for hazards such as air quality, radon, radiation and smoke, checking on buildings and inspect trouble sites remotely to reduce emergency site visits, monitoring wifi reception and sniff for problem spots, and sending for supplies and equipment from a partner on the other side of the building. A mobile robot equipped with a closed circuit television can be used to detect intrusion and hazard in a building.

Mobile robots can be used as automated home helpers. Some of the robots that can be bought ready-made can map the environment where they are going to work by driving around randomly or via remote control. They can use this map to travel to any given point in the mapped environment and avoid obstacles in their way. The mapping feature of mobile robots are not only useful for the robot navigation, but are also accurate enough to be used as the measurement of an area. They can reproduce the actual shape of a room independent of the geometry of the room. They can follow people around and can vocally communicate with people if their path is

completely blocked! They can serve food on a tray or carry drinks in their mobile refrigerator.

Security and surveillance is another application of mobile robots. They can watch for intruders to a secure remote site or a vacation home and vocally issue warnings if they encounter intruders. They can be scheduled to move around and take snapshots as they move and send the snapshots to a designated receiver.

To perform the above mentioned functions, a mobile robot must be able to sense the environment boundary and obstacles, decide how to move from some point to the other (motion planning), and finally, control its driving mechanism such that the planned motion is actually executed in reality. Some of the more common theories and methods for motion planning and control of the two common types of mobile robots, Hilare-type and car-like, are introduced in Chapters 4 and 6.

## 1.4 Autonomous Surface Vessels

An autonomous surface vessel (ASV) is a robotic boat or ship that can react to the environmental changes and accomplish a task with minimum human interference. Similar to many advanced systems that have civilian use today, the ASVs development started for military applications. An unmanned surface vessel is shown in Fig. 1.7.

### *1.4.1 Military and Security Applications*

One of the military applications of ASVs is reconnaissance and surveillance in the open ocean and coastal waters. ASVs can furnish situational awareness to remote command stations in real-time. ASVs operate via remote control from a command center on a ship, or a plane, or on land, or autonomously. They can send pictures, video, and other electronic data to a land-based, airborne, or ship-based command center. ASVs can be powered by diesel engines, electric motors, or even with wind

**Fig. 1.7** Silver Marlin unmanned surface vessel with autonomous obstacle avoidance, manufactured by Elbit Systems (Courtesy of Oreet International Media Ltd.)

power. They can be designed for speed and can perform over the horizon, long-duration missions.

The ASVs are usually equipped with suites of ocean surveillance equipment, for example, stabilized infrared, thermal imaging, and live video cameras. These devices are uplinked with satellite or line-of-sight radio to transmit information to a control platform in real-time. There is no personnel on-board of ASVs. Therefore, they can intercept the targets of interest with minimal human and financial resources, freeing more expensive vessels, helicopters or aircraft for other mission assignments.

Autonomous Surface Vessels can play an important role in homeland security, drug control, and search and rescue missions. ASVs can patrol coastal waters, ports, and sensitive facilities and surveil for law enforcement and drug trafficking control. By cooperation with manned or unmanned aerial vehicles, ASVs can closely observe suspicious activities in important maritime locations and passages and issue early warning of any hostile or illegal activity to a command center. ASVs can protect the sovereignty of the remote lands with environmental resources. Groups of ASVs can be used for a long and tiring search and rescue missions.

The ASVs can replace the manned vessels that work in hazardous and dangerous situations. For example, during torpedo or missile exercises, and gun shoots, the safety and security of the areas downrange of the test must be ensured. Doing this using manned vessels, helicopters, or aircraft puts the humans in danger. ASVs can do this job without any risk to humans. They can be equipped and programmed for checking the target areas for unauthorized vessels, as well as endangered marine species. The ASV can provide real-time videos and collect other required data by being accurately positioned near intended weapon impact points. This eliminates the risk to personnel or more expensive observation platforms.

Another application for ASVs is providing mine countermeasures. The areas that are cleared from mines must be monitored to make sure that new mines are not re-seeded. Because of their small size and draft, undetectable electronic and noise footprint, and minimal effect on the radar, ASVs are highly suitable for monitoring sensitive areas such as channels, harbor entrances, and seashore. Such missions do not risk on-board personnel, since the ASV is unmanned.

### *1.4.2 Civilian Applications*

One can think of many civilian applications for ASVs. Examples are protection of maritime industrial assets and valuable shipments, protecting platforms for undersea gas exploration, ocean survey and mapping, metrological data collection, fishery support, marine biological research, and even recreational boating.

Open-sea operations are conducted by many commercial industries, who have many valuable mobile (container ships or oil tankers) or stationary assets (piplines on the sea bottom and off-shore oil rigs). ASVs can surveil the shipping lanes, bottlenecks, and sensitive areas for the mobile assets and can monitor the security of

areas both close and far from the stationary assets. These applications rely on the advanced sensors that can be mounted on an ASV to provide real-time images and sensory information and can make a control center aware of any threats.

The ASVs can benefit the undersea oil and mineral exploration in many ways. The topographical mapping of the ocean floor, which is the beginning of any exploration before excavation, is done by using sound waves. The mapping demands strict adherence to a predetermined course by the surface vessel. The ability of precise navigation for ASV make them a perfect tool for topographical mapping. Also, ASVs can stay stationary at a given point for an extended period of time without the need to be anchored or tethered. This feature allows an ASV to be used as a mobile weather buoy for collecting weather and hydrological data, and for supporting oceanographic investigations.

Fishery Support or compliance can be another application for ASVs. They can be equipped with sonar sensors and deployed to search for areas with higher population of fish. After locating the fish, ASVs can direct fishing vessels to the more fish-populated areas. Equally, ASVs can be used for surveillance in the areas where fishing is prohibited for protection of the ecosystem and inform the authorities if illegal fishing activities are recognized.

To design and deploy ASVs, several subsystems must be designed and the subsystems must be integrated. Examples of these subsystems are the different sensory systems needed for different applications of the ASV, the structural and dynamic stability, the engine and the driveline, the autonomy for independently making decisions, and the controls that actually make the ASV perform its task as planned. Some part of the autonomy of an ASV that relates to path planning can be addressed by using the methods introduced in Chapter 4, especially the two-dimensional (2D) obstacle avoidance method. The nonlinear controls applicable to an ASV is introduced in Chapter 7. Since many applications discussed in this subsection can be accelerated by using multiple ASVs, some part of Chapter 7 is dedicated to formation control, with which a group of ASVs can move together with user-specified distances for accomplishing a cooperative task.

## 1.5 Autonomous Helicopters

### *1.5.1 Research Platforms*

Several aspects of the autonomy for autonomous helicopters have been and are still under active research in universities and research centers. Since a real-size helicopter can be very expensive to purchase, maintain, and fly, most researchers have developed their own experimental platforms. These platforms are usually developed by adoption of remote control small-size model helicopters that are available for hobbyists and aerial photographers. The adapted platform is then modified and equipped with navigational sensors (GPS receivers, accelerometers, rate gyros, electronic compasses, etc.) and on-board embedded control computers. Some

researchers, on the other hand, have used the commercially available model helicopters that are ready for autonomous flight. Some of these helicopters had been used for crop spraying in Japan via remote control. Autonomous helicopters bought from companies are usually several times more expensive than the cost of the in-house development of autonomous helicopters with the same size and specifications. There are also some commercial platforms specifically developed for military applications (Fig. 1.8).

Several projects for development and experimentation with autonomous model helicopters started in the 1990s and is continuing still. Several autonomous helicopter projects were carried out in the University of Southern California (USC) since 1991. The USC presented prototypes for Autonomous Vehicle Aerial Tracking and Retrieval/Reconnaissance (AVATAR) in 1994 and 1997. The AVATAR was ranked first in the Association for Unmanned Vehicle Systems International (AUSVI) robotics competition in 1994.

Some in-house made and commercially available autonomous helicopter platforms have been modified and used by the Carnegie Mellon's Robotics Institute. Their autonomous helicopter succeeded to achieve the first place in the AUSVI's specialized aerial robotic competition. The AeRobot project, (BEAR in short), at the University of Berkeley is one of the famous autonomous helicopter projects. The BEAR team has used their autonomous helicopter as a test platform for evaluating the feasibility and performance of their integrated approach to intelligent systems. In the last several years, many aerial robots and autonomous helicopter platforms have also been designed and built by the Unmanned Aerial Vehicle (UAV) research facility in the Georgia Institute of Technology (GIT). The GIT also achieved the first ranking in the AUVSI's specialized aerial robotics competition.

**Fig. 1.8** MQ-8B unmanned helicopter used by the US Marine Corps and the US Navy, manufacturered by Northrop Grumman
Source: http://en.wikipedia.org/wiki/Image:MQ-8B_Fire_Scout.jpeg

Many European research centers work on autonomous helicopters and aerial robots. The Wallenberg laboratory for research on Information Technology and Autonomous Systems (WITAS) has had collaborative projects with private companies on UAV research, in which commercial autonomous helicopters have been used. Furthermore, many European Universities have adapted the conventional radio controlled helicopters for experimention on coordination and control of multiple heterogeneous vehicles. One can name the Universidad Polit'cnica de Madrid and the Technical University of Berlin for their autonomous helicopter platforms, whose helicopter won the AUSVI's aerial robotics competitions in 2000.

### *1.5.2 Civilian Applications*

Unmanned small-size helicopters already have numerous civilian applications. These unmanned, helicopters are mostly remotely controlled, especially for take-off and landing. However, they extensively use computer-controlled mode for stabilization of the helicopter in hover and near hover maneuvers. Although the unmanned helicopters used for commercial applications are not autonomous (computer-stabilized would be a more accurate term), in the future, they will take advantage of the research being done about autonomy to accomplish their tasks easier and more accurately. Some of these applications are presented below.

Perhaps aerial photography and videography is the most common application of unmanned helicopters. The customers of such a service are as follows: real estate agencies who need the aerial photographs and videos of estates for promotional purposes and for providing virtual tours of inaccessible sites; publishers who use pictures of landscapes, bridges, tall buildings, etc., in their publications; movie companies who can film stunts and overhead angles with much lower cost than that of using a full-size helicopter; news companies who can use them for traffic monitoring and aerial coverage of news events.

Another common application of unmanned helicopters are for inspection and security. Regular aerial survey of properties, aerial inspection of bridges, utility towers, powerlines, piplines, rail roads, and structures are examples of the use of unmanned helicopters. Furthermore, the unmanned helicopters are used for search and rescue missions for natural disaster survivors, surveillance of sensitive suspected areas for criminal activities, and patrolling political boarders for suspicious traffics. They can be used for remote sensing and monitoring of biological, chemical, and nuclear weapons.

The aerial robots have some agricultural applications such as crop dusting, crop health monitoring, and mapping using infrared cameras. They can be used for carrying supplies from ground to higher floors, providing temporary and immediate platforms for communications, and for delivering mail to remote areas.

### *1.5.3 Security and Military Applications*

Drug control and search and rescue missions are ideal applications for autonomous helicopters. Autonomous helicopters can patrol boarders, ports, and sensitive

facilities and surveil for law enforcement. They can closely monitor suspicious activities in important border sections and passages and report any suspected hostile or illegal activity. They can be used to protect the sovereignty of the remote lands with environmental resources. Cooperative helicopters forming a group can be used for long and tiring search and rescue missions.

Reconnaissance and surveillance in remote areas is one of the most common military applications of autonomous helicopters. Autonomous helicopters can provide situational awareness in real-time via sending pictures, video, and other electronic data to a land-based or ship-based command center.

### *1.5.4 Mathematical Models and Methods*

One part of autonomy of an autonomous helicopter is its ability to plan its trajectory in an environment with obstacles. The material presented in Section 4.10 is one of the efficient ways for path planning for aerial robots working in environments with obstacles. After the trajectory of an autonomous helicopter is determined by path planning methods, there is a need for a closed-loop feedback control algorithm to ensure that the helicopter performs the calculated desired trajectory to avoid obstacles. The material presented in Chapter 8 are useful for that purpose. Also, one can see that many of the applications of autonomous helicopters mentioned in this section can be accelerated by using multiple cooperative autonomous helicopters. A part of Chapter 8 discusses methods with which the motion of a group of cooperative helicopters can be coordinated.

## 1.6 Summary

The types and application of different autonomous robots were discussed in this chapter. These types of robots are interdisciplinary machines and because of that many researchers and engineers study them in different contexts. Each of these contexts relate to one important part of what makes an autonomous robot. For example, mission planning, machine vision, global path planning and obstacle avoidance, cooperative work of multiple robots, human-machine interaction, local path planning and obstacle avoidance, middle level trajectory-tracking controller development, low-level actuator controller design, navigational sensors, and platform development are all active fields of research. Each of these fields have been approached differently by researchers and engineers. It is close to impossible to address all these fields and methods in one book. Therefore, this book aims at covering some of the methods related to modeling, global and local path planning, obstacle avoidance, and rather advanced control of autonomous robots as mechanical systems.

# Chapter 2
# Redundant Manipulators

## 2.1 Introduction

In the last decade, redundant manipulators have been the subject of study for many researchers and engineers.[1] While most manipulators have enough DOFs to perform tasks in their end-effector task space, that is, providing any desired pose (position and orientation), their workspace is limited due to mechanical constraints on joints and obstacles that may be present in the work area. Redundant manipulators have extra DOFs compared to the minimum DOFs required for reaching their task space. This allows the redundant manipulators to perform tasks that require high dexterity. They can use the extra DOFs in their benefit to avoid their joint limits and the obstacles in the workspace, while still reaching a desired end-effector pose in the task space.

Avoiding joint limits [70] and obstacles [19, 5] are two of the features of redundant manipulators that have been exploited more often. However, the dexterity of this kind of manipulators can be used to satisfy any desirable kinematic or dynamic characteristic. An example of desired kinematic characteristics is posture control [17], in which the manipulator is programmed to choose a desired set of poses out of all possible poses that can perform a desired task. Examples of desired dynamic characteristic are controlling the contact force of the end-effector [56] or selecting poses that have optimum inertia [71].

The dexterity of redundant manipulators is sometimes comparable to that of a human arm. Redundant manipulators are employed in very important applications where dexterity is required. Perphaps, one of the most famous applications of redundant manipulators is in the International Space Station, where the Special Purpose Dexterous Manipulator (SPDM) (also known as Dexter or Canada Arm 2 in short) is being employed.

Note that the extra DOFs of a redundant manipulator make it kinematically different than a non redundant one. Therefore, the mathematical methods developed

[1] The theories in chapter are strongly based on the second chapter of the book titled: "Control of Redundant Manipulators: Theory and Experiments," by R. V. Patel and F. Shadpey, Springer-Verlag Berlin Heidelberg 2005.

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_2

for non redundant manipulators are not applicable to a redundant one. Specifically, the "inverse kinematic" problem for a redundant manipulator has multiple solutions in general. Methods that deal with the multiple solutions of the "inverse kinematic" problem for redundant manipulators and can find the best solution that satisfies a desired criteria are known as "redundancy resolution" methods.

In this chapter, first, the kinematics of redundant manipulators is introduced. Then, the most common methods for redundancy resolution are discussed. Finally, the performance of different redundancy resolution methods are studied from two different view points of robustness with respect to algorithmic and kinematic singularity, and flexibility with respect to incorporation of different additional desired tasks, e.g., obstacle avoidance or joint limit avoidance.

### *2.1.1 Kinematics of Redundant Manipulators*

For a manipulator, the task space is the space that defines the pose (position and orientation) of the end-effector. For example, for a manipulator whose end-effector moves in a plane, the end-effector pose can be defined by two position components and one orientation angle. Hence, the task space dimension is three. The joint space for a manipulator is comprised of all the variables that define the configuration of the joints.

For a redundant manipulator, there are more joint variables than there are DOFs for the end-effector. In other words, when the dimension of the task space $m$ for a manipulator is larger than the dimension of the joint space $n$, the manipulator is said to be redundant.

Normally, the variables that define the pose of the end-effect with respect to a fixed frame of reference are gathered in one single vector as the end-effector pose. Here, this vector is denoted by the $(m \times 1)$ vector $\mathbf{x}$. Also, the variables that define the configuration of the joints are organized in a vector. Here, this vector is named $\mathbf{q}$, (which is $n \times 1$). The difference of the joint space dimension and the task space dimension is called the degree of redundancy, that is, $r = n - m, (r \geq 1)$ is the degree of redundancy.

As one can guess, the pose of the end-effector in space depends on the configuration of the joints. Mathematically, this can be expressed as a functional relation between the end-effector pose vector $\mathbf{x}$ and the joint variables vector $\mathbf{q}$ as

$$\mathbf{x} = \mathbf{f}(\mathbf{q}). \tag{2.1}$$

This relation is known as the forward kinematics relation. Also, the (linear and angular) velocity components for the end-effector can be related to the rate of change of the joint variables. This relation can be expressed in mathematical terms as

$$\dot{\mathbf{x}} = \mathbf{J}_e(\mathbf{q})\dot{\mathbf{q}}, \tag{2.2}$$

where $\dot{\mathbf{x}}$ contains the linear and angular velocity components of the end-effector, and $\mathbf{J}_e$ is the $(m \times n)$ Jacobian of the end-effector. Equation (2.2) is known as the differential kinematics of the manipulator.

Equation (2.2) has an interesting mathematical interpretation. All the possible joint variable velocities $\dot{\mathbf{q}}$ form an $n \times 1$-dimensional mathematical space that is a subset of $\Re^n$. Also, all the possible end-effector velocity vectors $\dot{\mathbf{x}}$ form an $m \times 1$-dimensional mathematical space that is a subset of $\Re^m$. Here, $\Re$ is a set of real numbers. With these definitions, at any fixed $\mathbf{q}$, the Jacobian matrix $\mathbf{J}_e(\mathbf{q})$ can be interpreted as a linear transformation that maps vectors from the space $\Re^n$ into the space $\Re^m$.

Similar to any other linear transformation, the input space $\Re^n$ of the Jacobian matrix has two important associated subspaces. These two subspaces are called the range and the null space (Fig. 2.1). The range of the Jacobian matrix is the subspace of $\Re^n$ that is covered by the transformation. Physically, these are joint velocities that are mechanically possible to be generated by the manipulator's drive mechanism. The range denoted by $\Re(\mathbf{J}_e)$ is mathematically defined by

$$\Re(\mathbf{J}_e) = \{\mathbf{J}_e\dot{\mathbf{q}} | \dot{\mathbf{q}} \in \Re^n\}. \tag{2.3}$$

The null space of the Jacobian matrix is a subset of the input space $\Re^n$ that is mapped to a zero vector in the output space $\Re^m$ by the Jacobian matrix. Physically, these are the achievable joint velocities that do not generate any velocity at the end-effector. The null space of the Jacobian matrix is denoted by $\aleph(\mathbf{J}_e)$ and can be mathematically defined by

$$\aleph(\mathbf{J}_e) = \{\dot{\mathbf{q}} \in \Re^n | \mathbf{J}_e\dot{\mathbf{q}} = \mathbf{0}\}. \tag{2.4}$$

More information about the mathematical definition of the null space can be found in Section A.1.

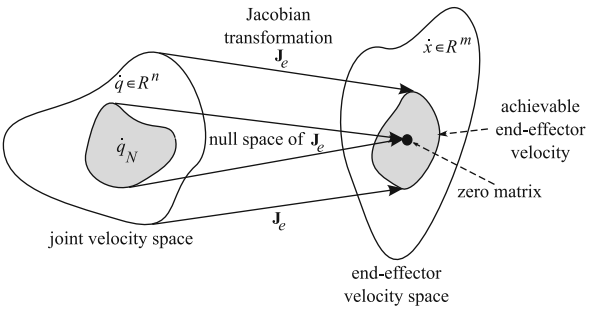


**Fig. 2.1** The Jacobian matrix $\mathbf{J}_e$ maps the joint velocity space onto the end-effector velocity space. The null space of the Jacobian matrix $\aleph(\mathbf{J}_e)$ maps a portion of the joint velocity space $\dot{\mathbf{q}}_\aleph$ onto zero end-effector velocity

The existence of the null space, as defined in Eq. (2.4), for the Jacobian matrix is the underlying mathematical basis for redundant manipulators. Physically, Eq. (2.4) implies that the velocities $\dot{\mathbf{q}}_{\aleph}$ picked from the null space $\aleph(\mathbf{J}_e)$ do not generate any velocity $\dot{\mathbf{x}}$ at the end-effector, i.e.,

$$\mathbf{J}_e \dot{\mathbf{q}}_{\aleph} = \mathbf{0}. \tag{2.5}$$

Although the velocities $\dot{\mathbf{q}}_{\aleph}$ do not generate any motion at the end-effector, they generate internal joint motions. Therefore, these velocities can be used to satisfy any requirement that the redundant manipulator must meet, for example, obstacle avoidance for the links, while the end-effector is performing its main task without being disturbed. This can be mathematically described as follows. Consider a desired end-effector velocity $\dot{\mathbf{x}}^d$ that can be generated by applying the joint rates $\dot{\mathbf{q}}^d$. This implies that

$$\dot{\mathbf{x}}^d = \mathbf{J}_e \dot{\mathbf{q}}^d. \tag{2.6}$$

Now, assume that the joint velocities $\dot{\mathbf{q}}_{\aleph}$ are selected from the null space $\aleph(\mathbf{J}_e)$ by an algorithm. The joint velocities $\dot{\mathbf{q}}^d + \alpha \dot{\mathbf{q}}_{\aleph}$, where $\alpha$ is a scalar multiplier, still generate the desired end-effector velocity because

$$\mathbf{J}_e(\dot{\mathbf{q}}^d + \alpha \dot{\mathbf{q}}_{\aleph}) = \mathbf{J}_e \dot{\mathbf{q}}^d + \mathbf{0} = \dot{\mathbf{x}}^d. \tag{2.7}$$

The dimension of the null space from which $\dot{\mathbf{q}}_{\aleph}$'s can be selected depends on the rank of the Jacobian matrix. If the Jacobian matrix $\mathbf{J}_e(\mathbf{q})$ has full column rank (see Section A.2) at a given joint position $\mathbf{q}$, then the dimension of the null space $\aleph(\mathbf{J}_e)$ is equal to the degree of redundancy. If the Jacobian matrix has a rank of $m' < m$, the dimension of $\aleph(\mathbf{J}_e)$ is equal to $(n - m')$.

Since the choice of velocities that belong to the null space is not unique, there are several ways in which the desired main task $\dot{\mathbf{x}}^d$ can be achieved. In other words, there are multiple solutions to the inverse kinematics problem for a redundant manipulator. These multiple solutions can be used wisely to the benefit of the user. To wisely use these multiple solutions, useful additional constraints can be defined. There are two approaches for defining additional constraints: global and local. Global approaches achieve optimal behavior along the whole trajectory which ensures superior performance over local methods [64, 44, 76]. However, their computational burden makes them unsuitable for real-time sensor-based manipulator control applications. For that reason, here, the local approaches, which lead to local optimal behavior, are discussed.

*Example 2.1.* Consider a planar Prismatic-Revolute-Revolute (PRR) 3-DOF manipulator with joint variables $q_1$, $q_2$, and $q_3$ (Fig. 2.2). The Cartesian coordinates of the end-effector $x_1$ and $x_2$ are assumed as the task space with two dimensions. The link lengths for the second and third links are $l_2$ and $l_3$, respectively.

(a) Determine the degree of redundancy of this manipulator.
(b) Derive the Jacobian matrix for this manipulator.

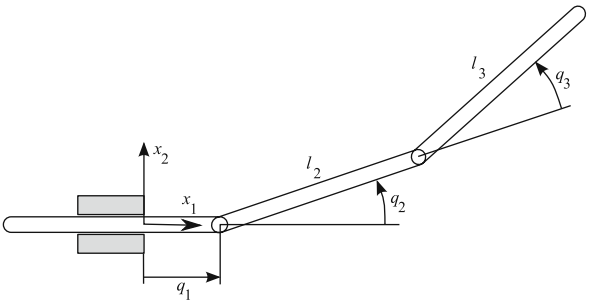


**Fig. 2.2** A redundant Prismatic-Revolute-Revolute (PRR) manipulator

*Solution.* The joint space vector and the task space vector are defined as $\mathbf{q} = [q_1, q_2, q_3]^T$ and $\mathbf{x} = [x_1, x_2]^T$, respectively.

(a) The dimension of the joint space and the task space are $n = 3$ and $m = 2$, respectively. Therefore, the degree of redundancy is $r = n - m = 1$.
(b) To find the Jacobian, first, the position of the end-effector is written as a function of joint parameters. By observing the geometry of the manipulator, one can write

$$x_1 = q_1 + l_2 \cos(q_2) + l_3 \cos(q_2 + q_3), \tag{2.8}$$
$$x_2 = l_2 \sin(q_2) + l_3 \sin(q_2 + q_3), \tag{2.9}$$

which can be written in the matrix form of Eq. (2.1) as

$$\mathbf{x} = \mathbf{f}(\mathbf{q}) = \begin{bmatrix} q_1 + l_2 \cos(q_2) + l_3 \cos(q_2 + q_3) \\ l_2 \sin(q_2) + l_3 \sin(q_2 + q_3) \end{bmatrix}. \tag{2.10}$$

Differentiating the above equations with respect to time yields

$$\dot{x}_1 = \dot{q}_1 - l_2 \dot{q}_2 \sin(q_2) - l_3(\dot{q}_2 + \dot{q}_3) \sin(q_2 + q_3), \tag{2.11}$$
$$\dot{x}_2 = l_2 \dot{q}_2 \cos(q_2) + l_3(\dot{q}_2 + \dot{q}_3) \cos(q_2 + q_3), \tag{2.12}$$

which can be written in the matrix form of Eq. (2.2) as

$$\dot{\mathbf{x}} = \mathbf{J}_e \dot{\mathbf{q}}, \tag{2.13}$$

where

$$\mathbf{J}_e = \begin{bmatrix} 1 & -l_2 \sin(q_2) - l_3 \sin(q_2 + q_3) & -l_3 \sin(q_2 + q_3) \\ 0 & +l_2 \cos(q_2) + l_3 \cos(q_2 + q_3) & +l_3 \cos(q_2 + q_3) \end{bmatrix}. \tag{2.14}$$

This completes the solution to the example.

## 2.2 Redundancy Resolution at the Velocity Level

Usually, in the real world applications of manipulators, the desired trajectory (timed position and orientation) of the end-effector is defined as the main task. For the control of the manipulator, however, the trajectory of the joint variables are required. Therefore, the solution to the inverse kinematics problem (more commonly referred to as redundancy resolution for redundant manipulators) is necessary. At the first step of redundancy resolution, the solution is done in the velocity level, that is, the desired joint rates that generate a desired velocity for the end-effector are calculated. This is known as the redundancy resolution at the velocity level.

The reader should be reminded that the redundancy resolution for a redundant manipulator is not trivial. Because of the kinematic redundancy, there are always more unknown joint velocities than there are equations. In this section, the mathematical methods that allow the solution for redundant manipulators at the velocity level are presented and discussed. The mathematical methods can be categorized under two types of exact and approximate solution methods.

### *2.2.1 Exact Solutions*

The pseudo-inverse method and the augmented Jacobian are two exact solution methods that are discussed in this section.

#### 2.2.1.1 Pseudo-Inverse Method

Here, a joint instantaneous velocity $\dot{\mathbf{q}}$ must be found such that it generates a given desired instantaneous velocity $\dot{\mathbf{x}}$ at the end-effector of a redundant manipulator. The instantaneous joint velocities can be found by seeking the exact solution of Eq. (2.2) for $\dot{\mathbf{q}}$ for a given $\dot{\mathbf{x}}$. One of the methods used for obtaining this exact solution is finding the pseudo-inverse of the matrix $\mathbf{J}_e$, denoted by $\mathbf{J}_e^{\dagger}$, and using it as

$$\dot{\mathbf{q}}_p = \mathbf{J}_e^{\dagger}\dot{\mathbf{x}}, \tag{2.15}$$

where the subscript $p$ indicated that this is the primary solution to Eq. (2.2). This solution can be later enhanced by adding solutions $\dot{\mathbf{q}}_{\aleph}$ from the null space of the Jacobian matrix $\mathbf{J}_e$. The pseudo-inverse of $\mathbf{J}_e$ can be written as

$$\mathbf{J}_e^{\dagger} = \mathbf{v}\boldsymbol{\sigma}^{*}\mathbf{u}^{T}, \tag{2.16}$$

where $\boldsymbol{\sigma}$, $\mathbf{v}$, and $\mathbf{u}$ are obtained from the singular-value decomposition (SVD) of $\mathbf{J}_e$ [35], and $\boldsymbol{\sigma}^{*}$ is the transpose of $\boldsymbol{\sigma}$ with all the non-zero values reciprocated (see Section A.3 for more details). Equation (2.16) can also be written in the following summation form

$$\mathbf{J}_e^{\dagger} = \sum_{i=1}^{m'} \frac{1}{\sigma_i} \hat{\mathbf{v}}_i \hat{\mathbf{u}}_i^T, \tag{2.17}$$

where $m'$ is the number of nonzero diagonal components of the matrix $\boldsymbol{\sigma}$, and $\sigma_i$ is the $i$-th nonzero diagonal element of the matrix $\boldsymbol{\sigma}$, and $\hat{\mathbf{v}}_i$ and $\hat{\mathbf{u}}_i$ are the $i$-th column of the matrices $\mathbf{v}$ and $\mathbf{u}$, respectively.

If $\mathbf{J}_e$ has full row rank, then its pseudo-inverse is given by

$$\mathbf{J}_e^{\dagger} = \mathbf{J}_e^T (\mathbf{J}_e \mathbf{J}_e^T)^{-1}. \tag{2.18}$$

The pseudo-inverse method can provide a solution independent of any unbalance in the number of equations and unknowns in Eq. (2.2). In other words, even if the forward kinematic equation is under-specified, square, or over-specified, the pseudo-inverse method can easily specify a solution. However, there are some disadvantages in using this simple method alone.

For example, as mentioned before, Eq. (2.15) only provides the primary solution, which is not in the null space of the Jacobian $\mathbf{J}_e$. This means that the redundancy of the manipulator, which has extra DOFs, cannot be exploited for any useful purpose that could be defined as additional task by a user. This problem can be solved by adding a joint velocity vector $\dot{\mathbf{q}}_{\aleph}$ that belongs to the null space of the Jacobian matrix $\mathbf{J}_e$ to the primary solution as [25]

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_p + \dot{\mathbf{q}}_{\aleph}. \tag{2.19}$$

As shown in Eq. (2.7), the new joint velocity $\dot{\mathbf{q}}$ still satisfies Eq. (2.2). The term $\dot{\mathbf{q}}_{\aleph}$ can be selected as

$$\dot{\mathbf{q}}_{\aleph} = (\mathbf{I} - \mathbf{J}_e^{\dagger} \mathbf{J}_e) \boldsymbol{\nu}, \tag{2.20}$$

where $\boldsymbol{\nu}$ is an arbitrary $n$-dimensional vector, which will be wisely chosen to satisfy a desired additional task. This desired additional task can be torque and acceleration minimization [70], singularity avoidance [63], or obstacle avoidance [19, 5]. To achieve any of these additional tasks, a cost function can be defined $\Phi(\mathbf{q})$, whose optimum value would ensure the desired additional task. Then, the arbitrary vector $\boldsymbol{\nu}$ can be selected such that the solution given by Eq. (2.19) converges to the optimum value of the cost function. This can be done by choosing the arbitrary vector as

$$\boldsymbol{\nu} = -\nabla \Phi(\mathbf{q}) = -\frac{\partial \Phi}{\partial \mathbf{q}} = -[\frac{\partial \Phi}{\partial q_1} \quad \dots \quad \dots \quad \frac{\partial \Phi}{\partial q_n}]^T. \tag{2.21}$$

Another problem with the solutions provided by Eq. (2.15) is that they may lead to singular configurations for the manipulator, at which the Jacobian matrix $\mathbf{J}_e$ does not have full rank [59]. At those singular configurations, the end-effector of the manipulator cannot generate velocity components in certain directions, which is not

desirable. Close to a singular position, very large joint rates are needed to generate an end-effector velocity in certain directions. Mathematically, for a singular posture, even the largest element of the matrix $\boldsymbol{\sigma}$ is very close to zero. Since the reciprocals of the elements of $\boldsymbol{\sigma}$ appear in the matrix $\boldsymbol{\sigma}^*$ in Eq. (2.16) or (2.17), the joint rates resulting from Eq. (2.15) are very large.

*Example 2.2.* Consider the PRR redundant manipulator of Example 2.1 (Fig. 2.2). If the link lengths $l_2$ and $l_3$ are 0.5 m, find

(a) the joint rates that generate an end-effector velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s, if the arm is at a non singular posture $\mathbf{q} = [0.25\text{ m}, \pi/12\text{ rad}, \pi/3\text{ rad}]^T$;
(b) the joint rates that generate an end-effector velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s, if the arm is at the singular posture $\mathbf{q} = [0.25\text{ m}, \pi/2\text{ rad}, 0\text{ rad}]^T$.

*Solution.* The Jacobian matrix of Eq. (2.14) derived in Example 2.1 is used in this example.

(a) The Jacobian matrix (2.14) at the non singular posture is

$$\mathbf{J}_e = \begin{bmatrix} 1.0000 & -0.6124 & -0.4830 \\ 0.0000 & 0.6124 & 0.1294. \end{bmatrix} \tag{2.22}$$

As seen from the above matrix, at a non singular posture, the Jacobian matrix has full row rank (the rank of $\mathbf{J}_e = 2$, which is equal to the number of rows of the Jacobian, $n = 2$). In this situation, the expression $\mathbf{J}_e\mathbf{J}_e^T$ is non singular and the pseudo-inverse $\mathbf{J}_e^\dagger$ can be calculated from

$$\mathbf{J}_e^\dagger = \mathbf{J}_e^T(\mathbf{J}_e\mathbf{J}_e^T)^{-1}. \tag{2.23}$$

For $q_1 = 1.25$ m, $q_2 = \pi/12$ rad, and $q_3 = \pi/3$ rad, this yields

$$\mathbf{J}_e^\dagger = \begin{bmatrix} 0.8931 & 0.9974 \\ 0.0634 & 1.6345. \\ -0.3023 & -0.0072 \end{bmatrix} \tag{2.24}$$

Since $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s,

$$\dot{\mathbf{q}}_p = \mathbf{J}_e^\dagger\dot{\mathbf{x}}^d = \begin{bmatrix} 0.4466 \\ 0.0319 \\ -0.1511 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s.} \\ \text{rad/s} \end{matrix} \tag{2.25}$$

(b) At a singular posture, the Jacobian matrix does not have full row rank (rank of $\mathbf{J}_e < n$). Calculating the Jacobian matrix (2.14) confirms that its rank is 1 ($m' = 1$).

$$\mathbf{J}_e = \begin{bmatrix} 1.00 & -1.00 & -0.50 \\ 0.00 & 0.00 & 0.00 \end{bmatrix} \tag{2.26}$$

Since the Jacobian matrix does not have full row rank, the expression $\mathbf{J}_e\mathbf{J}_e^T$ is singular (has no inverse) and the method used in the previous part of this example fails. In such a situation, the pseudo-inverse of the Jacobian matrix must be calculated using the SVD method. In this method, three matrices $\mathbf{u}$, $\boldsymbol{\sigma}$, and $\mathbf{v}$ are found such that $\mathbf{u}\boldsymbol{\sigma}\mathbf{v} = \mathbf{J}_e$.[2]

$$\mathbf{u} = \begin{bmatrix} -1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{bmatrix} \quad \boldsymbol{\sigma} = \begin{bmatrix} 1.5000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix} \tag{2.27}$$

$$\mathbf{v} = \begin{bmatrix} -0.6667 & -0.7452 & -0.0000 \\ 0.6667 & -0.5963 & -0.4472 \\ 0.3333 & -0.2981 & 0.8944 \end{bmatrix}. \tag{2.28}$$

The matrix $\boldsymbol{\sigma}^*$ is

$$\boldsymbol{\sigma}^* = \begin{bmatrix} 0.6667 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \end{bmatrix}. \tag{2.29}$$

The pseudo-inverse of the Jacobian is calculated as follows

$$\mathbf{J}_e^\dagger = \mathbf{v}\boldsymbol{\sigma}^*\mathbf{u}^T, \tag{2.30}$$

which results in

$$\mathbf{J}_e^\dagger = \begin{bmatrix} 0.4444 & 0.0000 \\ -0.4444 & 0.0000 \\ -0.2222 & 0.0000 \end{bmatrix}. \tag{2.31}$$

Finally the joint rates are

$$\dot{\mathbf{q}}_p = \mathbf{J}_e^\dagger\dot{\mathbf{x}} = \begin{bmatrix} 0.2222 \\ -0.2222 \\ -0.1111 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s.} \\ \text{rad/s} \end{matrix} \tag{2.32}$$

This completes the solution to the example.

[2] The following command in MATLAB calculates the SVD matrices: $[\mathbf{u}, \boldsymbol{\sigma}, \mathbf{v}] = \text{SVD}(\mathbf{J}_e)$.

#### 2.2.1.2 Augmented Jacobian Method

In the augmented Jacobian method [27, 67], for a redundant manipulator with the degree of redundancy of $r = n - m$, $r$ additional tasks are defined. These additional tasks, which are a function of joint variables $\mathbf{q}$, are organized in an $r \times 1$ vector, represented by $\mathbf{z}$. Since the additional task $\mathbf{z}$ is a function of the joint variables vector $\mathbf{q}$, an $r \times n$ Jacobian matrix $\mathbf{J}_c$, known as the Jacobian of the additional task, can be defined that relates their rate of change as

$$\dot{\mathbf{z}} = \mathbf{J}_c \dot{\mathbf{q}}. \tag{2.33}$$

Equation (2.33) adds $r$ equations to the forward kinematics equations (2.2), which brings the total number of equations to $n$. Since there are $n$ unknown joint rates in $\dot{\mathbf{q}}$, as long as the derivative of the additional task $\dot{\mathbf{z}}$ is defined, the number of equations and unknowns are balanced. This can be mathematically expressed by defining an augmented task vector as

$$\mathbf{y} = \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix}, \tag{2.34}$$

and expressing the augmented task in terms of the joint rate vector as

$$\dot{\mathbf{y}} = \begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{z}} \end{bmatrix} = \mathbf{J}_a \dot{\mathbf{q}}, \tag{2.35}$$

where $\mathbf{J}_a$ is the $(n \times n)$ augmented Jacobian matrix,

$$\mathbf{J}_a = \begin{bmatrix} \mathbf{J}_e \\ \mathbf{J}_c \end{bmatrix}. \tag{2.36}$$

$\mathbf{J}_e$ and $\mathbf{J}_c$ are the $(m \times n)$ and $(r \times n)$ Jacobian matrices of the main and additional tasks, respectively. Once again, $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ are the task vectors of the main, augmented, and additional tasks, respectively.

Since the augmented Jacobian matrix in Eq. (2.35) is square, the solution for the joint rates $\dot{\mathbf{q}}$ can be simply found by using the inverse of $\mathbf{J}_a$. This is a really simple approach, however, there are two major disadvantages associated with this method [71].

Calculation of the inverse of the augmented Jacobian matrix is required in this method. For the inverse of the augmented Jacobian matrix to exist at all times, the additional tasks must be defined at all times. In other words, part-time additional tasks such as obstacle avoidance or joint limit avoidance that are defined based on some conditions that may not exist at all times cannot be used as additional tasks. Hence, this method is not suitable for part-time tasks.

Another disadvantage is that extra singularities can be introduced into the kinematics of the redundant manipulator by defining the additional task. This may be caused by extra singularities that are a consequence of possible rank deficiencies

of the additional task Jacobian $\mathbf{J}_c$ at certain postures. Or this can be caused by an unwanted conflict between the main and the additional task at certain postures, at which the rows of $\mathbf{J}_e$ or $\mathbf{J}_c$ become linearly dependent. This linear dependency, which leads to singularity in the matrix $\mathbf{J}_a$, is task dependent and very hard to predict. In this situation, the solution of Eq. (2.35) based on the inverse of the extended Jacobian $\mathbf{J}_a$ may result in instability near a singular configuration.

*Example 2.3.* Consider the PRR redundant manipulator of Example 2.1 (Fig. 2.2) at a posture $q_1 = 0.25\,\text{m}$, $q_2 = \pi/12$ rad, and $q_3 = \pi/3$ rad. If the end-effector's angular velocity is defined as an additional task, find the joint rates required to generate a velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s and an angular velocity of $\omega_3^d = \pi/12$ rad/s for the end-effector.

*Solution.* The additional task is defined based on the desired angular velocity of the end-effector as

$$\dot{z}_1 = \omega_3 = \dot{q}_2 + \dot{q}_3 = \mathbf{J}_c\dot{\mathbf{q}}, \tag{2.37}$$

where the additional task's Jacobian matrix is

$$\mathbf{J}_c = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix}. \tag{2.38}$$

The augmented Jacobian for the manipulator becomes

$$\mathbf{J}_a = \begin{bmatrix} \mathbf{J}_e \\ \mathbf{J}_c \end{bmatrix} = \begin{bmatrix} 1 & -l_2\sin(q_2) - l_3\sin(q_2+q_3) & -l_3\sin(q_2+q_3) \\ 0 & +l_2\cos(q_2) + l_3\cos(q_2+q_3) & +l_3\cos(q_2+q_3) \\ 0 & 1 & 1 \end{bmatrix}. \tag{2.39}$$

Substituting the values for $q_1$, $q_2$, and $q_3$ and inverting $\mathbf{J}_a$ yields

$$\mathbf{J}_a^{-1} = \begin{bmatrix} 1 & 0.2979 & 0.4483 \\ 0 & 2.0706 & -0.2679. \\ 0 & -2.0706 & 1.2679 \end{bmatrix} \tag{2.40}$$

The first time derivative of the augmented task $\mathbf{y}$ is

$$\dot{\mathbf{y}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z}_1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.0 \\ \pi/12 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{m/s}\,. \\ \text{rad/s} \end{matrix} \tag{2.41}$$

This results in the following joint rates

$$\dot{\mathbf{q}} = \mathbf{J}_a^{-1}\dot{\mathbf{y}} = \mathbf{J}_a^{-1}\begin{bmatrix} 0.5 \\ 0.0 \\ \pi/12 \end{bmatrix} = \begin{bmatrix} 0.6174 \\ -0.7010 \\ 0.3319 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s.} \\ \text{rad/s} \end{matrix} \tag{2.42}$$

This completes the solution to the example.

## 2.2.2 Approximate Solution Methods

### 2.2.2.1 Singularity Avoidance

Singular postures for a manipulator is not desirable because at a singular posture, the inverse kinematics of a manipulator does not have a meaningful solution. In practice, this means that close to a singular posture, generating a velocity component in certain directions at the end-effector of a manipulator requires very high joint rates, which are not physically possible for the joints to afford. A redundant manipulator can avoid singular postures by exploiting its extra DOFs than that required for a given main task. Here, the use of this feature of redundant manipulators is discussed.

If $\dot{\mathbf{x}}^d$ is the desired main task, the redundancy resolution problem can be formulated as finding the joint rate $\dot{\mathbf{q}}$ that approximately satisfies Eq. (2.2) by minimizing the cost function

$$F = \|\mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d\|^2. \tag{2.43}$$

To avoid the singularities, the weighted norm of the joint rates is added to the above cost function. That way, high joint rates are penalized, causing the manipulator not to move close to the singularity posture.

$$F = \|\mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d\|^2 + \|\lambda\dot{\mathbf{q}}\|^2 \tag{2.44}$$

This cost function can be expanded in the following form

$$\begin{aligned} F &= (\mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d)^T(\mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d) + \lambda^2\dot{\mathbf{q}}^T\dot{\mathbf{q}}, \\ &= \dot{\mathbf{q}}^T\mathbf{J}_e^T\mathbf{J}_e\dot{\mathbf{q}} - 2\dot{\mathbf{q}}^T\mathbf{J}_e^T\dot{\mathbf{x}}^d + (\dot{\mathbf{x}}^d)^T\dot{\mathbf{x}}^d + \lambda^2\dot{\mathbf{q}}^T\dot{\mathbf{q}}. \end{aligned} \tag{2.45}$$

The partial derivate of the cost function $F$ with respect to $\dot{\mathbf{q}}^T$ vanishes for $\dot{\mathbf{q}}$ that minimizes $F$.

$$\frac{\partial F}{\partial \dot{\mathbf{q}}^T} = 2(\mathbf{J}_e^T\mathbf{J}_e\dot{\mathbf{q}} + \lambda^2\dot{\mathbf{q}} - \mathbf{J}_e^T\dot{\mathbf{x}}^d) = 0 \tag{2.46}$$

Solving the partial derivative of the cost function $F$ for the unknown $\dot{\mathbf{q}}$ results in

$$\dot{\mathbf{q}}_\lambda = (\mathbf{J}_e^T\mathbf{J}_e + \lambda^2\mathbf{I})^{-1}\mathbf{J}_e^T\dot{\mathbf{x}}^d. \tag{2.47}$$

The solution to Eq. (2.47), which is unique, closely approximates the exact solution while avoids high joint rates.

One can compare this approximate result with the exact solution by studying the SVD of the exact pseudo-inverse (Eq. 2.17) and that of the coefficient matrix of $\dot{\mathbf{x}}^d$ in Eq. (2.47). The singular values of the exact and the approximate solution are

$$\frac{1}{\sigma_i}, \quad \frac{\sigma_i}{\sigma_i^2 + \lambda^2}, \tag{2.48}$$

respectively. The weight $\lambda$ is what makes the actual difference between the singular values of the two solutions. Since $\lambda \neq 0$, there are no singularities for the approximate solution. Also, if $\lambda$ is selected to be small, when the manipulator is far from a singular posture and $\sigma_i$'s are large, the singular values of the exact and approximate solutions are very close, causing close solutions for $\dot{\mathbf{q}}$. Furthermore, when the manipulator is close to a singular posture, the singular values have the same order as $\lambda$. In these cases, the weight $\lambda^2$ in the denominator reduces the potentially high norm joint rates.

*Example 2.4.* Consider the PRR redundant manipulator of Example 2.1. Using the approximate solution method, find the joint rates that generate a velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s for the end-effector when the manipulator is

(a) at a non singular posture of $q_1 = 0.25$ m, $q_2 = \pi/12$ rad, and $q_3 = \pi/3$ rad;
(b) at a singular posture of $q_1 = 0.25$ m, $q_2 = \pi/2$ rad, and $q_3 = 0$ rad.

*Solution.* Assume $\lambda = 0.1$.

(a) The Jacobian matrix derived in Eq. (2.14) at the non singular posture is

$$\mathbf{J}_e = \begin{bmatrix} 1.0000 & -0.6124 & -0.4830 \\ 0.0000 & 0.6124 & 0.1294 \end{bmatrix} \tag{2.49}$$

The approximate solution is

$$\dot{\mathbf{q}}_p = (\mathbf{J}_e^T \mathbf{J}_e + \lambda^2 \mathbf{I})^{-1} \mathbf{J}_e^T \dot{\mathbf{x}}^d = \begin{bmatrix} 0.4379 \\ 0.0239 \\ -0.1498 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s}, \\ \text{rad/s} \end{matrix} \tag{2.50}$$

which is close to the exact solution obtained in Example 2.1 (Eq. 2.25). The error in the end-effector's velocity introduced due to the approximate solution is

$$\mathbf{e}_{\dot{x}} = \mathbf{J}_e \dot{\mathbf{q}}_p - \dot{\mathbf{x}}^d = \begin{bmatrix} -0.0044 \\ -0.0048 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{m/s}. \end{matrix} \tag{2.51}$$

This error is negligible.

(b) The Jacobian matrix derived in Eq. (2.14) at the singular posture is

$$\mathbf{J}_e = \begin{bmatrix} 1.0000 & -1.0000 & -0.5000 \\ 0.0000 & 0.0000 & 0.0000 \end{bmatrix}. \tag{2.52}$$

The approximate solution is

$$\dot{\mathbf{q}}_p = (\mathbf{J}_e^T \mathbf{J}_e + \lambda^2 \mathbf{I})^{-1} \mathbf{J}_e^T \dot{\mathbf{x}}^d = \begin{bmatrix} 0.2212 \\ -0.2212 \\ -0.1106 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s}, \\ \text{rad/s} \end{matrix} \tag{2.53}$$

which is close to the exact solution obtained in Example 2.1 (Eq. 2.32). The error in the end-effector's velocity introduced due to the approximate solution is

$$\mathbf{e}_{\dot{x}} = \mathbf{J}_e\dot{\mathbf{q}}_p - \dot{\mathbf{x}}^d = \begin{bmatrix} -0.0022 \\ 0.0000 \end{bmatrix} \text{m/s}. \tag{2.54}$$

This error is negligible. Since the posture under consideration is a singular configuration for the PRR manipulator, the method would not result in a solution if $\lambda$ was assumed to be zero.

#### 2.2.2.2 Configuration Control

The configuration control is another approximate solution method. It is derived by using the same idea of minimization of a cost function as was used for singularity avoidance [69, 70, 68]. In the configuration control method, in addition to the main task $\dot{\mathbf{x}}$, an additional task $\dot{\mathbf{z}}$, and a singularity avoidance task are considered.

$$\dot{\mathbf{x}} = \mathbf{J}_e\dot{\mathbf{q}} \tag{2.55}$$

$$\dot{\mathbf{z}} = \mathbf{J}_c\dot{\mathbf{q}} \tag{2.56}$$

Note that there is no restriction on the dimension of the additional task unlike for the augmented Jacobian method of Section 2.2.1.2.

The desired velocities for the main and additional tasks are defined as $\dot{\mathbf{x}}^d$ and $\dot{\mathbf{z}}^d$, respectively. Then, the joint rates $\dot{\mathbf{q}}$ are found such that the error for the main and the additional tasks are minimized while high joint rates are penalized. To implement this idea, a cost function is defined as follows

$$F = (\mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d)^T\mathbf{W}_e(\mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d) + (\mathbf{J}_c\dot{\mathbf{q}} - \dot{\mathbf{z}}^d)^T\mathbf{W}_c(\mathbf{J}_c\dot{\mathbf{q}} - \dot{\mathbf{x}}^d) + \dot{\mathbf{q}}^T\mathbf{W}_v\dot{\mathbf{q}}, \tag{2.57}$$

where $\mathbf{W}_e(m \times m)$ , $\mathbf{W}_c(k \times k)$, and $\mathbf{W}_v(n \times n)$ are diagonal positive-definite weighting matrices that assign priority to the main, additional, and singularity avoidance tasks. In Eq. (2.57), the first term penalizes the error in the main task's velocity, the second term penalizes the error in the additional task's velocity, and the third term penalizes high joint rates, hence, causes the manipulator to avoid singularities.

The joint rates that minimize the cost function (2.57) can be found by equating the derivative of $F$ to zero. The derivate of the cost function is

$$\frac{\partial F}{\partial \dot{\mathbf{q}}^T} = 2(\mathbf{J}_e^T\mathbf{W}_e\mathbf{J}_e + \mathbf{J}_c^T\mathbf{W}_c\mathbf{J}_c + \mathbf{W}_v)\dot{\mathbf{q}} - 2(\mathbf{J}_e^T\mathbf{W}_e\dot{\mathbf{x}}^d + \mathbf{J}_c^T\mathbf{W}_c\dot{\mathbf{z}}^d). \tag{2.58}$$

The joint rates are

$$\dot{\mathbf{q}} = (\mathbf{J}_e^T\mathbf{W}_e\mathbf{J}_e + \mathbf{J}_c^T\mathbf{W}_c\mathbf{J}_c + \mathbf{W}_v)^{-1}(\mathbf{J}_e^T\mathbf{W}_e\dot{\mathbf{x}}^d + \mathbf{J}_c^T\mathbf{W}_c\dot{\mathbf{z}}^d). \tag{2.59}$$

Note that there is no restriction on the dimension of the additional task unlike for the augmented Jacobian method of Section 2.2.1.2. Therefore, the disadvantages of the augmented Jacobian method do not exist for the configuration control method. Any part-time additional task, for example, joint limit avoidance or obstacle avoidance, can be defined as the additional task. When the additional task is not active, for example, the joints are not close to their limits, there are not as many active tasks as the degree of redundancy ($k < r$). In these situations, Eq. (2.59) provides a solution similar to that of the singularity avoidance method. When the additional task is active, for example, when some of the joints are close to their limits, the number of active additional tasks can be larger than the degree of redundancy ($k > r$). In those cases, Eq. (2.59) gives the best solution that minimizes the cost function $F$.

Since there are no hard limitations on the dimension of the additional tasks, the method of configuration control is very powerful and flexible. Any desirable kinematics for the manipulator (such as posture control, joint limit avoidance, and obstacle avoidance [70]), or any dynamic measure of performance that can be formulated as a kinematic function of joint positions or rates (such as contact force, inertia, etc. [71]) can be used as an additional task.

*Example 2.5.* Consider the PRR redundant manipulator of Example 2.1 and the additional task introduced in Example 2.3. Assume that the main task is three times more important than the additional task and 30 times more important than singularity avoidance. Find the joint rates that generate a velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s and an angular velocity of $\omega_3^d = \pi/12$ rad/s for the end-effector, if the manipulator is at

(a) a non singular posture of $q_1 = 0.25$ m, $q_2 = \pi/12$ rad, and $q_3 = \pi/3$ rad;
(b) a singular posture of $q_1 = 0.25$ m, $q_2 = \pi/2$ rad, and $q_3 = 0$ rad.

*Solution.* The dimensions of the main task $\mathbf{x}$, the additional task $\mathbf{z}$, and the joint space are $m = 2, k = 1$, and $n = 3$, respectively. The weight matrices are defined as

$$\mathbf{W}_e = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}, \quad \mathbf{W}_c = \begin{bmatrix} 1 \end{bmatrix}, \quad \mathbf{W}_v = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}. \tag{2.60}$$

(b) Using Eq. (2.59) with $\mathbf{J}_e$ (Eq. 2.14) and $\mathbf{J}_c$ (Eq. 2.38) and $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s and $\dot{\mathbf{z}}^d = [\pi/12]$ rad/s results in

$$\dot{\mathbf{q}} = \begin{bmatrix} 0.5764 \\ -0.0283 \\ 0.2338 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s.} \\ \text{rad/s} \end{matrix} \tag{2.61}$$

The errors in the main and augmented tasks are

$$\dot{\mathbf{e}}_e = \mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d = \begin{bmatrix} -0.0192 \\ 0.0129 \end{bmatrix} \text{m/s} \\ \text{m/s'}, \tag{2.62}$$

$$\dot{\mathbf{e}}_c = (\dot{q}_2 + \dot{q}_3) - \dot{z}_1^d = -0.0562 \text{ rad/s}. \tag{2.63}$$

(b) Using Eq. (2.59) with $\mathbf{J}_e$ (Eq. 2.14) and $\mathbf{J}_c$ (Eq. 2.38) and $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s and $\dot{\mathbf{z}}^d = [\pi/12]$ rad/s results in

$$\dot{\mathbf{q}} = \begin{bmatrix} 0.5669 \\ -0.0373 \\ 0.2461 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s.} \\ \text{rad/s} \end{matrix} \tag{2.64}$$

The errors in the main and augmented tasks are

$$\dot{\mathbf{e}}_e = \mathbf{J}_e\dot{\mathbf{q}} - \dot{\mathbf{x}}^d = \begin{bmatrix} -0.0189 \\ 0.0000 \end{bmatrix} \text{m/s} \\ \text{m/s'}, \tag{2.65}$$

$$\dot{\mathbf{e}}_c = (\dot{q}_2 + \dot{q}_3) - \dot{z}_1^d = -0.0530 \text{ rad/s}. \tag{2.66}$$

This completes the solution to this example.

## 2.3 Redundancy Resolution at the Position Level

So far, different methods were studied with which required joint velocities can be determined such that a given velocity can be achieved at the end-effector of a redundant manipulator. These methods by themselves are not able to provide a set of joint positions that result in a desired postion for the end-effector. To find the joint postions required to bring the end-effector to a given position, the joint rates calculated by the redundancy resolution methods at the velocity level must be integrated. In this subsection, the integration procedure is presented.

The mathematical formulation of the redundancy resolution problem in the position level is described as finding $\mathbf{q}$ such that

$$\mathbf{x}^d = \mathbf{f}(\mathbf{q}), \tag{2.67}$$

where $\mathbf{x}^d$ is the desired position of the end-effector. Since this problem is to be solved by integrating the joint velocities, an initial condition is needed. In the physical sense, this initial condition represents the initial posture of the manipulator from which the motion toward the desired position starts. Assume that the initial posture of the manipulator is described by $\mathbf{q}_1$. At this initial posture, the end-effector is located at $\mathbf{x}_1$, where

$$\mathbf{x}_1 = \mathbf{f}(\mathbf{q}_1) \tag{2.68}$$

A path lying in the workspace of the manipulator must be assumed. The path starts from the initial position of the end-effector, $\mathbf{x}_1$, to its desired position, $\mathbf{x}_d$. The simplest assumption for this path is a line segment connecting the two positions in the Euclidean space. This line segment is divided into $N$ smaller line segments for numerical integration. At any integration step, the joint rates required to move the end-effector along this line are calculated. These joint velocities are integrated sequentially until the end-effector reaches the desired point. The integration algorithm follows.

1. Assume an initial posture, $\mathbf{q}_1$, for the manipulator and calculate the initial position of the end-effector, $\mathbf{x}_1$, from Eq. (2.68).
2. Plan a trajectory from $\mathbf{x}_1$ to $\mathbf{x}_{N+1} = \mathbf{x}^d$ with $N$ intervals, and assume the period of the motion, $T$.
3. Calculate a planned velocity at the interval $k$ that moves the end-effector toward the desired position as

$$\dot{\mathbf{x}}_k = \alpha \frac{\mathbf{x}^d - \mathbf{x}_k}{(N+1-k)\Delta t}, \quad \Delta t = \frac{T}{N}, \tag{2.69}$$

   where $\alpha$ is the deceleration factor greater than 1, which results in faster velocities at the beginning of the motion and slower velocities closer to the desired position.
4. Find the joint rates that generate the planned end-effector velocity at step $k$. Any of the methods discussed so far can be used for joint rate calculation, for example, the exact redundancy resolution method using a pseudo-inverse Jacobian matrix.

$$\dot{\mathbf{q}}_k = \mathbf{J}_e^{\dagger}(\mathbf{q}_k)\dot{\mathbf{x}}_k \tag{2.70}$$

5. Find $\mathbf{q}_k$ at the next interval by numerically integrating (2.70). Any method of integration can be used, for example the simple method of Euler.

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \dot{\mathbf{q}}_k \Delta t \tag{2.71}$$

6. Find the new end-effector position

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{q}_{k+1}) \tag{2.72}$$

7. Repeat steps 3 to 6 for $k = 1, \ldots, N$.

When the above algorithm runs to the end, when $k = N$, the last step represents $\mathbf{x}_{N+1} = \mathbf{x}^d = \mathbf{f}(\mathbf{q}_{N+1})$. This indicates that the joint posture corresponding to the last step, $\mathbf{q}_{N+1}$, is the solution to the redundancy resolution problem at the position level.

The solution to the redundancy resolution at the position level via integration can be used in two different ways. Note that the above algorithm actually results in the time history of the joint positions and velocities (joint trajectories) that cause the end-effector to move along the planned path from the initial position, $\mathbf{x}_1$, to the desired position, $\mathbf{x}^d$. If the planned path is important to the user, the history of the joint position and velocities derived from this algorithm must be used to control the manipulator on the planned path. If the planned path is not important to the user and only the desired position is important, simpler joint trajectories can be used to move the manipulator joints form the initial posture to the desired posture, $\mathbf{q}_{N+1}$, found at the $N$th step of the above algorithm.

*Example 2.6.* Consider the PRR redundant manipulator of Example 2.1 at an initial posture $\mathbf{q}_1 = [0, 0, 0]^T$. Find the trajectory of the joints required to move the end-effector of the manipulator from its initial position to the desired position $\mathbf{x}^d = [1.25, 0.25]^T$ m. Assume a period of $T = 10$ s for this motion. Use $N = 1,000$ integration intervals and a deceleration factor of $\alpha = 2$.

*Solution.* The redundancy resolution algorithm presented in this section is used to solve this example. The approximate redundancy resolution method with singularity avoidance (Eq. 2.47 with $\lambda = 0.1$) is used for calculating the joint rates at Step 4 of the algorithm. Application of the algorithm introduced in this section results in the joint trajectories shown in Fig. 2.3. As can be seen in Fig. 2.4, these joint trajectories cause the end-effector of the manipulator to move from the initial position $\mathbf{x}_1 =$

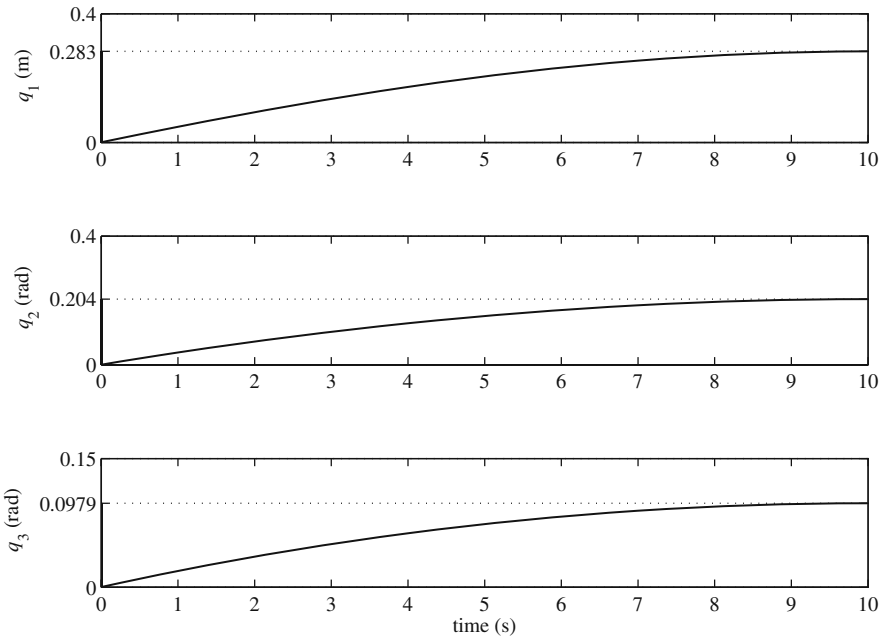


**Fig. 2.3** The joint trajectories for the PRR manipulator via the approximate solution method

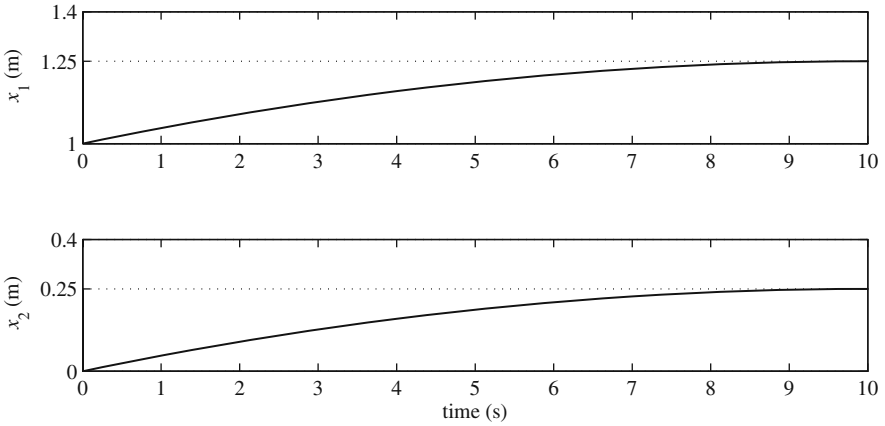


**Fig. 2.4** The end-effector trajectory for the PRR manipulator via the approximate solution method

$[1.0, 0.0]^T$ m to the desired position $\mathbf{x}^d = [1.25, 0.25]^T$ m in 10 s. Because of the application of the deceleration factor $\alpha$, the manipulator slows down when the end-effector gets closer to the desired position. The posture of the manipulator, $\mathbf{q}$, and the position of the end-effector, $\mathbf{x}$, at the end of the motion are

$$\mathbf{q} = \begin{bmatrix} 0.2830 \\ 0.2040 \\ 0.0979 \end{bmatrix} \begin{matrix} \text{m} \\ \text{rad}, \\ \text{rad} \end{matrix} \qquad \mathbf{x} = \begin{bmatrix} 1.2500 \\ 0.2500 \end{bmatrix} \begin{matrix} \text{m} \\ \text{m}. \end{matrix} \tag{2.73}$$

The manipulator at the end of the motion and the path of the end-effector are shown in Fig. 2.5. Since the approximate solution with a small $\lambda$ (0.1) has been used for redundancy resolution at Step 4 of the algorithm, the path of the end-effector is very close to the planned linear path from the initial to the desired position of the end-effector.

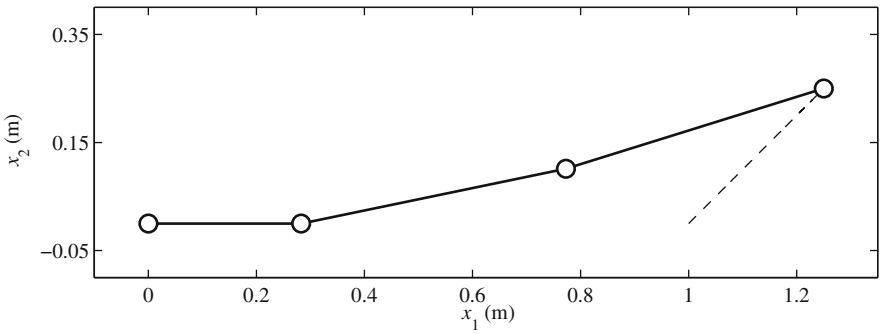


**Fig. 2.5** The final posture of the PRR manipulator and the path of the end-effector via the approximate solution method

## 2.4 Joint Limit Avoidance and Obstacle Avoidance

Joint Limit Avoidance (JLA) and Obstacle Avoidance are two important issues that have to be dealt with when planning the trajectory of the manipulator joints for a certain task at the position level. These two issues can be addressed by using the Configuration Control method, which can serve as a general framework for resolving redundancy. In this section, two general approaches for representing additional tasks are formulated.

### *2.4.1 Joint Limit Avoidance (JLA)*

There are two methods with which the joint limits of a redundant manipulator can be considered when finding joint trajectories that satisfy a given task.

#### 2.4.1.1 Inequality Constraints

In this method, the limits for the joints are defined by part-time constraints as additional tasks. These part-time additional tasks are active for a joint, when the joint position is close to the joint limit. When a joint position is far from the joint limit, the JLA additional task becomes inactive for that joint. In this case, the redundancy can be used for other additional tasks.

A JLA additional task is activated and deactivated by wisely selecting its corresponding weight matrix in the configuration control formulation ($\mathbf{W}_c$ in Eq. (2.59)). It is always a good idea to define a continuous weight for each joint to ensure a smooth joint trajectory. Usually, the weight of the JLA task for a joint is selected to be zero in a region of the joint motion around the center of the joint range. Then, a "buffer" region is assumed with a width $\tau_i$. When the joint position enters this region, the weight of the JLA task is increased from zero to a maximum at the lower limit ($q_{imin}$) or upper limit ($q_{i\,\max}$). The $i$-th diagonal entry of the weight matrix $\mathbf{W}_c$ corresponding to joint $i$ is

$$W_{c_{ii}} = \begin{cases} W_0 & \text{if } q_i < q_{i\,\min} \\ \frac{W_0}{2}[1+\cos(\pi(\frac{q_i - q_{i\,\min}}{\tau_i}))] & \text{if } q_{i\,\min} \le q_i \le q_{i\,\min} + \tau_i \\ 0 & \text{if } q_{i\,\min} + \tau_i < q_i < q_{i\,\max} - \tau_i \\ \frac{W_0}{2}[1+\cos(\pi(\frac{q_{i\,\max} - q_i}{\tau_i}))] & \text{if } q_{i\,\max} - \tau_i \le q_i \le q_{i\,\max} \\ W_0 & \text{if } q_i > q_{i\,\max} \end{cases}, \tag{2.74}$$

where $W_0$ is a user-defined constant representing the coefficient for the weight. This coefficient is normally selected much larger than that of the main task and the singularity avoidance task.

The weight matrix (2.74) is accompanied with an additional task. Since all the joints need to be monitored for the limits, the additional task is defined as a one-to-one function of the joint positions.

$$\mathbf{z} = \mathbf{q} \tag{2.75}$$

With this definition, the corresponding Jacobian for the additional task, $\mathbf{J}_c$, is defined by

$$\mathbf{J}_c = \frac{\partial \mathbf{z}}{\partial \mathbf{q}} = \mathbf{I}. \tag{2.76}$$

Also, since the joint rates must vanish when the joint limits are reached, the desired joint rates when the JLA additional task is active must be selected to be zero.

$$\dot{\mathbf{z}}^d = \mathbf{0} \tag{2.77}$$

The Jacobian for the additional task (2.76) and a weight matrix whose entries are defined by Eq. (2.74) are used with the configuration control method, representred by Eq. (2.59) for joint limit avoidance. An illustrative example follows.

*Example 2.7.* Consider the redundant PRR manipulator of Example 2.1. Assume a joint limit of $q_{2\max} = 0.1$ rad for the robot's second joint with an activation buffer of $\tau_2 = 0.02$ rad. If the manipulator is at the initial posture of $\mathbf{q}_1 = [0, 0, 0]^T$, find the joint trajectories that cause the end-effector to move to a desired position of $\mathbf{x}^d = [1.25, 0.25]^T$ m.

*Solution.* Since, in general, it could be assumed that all the joints have a limited range of motion, the additional task, $\mathbf{z}$, contains all the joint variables $q_1$, $q_2$, and $q_3$. Also, the desired additional task rate, $\dot{\mathbf{z}}^d$, for joint limit application is always zero. That is

$$\mathbf{z} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}, \quad \dot{\mathbf{z}}^d = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{2.78}$$

With the above definition, the Jacobian of the additional task is derived as

$$\mathbf{J}_c = \frac{\partial \mathbf{z}}{\partial \mathbf{q}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.79}$$

The diagonal additional task's weight matrix is chosen such that, in this example, only the second diagonal component is non zero, reflecting the fact that only the second joint is assumed to have limited range of motion.

$$\mathbf{W}_c = \begin{bmatrix} 0 & 0 & 0 \\ 0 & W_{c_{22}} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{2.80}$$

where

$$W_{c_{22}} = \begin{cases} 0 & \text{if } q_2 < 0.08 \\ \frac{50}{2}[1 + \cos(\pi(\frac{0.1-q_2}{0.02}))] & \text{if } 0.08 \leq q_2 \leq 0.1 \\ 50 & \text{if } q_2 > 0.1 \end{cases}. \tag{2.81}$$

To find the joint trajectories, the algorithm presented in Section 2.3 must be used. However, the above additional task must be incorporated in the algorithm. Furthermore, since the start posture is a singular configuration, a singularity avoidance task must also be incorporated into the algorithm. These are done by replacing the joint rate formula in Step 4 of the algorithm by Eq. (2.59). Since $\dot{\mathbf{z}}^d = [0, 0, 0]^T$, we have

$$\dot{\mathbf{q}}_k = [\mathbf{J}_e^T \mathbf{W}_e \mathbf{J}_e + \mathbf{J}_c^T \mathbf{W}_c \mathbf{J}_c + \mathbf{W}_v]^{-1} \mathbf{J}_e^T \mathbf{W}_e \dot{\mathbf{x}}_k, \tag{2.82}$$

in which the following weight matrices for the main task and the singularity avoidance task are assumed.

$$\mathbf{W}_e = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}, \quad \mathbf{W}_v = \begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}. \tag{2.83}$$

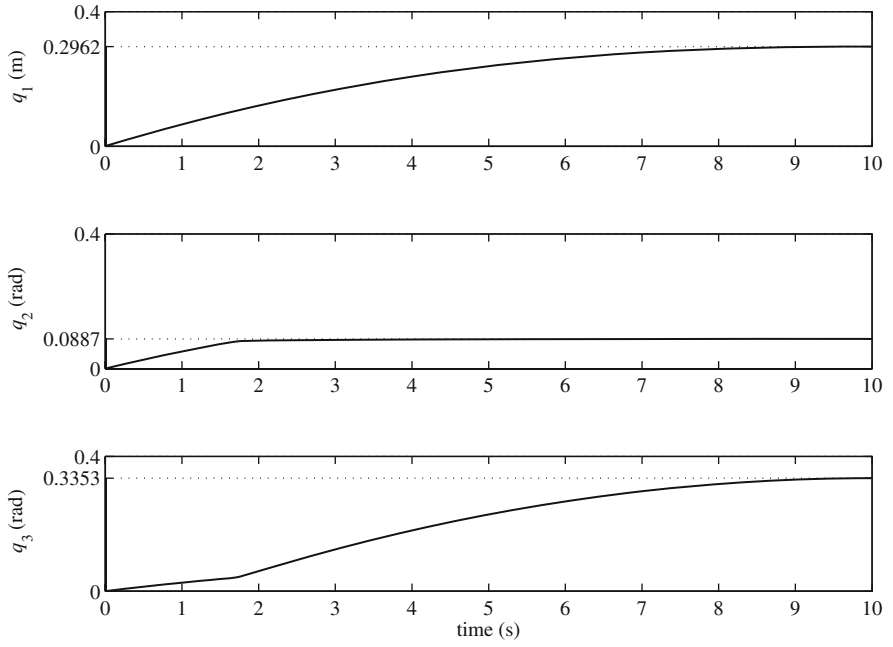


**Fig. 2.6** The joint trajectories for the PRR manipulator with JLA via inequality constraints

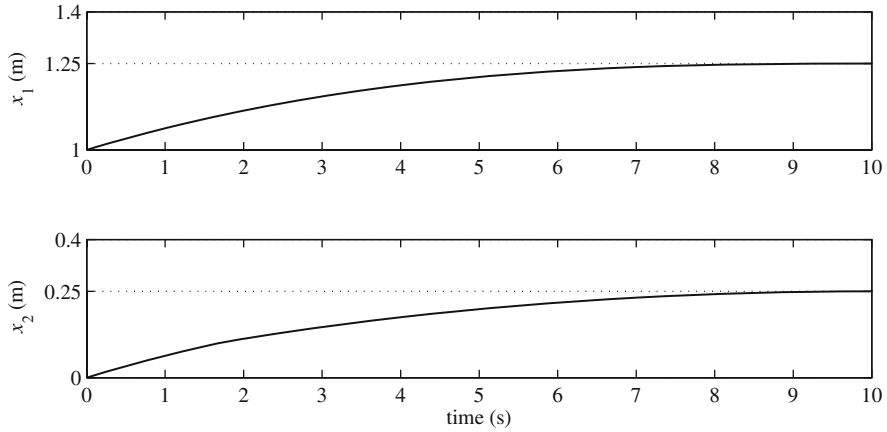


**Fig. 2.7** The end-effector trajectory for the PRR manipulator with JLA via inequality constraints

The application of the algorithm in Section 2.3 with the described modifications results in the joint trajectories shown in Fig. 2.6. Because of the application of the joint limit additional task, the second joint stops close to its limit at $q_2 \approx 0.0887$ rad while the other joints continue their motions until the desired end-effector position is achieved. As can be seen in Fig. 2.7, these joint trajectories cause the end-effector of the manipulator to move from the initial position $\mathbf{x}_1 = [1.0, 0.0]^T$ m to the desired position $\mathbf{x}^d = [1.25, 0.25]^T$ m in 10 s. Since a deceleration factor of $\alpha = 2$ has been used, the manipulator slows down when the end-effector gets closer to the desired position. The posture of the manipulator, $\mathbf{q}$, and the position of the end-effector, $\mathbf{x}$, at the end of the motion are

$$\mathbf{q} = \begin{bmatrix} 0.2962 \\ 0.0887 \\ 0.3353 \end{bmatrix} \begin{matrix} \text{m} \\ \text{rad}, \\ \text{rad} \end{matrix} \qquad \mathbf{x} = \begin{bmatrix} 1.2500 \\ 0.2500 \end{bmatrix} \begin{matrix} \text{m} \\ \text{m}. \end{matrix} \tag{2.84}$$

The manipulator at the end of the motion and the path of the end-effector are shown in Fig. 2.8. Since the weighted additional task and singularity avoidance has been introduced at Step 4 of the algorithm for redundancy resolution, the solution for the main task is approximate and the path of the end-effector is not too close to the planned linear path from the initial to the desired position of the end-effector.

#### 2.4.1.2 Kinematic Optimization

In the inequality constraint method, a part-time JLA task was defined. A disadvantage of that method is that the joints move freely until they are close to their limits. There is no sophisticated planning to prevent the joints from reaching their limits. Here, another method for JLA is discussed, in which the JLA task is a full-time task. This method tends to monitor the joints at all times to prevent them from getting close to their limits.

This method takes advantage of the null space of the Jacobian of a redundant manipulator. Normally, an exact solution to the redundancy resolution problem is

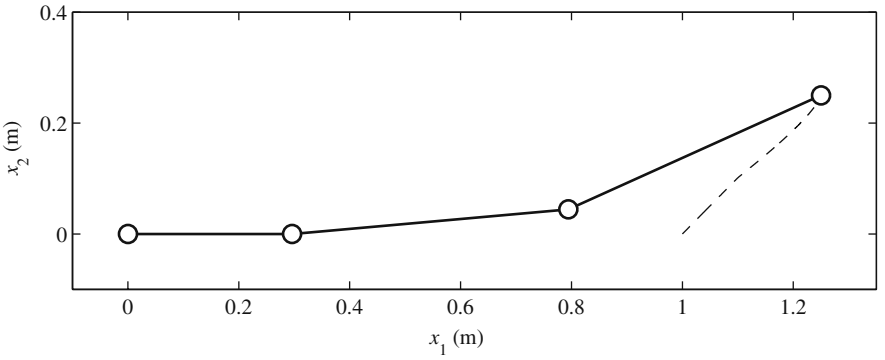


**Fig. 2.8** The final posture of the PRR manipulator and the path of the end-effector with JLA via inequality constraints

found using Eq. (2.15). This solution for the joint rates generates the desired motion at the end-effector. Then, the exact joint rate solution is completed by adding a joint rate solution belonging to the null space of the manipulator's Jacobian as shown by Eq. (2.19). The null space solution is found using Eq. (2.20) along with Eq. (2.21). The arbitrary vector $\boldsymbol{\nu}$ in the null space solution comes from optimizing a cost function.

Here, since the goal is to keep the joint far from their limits, a representative of the difference of the position of a joint $i$ to the center $q_{c_i}$ of the joint range $\Delta q_i$ is defined as a cost function to be minimized. The simplest form of such a function is the quadratic form

$$\Phi(\mathbf{q}) = \sum_{i=1}^{n} \left[ \frac{q_i - q_{c_i}}{\Delta q_i} \right]^2. \tag{2.85}$$

The JLA is now achieved by finding a $\boldsymbol{\nu}$ that optimizes Eq. (2.85), while the end-effector motion is generated by the exact solution. The cost function (2.85) tends to keep the trajectory for the joints around the center of their ranges at all times.

The JLA cost function can be defined with a different view. Similar to the JLA through the inequality constraint method, one may decide to only focus on the joint that is farthest from its center of the range compared to all other joints [48]. This can be translated into the following mathematical relation

$$\Phi(\mathbf{q}) = \max \frac{|q_i - q_{c_i}|}{\Delta q_i} = \| \frac{\mathbf{q} - \mathbf{q}_c}{\Delta \mathbf{q}} \|_\infty, \tag{2.86}$$

which is known as the infinity norm. Although concentrating on the joint closet to its limit using the infinity norm as a cost function is intuitive, it leads to a mathematical complication because the infinity norm is not differentiable. In mathematics, the

$p$-norm of a vector defined by

$$\|\mathbf{x}\|_p = \left( \sum_{i=1}^{n} |x_i|^p \right)^{1/p} \tag{2.87}$$

is an acceptable approximation for the infinity norm. Using the $p$-norm, a proper cost function for JLA via kinematic optimization can be defined as

$$\Phi(\mathbf{q}) = \| \frac{\mathbf{q} - \mathbf{q}_c}{\Delta \mathbf{q}} \|_p. \tag{2.88}$$

Theoretically, the higher $p$ is, the closer the cost function is to the infinity norm. However, $p = 6$ is sufficient for most practical cases. Also, in some practical cases, avoiding joint limits is more importance for certain joints. In such cases, a weighted version of Eq. (2.88) is used.

$$\Phi(\mathbf{q}) = \|\mathbf{K}(\frac{\mathbf{q} - \mathbf{q}_c}{\Delta \mathbf{q}})\|_p \tag{2.89}$$

where $\mathbf{K}$ is an $n \times n$ diagonal matrix.

*Example 2.8.* Consider the redundant PRR manipulator of Example 2.1 with a limited joint range for the second joint of $q_2 \in [-0.1, 0.1]$ rad. Use the 2-norm function to incorporate this joint motion limitation in the kinematic redundancy resolution via kinematic optimization. Assume that the manipulator is at the initial posture of $\mathbf{q}_1 = [0, 0, 0]^T$. Find the joint trajectories that cause the end-effector to move to the desired position $\mathbf{x}^d = [1.25, 0.25]^T$ m.

*Solution.* The 2-norm function for optimization is defined based on Eq. (2.89) as

$$\Phi = \|\mathbf{K}(\frac{\mathbf{q} - \mathbf{q}_c}{\Delta \mathbf{q}})\|_2 = \left( \sum_{i=1}^{3} |K_{ii} \frac{q_i - q_{c_i}}{\Delta q_i}|^2 \right)^{1/2}. \tag{2.90}$$

Since the joint limitation is assumed to exist only for the second joint, $K_{11}$ and $K_{33}$ are selected to be zero. Only $K_{22}$ is assigned a non zero value of 0.01. Furthermore, since the range of the motion of the second joint is $q_2 \in [-0.1, 0.1]$ rad, the center of the range is $q_{c_2} = 0$ and the range is $\Delta q_2 = 0.2$ rad. With these definitions, Eq. (2.90) is reduced to

$$\Phi = |K_{22} \frac{q_2 - q_{c_2}}{\Delta q_2}| = \begin{cases} K_{22} \frac{q_2 - q_{c_2}}{\Delta q_2} & \text{if } q_2 \geq q_{c_2} \\ K_{22} \frac{q_{c_2} - q_2}{\Delta q_2} & \text{if } q_2 < q_{c_2} \end{cases}. \tag{2.91}$$

The gradient of this cost function is needed for taking advantage of the null space of the manipulator's Jacobian to incorporate the joint limit via kinematic optimization. It is calculated as

$$\boldsymbol{v} = -\nabla\Phi = -[\frac{\partial\Phi}{\partial q_1}, \frac{\partial\Phi}{\partial q_2}, \frac{\partial\Phi}{\partial q_3}]^T = [0, -\frac{\partial\Phi}{\partial q_2}, 0]^T, \tag{2.92}$$

where

$$\frac{\partial\Phi}{\partial q_2} = \begin{cases} \frac{K_{22}}{\Delta q_2} & \text{if } q_2 \geq q_{c_2} \\ -\frac{K_{22}}{\Delta q_2} & \text{if } q_2 < q_{c_2} \end{cases}. \tag{2.93}$$

The algorithm presented in Section 2.3 must be used for redundancy resolution in the position level and finding the required joint trajectories. Note that the Step 4 of the presented algorithm, at which the joint rates are calculated, must be modified to address the joint limit avoidance via kinematic optimization. The joint rates required to perform the main task, $\dot{\mathbf{q}}_p$, are calculated via the approximate solution (Eq. 2.47)

$$\dot{\mathbf{q}}_p = (\mathbf{J}_e^T\mathbf{J}_e + \lambda^2\mathbf{I})^{-1}\mathbf{J}_e^T\dot{\mathbf{x}}^d, \tag{2.94}$$

which also includes singularity avoidance. A null space solution, $\dot{\mathbf{q}}_{\aleph}$ from Eq. (2.20), is added to the main task solution. This null space solution does not affect the end-effector's motion, however, it implements the joint limit requirement.

$$\dot{\mathbf{q}}_{\aleph} = (\mathbf{I} - \mathbf{J}_e^{\dagger}\mathbf{J}_e)\boldsymbol{v}, \tag{2.95}$$

where the pseudo-inverse of the Jacobian in the above equation is calculated via approximate solution for singularity avoidance, that is,

$$\mathbf{J}_e^{\dagger} = (\mathbf{J}_e^T\mathbf{J}_e + \lambda^2\mathbf{I})^{-1}\mathbf{J}_e^T. \tag{2.96}$$

Finally, the joint rates that replace Step 4 of the position level redundancy resolution algorithm are

$$\dot{\mathbf{q}}_k = (\mathbf{J}_e^T\mathbf{J}_e + \lambda^2\mathbf{I})^{-1}\mathbf{J}_e^T\dot{\mathbf{x}}_k + (\mathbf{I} - \mathbf{J}_e^{\dagger}\mathbf{J}_e)\boldsymbol{v}. \tag{2.97}$$

The application of the algorithm in Section 2.3 with the described modifications results in the joint trajectories shown in Fig. 2.9. Because of the application of the joint limit via kinematic optimization, the second joint operates well within its limits $q_2 \in [-0.1, 0.1]$ rad. Also, since the optimization criterion is active during the whole motion, the trajectory of the joints are smoother compared to the previous example, in which the joint limit criterion is only active in the "buffer" zone. As can be seen in Fig. 2.10, these joint trajectories cause the end-effector of the manipulator to move from the initial position $\mathbf{x}_1 = [1.0, 0.0]^T$ m to the desired position $\mathbf{x}^d = [1.25, 0.25]^T$ m in 10 s. Since a deceleration factor of $\alpha = 3$ has been used, the manipulator slows down when the end-effector gets closer to the desired position. The posture of the manipulator, $\mathbf{q}$, and the position of the end-effector, $\mathbf{x}$, at the end of the motion are

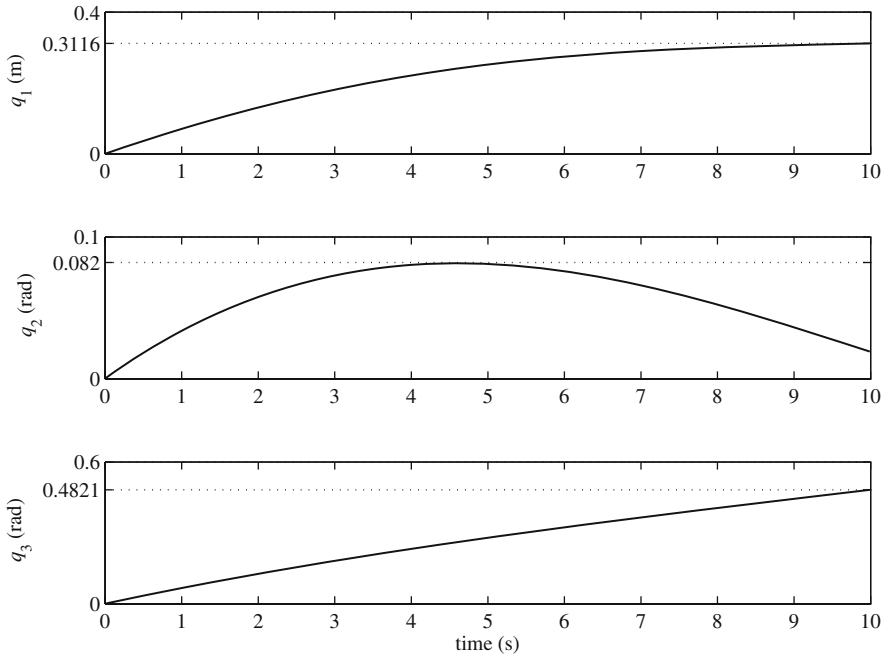


**Fig. 2.9** The joint trajectories for the PRR manipulator with JLA via kinematic optimization

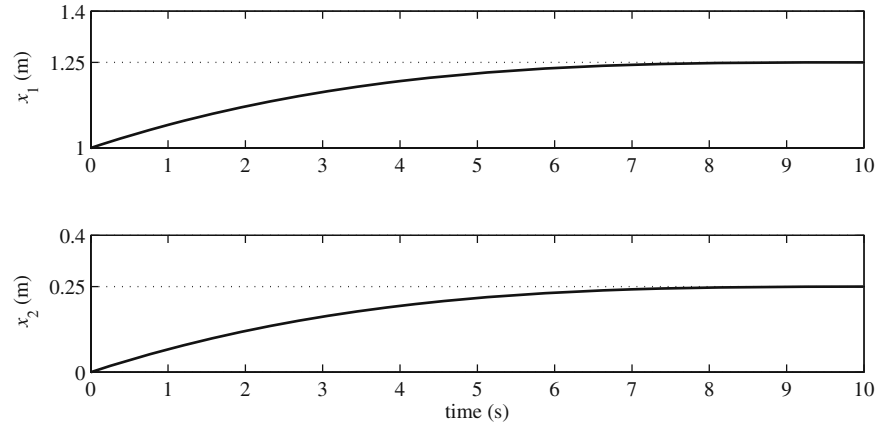


**Fig. 2.10** The end-effector trajectory for the PRR manipulator with JLA via kinematic optimization

$$\mathbf{q} = \begin{bmatrix} 0.3116 \\ 0.0193 \\ 0.4821 \end{bmatrix} \begin{matrix} \text{m} \\ \text{rad}, \\ \text{rad} \end{matrix} \qquad \mathbf{x} = \begin{bmatrix} 1.2500 \\ 0.2500 \end{bmatrix} \begin{matrix} \text{m} \\ \text{m}. \end{matrix} \tag{2.98}$$

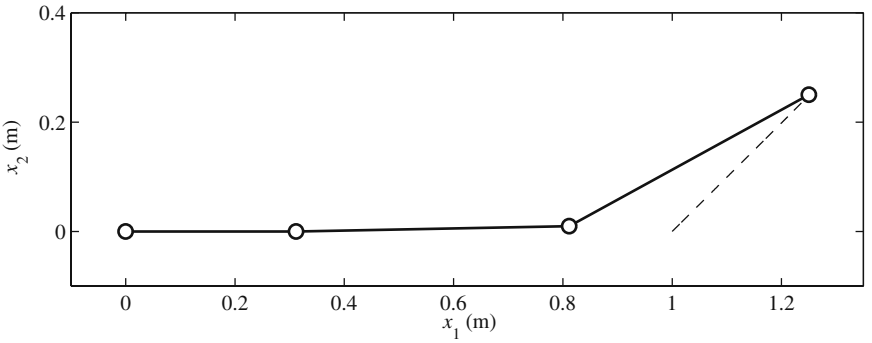


**Fig. 2.11** The final posture of the PRR manipulator and the path of the end-effector with JLA via kinematic optimization

The manipulator at the end of the motion and the path of the end-effector are shown in Fig. 2.11. Since the singularity avoidance factor $\lambda$ has been selected very small (0.1) and the joint limit avoidance has been incorporated as a null space solution, the joints rates calculated at Step 4 of the algorithm for redundancy resolution are very close to the exact solution. As a consequence, the path of the end-effector is very close to the linear path planned from the initial to the desired position of the end-effector.

## *2.4.2 Obstacle Avoidance*

In this section, an outline of an obstacle avoidance algorithm for the 2D workspace of a planar redundant manipulator, applicable to both static and moving obstacles, is given.

### 2.4.2.1 Algorithm Description

Obstacle avoidance, similar to JLA, is a part-time task, which is only activated when a possibility of collision is detected. The configuration control method, which is useful when part-time additional tasks are involved, is used here for redundancy resolution.

The obstacle avoidance algorithm has three layers [19]. First, the distance of all the manipulators' links to the obstacles must be calculated at any given time. Second, for each link, a decision must be made based on the distance of the link to the obstacle to determine if the obstacle avoidance additional task must be activated for the link. And third, the configuration control method must be used for redundancy resolution to find the joint rates that avoid collision of the links with obstacles.

The distance of a link to an obstacle is calculated through some simplifying assumptions (Fig. 2.12). The links of the planar manipulator are considered as straight

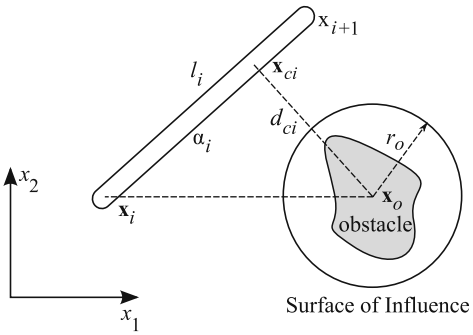


**Fig. 2.12** Determination of the critical point and the distance between a link and a Surface of Influence

lines connecting the center of joint coordinate frames. Obstacles are enclosed in circles with diameters larger than the largest obstacle dimension. The thickness of the manipulator links should also be added to the radii of these circles. These circles are called the Surface of Influence (SOI).

With these assumptions, the calculation of the location of the potential point of collision, also known as the critical point, becomes rather simple. The calculation procedure for finding the location of the critical point follows.

If link $i$ is modeled by a line from joint $i$ at the Cartesian coordinates $\mathbf{x}_i$ to joint $i+1$ at the Cartesian coordinates $\mathbf{x}_{i+1}$, the unit vector representing the direction of the link with length $l_i$ is

$$\hat{\mathbf{e}}_i = \frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{l_i}. \tag{2.99}$$

Assume that the center of the SOI is at the Cartesian coordinates $\mathbf{x}_o$. The projection of a line from joint $i$ to the center of the SOI on the link $i$ is

$$\alpha_i = \hat{\mathbf{e}}_i^T(\mathbf{x}_o - \mathbf{x}_i), \tag{2.100}$$

which represents the dot product of the unit vector $\hat{\mathbf{e}}_i$ and the vector $\mathbf{x}_o - \mathbf{x}_i$ in the matrix notation. Now, the Cartesian coordinates of the critical point can be calculated as

$$\mathbf{x}_{c_i} = \mathbf{x}_i + \alpha_i \hat{\mathbf{e}}_i. \tag{2.101}$$

The distance of the critical point with the center of the SOI is

$$d_{c_i} = \|\mathbf{x}_{c_i} - \mathbf{x}_o\|. \tag{2.102}$$

Based on this distance, the unit vector pointing from the critical point to the center of the obstacle is determined.

$$\hat{\mathbf{u}}_i = \frac{\mathbf{x}_{c_i} - \mathbf{x}_o}{d_{c_i}}, \tag{2.103}$$

or

$$d_{c_i} = \hat{\mathbf{u}}_i^T (\mathbf{x}_{c_i} - \mathbf{x}_o). \tag{2.104}$$

All the links and SOIs are considered, and a critical distance $d_{ci}$ is calculated for each. If, for a link $i$, this critical distance is smaller than the radius of the SOI (or $r_o - d_{ci} > 0$), the obstacle avoidance additional task for that link is activated. For each link $i$, the obstacle avoidance additional task can be defined as the normal distance of the link to the SOI. That is

$$z_i = f_i(\mathbf{q}, t) = r_o - d_{c_i}. \tag{2.105}$$

Using Eq. (2.104), one can write the derivative of the additional task as

$$\dot{z}_i = -\frac{d}{dt}(d_{c_i}) = -\hat{\mathbf{u}}_i^T \left( \frac{\partial \mathbf{x}_{c_i}}{\partial \mathbf{q}} \dot{\mathbf{q}} - \dot{\mathbf{x}}_o \right), \tag{2.106}$$

where $\dot{\mathbf{x}}_o$ is the velocity of the center of the SOI, or, in other words, the obstacle [2].

The obstacle avoidance additional task must be defined such that a link $i$ does not enter the SOI of its corresponding obstacle. Therefore, when the additional task becomes active, the desired value for the additional task is

$$z_i^d = 0, \tag{2.107}$$

which also implies that

$$\dot{z}_i^d = \ddot{z}_i^d = 0. \tag{2.108}$$

The Jacobian of the obstacle avoidance additional task must be calculated. Since each link $i$ has its own unique condition regarding the obstacles, each row $i$ of the Jacobian matrix for the additional task is calculated separately based on the position of the critical point on link $i$. The $i$-th row of the Jacobian is derived by observing Eq. (2.106).

$$\mathbf{J}_{c_i} = -\hat{\mathbf{u}}_i^T \mathbf{J}_{\mathbf{x}_{c_i}}, \tag{2.109}$$

where

$$\mathbf{J}_{\mathbf{x}_{c_i}} = \frac{\partial \mathbf{x}_{c_i}}{\partial \mathbf{q}}. \tag{2.110}$$

The Jacobian of the additional task is assembled from $\mathbf{J}_{c_i}$ of each link that is facing a SOI.

*Example 2.9.* Assume an obstacle at the Cartesian coordinate $(0.6, 0.0)$ m in the workspace of the redundant PRR manipulator of Example 2.1. Assume a Surface of Influence with radius $r_0 = 0.15$ m for the obstacle. The manipulator is at an initial posture of $\mathbf{q}_1 = [0, 0.5, 0.5]^T$. The end-effector must reach the desired position of $\mathbf{x}^d = [1.15, 0.15]^T$ m in 10 s. Determine the joint trajectories for the redundant manipulator.

*Solution.* To simplify the solution to this example, it is assumed that only link 2 may collide with the obstacle. First, the distance of the critical point on link 2 with the center of the SOI is calculated and an additional task is defined based on this distance. Then, the Jacobian of the additional task is determined. Finally, this augmented task is incorporated into the redundancy resolution via the configuration control method. The details of the solution procedure follow.

The Cartesian coordinates of joints 2 and 3 are written as

$$\mathbf{x}_2 = \begin{bmatrix} q_1 \\ 0 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} q_1 + l_2 \cos q_2 \\ l_2 \sin q_2 \end{bmatrix}. \tag{2.111}$$

The unit vector representing the direction of link 2, $\hat{\mathbf{e}}_2$, and the distance of the critical point with joint 2, $\alpha_2$, can be derived as

$$\hat{\mathbf{e}}_2 = \frac{\mathbf{x}_3 - \mathbf{x}_2}{l_2} = \begin{bmatrix} \cos q_2 \\ \sin q_2 \end{bmatrix}, \tag{2.112}$$

$$\alpha_2 = \hat{\mathbf{e}}_2^T (\mathbf{x}_o - \mathbf{x}_2). \tag{2.113}$$

The Cartesian coordinates of the critical point, $\mathbf{x}_{c_2}$, and the distance of the critical point to the center of the SOI, $d_{c_i}$, are

$$\mathbf{x}_{c_2} = \mathbf{x}_2 + \alpha_2 \hat{\mathbf{e}}_2 = \begin{bmatrix} q_1 + \alpha_2 \cos q_2 \\ \alpha_2 \sin q_2 \end{bmatrix}, \tag{2.114}$$

$$d_{c_i} = \|\mathbf{x}_{c_2} - \mathbf{x}_o\| = \sqrt{(x_{c_2,1} - x_{o,1})^2 + (x_{c_2,2} - x_{o,2})^2}. \tag{2.115}$$

The unit vector pointing from the critical point to the obstacle center is

$$\hat{\mathbf{u}}_2 = \frac{\mathbf{x}_{c_2} - \mathbf{x}_o}{d_{c_2}}. \tag{2.116}$$

The Jacobian of the critical point is calculated as

$$\mathbf{J}_{\mathbf{x}_{c_2}} = \frac{\partial \mathbf{x}_{c_2}}{\partial \mathbf{q}} = \begin{bmatrix} 1 & -\alpha_2 \sin q_2 & 0 \\ 0 & \alpha_2 \cos q_2 & 0 \end{bmatrix}. \tag{2.117}$$

The second column of the Jacobian of the additional task, which corresponds to obstacle avoidance for link 2, is

$$\mathbf{J}_{c_2} = \begin{cases} [0,0,0] & \text{if } d_{c_2} > r_o \\ -\hat{\mathbf{u}}_2^T \mathbf{J}_{\mathbf{x}_{c_2}} & \text{if } d_{c_2} \leq r_o \end{cases}. \tag{2.118}$$

This means that, the additional task corresponding to link 2 is only active when the distance of the critical point to the center of the SOI, $d_{c_2}$, is smaller than the radius of the SOI for the obstacle, $r_o$.

Now, the Jacobian for the additional task for the whole manipulator, $\mathbf{J}_c$, must be assembled using the augmented tasks corresponding to each link. Note that the first and the third row of the Jacobian of the additional task, $\mathbf{J}_c$, have all zero components, because no obstacle avoidance is being considered for links 1 and 3 in this example for simplicity. The above note dictates that the Jacobian of the additional task, $\mathbf{J}_c$, becomes

$$\mathbf{J}_c = \begin{bmatrix} 0 & 0 & 0 \\ J_{c_2,1} & J_{c_2,2} & J_{c_2,3} \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.119}$$

The Step 4 of redundancy resolution algorithm the position level presented in section 2.3 must be modified such that the incorporation of the additional task is possible. This is done by using Eq. (2.59). Since the desired rate for the additional task, $\dot{\mathbf{z}}^d$, is zero, Eq. (2.59) is reduced to

$$\dot{\mathbf{q}}_k = [\mathbf{J}_e^T \mathbf{W}_e \mathbf{J}_e + \mathbf{J}_c^T \mathbf{W}_c \mathbf{J}_c + \mathbf{W}_v]^{-1} \mathbf{J}_e^T \mathbf{W}_e \dot{\mathbf{x}}_k. \tag{2.120}$$

The following weight matrices are assumed

$$\mathbf{W}_e = \mathbf{I}, \quad \mathbf{W}_c = (1000)\mathbf{I}, \quad \mathbf{W}_v = (0.1)\mathbf{I}. \tag{2.121}$$

where $\mathbf{I}$ is a $3 \times 3$ identity matrix. The application of the algorithm in Section 2.3 with the described modifications results in the joint trajectories shown in Fig. 2.13. Because of the implementation of the obstacle avoidance for link 2 via configuration control method, the second link stops far from the obstacle. Also, in this example, the deceleration factor $\alpha$ was chosen equal to 3.0, which makes the joint speeds much slower toward the end of the motion compared to the joint speeds at the beginning of the motion. As can be seen in Fig. 2.14, these joint trajectories cause the end-effector of the manipulator to move from its initial position to the desired position $\mathbf{x}^d = [1.15, 0.15]^T$ m in 10 s. The posture of the manipulator, $\mathbf{q}$, and the position of the end-effector, $\mathbf{x}$, at the end of the motion are

$$\mathbf{q} = \begin{bmatrix} 0.1841 \\ 0.3668 \\ -0.4254 \end{bmatrix} \begin{matrix} \text{m} \\ \text{rad}, \\ \text{rad} \end{matrix} \qquad \mathbf{x} = \begin{bmatrix} 1.1500 \\ 0.1500 \end{bmatrix} \begin{matrix} \text{m} \\ \text{m}. \end{matrix} \tag{2.122}$$

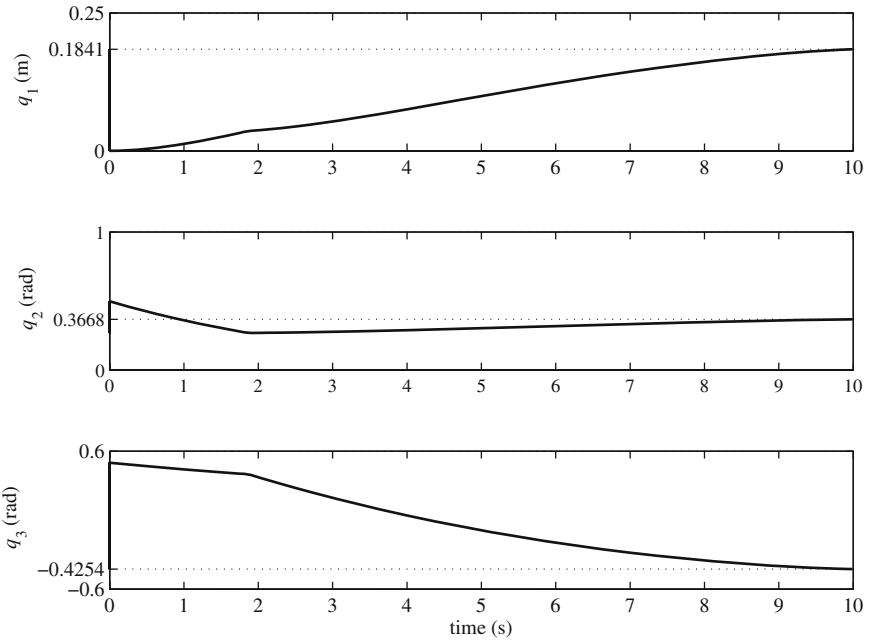


**Fig. 2.13** The joint trajectories for the PRR manipulator with obstacle avoidance via configuration control

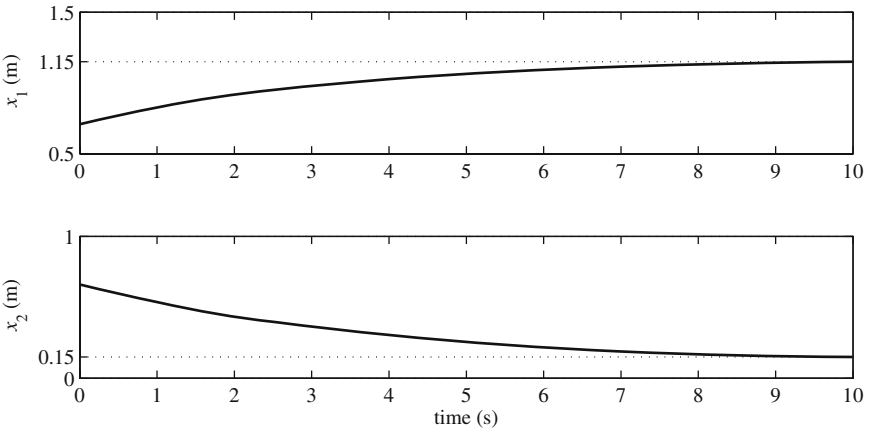


**Fig. 2.14** The end-effector trajectory for the PRR manipulator with obstacle avoidance via configuration control

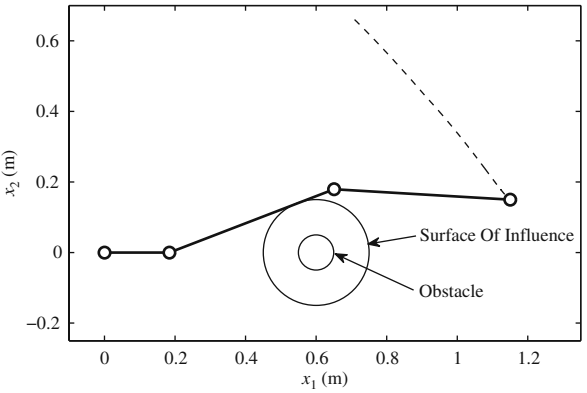


**Fig. 2.15** The final posture of the PRR manipulator and the end-effector path with obstacle avoidance via configuration control

The manipulator at the end of the motion and the path of the end-effector are shown in Fig. 2.15. Since the obstacle avoidance has been implemented via the configuration control method, the solution to joint rates used in Step 4 of the algorithm (based on Eq. 2.59) is a trade-off between the main and the additional tasks, with a higher weight for the additional task. As a consequence, the path of the end-effector is far from the linear path planned from the initial to the desired position of the end-effector.

## 2.5 Summary

In this chapter, the basic issues needed for the analysis of kinematically redundant manipulators were presented. Different redundancy resolution schemes were reviewed. The formulation of the additional tasks to be used by the redundancy resolution module were presented in this chapter. Joint limit avoidance, which is one of the most useful additional tasks, was studied in detail. The basic formulation of static and moving obstacle collision avoidance task in 2D workspace was presented.

## Problems

**Problem 2.1.** Consider the 3DOF planar Revolute-Revolute-Prismatic (RRP) manipulator shown in Fig. 2.16 with joint variables $q_1$, $q_2$, and $q_3$. The Cartesian coordinates of the end-effector $x_1$ and $x_2$ are assumed as the task space with two dimensions. The link lengths for the first, second, and third links are $l_1$, $l_2$, and $l_3$, respectively.

(a) Determine the degree of redundancy of this manipulator.
(b) Derive the Jacobian matrix for this manipulator.

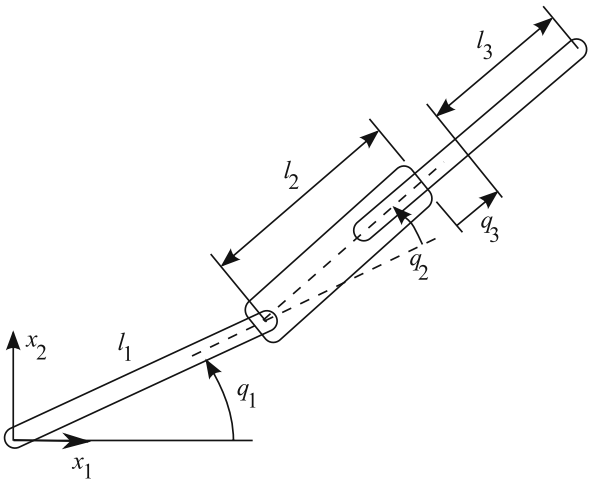


**Fig. 2.16** A RRP manipulator

**Problem 2.2.** Consider the RRP redundant manipulator of Problem 2.1. If the link length $l_1$ is 1.0 m and the lengths $l_2$ and $l_3$ are 0.5 m, find

(a) the joint rates that generate an end-effector velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s, if the arm is at a posture $\mathbf{q} = [\pi/12 \text{ rad}, \pi/3 \text{ rad}, 0.25 \text{ m}]^T$;
(b) the joint rates that generate an end-effector velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s, if the arm is at a posture $\mathbf{q} = [\pi/2 \text{ rad}, 0 \text{ rad}, 0.25 \text{ m}]^T$.

**Problem 2.3.** Consider the RRP redundant manipulator of Problem 2.1 at a posture $q_1 = \pi/6$ rad, $q_2 = \pi/12$ rad, and $q_3 = 0.25$ m. If the end-effector's angular velocity is defined as an additional task, find the joint rates required to generate a velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s and an angular velocity of $\omega_3^d = \pi/12$ rad/s for the end-effector.

**Problem 2.4.** Consider the RRP redundant manipulator of Problem 2.1. Using the approximate solution method, find the joint rates that generate a velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s for the end-effector when the manipulator is

(a) at a posture of $q_1 = \pi/12$ rad, $q_2 = \pi/3$ rad, and $q_3 = 0.25$ m.
(b) at a posture of $q_1 = \pi/2$ rad, $q_2 = 0$ rad, and $q_3 = 0.25$ m.

**Problem 2.5.** Consider the RRP redundant manipulator of Problem 2.1 and the additional task introduced in Problem 2.3. Assume that the main task is three times more important than the additional task and 30 times more important than singularity avoidance. Find the joint rates that generate a velocity of $\dot{\mathbf{x}}^d = [0.5, 0.0]^T$ m/s

and an angular velocity of $\omega_3^d = \pi/12$ rad/s for the end-effector, if the manipulator is at

(a) a posture of $q_1 = \pi/3$ rad, $q_2 = \pi/12$ rad, and $q_3 = 0.25$ m;
(b) a posture of $q_1 = \pi/2$ rad, $q_2 = 0$ rad, and $q_3 = 0$ m.

**Problem 2.6.** Consider the 3DOF planar RRP manipulator of Problem 2.1. The joint variables are $q_1$, $q_2$, and $q_3$. The Cartesian coordinates of the end-effector $x_1$ and $x_2$ are assumed as the main task space with two dimensions. The link lengths are $l_1 = 1$ m and $l_2 = l_3 = 0.5$ m.

Assume a joint limit of $q_{3\,\min} = -0.3$ m and $q_{3\,\max} = 0.3$ m for the third joint with an activation buffer of $\tau_3 = -0.02$ m. If the manipulator is at the initial posture of $\mathbf{q}_1 = [-0.86, 0.15, 0.00]^T$, find the joint trajectories that cause the end-effector to move to a desired position of $\mathbf{x}^d = [1.78, 1.42]^T$ m. Plot the components of $\mathbf{q}$, $\dot{\mathbf{q}}$, $\mathbf{x}$, $\dot{\mathbf{x}}$, and the path of the end-effector.

(a) Use the Inequality Constraint method.
(b) Use the Kinematic Optimization method.
(c) Discuss the difference in the obtained plots in part (a) and (b).

**Problem 2.7.** Assume an obstacle at the Cartesian coordinate $(1.25, 0.0)$ m in the workspace of the redundant RRP manipulator of Problem 2.1. Assume a SOI with radius $r_0 = 0.15$ m for the obstacle. The manipulator is at an initial posture of $\mathbf{q}_1 = [\pi/2, 0, 0]^T$. The end-effector must reach the desired position of $\mathbf{x}^d = [2, 0]^T$ m in 15 s. Determine the joint trajectories for the redundant manipulator.

# Chapter 3
# Hyper-Redundant Manipulators

## 3.1 Introduction

A hyper-redundant manipulator has many more kinematic DOFs than the number of its task space coordinates. Therefore, the classical methods cannot be used for solving their inverse kinematics. Many investigations have been focused on the redundancy resolution of this type of manipulators based on the manipulator Jacobian pseudo-inverse [47]. Singularity avoidance [81, 4], obstacle avoidance [48, 60, 5], and keeping joint variables in their physical limitation [47] are some examples of supplementary tasks. Extended Jacobian inverse [4] and augmented inverse Jacobian [69] are also utilized extensively. In the case of spatial hyper-redundant manipulators with hundreds of DOFs, the computational burden of pseudo-inverse Jacobian becomes prohibitive, despite of proposed improvements [36]. Furthermore, most of the proposed schemes handle the inverse kinematic problem at velocity level only.

The modal approach, which is the focus of this chapter, has been presented as a unique method for redundancy resolution of hyper-redundant manipulators [14, 15]. In this method, a backbone curve is defined as a piecewise continuous curve that captures the important macroscopic geometric features of a hyper-redundant robot. The backbone curve is restricted by a set of intrinsic shape functions to a modal form. The mode shape functions are arbitrary and lead to an efficient inverse kinematics solution at the position level. Once the backbone curve is determined for an assumed location of end-effector, depending on the physical implementation morphology of a particular manipulator, various fitting algorithms can be developed. This method has been utilized to form the foundation for obstacle avoidance [13], locomotion [16], and motion control [58, 62].

In this chapter, a spatial hyper-redundant robot that does not have macroscopic branches[1] or closed loops is considered. Also, it is assumed that the robot has a sufficiently large number of links so that its geometry can be nominally captured by a spatial curve closely, though not exactly. This curve, called the backbone curve, is

[1] A manipulator is said to have a macroscopic branch when a series of links and joints branch out from one of the manipulator's intermediate joints. Such a manipulator has more than one end-effector.

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_3

piecewise continuous and captures the important macroscopic geometric features of the hyper-redundant robot.

The modal approach is utilized to resolve the redundancy of a spatial hyper-redundant manipulator. Useful shape functions are introduced to achieve a more complete workspace. Two fitting methods are introduced, the Constrained Least Square Fitting Method (CLSFM) and the Recursive Fitting Method (RFM). The application of the CLSFM is only discussed for planar manipulators due to its high algebraic and numerical computation time. However, the application of the RFM is presented for spatial hyper-redundant manipulators with universal joints. The RFM solves a single nonlinear algebraic equation per link and avoids systems of nonlinear simultaneous equations. This method can be used for spatial universal-jointed robots with any number of links without requiring a heavy computation. Finally, the velocity property of the backbone curve is investigated and an inverse velocity propagation scheme is introduced. The inverse velocity propagation scheme uses velocity information of the backbone curve. This scheme is recursive and free from singularity in the workspace, and can be applied easily to spatial arms with an arbitrary number of links.

## 3.2 Parameterization of the Backbone Curve

A spatial (backbone) curve can be parameterized as follows

$$\mathbf{x}(s) = \int_0^s L\mathbf{u}(\sigma)d\sigma, \tag{3.1}$$

where $s \in [0, 1]$ is the curve length parameter, $\mathbf{u}(\sigma)$ is the unit vector tangent to the curve at $\sigma$, and $L$ is the length of the curve. The parameterization shown in Fig. 3.1 has been used for $\mathbf{u}(s)$. Therefore, Eq. (3.1) expands to

$$\mathbf{x}(s) = \begin{bmatrix} \int_0^s L\sin\phi(\sigma)\cos\psi(\sigma)d\sigma \\ \int_0^s L\cos\phi(\sigma)\cos\psi(\sigma)d\sigma \\ \int_0^s L\sin\psi(\sigma)d\sigma \end{bmatrix}. \tag{3.2}$$

The functions $\phi(s)$ and $\psi(s)$ may be represented as a linear combination of mode shapes, as follows

$$\begin{aligned} \phi(s) &= \sum_{i=1}^{n_1} a_i f_i(s) + \sum_{i=1}^{2} b_{i\phi} g_i(s), \\ \psi(s) &= \sum_{i=n_1+1}^{n_2} a_i f_i(s) + \sum_{i=1}^{2} b_{i\psi} g_i(s), \end{aligned} \tag{3.3}$$

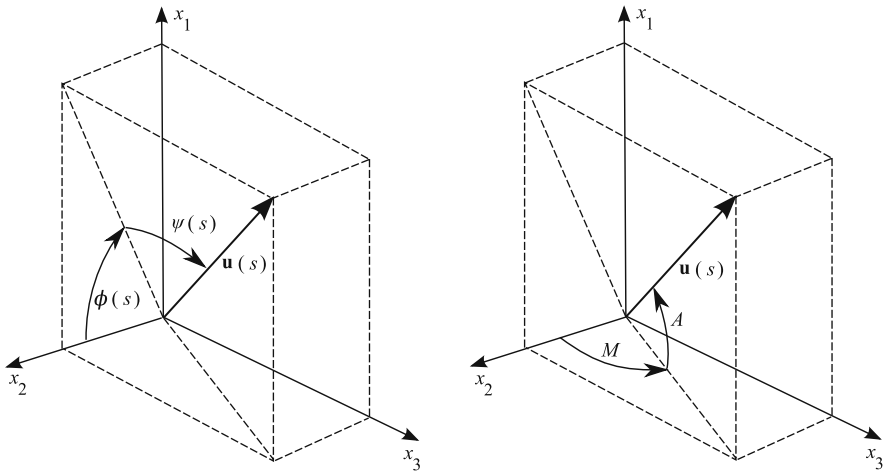


**Fig. 3.1** Parameterization of $\mathbf{u}(s)$, curve parameters, and azimuth and meridian angles

where $[f_i(s)]$ are the mode shapes, $a_i$'s are the mode participation factors, $n_1$ is the number of mode shapes corresponding to $\phi(s)$, $n_2$ is the total number of mode shapes, and $[g_i(s)]$ and $\left[b_{i\phi}, b_{i\psi}\right]$ are used to specify the orientation of the backbone curve at the start and end points. This modal approach reduces the inverse kinematic problem to the determination of $a_i$'s that satisfy the task constraints, i.e., $\mathbf{x}(1) = \mathbf{x}_D$, where $\mathbf{x}_D$ represents the desired position vector of the backbone curve end point.

*Example 3.1.* Consider a spatial hyper-redundant manipulator. Specify the mode shapes.

*Solution.* Since the end point of the corresponding backbone curve is determined by three coordinate components, there are three task constraints. Therefore, three mode shapes and three mode participation factors are needed. The three mode shapes can be selected as follows

$$\begin{aligned}\phi(s) &= a_1 \sin(2\pi s) + a_2(1 - \cos(2\pi s)) \\ &\quad + b_{1\phi}(1 - \sin(\pi s/2)) + b_{2\phi} \sin(\pi s/2), \qquad (3.4)\\ \psi(s) &= a_3(1 - \cos(2\pi s)) \\ &\quad + b_{1\psi}(1 - \sin(\pi s/2)) + b_{2\psi} \sin(\pi s/2), \qquad (3.5)\end{aligned}$$

where $\left[b_{1\phi}, b_{1\psi}\right] = [\phi(0), \psi(0)]$ and $\left[b_{2\phi}, b_{2\psi}\right] = [\phi(1), \psi(1)]$. This completes the solution to this example.

Once the mode shapes are selected, the vector of the modal participation factors, $\mathbf{a}$, can be found by using the following iterative approximation

$$\mathbf{a}_{m+1} = \mathbf{a}_m + \alpha \mathbf{J}_a^{-1}(\mathbf{a}_m, 1)[\mathbf{x}_D - \mathbf{x}_m], \qquad (3.6)$$

where $\alpha$ is a constant that controls the convergence rate and $m$ is the iteration counter, and the modal Jacobian is

$$\mathbf{J}_a(\mathbf{a}, s) = \frac{\partial \mathbf{x}(s)}{\partial \mathbf{a}^T} = \begin{bmatrix} \frac{\partial x_1}{\partial a_1} & \cdots & \frac{\partial x_m}{\partial a_{n_2}} \\ \vdots & \cdots & \vdots \\ \frac{\partial x_m}{\partial a_1} & \cdots & \frac{\partial x_m}{\partial a_{n_2}} \end{bmatrix}. \tag{3.7}$$

The modal Jacobian matrix, $\mathbf{J}_a(\mathbf{a}, s)$, is evaluated at $s = 1$ in Eq. (3.6). This matrix is the Jacobian for the kinematics equation $\mathbf{x}(1) = \mathbf{x}_D$, where $\mathbf{x}_D$ represents the desired position vector of the backbone curve end point. The partial derivatives of the elements of $\mathbf{x}(1)$ are derived using Eq. (3.3) and evaluated by numerical integration. Note that the numerical solution presented in Eq. (3.6) is independent of the number of links.

*Example 3.2.* Assuming the mode shapes given in Eq. (3.5), compute the mode participation factors, if the end point of a backbone curve with 1-m length is to be located at $\mathbf{x}_D = \begin{bmatrix} 0.4 & 0.2 & 0.2 \end{bmatrix}^T$ m. Also assume that the backbone curve has a tangent defined by $\phi(0) = 255^\circ$ and $\psi(0) = 0^\circ$ at the start point. Consider a tangent to the endpoint of the backbone curve defined by $\phi(1) = 0^\circ$, and $\psi(1) = 0^\circ$. See Fig. 3.1.

*Solution.* According to the problem specifications, one can write

$$\begin{bmatrix} b_{1\phi}, b_{1\psi} \end{bmatrix} = [255\pi/180, 0], \qquad \begin{bmatrix} b_{2\phi}, b_{2\psi} \end{bmatrix} = [0, 0].$$

Now, the modal Jacobian matrix, $\mathbf{J}_a(\mathbf{a}, s)$, which was introduced in Eq. (3.6), can be computed. This matrix is defined by

$$\mathbf{J}_a(\mathbf{a}, s) = \frac{\partial \mathbf{x}(s)}{\partial \mathbf{a}^T} = \begin{bmatrix} \frac{\partial x_1}{\partial a_1} & \frac{\partial x_1}{\partial a_2} & \frac{\partial x_1}{\partial a_3} \\ \frac{\partial x_2}{\partial a_1} & \frac{\partial x_2}{\partial a_2} & \frac{\partial x_2}{\partial a_3} \\ \frac{\partial x_3}{\partial a_1} & \frac{\partial x_3}{\partial a_2} & \frac{\partial x_3}{\partial a_3} \end{bmatrix}. \tag{3.8}$$

By using the above relation, one can show that the components of the modal Jacobian matrix are as follows

$$J_{11}(\mathbf{a}, s) = \int_0^s \cos\phi(\sigma)\cos\psi(\sigma)\sin(2\pi\sigma)d\sigma,$$

$$J_{12}(\mathbf{a}, s) = \int_0^s \cos\phi(\sigma)\cos\psi(\sigma)(1 - \cos(2\pi\sigma))d\sigma,$$

$$J_{13}(\mathbf{a}, s) = \int_0^s -\sin\phi(\sigma)\sin\psi(\sigma)(1 - \cos(2\pi\sigma))d\sigma,$$

$$J_{21}(\mathbf{a}, s) = \int_0^s -\sin\phi(\sigma)\cos\psi(\sigma)\sin(2\pi\sigma)d\sigma,$$

$$J_{22}(\mathbf{a}, s) = \int_0^s -\sin\phi(\sigma)\cos\psi(\sigma)(1-\cos(2\pi\sigma))d\sigma,$$

$$J_{23}(\mathbf{a}, s) = \int_0^s -\cos\phi(\sigma)\sin\psi(\sigma)(1-\cos(2\pi\sigma))d\sigma,$$

$$J_{31}(\mathbf{a}, s) = 0,$$

$$J_{32}(\mathbf{a}, s) = 0,$$

$$J_{33}(\mathbf{a}, s) = \int_0^s \cos\psi(\sigma)(1-\cos(2\pi\sigma))d\sigma.$$

At the first iteration, $m$ in Eq. (3.6) is equal to 0. Also, a first guess for mode particpation factors is assumed as $\mathbf{a}_0 = \begin{bmatrix}1.0 & 1.0 & 0.5\end{bmatrix}^T$. Then, $\mathbf{x}_0$ can be computed by using the first guess, $\mathbf{a}_0$, in Eq. (3.2) with $s = 1$. Now that all the right hand terms of Eq. (3.6) are at hand, a new value for $\mathbf{a}_1$ (second iteration) is obtained. This procedure continues until $\mathbf{x}_m$ converges to $\mathbf{x}_D$. The final value of the mode participation vector is

$$\mathbf{a} = \begin{bmatrix}0.54 & 0.18 & 0.20\end{bmatrix}^T.$$

The corresponding backbone curve is shown in Fig. 3.2. This completes the inverse kinematics problem at the backbone curve level.

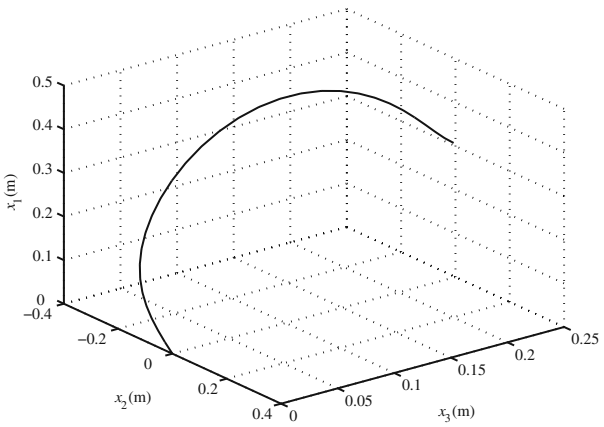


**Fig. 3.2** The backbone curve obtained in Example 3.2

### 3.2.1 Workspace Considerations

The workspace of the standard mode shapes will be limited if the backbone curve at its start point is assumed to be always tangent to the $Y$-axis. The workspace will be improved by setting the meridian angle of tangent at the start point to zero ($M_s = 0$) and its azimuth angle, $A_s$, to

$$A_s = \beta + \pi(1 - \delta), \qquad M_s = 0, \tag{3.9}$$

$$\delta = \frac{\sqrt{x_1(1)^2 + x_2(1)^2 + x_3(1)^2}}{L}, \qquad \beta = \arctan\left(\frac{x_1(1)}{\sqrt{x_2(1)^2 + x_3(1)^2}}\right). \tag{3.10}$$

where

$$\tan\phi(s) = \tan A / \cos M \qquad \sin\psi(s) = \sin M \cos A \tag{3.11}$$

Please see Fig. 3.1. $(x_1(1), x_2(1), x_3(1))$ is the specified position of the curve end point and $L$ is the backbone curve length. In Eq. (3.9), $\beta$ makes the tangent at the start point coincident with the line of view of the end-effector from the base (Fig. 3.3). The second term in Eq. (3.9) rotates the tangent away from the end-effector, depending on the distance between the end-effector and the base. When $\delta \approx 1$, the end-effector is at the maximum distance from the base and the backbone curve tends to become a straight line. When $\delta \ll 1$, the end-effector is near the base. Therefore, the second term rotates the tangent to allow more space for the end-effector (Fig. 3.3).

One can show that using these mode shapes the workspace is much improved. The larger workspace avoids mode switching, which imposes more computational effort, particularly when it is required to simultaneously fix the end-effector [30].

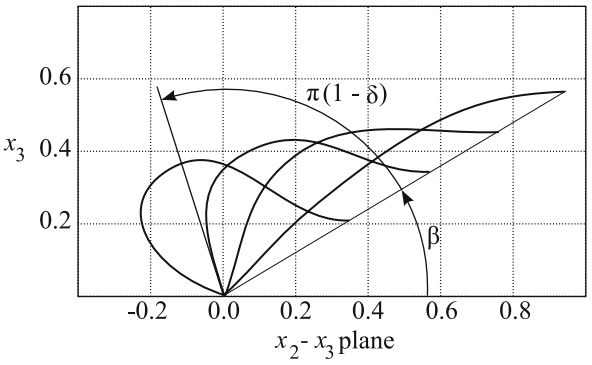


**Fig. 3.3** Determining the azimuth angle of the start point tangent

## 3.3 Fitting Methods

Once the backbone curve, which captures the gross configuration of a hyper-redundant manipulator, is determined, there is a need for an algorithm that determines the joint postures of the manipulator such that the segmented manipulator is positioned in a configuration as close as possible to the backbone curve. The algorithm that determines the joint angles is called the fitting algorithm. In this section, we discuss two different methods of deriving such an algorithm, the CLSFM and the RFM. Since the CLSFM has a high algebraic and numerical computation load, only its 2D implementation is presented. For the RFM, which is computationally more efficient than the CLSFM, the spatial case is presented, from which the reader can derive the planar case implementation. At the end of this section, the computational load of the two methods are compared for 2D hyper-redundant manipulators.

### *3.3.1 Constraint Least Square Fitting Method (CLSFM)*

A planar $n$-link manipulator is considered. It is assumed that the link lengths are the same for all the links, with a value of $1/n$. The first joint is located at $(x_1, x_2) = (0, 0)$, where $(x_1, x_2)$ are the Cartesian coordinates describing the 2D plane in which the manipulator lies. The Cartesian components of the position of the end-effector, $x_1$ and $x_2$, and the orientation of the end-effector with respect to the $x_2$ axis, $z_1$, are defined as the manipulator's task space.

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & z_1 \end{bmatrix}^T . \tag{3.12}$$

If $q_i$ is the angle that link $i$ makes with the $x_2$-axis, the forward kinematics of the manipulator simply is

$$x_1 = \frac{1}{n}\sum_{i=1}^{n} \sin q_i, \quad x_2 = \frac{1}{n}\sum_{i=1}^{n} \cos q_i, \quad z_1 = q_n, \tag{3.13}$$

where $z_1$ is the extra task added so that one can specify the end-effector's direction. Our goal is to determine $q_i$'s such that the end-effector is at the desired point in the task space, while the joints of the manipulator are positioned as close as possible to their corresponding point on the backbone curve. To position the joints as close as possible to their corresponding point on the backbone curve, a fitting error function is defined as follows

$$G = \frac{1}{2}\sum_{i=1}^{n} [(x_1(s_i) - x_{1i})^2 + (x_2(s_i) - x_{2i})^2], \tag{3.14}$$

where $x_1(s_i)$ and $x_2(s_i)$ are the $x_1$ and $x_2$ position components of the points on the continuous backbone curve at a length parameter of $s_i = i/n$ for $i = 1, \ldots, n$,

respectively. The location of joint $i+1$ (or the end-effector tip, if $i = n$) can be calculated as

$$x_{1i} = \frac{1}{n}\sum_{j=1}^{i} \sin q_j, \quad x_{2i} = \frac{1}{n}\sum_{j=1}^{i} \cos q_j. \tag{3.15}$$

Finding the $q_i$'s that minimize the quadratic fitting cost function defined in Eq. (3.14) is done numerically. To simplify the formulation and the linearization procedure, an estimation for the solution, $\tilde{q}_j$, is considered and the procedure is formulated based on corrections, $\epsilon_j$, that are applied to the estimated solution.

$$q_j = \tilde{q}_j + \epsilon_j. \tag{3.16}$$

For the estimated solution, an estimation of the slope of the backbone curve at the point corresponding to the $j$th joint is used. When a large $n$ is assumed, the $j$th joint angle might be approximated by

$$\tilde{q}_j = \arctan\left(\frac{x_1(s_j) - x_1(s_{j-1})}{x_2(s_j) - x_2(s_{j-1})}\right), \quad j = 1, \ldots, n, \tag{3.17}$$

where $x_1(s_0)$ and $x_2(s_0)$ are the position components of the manipulator's base, i.e., $(0, 0)$. The correction angle $\epsilon_j$ is assumed to be small, and one can linearize the fitting error and the end-effector position constraint equations with respect to $\epsilon_j$, which results in

$$G = \frac{1}{2}\sum_{i=1}^{n}[(x_1(s_i) - \frac{1}{n}\sum_{j=1}^{i}(\sin\tilde{q}_j + \epsilon_j\cos\tilde{q}_j))^2 + (x_2(s_i) - \frac{1}{n}\sum_{j=1}^{i}(\cos\tilde{q}_j - \epsilon_j\sin\tilde{q}_j))^2]. \tag{3.18}$$

Now, the fitting method problem reduces to finding $\epsilon_j$'s that minimize the cost function defined in Eq. (3.18) subject to the following constraints

$$\begin{aligned} g_1 &= x_1^d - x_{1n} \approx x_1^d - \frac{1}{n}\sum_{j=1}^{n}(\sin\tilde{q}_j + \epsilon_j\cos\tilde{q}_j) = 0, \\ g_2 &= x_2^d - x_{2n} \approx x_2^d - \frac{1}{n}\sum_{j=1}^{n}(\cos\tilde{q}_j - \epsilon_j\sin\tilde{q}_j) = 0, \\ g_3 &= z_1^d - (\tilde{q}_n + \epsilon_n) = 0. \end{aligned} \tag{3.19}$$

These constraints guarnatee that the end-effector is fixed at its desired position. To minimze the cost function (3.19), its gradient with respect to the correction angles must be put equal to zero and the above constraints must be applied using Lagrange multipliers. With these conditions, a necessary condition for the constrainted minima of (3.19) is

$$\frac{\partial G}{\partial \epsilon_k} + \sum_{l=1}^{3} \lambda_l \frac{\partial g_l}{\partial \epsilon_k} = 0, \quad k = 1, \ldots, n. \tag{3.20}$$

Equation (3.20) and the constraint Eq. (3.19) provide $n + 3$ linear equations for the $n$ unknown correction angles $\epsilon_j$ and three Lagrange multipliers $\lambda_l$'s. These can be solved for $\epsilon_j$'s that minimize Eq. (3.18).

Equation (3.20) expands as follows

$$\begin{aligned}
\sum_{i=1}^{n}[(x_1(s_i) - \frac{1}{n}\sum_{j=1}^{i}(\sin\tilde{q}_j + \epsilon_j \cos\tilde{q}_j))(\frac{-1}{n}\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m) \\
+(x_2(s_i) - \frac{1}{n}\sum_{j=1}^{i}(\cos\tilde{q}_j - \epsilon_j \sin\tilde{q}_j))(\frac{1}{n}\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m)] \\
-\frac{\lambda_1}{n}\sum_{j=1}^{n}\delta_{jk}\cos\tilde{q}_j + \frac{\lambda_2}{n}\sum_{j=1}^{n}\delta_{jk}\sin\tilde{q}_j - \lambda_3\delta_{nk} = 0, \quad k = 1, \ldots, n,
\end{aligned} \tag{3.21}$$

where

$$\delta_{jk} = \frac{\partial \epsilon_j}{\partial \epsilon_k} = \begin{cases} 0 & j \neq k \\ 1 & j = k \end{cases}. \tag{3.22}$$

Rearranging Eq. (3.21) in terms of $\epsilon_j$'s yields

$$\begin{aligned}
\sum_{i=1}^{n}[\frac{1}{n^2}\sum_{j=1}^{i}\epsilon_j(\cos\tilde{q}_j\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m + \sin\tilde{q}_j\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m)] \\
+\sum_{i=1}^{n}[x_1(s_i)(\frac{-1}{n}\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m) + x_2(s_i)(\frac{1}{n}\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m)] \\
+\sum_{i=1}^{n}[\frac{1}{n^2}(\sum_{j=1}^{i}\sin\tilde{q}_j(\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m) - \sum_{j=1}^{i}\cos\tilde{q}_j(\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m))] \\
-[\frac{1}{n}\sum_{j=1}^{n}\delta_{jk}\cos\tilde{q}_j]\lambda_1 + [\frac{1}{n}\sum_{j=1}^{n}\delta_{jk}\sin\tilde{q}_j]\lambda_2 - [\delta_{nk}]\lambda_3 = 0, \\
k = 1, \ldots, n.
\end{aligned} \tag{3.23}$$

Equation (3.23) represents $n$ equations for $k = 1, \ldots, n$. The coefficients of $\epsilon_p$'s in the $k$th equation are ($p = 1, \ldots, n$) denoted as:

$$A_{kp} = \sum_{i=1}^{n}[\frac{1}{n^2}\sum_{j=1}^{i}\delta_{jp}(\cos\tilde{q}_j\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m + \sin\tilde{q}_j\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m)]. \quad (3.24)$$

The coefficients of $\lambda_l$'s ($l = 1, \ldots, 3$) in the $k$th ($k = 1, \ldots, n$) equation in (3.23) are denoted as

$$A_{k(n+1)} = \frac{-1}{n}\sum_{j=1}^{n}\delta_{jk}\cos\tilde{q}_j, \quad (3.25)$$

$$A_{k(n+2)} = \frac{1}{n}\sum_{j=1}^{n}\delta_{jk}\sin\tilde{q}_j, \quad (3.26)$$

$$A_{k(n+3)} = -\delta_{nk}. \quad (3.27)$$

The coefficients of $\epsilon_j$'s ($j = 1, \ldots, n$) in Eq. (3.19) are denoted as

$$A_{(n+1)j} = \frac{-1}{n}\cos\tilde{q}_j, \quad (3.28)$$

$$A_{(n+2)j} = \frac{1}{n}\sin\tilde{q}_j, \quad (3.29)$$

$$A_{(n+3)j} = -\delta_{nj}. \quad (3.30)$$

The constant term in the $k$th ($k = 1, \ldots, n$) equation in (3.23) is denoted as

$$B_k = \sum_{i=1}^{n}[x_1(s_i)(\frac{-1}{n}\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m) + x_2(s_i)(\frac{1}{n}\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m)]$$
$$+\sum_{i=1}^{n}[\frac{1}{n^2}(\sum_{j=1}^{i}\sin\tilde{q}_j(\sum_{m=1}^{i}\delta_{mk}\cos\tilde{q}_m) - \sum_{j=1}^{i}\cos\tilde{q}_j(\sum_{m=1}^{i}\delta_{mk}\sin\tilde{q}_m))], \quad (3.31)$$
$$k = 1, \ldots, n.$$

The constant terms in Eq. (3.19) are denoted as

$$B_{n+1} = x_1^d - \frac{1}{n}\sum_{j=1}^{n}\sin\tilde{q}_j, \quad (3.32)$$

$$B_{n+2} = x_2^d - \frac{1}{n}\sum_{j=1}^{n}\cos\tilde{q}_j, \quad (3.33)$$

$$B_{n+3} = z_1^d - \tilde{q}_n. \quad (3.34)$$

By using the definitions in Eqs. (3.24), (3.25), (3.26), (3.27), (3.28), (3.29), (3.30), (3.31), (3.32), (3.33), and (3.34), one can form a $(n + 3) \times (n + 3)$ linear system of algebraic equations in terms of $n$ unknown $\epsilon_j$'s and three unknown $\lambda_l$'s in the following matrix format

$$\mathbf{A}_{(n+3)\times(n+3)}\begin{bmatrix}\boldsymbol{\epsilon}_{n\times 1}\\ \boldsymbol{\lambda}_{3\times 1}\end{bmatrix}+\mathbf{B}_{(n+3)\times 1}=\mathbf{0}. \tag{3.35}$$

The numerical solution procedure for the above optimization problem can be described as follows.

1. For a desired augmented task space of the end-effector, $\mathbf{x}^d = [x_1^d, x_2^d, z_1^d]^T$, a backbone curve is determined.
2. Based on the backbone curve, the desired joint positions of the $n$ joints of the manipulator, $(x_1(s_i), x_2(s_i)$ for $i = 1, \ldots, n)$, are calculated.
3. Based on the approximate backbone curve slope at the desired joint positions, the joint angles are approximated as $\tilde{q}_j$'s (Eq. 3.17).
4. With the approximate joint angles, the matrices $\mathbf{A}$ and $\mathbf{B}$ are calculated (Eqs. 3.24 to 3.34), and the joint correction angles are found using Eq. (3.35).
5. If the joint angle corrections $\epsilon_j$'s are small enough, the corrected joint angles $q_j = \tilde{q}_j + \epsilon_j$ are the final solution for the manipulator's joint postures.
6. If the calculated $\epsilon_j$'s are not small enough, the corrected joint angles $q_j = \tilde{q}_j + \epsilon_j$ are used as the new joint angle estimate $\tilde{q}_j$ and this procedure is repeated starting from Step 4.

*Example 3.3.* Consider a 10-link manipulator with a total length of 1 m. If the base of the manipulator is fixed at $(x_1, x_2) = (0, 0)$ m, compute the manipulator's joint angles such that the end-effector makes a $90^\circ$ angle with the $x_2$ coordinate axis and its tip positions at $(x_1, x_2) = (0.4, 0.5)$ m. Assume that the first link approximately makes a $-15^\circ$ angle with the $x_2$ coordinate axis.

*Solution.* For solving this example, the procedure explained above must be used. The details of the solution procedure follows:

1. Since the manipulator is a planar one, the planar Cartesian coordinates $\mathbf{x} = [x_1, x_2]^T$ can adequately describe the position of the end-effector. A backbone curve that lies in the plane of motion of the manipulator is defined as

$$\mathbf{x}(s)=\begin{bmatrix}\int_0^s L\sin\phi(\sigma)d\sigma\\ \int_0^s L\cos\phi(\sigma)d\sigma\end{bmatrix}, \tag{3.36}$$

where the manipulator length is $L = 1$ m. Since the manipulator is planar, only two mode shapes and two mode participation factors ($\mathbf{a} = [a_1, a_2]^T$) are used to define the backbone curve. The first equation in (3.3) becomes

$$\begin{aligned}\phi(s) &= a_1\sin(2\pi s)+a_2(1-\cos(2\pi s))\\ &\quad + b_{1\phi}(1-\sin(\pi s/2))+b_{2\phi}\sin(\pi s/2),\end{aligned} \tag{3.37}$$

where according to the problem definition, one must set $b_{1\phi} = -\pi/12$ rad, $b_{2\phi} = \pi/2$ rad, and $\mathbf{x}_D = [0.4, 0.5]^T$ m. After the Jacobian is derived from Eq. (3.7), the mode participation factors can be found by using the iterative Eq. (3.6) as

**Table 3.1** Desired joint positions ($x_i(s_j)$) and the results for joint angle estimates and corrections for the first iteration

| Link $j$ | $s_j$ | $x_1(s_j)(m)$ | $x_2(s_j)$ (m) | $\tilde{q}_j$ (deg) | $\epsilon_j$ (deg) | $q_j$ (deg) |
|---|---|---|---|---|---|---|
| 1 | 0.1 | −0.029 | 0.096 | −16.9 | 0.1 | −16.8 |
| 2 | 0.2 | −0.063 | 0.190 | −20.1 | −0.2 | −20.3 |
| 3 | 0.3 | −0.093 | 0.285 | −17.3 | −0.3 | −17.7 |
| 4 | 0.4 | −0.100 | 0.384 | −4.0 | −0.3 | −4.3 |
| 5 | 0.5 | −0.066 | 0.477 | 19.9 | −0.0 | 19.9 |
| 6 | 0.6 | 0.009 | 0.540 | 50.2 | 0.5 | 50.7 |
| 7 | 0.7 | 0.107 | 0.559 | 79.4 | 0.7 | 80.1 |
| 8 | 0.8 | 0.205 | 0.542 | 99.7 | 0.4 | 100.2 |
| 9 | 0.9 | 0.301 | 0.514 | 106.2 | 8.0 | 114.2 |
| 10 | 1.0 | 0.400 | 0.500 | 98.1 | −8.1 | 90.0 |

$$\mathbf{a} = [-0.5350, -0.2150]^T. \tag{3.38}$$

2. Using Eq. (3.36), the coordinates of 10 equi-distant points along the backbone curve, $x_i(s_j)$, are determined. These coordinates correspond to the desired position of the link $j$'s end-point. The results are listed in the third and forth column of Table 3.1.
3. Using Eq. (3.17), the first estimations of the joint angles $\tilde{q}_j$ are calculated. The results are listed in the fifth column of Table 3.1.
4. The matrices **A** and **B** are formed, and the first correction angles $\epsilon_j$ are found by solving Eq. (3.35). The results for the first iteration are listed in the sixth column of Table 3.1.
5. As can be seen in the sixth column of Table 3.1, some of the calculated joint angle corrections are quite large. Therefore, the solution procedure must be iterated for a better convergence. A convergence condition must be defined. Limiting the average per joint of the 2-norm of the correction angles, $\|\boldsymbol{\epsilon}\|_2/n$, to $10^{-4}$ rad seems to be a good choice. This means the average correction angle for the final solution will be less than 0.0057 degrees.
6. For each new iteration, the new estimated joint angles are set as $\tilde{q}_j + \epsilon_j$. After 4 iterations, the average per joint of the 2-norm of the correction angles becomes $\|\boldsymbol{\epsilon}\|_2/n = 2.7372 \times 10^{-5}$ rad, which is less than the limit selected in the previous step. The joint angles, $q_j = \tilde{q}_j + \epsilon_j$, after the 4th iteration, as the final solution, are

$$\mathbf{q} = [-16.8, -20.2, -17.6, -4.2, 20.2, 51.1, 80.1, 100.5, 113.4, 90.0]^T \text{ deg.} \tag{3.39}$$

Note that the results for the angles are converted to degrees only for ease of imagination. All the angles used in the actual calculations are in radians. The manipulator at a posture corresponding to the final solution and its corresponding backbone curve are shown in Fig. 3.4.

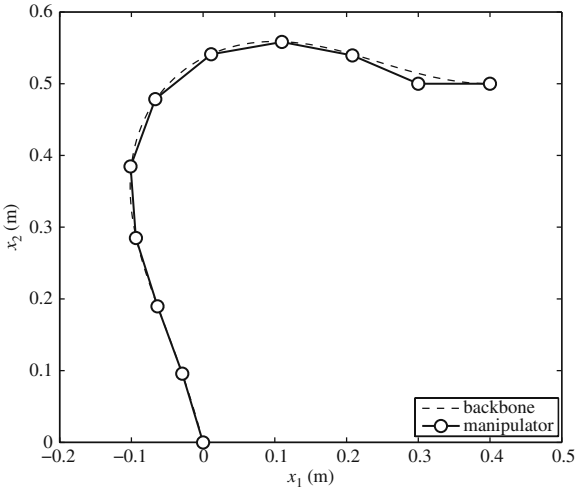


**Fig. 3.4** A 10-link hyper-redundant manipulator fitted on its backbone curve via the CLSFM

### *3.3.2 Recursive Fitting Method (RFM)*

A spatial hyper-redundant robot that consists of $n$ links connected by $n$ universal joints (Fig. 3.5) is considered. The universal joint frames are shown in Fig. 3.6, where the frames are attached using the Denavit-Hartenberg convention. The local coordinate systems, $\{k\}$ and $\{k'\}$, are defined such that the $X_k$-axis is along the link $k$ and the two angles of the universal joint, $\theta_k$ and $\gamma_k$, are about the local $Z_{k'}$ and $Z_k$ axes, respectively. A RFM is introduced to determine the joint positions and the joint angles that will fit the links on the backbone curve.

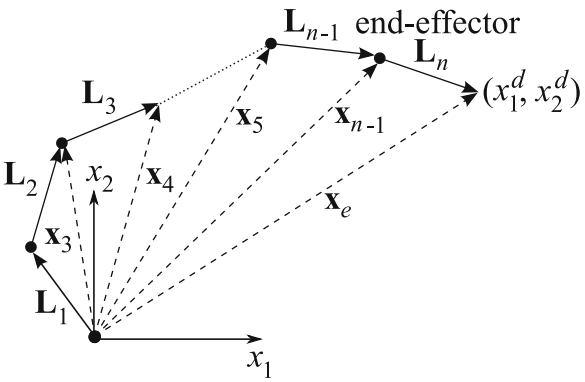


**Fig. 3.5** The vectors representing the links of a spatial hyper-redundant robot

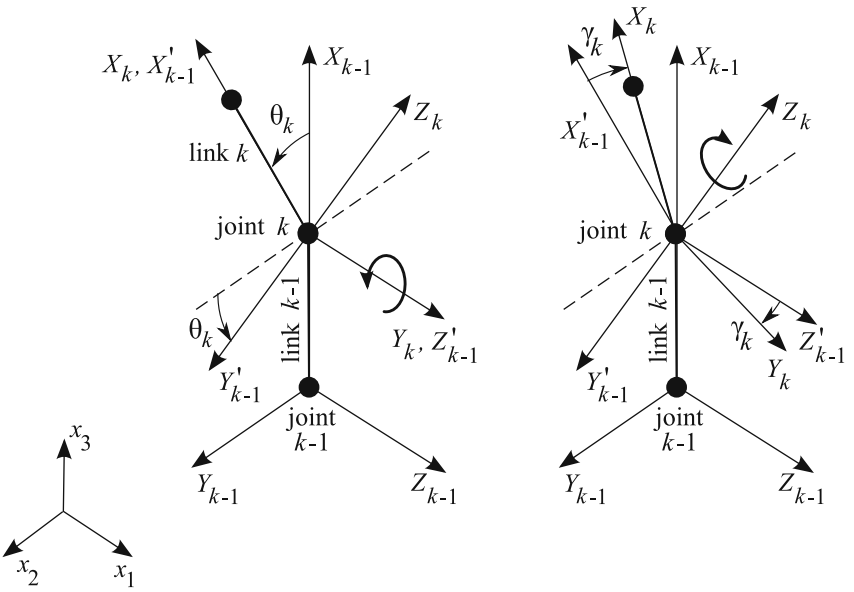


**Fig. 3.6** The joint frames for a universal jointed hyper-redundant manipulator. Frame $\{k-1\}$ is attached to link $k-1$. Frame $\{k'\}$ is the intermediate universal joint frame, representing the rotation for the first joint parameter $\theta_k$. Frame $\{k\}$ is attached to link $k$. This frame represents the rotation for the second joint parameter $\gamma_k$

#### 3.3.2.1 Joint Positions on the Backbone Curve

Since the position, $\mathbf{x}_e$, and the orientation of the end-effector are known, the coordinates of the last joint can be obtained as $\mathbf{x}_n = \mathbf{x}_e - \mathbf{L}_n$, (Fig. 3.5). Next, a backbone curve is introduced. The backbone curve ends at the last joint ($\mathbf{x}_D = \mathbf{x}_n$) with its orientation tangent to the end-effector and has a length of $L$ at least equal to the sum of the robot link lengths excluding the end-effector. The position of the remaining joints on the backbone curve, $\mathbf{x}_k = \mathbf{x}(s_k)$, are determined by

$$\|\mathbf{x}_{k+1} - \mathbf{x}(s_k)\| = l_k \qquad k = n-1, \ldots, 3, \tag{3.40}$$

where $l_k$ is the length of link $k$. The above nonlinear equations are solved recursively backward for $s_k$ using a numerical method, such as the bisection method (see Section A.5).

*Example 3.4.* Consider an 11-link hyper-redundant manipulator ($n = 11$) with link lengths equal to 0.1 m. Let us specify the end-effector tip position as $\mathbf{x}_e = \begin{bmatrix}0.4 & 0.3 & 0.2\end{bmatrix}$ m. The end-effector is oriented such that both its azimuth, $A$, and meridian angle, $M$, are zero (see Fig. 3.1 for the definition of angles). Compute the position of the joints on the backbone curve.

*Solution.* Using the information given, one can compute the vector representing the end-effector as

$$\mathbf{L}_n = l_{11}\left[\sin A \ \cos A \cos M \ \cos A \sin M\right]^T = \left[0.0 \ 0.1 \ 0.0\right]^T \text{ m}. \tag{3.41}$$

Therefore, the position of the end-effector's joint, $j_{11}$, is

$$\mathbf{x}_{11} = \mathbf{x}_e - \mathbf{L}_{11} = \left[0.4 \ 0.2 \ 0.2\right]^T \text{ m}. \tag{3.42}$$

Now, we consider a backbone curve with unit length corresponding to the 10 links (excluding the end-effector). The end of this backbone curve coincides with the end-effector joint. The mode participation factor of Example 3.2 can be used here. The position of link 10 joint, $j_{10}$, on the backbone curve will be determined by using Eq. (3.40) as

$$\|\mathbf{x}_{11} - \mathbf{x}(s_{10})\| = l_{10}. \tag{3.43}$$

If the backbone curve was a straight line, $s_{10}$ would be equal to 0.9. Therefore, we choose $s_{10} = 0.9$ as the first guess for the solution to Eq. (3.43). Using Eq. (3.1), we evaluate $\mathbf{x}(s_{10})$ and using Eq. (3.43), we compute $l_{10}$. We iterate Eq. (3.43) by utilizing the bisection method (see Section A.5) until $l_{10}$ converges to 0.1. The final value for $s_{10}$ is 0.8999, which corresponds to the joint position

$$\mathbf{x}(s_{10}) = \mathbf{x}_{10} = \left[0.4132 \ 0.1009 \ 0.1993\right]^T \text{ m}. \tag{3.44}$$

This procedure is repeated for other links except the first and second links. The results are given in Table 3.2. In this fitting method, the backbone curve is approximated with a number of lines with specified lengths. The backbone curve length $L$ is chosen at least equal to the sum of the robot link lengths excluding the end-effector. Each link consumes a portion of the backbone curve that is longer than the link length. Consequently, when the third joint is located on the backbone curve, the remaining portion of the backbone curve will be shorter than the sum of the remaining two links lengths. Hence, it is guaranteed that one can fit the first two links between the third joint, located at $\mathbf{x}(s_3)$. To position the first two links, we

**Table 3.2** Joints length parameters and positions for Examples 3.4 and 3.5

| $k$ | $s$ | $(x_k)_1$ (m) | $(x_k)_2$ (m) | $(x_k)_3$ (m) |
|---|---|---|---|---|
| 11 | 1.0000 | 0.4000 | 0.2000 | 0.2000 |
| 10 | 0.8999 | 0.4132 | 0.1009 | 0.1993 |
| 9 | 0.7999 | 0.4409 | 0.0052 | 0.1908 |
| 8 | 0.6999 | 0.4561 | −0.0915 | 0.1704 |
| 7 | 0.5989 | 0.4339 | −0.1836 | 0.1384 |
| 6 | 0.4975 | 0.3670 | −0.2468 | 0.0993 |
| 5 | 0.3962 | 0.2757 | −0.2591 | 0.0605 |
| 4 | 0.2954 | 0.1880 | −0.2226 | 0.0290 |
| 3 | 0.1951 | 0.1156 | −0.1565 | 0.0092 |
| 2 | – | 0.0396 | −0.0916 | 0.0054 |
| 1 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

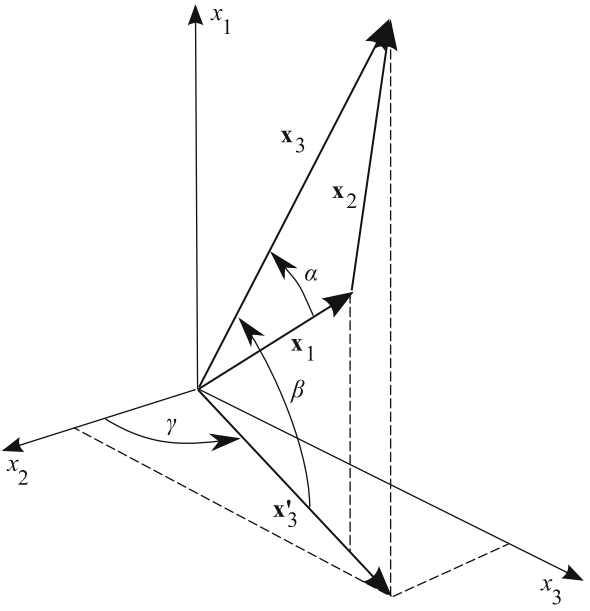


**Fig. 3.7** Notation used in Example 3.5

arbitrarily assume that they are always in the same plane, i.e., $\theta_2$ and $\dot{\theta}_2$ are always zero (other assumptions are also possible). The vectors representing the first and the second links are shown in Fig. 3.7. Note that the vector $\mathbf{x}_3$, showing the position of joint 3 with respect to the inertial coordinate system, is already determined in the previous example. With the aid of the angles shown in Fig. 3.7, one can find $\mathbf{x}_2$, which is the vector representing the position of joint 2 with respect to the inertial coordinate system.

*Example 3.5.* Use the position of the third joint obtained in Example 3.4 and compute the position of the second joint. Assume that the first and second link lay in the same plane perpendicular to the $x_2 - x_3$ plane and have the same length $l$. Joint one is located at the origin of the coordinate system.

*Solution.* The length of vector $\mathbf{x}_3$, defining the poisition of the third joint with respect to the origin of the inertial frame is given as

$$|\mathbf{x}_3| = \sqrt{(x_3)_1^2 + (x_3)_2^2 + (x_3)_3^2} = 0.1948 \text{ m}.$$

The length of the projection of $\mathbf{x}_3$ on the *YZ* plane is

$$|\mathbf{x}'_3| = \sqrt{(x_3)_2^2 + (x_3)_3^2} = 0.1568 \text{ m}.$$

The angles shown in Fig. 3.7 may be computed as

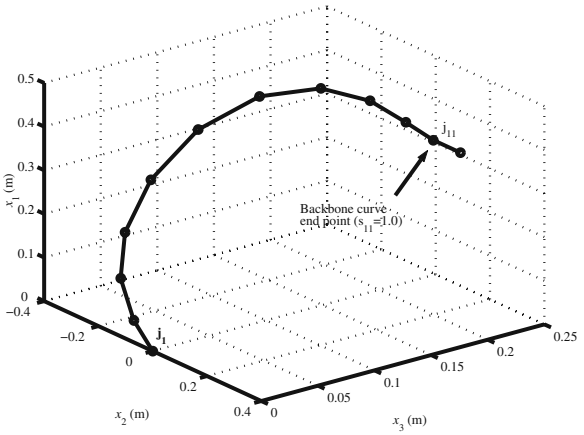


**Fig. 3.8** Links and joints based on the position data listed in Table 3.2

$$\alpha = \arccos\left(\frac{|\mathbf{x}_3|/2}{l}\right) = 13.05^\circ,$$
$$\beta = \arctan\left(\frac{x_3 - x_1}{|\mathbf{x}_3'|}\right) = 36.41^\circ,$$
$$\gamma = \arctan\left(\frac{(x_3)_3}{(x_3)_2}\right) = 176.62^\circ,$$

Finally, the position of the second joint can be obtained as

$$(x_3)_1 = l\sin(\beta - \alpha) = 0.0396 \text{ m},$$
$$(x_3)_2 = l\cos(\beta - \alpha)\cos(\gamma) = -0.0916 \text{ m},$$
$$(x_3)_3 = l\cos(\beta - \alpha)\sin(\gamma) = 0.0054 \text{ m}.$$

These values are listed in Table 3.2. Note that since the second joint is not on the backbone curve, there is no corresponding backbone curve length parameter $s$ for this joint. Figure 3.8 shows the links and joint based on the data listed in Table 3.2.

### 3.3.2.2 Joint Angles from Joint Positions

The universal joint angles can be calculated when the joint positions are known. The vector representing each link $k$, $\mathbf{L}_k$, is written in the link $k$'s local frame $\{k\}$ as $\mathbf{L}_k^{(k)} = \begin{bmatrix} l_k & 0 & 0 \end{bmatrix}^T$. The same vector can be written in frame $\{k-1\}$ as $\mathbf{L}_k^{(k-1)} = {}_k^{k-1}\mathbf{R}\mathbf{L}_k^{(k)}$, where ${}_k^{k-1}\mathbf{R}$ is the rotation matrix that relates the orientation of frames $\{k\}$ and $\{k-1\}$. $\mathbf{L}_k$ and $\mathbf{L}_k^{(k)}$ are related as

$${}_k^{k-1}\mathbf{R}\mathbf{L}_k^{(k)} = {}_{k-1}^{0}\mathbf{R}^{-1}\mathbf{L}_k, \qquad k = 1, 2, \ldots, n. \tag{3.45}$$

Equation (3.45) can be solved for the angles of the $k$-th universal joint, $\theta_k$ and $\gamma_k$.

*Example 3.6.* Using the joint positions listed in Table 3.2, compute the joint angles of the first joint.

*Solution.* For the first link, $k = 1$. Therefore, Eq. (3.45) reduces to

$$ {}^{0}_{1}\mathbf{R}\mathbf{L}_1^{(1)} = {}^{0}_{0}\mathbf{R}^{-1}\mathbf{L}_1. $$

Note that ${}^{0}_{0}\mathbf{R}$ is the identity matrix. ${}^{0}_{1}\mathbf{R}$, which transfers the reference frame to the frame of link one, consists of two rotation matices. The first rotation matrix, ${}^{0}_{1'}\mathbf{R}$, transfers the reference frame to the first joint frame of joint one, $\{1'\}$. The second rotation matrix, ${}^{1'}_{1}\mathbf{R}$, transfers the first joint frame of joint one to second joint frame of joint one, $\{1\}$. According to Fig. 3.6 and by using the Denavit-Hartenberg frame convention, one can show that the rotation matrices are

$$ {}^{0}_{1'}\mathbf{R} = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad {}^{1'}_{1}\mathbf{R} = \begin{bmatrix} \cos(-\gamma_1) & 0 & -\sin(-\gamma_1) \\ 0 & 1 & 0 \\ \sin(-\gamma_1) & 0 & \cos(-\gamma_1) \end{bmatrix}, $$

and

$$ {}^{1'}_{1}\mathbf{R}\,{}^{0}_{1'}\mathbf{R} = {}^{0}_{1}\mathbf{R}. $$

Also, the vector defining the link in joint one frame and in the reference frame are

$$ \mathbf{L}_1^{(1)} = \begin{bmatrix} l_1 & 0 & 0 \end{bmatrix}^T, \qquad \mathbf{L}_1 = \begin{bmatrix} (x_2)_1 - (x_1)_1 & (x_2)_2 - (x_1)_2 & (x_2)_3 - (x_1)_3 \end{bmatrix}^T, $$

respectively. Note that the first joint is always assumed to be at the origin of the inertial coordinate system therefore, $\mathbf{x}_1 = [(x_1)_1, (x_1)_2, (x_1)_3]^T = [0, 0, 0]^T$. Using Eq. (3.45) yields

$$ \begin{bmatrix} l_1 \cos(\theta_1)\cos(-\gamma_1) \\ l_1 \sin(\theta_1) \\ l_1 \cos(\theta_1)\sin(-\gamma_1) \end{bmatrix} = \begin{bmatrix} (x_2)_1 - (x_1)_1 \\ (x_2)_2 - (x_1)_2 \\ (x_2)_3 - (x_1)_3 \end{bmatrix}. $$

Solving the above equations in terms of $\theta_1$ and $\gamma_1$, one can show that

$$ \theta_1 = \arcsin\left(\frac{(x_2)_2}{l_1}\right) = -66.35^\circ, \qquad \gamma_1 = -\arctan\left(\frac{(x_2)_3}{(x_2)_1}\right) = -7.77^\circ. $$

After computing $\theta_1$ and $\gamma_1$, the matrix ${}^{0}_{1}\mathbf{R}$ is known and can be used in the following relation to compute $\theta_2$ and $\gamma_2$.

$$ {}^{1}_{2}\mathbf{R}\mathbf{L}_2^{(2)} = {}^{0}_{1}\mathbf{R}^{-1}\mathbf{L}_2 $$

**Table 3.3** Joints angles for Example 3.6

| $k$ | $\theta_k$ (deg) | $\gamma_k$ (deg) |
|---|---|---|
| 1 | −66.42 | −7.77 |
| 2 | 25.80 | 4.13 |
| 3 | −0.75 | −11.51 |
| 4 | 20.31 | −6.59 |
| 5 | 29.22 | −3.41 |
| 6 | 34.23 | −0.26 |
| 7 | 32.88 | 4.95 |
| 8 | 22.88 | 12.38 |
| 9 | 5.67 | 18.16 |
| 10 | −10.35 | 16.79 |
| 11 | −8.79 | 19.14 |

Using the same procedure discussed in this example, one can compute the other joint angles. All joint angles are listed in Table 3.3.

### *3.3.3 Comparison Between the CLSFM and the RFM*

The CLSFM method has a much higher computational cost compared to that of the RFM. Because of the high computational load of the CLSFM, the real-time implementation of this method for spatial hyper-redundant manipulators is very difficult to realize. That is the reason why the application of this method has only been presented for the planar hyper-redundant manipulators.

The dimensions of the coefficient matrix **A** and the constant matrix **B** for the CLSFM depend on the manipulator's number of joints. A spatial manipulator with $n$ links has $2n$ joint parameters. For each iteration for finding the joint angle corrections in the CLSFM for spatial case, the $2n$ by $2n$ linear system of Eq. (3.35) must be solved. This solution has a computational load proportional to $4n^2$. In the RFM, the nonlinear Eq. (3.40) is solved via iterations for $n-2$ Cartesian joint positions. However, since there are extra calculations needed for finding the end-effector's and the second joint's Cartesian positions, one can assume the computation load for finding the joint Cartesian positions is proportional to $n$. Finding the joint angles in the RFM is proportional to $2n$ for the spatial case, since there are two joint angles to be determined for each link. Therefore, finding the joint angles via the RFM has a computational load proportional to $3n$, which is much less than that of the CLSFM.

The above argument only considers the computational load for one iteration. The nature of equations to be solved iteratively in the RFM (e.g., Eq. 3.40) is such that they converge faster to a given accuracy tolerance compared to the CLSFM. In other words, CLSFM needs more iterations to achieve a given solution convergence tolerance than the RFM.

Another important feature of the RFM is that it allows the inverse kinematic solution at the velocity level. This is because, in this method, the joints are located on the backbone curve at all times. Therefore, the velocity information of the points on the backbone curve provide means to calculate joint velocities. This is discussed in more details in the next section.

## 3.4 Inverse Velocity Propagation

In the inverse velocity problem, the joint rotation rates must be determined such that a given velocity vector is generated at the end-effector. Here, the velocity propagation problem is solved by utilizing the results of the RFM. The solution procedure is as follows. First, the linear velocity of an arbitrary point on the backbone curve is calculated. Next, the linear velocities of the joints on the backbone curve are computed. Finally, the equations for the calculation of the joint angular velocities are presented.

### *3.4.1 Velocity of a Point on the Backbone Curve*

The linear velocity of joint $k$, $\mathbf{v}_k$, and the linear velocity of its corresponding point on the backbone curve, $\mathbf{w}_k$, are related as

$$\mathbf{v}_k = \mathbf{w}_k + \dot{\mathbf{S}}_k, \tag{3.46}$$

where $\dot{\mathbf{S}}_k = \dot{S}_k \mathbf{u}_k$ denotes the relative velocity of joint $k$ and its corresponding point on the backbone curve in the direction $\mathbf{u}_k$. $\mathbf{w}_k$ is calculated by taking the time derivative of Eq. (3.2),

$$\mathbf{w}_k = \dot{\mathbf{x}}(s_k) = \mathbf{J}_a(\mathbf{a}, s_k)\dot{\mathbf{a}} + \mathbf{J}_{b1}(s_k)\dot{\mathbf{b}}_1 + \mathbf{J}_{b2}(s_k)\dot{\mathbf{b}}_2, \tag{3.47}$$

where $\mathbf{J}_a(\mathbf{a}, s_k)$ is the modal Jacobian matrix, $\mathbf{J}_{b1}(s_k)$ and $\mathbf{J}_{b2}(s_k)$ are $3 \times 2$ Jacobians with respect to $\mathbf{b}_1 = [b_{1\phi}, b_{1\psi}]^T$ and $\mathbf{b}_2 = [b_{2\phi}, b_{2\psi}]^T$, and $s_k$ is the curve length parameter derived in Eq. (3.40). Note that the linear velocity of joint $n$, located at the end of the backbone curve ($s_n = 1$), can be computed by specifying the linear velocity of the tip of the end-effector and the rate of change of its azimuth and meridian angles. Then, the modal participation velocity vector $\dot{\mathbf{a}}$ can be computed, given $\dot{\mathbf{x}}(1)$ and rearranging Eq. (3.47) as

$$\dot{\mathbf{a}} = [\mathbf{J}_a(\mathbf{a}, 1)]^{-1} \left( \dot{\mathbf{x}}(1) - \mathbf{J}_{b1}\dot{\mathbf{b}}_1 - \mathbf{J}_{b2}\dot{\mathbf{b}}_2 \right). \tag{3.48}$$

*Example 3.7.* Derive the Jacobian matrices introduced in Eq. (3.47). Assume the mode shapes given in Eqs. (3.3).

*Solution.* The Jacobian matrices are defined as follows

$$\mathbf{J}_a(\mathbf{a}, s) = \frac{\partial \mathbf{x}(s)}{\partial \mathbf{a}^T},$$

where $\mathbf{x}(s)$ is defined in Eq. (3.2). The above equation for $\mathbf{J}_a(\mathbf{a}, s)$ is similar to the Jacobian matrix derived in Example 3.2. There is no need to derive the components again. The Jacobian matrix corresponding to $\mathbf{b}_1$ is

$$\mathbf{J}_{b1}(s) = \frac{\partial \mathbf{x}(s)}{\partial \mathbf{b}_1^T} = \begin{bmatrix} \frac{\partial x_1}{\partial b_{1\phi}} & \frac{\partial x_1}{\partial b_{1\psi}} \\ \frac{\partial x_2}{\partial b_{1\phi}} & \frac{\partial x_2}{\partial b_{1\psi}} \\ \frac{\partial x_3}{\partial b_{1\phi}} & \frac{\partial x_3}{\partial b_{1\psi}} \end{bmatrix},$$

where

$$\frac{\partial x_1}{\partial b_{1\phi}} = \int_0^s L(1 - \sin(\pi s/2)) \cos\phi(\sigma) \cos\psi(\sigma) d\sigma,$$

$$\frac{\partial x_1}{\partial b_{1\psi}} = \int_0^s -L(1 - \sin(\pi s/2)) \sin\phi(\sigma) \sin\psi(\sigma) d\sigma,$$

$$\frac{\partial x_2}{\partial b_{1\phi}} = \int_0^s -L(1 - \sin(\pi s/2)) \sin\phi(\sigma) \cos\psi(\sigma) d\sigma,$$

$$\frac{\partial x_2}{\partial b_{1\psi}} = \int_0^s -L(1 - \sin(\pi s/2)) \cos\phi(\sigma) \sin\psi(\sigma) d\sigma,$$

$$\frac{\partial x_3}{\partial b_{1\phi}} = 0,$$

$$\frac{\partial x_3}{\partial b_{1\psi}} = \int_0^s L(1 - \sin(\pi s/2)) \cos\psi(\sigma) d\sigma.$$

The Jacobian matrix corresponding to $\mathbf{b}_2$ is

$$\mathbf{J}_{b2}(s) = \frac{\partial \mathbf{x}(s)}{\partial \mathbf{b}_2^T} = \begin{bmatrix} \frac{\partial x_1}{\partial b_{2\phi}} & \frac{\partial x_1}{\partial b_{2\psi}} \\ \frac{\partial x_2}{\partial b_{2\phi}} & \frac{\partial x_2}{\partial b_{2\psi}} \\ \frac{\partial x_3}{\partial b_{2\phi}} & \frac{\partial x_3}{\partial b_{2\psi}} \end{bmatrix},$$

where

$$\frac{\partial x_1}{\partial b_{2\phi}} = \int_0^s L \sin(\pi s/2) \cos\phi(\sigma) \cos\psi(\sigma) d\sigma,$$

$$\frac{\partial x_1}{\partial b_{2\psi}} = \int_0^s -L \sin(\pi s/2) \sin\phi(\sigma) \sin\psi(\sigma) d\sigma,$$

$$\frac{\partial x_2}{\partial b_{2\phi}} = \int_0^s -L \sin(\pi s/2) \sin\phi(\sigma) \cos\psi(\sigma) d\sigma,$$

$$\frac{\partial x_2}{\partial b_{2\psi}} = \int_0^s -L \sin(\pi s/2) \cos\phi(\sigma) \sin\psi(\sigma) d\sigma,$$

$$\frac{\partial x_3}{\partial b_{2\phi}} = 0,$$

$$\frac{\partial x_3}{\partial b_{2\psi}} = \int_0^s L \sin(\pi s/2) \cos\psi(\sigma) d\sigma.$$

For every given backbone curve configuration, **a**, and at any given point on the backbone curve defined by $s_k$, one can evaluate these integrals numerically.

*Example 3.8.* Assume the azimuth and the meridian angles of the tangent to the end point of a backbone curve as is given in Eq. (3.9). Compute $\dot{\mathbf{b}}_1$ and $\dot{\mathbf{b}}_2$ introduced in Eq. (3.47).

*Solution.* Equations (3.10) are rewritten for the start point of the backbone curve ($s = 0$).

$$\phi(0) = b_{1\phi} = \arctan\left(\frac{\tan A_s}{\cos M_s}\right)$$

$$\psi(0) = b_{1\psi} = \arcsin(\sin M_s \cos A_s)$$

Substituting Eqs. (3.9) in the above equations yields

$$\begin{aligned} b_{1\phi} &= \beta + \pi(1 - \delta), \\ b_{1\psi} &= 0. \end{aligned}$$

Therefore,

$$\dot{\mathbf{b}}_1 = \begin{bmatrix} \dot{\beta} + \pi(1 - \dot{\delta}) \\ 0 \end{bmatrix}.$$

Note that $\dot{\beta}$ and $\dot{\delta}$ can be computed in terms of $\dot{x}_1(1)$, $\dot{x}_2(1)$, and $\dot{x}_3(1)$ by differentiating Eqs. (3.10). This means that when the linear velocity of the backbone curve end point ($s = 1$) is known, the numerical value of $\dot{\mathbf{b}}_1$ can be evaluated.

Now, Eqs. (3.10) are rewritten for the end point of the backbone curve ($s = 1$).

$$\phi(1) = b_{2\phi} = \arctan\left(\frac{\tan A_e}{\cos M_e}\right),$$

$$\psi(1) = b_{2\psi} = \arcsin\left(\sin M_e \cos A_e\right),$$

where $A_e$ and $M_e$ are the azimuth and meridian angle of the tangent to the backbone curve at its end point. Differentiating the above equations gives

$$\dot{\mathbf{b}}_2 = \begin{bmatrix} \dot{b}_{2\phi} \\ \dot{b}_{2\psi} \end{bmatrix}.$$

Note that $\dot{b}_{2\phi}$ and $\dot{b}_{2\psi}$ can be written in terms of $\dot{A}_e$ and $\dot{M}_e$ by differentiating $b_{2\phi}$ and $b_{2\psi}$, respectively. This means that when the changes in azimuth and meridian angle of the tangent to the backbone curve at the end point are given, the numerical value of $\dot{\mathbf{b}}_2$ can be evaluated.

*Example 3.9.* Compute the linear velocity of a point on the backbone curve in Example 3.4 coincident with joint ten. The velocity of the curve end point is specified as

$$\dot{\mathbf{x}}(1) = \begin{bmatrix} 0.1175 & 0.1000 & 0.0825 \end{bmatrix}^T \text{ m/s}.$$

Also, assume that the change in the azimuth and meridian angle of the tangent at the curve end point is specified as

$$\dot{A}_e = (\pi/18) \text{ rad/s}, \qquad \dot{M}_e = (\pi/18) \text{ rad/s}.$$

*Solution.* $\dot{\mathbf{b}}_1$ can be calculated based on the curve end point linear velocity.

$$\dot{\mathbf{b}}_1 = \begin{bmatrix} -0.6121 \\ 0 \end{bmatrix} \text{ rad/s}.$$

Also, $\dot{\mathbf{b}}_2$ is computed using the values of $\dot{A}_e$ and $\dot{M}_e$ as

$$\dot{\mathbf{b}}_2 = \begin{bmatrix} 0.1745 \\ 0.1745 \end{bmatrix} \text{ rad/s}.$$

Now, Eq. (3.48) is used to evaluate the rate of change of the mode participation vector as

$$\dot{\mathbf{a}} = \begin{bmatrix} 0.0535 & -0.0109 & -0.0262 \end{bmatrix}^T \text{ 1/s}.$$

Substituting the obtained $\dot{\mathbf{a}}$ in Eq. (3.47) along with $s_{10} = 0.8999$ for joint ten (from Table 3.2) gives

**Table 3.4** Linear velocity of points on the backbone curve coincident with joints

| $k$ | $s$ | $(w_k)_1$ (m/s) | $(w_k)_2$ (m/s) | $(w_k)_3$ (m/s) |
|---|---|---|---|---|
| 11 | 1.0000 | 0.1175 | 0.1000 | 0.0825 |
| 10 | 0.8999 | 0.1022 | 0.0981 | 0.0654 |
| 9 | 0.7999 | 0.0918 | 0.0965 | 0.0496 |
| 8 | 0.6999 | 0.0865 | 0.0983 | 0.0363 |
| 7 | 0.5989 | 0.0875 | 0.1013 | 0.0261 |
| 6 | 0.4975 | 0.0929 | 0.0998 | 0.0184 |
| 5 | 0.3962 | 0.0963 | 0.0902 | 0.0126 |
| 4 | 0.2954 | 0.0909 | 0.0740 | 0.0079 |
| 3 | 0.1951 | 0.0732 | 0.0537 | 0.0040 |
| 2 | – | – | – | – |
| 1 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

$$\mathbf{w}_{10} = \dot{\mathbf{x}}(s_{10}) = \begin{bmatrix} 0.1022 & 0.0981 & 0.0654 \end{bmatrix}^T \text{ m/s}.$$

Note that the same $\dot{\mathbf{a}}$ can be used to evaluate the velocity of the other points on the backbone curve, including but not limited to the points coincident with the other joints. The linear velocity of the points on the backbone curve coincident with the other joints are calculated by using the discussed procedure and are listed in Table 3.4.

### *3.4.2 Linear Velocity of Joints Located on the Backbone Curve*

The linear velocities of the joints at the two ends of each link $k$, $\mathbf{v}_{k+1}$ and $\mathbf{v}_k$, are related as

$$\mathbf{v}_k = \mathbf{v}_{k+1} - \boldsymbol{\omega}_k \times \mathbf{L}_k, \qquad k = n-1, \ldots, 3, \tag{3.49}$$

where

$$\boldsymbol{\omega}_k = \dot{M}_k \hat{\mathbf{i}} + \dot{A}_k (\sin M_k \hat{\mathbf{j}} - \cos M_k \hat{\mathbf{k}}). \tag{3.50}$$

$\dot{A}_k$ and $\dot{M}_k$ are the rates of change of azimuth and meridian angles of link $k$, and $\boldsymbol{\omega}_k$ is the angular velocity vector of link $k$. Substituting from Eq. (3.46) into Eq. (3.49) and rearranging leads to

$$\mathbf{v}_{k+1} - \mathbf{w}_k = \boldsymbol{\omega}_k \times \mathbf{L}_k + \dot{S}_k \mathbf{u}_k, \qquad k = n-1, \ldots, 3. \tag{3.51}$$

Equation (3.51) represents three linear simultaneous equations, which can be solved for $\dot{A}_k$, $\dot{M}_k$, and $\dot{S}_k$. Then, these results are substituted back into Eq. (3.51) and (3.50) to evaluate $\mathbf{v}_k$ and $\boldsymbol{\omega}_k$, for $k = n-1, \ldots, 3$.

*Example 3.10.* Calculate the linear velocity of joint ten of the manipulator described in Example 3.4. Consider the velocity of the backbone curve end point (joint eleven) given in Example 3.9.

*Solution.* First, Eq. (3.51) is expanded. It can be shown that

$$\mathbf{v}_{k+1} - \mathbf{w}_k = \begin{bmatrix} ((l_k)_1 \sin M_k + (l_k)_2 \cos M_k)\dot{A}_k \\ (-(l_k)_1 \cos M_k)\dot{A}_k + (l_k)_3 \dot{M}_k \\ (-(l_k)_1 \sin M_k)\dot{A}_k + (l_k)_2 \dot{M}_k \end{bmatrix} + \begin{bmatrix} \dot{S}_k (u_k)_1 \\ \dot{S}_k (u_k)_2 \\ \dot{S}_k (u_k)_3 \end{bmatrix}, \tag{3.52}$$

where according to Eqs. (3.1) and (3.2)

$$\begin{bmatrix} (u_k)_1 \\ (u_k)_2 \\ (u_k)_3 \end{bmatrix} = \begin{bmatrix} L \sin\phi(s_k) \cos\psi(s_k) \\ L \cos\phi(s_k) \cos\psi(s_k) \\ L \sin\psi(s_k) \end{bmatrix},$$

and

$$\mathbf{L}_k = \begin{bmatrix} (x_{k+1})_1 - (x_k)_1 \\ (x_{k+1})_2 - (x_k)_2 \\ (x_{k+1})_3 - (x_k)_3 \end{bmatrix} = \begin{bmatrix} (l_k)_1 \\ (l_k)_2 \\ (l_k)_3 \end{bmatrix}.$$

Equation (3.52) can be rearranged in terms of unknowns, $\dot{A}_k$, $\dot{M}_k$ and $\dot{S}_k$ as follows

$$\mathbf{v}_{k+1} - \mathbf{w}_k = \begin{bmatrix} (l_k)_3 \sin M_k + (l_k)_2 \cos M_k & 0 & (u_k)_1 \\ -(l_k)_1 \cos M_k & (l_k)_3 & (u_k)_2 \\ -(l_k)_1 \sin M_k & (l_k)_2 & (u_k)_3 \end{bmatrix} \begin{bmatrix} \dot{A}_k \\ \dot{M}_k \\ \dot{S}_k \end{bmatrix}. \tag{3.53}$$

From Table 3.2, one can see

$$\mathbf{L}_{10} = \mathbf{x}_{11} - \mathbf{x}_{10} = \begin{bmatrix} -0.0132 & 0.0991 & 0.0007 \end{bmatrix}^T \text{ m}.$$

Also, from Example 3.9, one can write

$$\mathbf{w}_{10} = \begin{bmatrix} 0.1022 & 0.0981 & 0.0654 \end{bmatrix}^T \text{ m/s}.$$

The linear velocity of the backbone curve end point was given as

$$\mathbf{x}_{11} = \dot{\mathbf{x}}(1) = \begin{bmatrix} 0.1175 & 0.1000 & 0.0825 \end{bmatrix}^T \text{ m/s}.$$

Substituting the above numerical values along with $k = 10$ in Eq. (3.53) and solving for $\dot{A}_k$, $\dot{M}_k$ and $\dot{S}_k$ yields

$$\dot{A}_{10} = -0.1537 \text{ rad/s}, \qquad \dot{M}_{10} = 0.1729 \text{ rad/s}, \qquad \dot{S}_{10} = 0.0000 \text{ m/s}.$$

Now, $\boldsymbol{\omega}_{10}$ is evaluated by using Eq. (3.50) as

**Table 3.5** Linear velocity of joints on the backbone curve

| $k$ | $s$ | $\dot{A}_k$ (rad/s) | $\dot{M}_k$ (rad/s) | $\dot{S}_k$ (m/s) | $v_{kx}$ (m/s) | $v_{ky}$ (m/s) | $v_{kz}$ (m/s) |
|---|---|---|---|---|---|---|---|
| 10 | 0.8999 | −0.1537 | 0.1729 | 0.0000 | 0.1022 | 0.0981 | 0.0654 |
| 9 | 0.7999 | −0.1086 | 0.1623 | 0.0000 | 0.0918 | 0.0965 | 0.0496 |
| 8 | 0.6999 | −0.0533 | 0.1349 | 0.0001 | 0.0865 | 0.0984 | 0.0364 |
| 7 | 0.5989 | 0.0117 | 0.1093 | 0.0004 | 0.0877 | 0.1017 | 0.0262 |
| 6 | 0.4975 | 0.0789 | 0.0743 | 0.0008 | 0.0935 | 0.1001 | 0.0188 |
| 5 | 0.3962 | 0.0928 | −0.1935 | 0.0011 | 0.0973 | 0.0900 | 0.0130 |
| 4 | 0.2954 | −0.1097 | −0.3032 | 0.0014 | 0.0920 | 0.0732 | 0.0083 |
| 3 | 0.1951 | −0.2581 | −0.1427 | 0.0016 | 0.0742 | 0.0525 | 0.0042 |
| 2 | – | – | – | – | – | – | – |
| 1 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

$$\boldsymbol{\omega}_{10} = \begin{bmatrix} 0.1729 & 0.0010 & -0.1537 \end{bmatrix}^T \text{ rad/s}.$$

Finally, Eq. (3.49) with $k = 10$ results in the joint ten linear velocity,

$$\mathbf{v}_{10} = \begin{bmatrix} 0.1022 & 0.0981 & 0.0654 \end{bmatrix}^T \text{ m/s}.$$

By using the same procedure discussed above, other joints' linear velocities can be computed. The results are listed in Table 3.5.

### *3.4.3 Joint Angular Velocities*

The first two joint velocities are determined separately since the second joint is not on the backbone curve in the RFM. Starting from the base, the linear velocity vector of the third joint is written in terms of the angular velocities of the first two joints as

$$\mathbf{v}_3 = \frac{\partial(\mathbf{L}_1 + \mathbf{L}_2)}{\partial \boldsymbol{\Theta}^T} \cdot \boldsymbol{\Theta}, \tag{3.54}$$

where $\boldsymbol{\Theta} = \begin{bmatrix} \theta_1 & \gamma_1 & \gamma_2 \end{bmatrix}^T$. Transforming Eq. (3.54) to frame {2} results in

$$\begin{bmatrix} l\dot{\gamma}_1 \sin\gamma_2 \\ l(\dot{\gamma}_1 \cos\gamma_2 + \dot{\gamma}_1 + \dot{\gamma}_2) \\ -l\dot{\theta}_1(\cos\gamma_1 + \cos(\gamma_1 + \gamma_2)) \end{bmatrix} = {}^2_0\mathbf{R}\mathbf{v}_3, \tag{3.55}$$

which is solved for $\dot{\theta}_1$, $\dot{\gamma}_1$, and $\dot{\gamma}_2$. Note that $\dot{\theta}_2 = 0$ (see Section 3.3.2.1). The link lengths are assumed to be equal to $l$.

The linear velocity of joint $k+1$ in inertial frame {0}, $\mathbf{v}_{k+1}$, is related to joint $k$ velocity, $\mathbf{v}_k$, as

$${}^0_k\mathbf{R}^{-1}\mathbf{v}_{k+1} = l({}^k\mathbf{J}_{k+1}\Omega_k + {}^k\mathbf{K}_{k+1}\boldsymbol{\omega}_{k-1}) + {}^k\mathbf{K}_{k+1}\mathbf{v}_k, \qquad k = 3, \ldots, n, \tag{3.56}$$

where

$$
{}^{k}\mathbf{J}_{k+1} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ -\cos\gamma_k & 0 \end{bmatrix}, \qquad {}^{k}\mathbf{K}_{k+1} = \begin{bmatrix} \cos\gamma_k & 0 & \sin\gamma_k \\ -\sin\gamma_k & 0 & \cos\gamma_k \\ 0 & 1 & 0 \end{bmatrix},
$$
$$
{}^{k}\mathbf{H}_{k+1} = \begin{bmatrix} 0 & 0 & 0 \\ \sin\theta_k & 0 & \cos\theta_k \\ \cos\theta_k \sin\gamma_k & \cos\gamma_k & -\sin\gamma_k \sin\theta_k \end{bmatrix}. \tag{3.57}
$$

Since the relative linear velocity of joint $k+1$ with respect to joint $k$ is always zero along the local $X_k$-axis (i.e., along the link), the first component of Eq. (3.56) is identically satisfied. Therefore, the second and third components of Eq. (3.56) can be solved for $\Omega_k = \begin{bmatrix} \dot{\theta}_k & \dot{\gamma}_k \end{bmatrix}^T$ as long as the lower $2 \times 2$ part of matrix ${}^{k}\mathbf{J}_{k+1}$ is nonsingular.

Derivation of Eqs. (3.56) to (3.57) are left to the reader as an exercise. These relations can be derived by writing the joint transformation matrices.

*Example 3.11.* Using Eqs. (3.56) to (3.57) evaluate the joint angular velocities.

*Solution.* Note that ${}^{2}_{0}\mathbf{R}$ and $\mathbf{v}_3$ in Eq. (3.55) are known from Examples 3.6 and 3.10, respectively. Therefore, Eq. (3.56) can be solved in terms of $\dot{\theta}_1$, $\dot{\gamma}_1$, and $\dot{\gamma}_2$.

$$
\dot{\theta}_1 = 2.91 \text{ deg/s}, \qquad \dot{\gamma}_1 = 29.07 \text{ deg/s}, \qquad \dot{\gamma}_2 = -4.78 \text{ deg/s}.
$$

Then, Eq. (3.56) is rewritten for $k = 3$ to $k = 11$ and is solved for remaining unknown joint angular velocities two at a time. The results are listed in Table 3.6.

### 3.4.4 Singularity Considerations in Inverse Velocity Propagation

The possible sources of singularity are Eqs. (3.48), (3.51), and (3.55). If the position problem has a solution, the modal Jacobian, $\mathbf{J}_a(\mathbf{a}, 1)$, is nonsingular and Eq. (3.48)

**Table 3.6** Joints angular velocities for Example 3.11

| $k$ | $\dot{\theta}_k$ (deg/s) | $\dot{\gamma}_k$ (deg/s) |
|---|---|---|
| 1 | 2.91 | 29.07 |
| 2 | 0.00 | −4.78 |
| 3 | −3.76 | −9.15 |
| 4 | −1.84 | −5.20 |
| 5 | −1.56 | −4.45 |
| 6 | −1.31 | −4.52 |
| 7 | −18.04 | 51.27 |
| 8 | 16.90 | −61.96 |
| 9 | 1.48 | −3.07 |
| 10 | 1.81 | 0.00 |
| 11 | 0.00 | 0.00 |

will always have a solution. For the other equations, one must consider that the mode shapes presented in Eqs. (3.4) and (3.5) generate low curvatures, i.e., $|\theta_k|$ and $|\gamma_k| \ll \pi/2$.

Equation (3.51) has a solution as long as $\boldsymbol{\omega}_k \times \mathbf{L}_k$ and $\mathbf{u}_k$ are not parallel. They are never parallel since $\mathbf{L}_k$ is always nearly fitted tangent to the backbone curve (nearly parallel to $\mathbf{u}_k$). In Eq. (3.55), singularity occurs if $\gamma_2 = 0$, $\gamma_2 = \pi$, or $|2\gamma_1 + \gamma_2| = \pi$. However, $\gamma_2 \neq 0$ in the RFM (see Section 3.3.2.1). Also, $\gamma_2 \neq \pi$ and $|2\gamma_1 + \gamma_2| << \pi$, since all joint angles are small. Finally, singularity in Eq. (3.56) occurs only if the lower $2 \times 2$ part of matrix ${}^k\mathbf{J}_{k+1}$ is singular (i.e., $\cos\gamma_k = 0$), which again is not possible since $|\gamma_k| \ll \pi/2$.

## 3.5 Summary

The modal approach was used to resolve the redundancy of spatial hyper-redundant manipulators. New shape functions were introduced to achieve a more complete workspace. The CLSFM and RFM were presented for planar and spatial manipulators, respectively. The CLSFM was not efficient for spatial manipulators due to its highly complex formulation and a high computational burden. The computational cost of the CLSFM is high because this method requires the solution to a system of nonlinear simultaneous equations. The RFM, on the other hand, was much simpler to formulate and required a rather low computational effort. The RFM needed to solve a single nonlinear algebraic equation per link thus avoiding systems of nonlinear simultaneous equations. Finally, the velocity property of the backbone curve was investigated and an inverse velocity propagation scheme was introduced. The fitting method was shown to guarantee the existence of the inverse kinematics solution at velocity level. This scheme is recursive and free from singularity as long as the position problem has a solution, and can be easily applied to spatial universal-jointed arms with an arbitrary number of links.

## Problems

**Problem 3.1.** Consider a planar hyper-redundant manipulator lying the $x_1$-$x_2$ plane.

(a) Write an appropriate backbone curve parameterized equation in term of the curve length parameter $s$ for the manipulator (see Eq. 3.1).
(b) Define appropriate mode shape functions for the manipulator (see Eq. 3.3).
(c) Determine the modal Jacobian for the planar manipulator (see Eq. 3.7).

**Problem 3.2.** Consider a spatial backbone curve for a hyper-redundant manipulator with the mode shape functions defined as in Eqs. (3.4) and (3.5). If the desired azimuth and meridian angles at the start and the end point of the backbone curve are

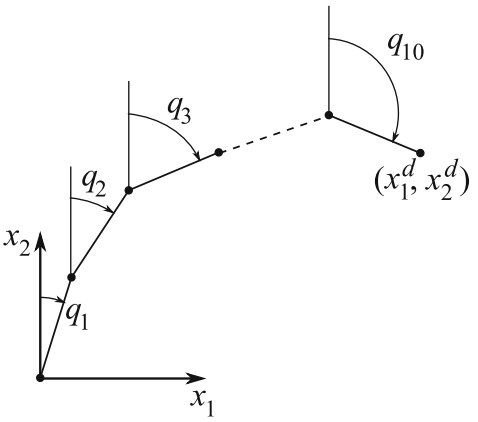


**Fig. 3.9** A ten-link planar hyper-redundant manipulator

$$M_s = 10^\circ \quad A_s = 35^\circ,$$
$$M_e = 25^\circ \quad A_e = 15^\circ,$$

find the slope parameters for the backbone curve at its start $(b_{1\phi}, b_{1\psi})$ and end $(b_{2\phi}, b_{2\psi})$ points.

**Problem 3.3.** Consider a planar 10-link manipulator with a total length of 2 m and the joint parameters shown in Fig. 3.9. If the base of the manipulator is at $[x_1, x_2]^T = [0, 0]^T$, compute the joint angles of the manipulator such that its end-effector makes a 105° angle with the positive $x_2$ axis and its end-effector positions at $[x_1^d, x_2^d]^T = [0.8, 1.0]^T$ m. Assume that the first link is approximately aligned with the $x_2$ axis. Use a 2D backbone curve and the CLSFM.

*Hint*: Note that the derivations in Section 3.3.1 assumes a manipulator with a unit length, which do not apply to a manipulator with a different length. Scale down the manipulator of this problem into a manipulator with 1-m length and recalculate the desired end-effector position accordingly for joint angle calculations instead of reformulating the equations derived in Section 3.3.1!

**Problem 3.4.** Consider a 16-link planar hyper-redundant manipulator ($n = 16$) with link lengths equal to 0.1 m. Assume the end-effector tip position as $\mathbf{x}_e = \begin{bmatrix}0.4 & 0.3\end{bmatrix}$ m. The end-effector is oriented such that it is parallel to the $x_2$ axis. Compute the position of the joints on the backbone curve.

**Problem 3.5.** Consider the 11-link spatial manipulator of Example 3.4. Use the positions of the joints six and seven obtained in Example 3.4 and listed in Table 3.2 and compute the angles of the universal joint six, $\theta_6$ and $\gamma_6$.

**Problem 3.6.** Consider the 10-link planar manipulator introduced in Problem 3.3. Assume that at the desired posture of $[x_1^d, x_2^d]^T = [0.8, 1.0]^T$ m, the end-effector

velocity is $[\dot{x}_1^d, \dot{x}_2^d]^T = [0.0, 1.0]^T$ m/s. The angular velocity of the end-effector link is $\pi/12$ rad/s.

(a) Derive the orientation Jacobian matrices introduced in Eq. (3.47).
(b) Calculate the rate of the mode participation vector $\mathbf{a}$.
(c) Compute the velocity of the backbone curve at its midpoint.
(d) Determine the linear velocity of joint six and the angular velocity of link six of the manipulator, if the linear velocity of joint seven is $\mathbf{v}_7 = [0.0875, 0.1013]^T$ m/s.

# Chapter 4
# Obstacle Avoidance Using Harmonic Potential Functions

## 4.1 Introduction

Path planning for single mobile robots has been the essential first step for realizing autonomous ground vehicles. There are numerous methods and algorithms initially developed for single mobile robots working in environments containing static obstacles. One can name the roadmap, cell decomposition, and potential field methods [10, 84, 50] among others.

The roadmap and cell decomposition methods rely on rules that are derived using the geometry of the obstacle field. Many problems such as motion planning for a number of circular [66] or rectangular [40] objects bounded by walls and motion planning for multiple robots [51, 38, 53] have been solved using the geometrical methods. These methods have even been extended to the case moving obstacles [8].

Different control theories have also been used for path planning for groups of mobile robots. The dynamic motion planning problem has been transformed into a conventional optimal control problem [43]. In another work, a decoupled controller, consisting of a force controller and a torque controller, has been introduced to address the presence of multiple obstacles in a robot's workspace [55]. A centralized control approach for keeping a group of robots in desired formation in presence of obstacles has also been used as path planning method [24, 39]. Several different control laws have been developed for decentralized control including hybrid control algorithms [52, 80, 72]. Behavior-based approach has also been introduced to simplify the definition of control laws in the decentralized control approach [6, 11].

Different path planning approaches have been integrated in a hierarchical manner to address obstacle avoidance for mobile robots in cluttered environments. For example, a three-layer hierarchical path planning system has been presented that consists of global planning, local navigator, and collision avoidance algorithms [37]. Also, a two-layered hierarchical path planning for multiple robots has been introduced, where the second layer mimics the behavior of a mass-spring-damper system to modify a globally planned path of a robot when it is close to other robots or stationary obstacles [3].

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_4

Another approach that has been extensively used for obstacle avoidance for single mobile robots, multiple mobile robots, and moving obstacles is the potential field approach. In the potential field method, an artificial potential field is assigned to the area where a robot works. The obstacles in the area are assigned a repulsive potential while the goal position of the robot is described by an attractive potential. Then, the path of the robots is calculated by using the gradient of the total artificial potential. Different mathematical definitions have been used for defining the artificial potentail fields and different strategies have been introduced for using the gradient of the total potential to find a path for the robot.

To extend the application of the potential field approach to the case of multiple robots, where the timing of the robots motions is important for collision avoidance, artificial potential fields in the robots' extended configuration space-time have been introduced [61]. Priority assignment to the robots of a group has been able to enhance path planning for multiple mobile robots in configuration space-time [78]. Another method for addressing multiple mobile robots using the potential field approach is using potentials that are functions of the relative speeds of the robots as well as their distances [33].

The possibility of the existence of local minima in the artificial potential field could be one of the drawbacks of the potential field methods. A local minimum can attract and trap the robot, preventing it from reaching its final goal. Search methods have been introduced to address this problem at a high computational cost [73, 82]. Another method for avoiding the generation of local miming is adding multiple auxiliary attraction potentials, whose positions are determined by a genetic algorithm [21]. Also, a set of analytical guidelines have been given for designing potential functions to avoid local minima for a number of representative scenarios [45].

It has been shown that harmonic potential functions do not suffer from local minima [12, 46] and lead to unique solutions. This property of harmonic potential functions allows the potentials to be defined in Euclidean space rather than the configuration space [20]. The application of harmonic potentials to the case of moving obstacles has also been reported [1, 75]. Harmonic potential fields have been utilized for single mobile robots and planar manipulators with low degrees of redundancy in known environments containing stationary obstacles by employing the panel method known in fluid mechanics [49, 31]. The panel method is quite suitable for real-time applications. The panel method has also been extended to the case of unknown environments [85] and 3D environments [86].

In this chapter, harmonic potential functions are introduced and the panel method is discussed. A complement to the traditional panel method [46] is presented to generate a more effective harmonic potential field for obstacle avoidance. This makes the traditional panel method suitable for dynamically changing environments. The application of the panel method to the case of a single mobile robot working in an environment with static obstacles is shown. The extension of the panel method to the case of multiple robots and moving obstacles is discussed. Finally, the 3D version of the panel method, which is applicable to path planning for aerial robots, is presented.

## 4.2 Potential Theory and Harmonic Functions

Potential theory is used in describing many conservative systems such as irrotational fluid flows. In the absence of viscous effects and rotational force, the originally irrotational flow will remain so in the region around a body inside the flow field. Let us denote the vectorial velocity field in this region by $\mathbf{V}$. Vorticity vanishes when the flow is irrotational. That is,

$$\text{curl } \mathbf{V} = \nabla \times \mathbf{V} = \mathbf{0}, \tag{4.1}$$

where

$$\nabla = \frac{\partial}{\partial x_1}\hat{\mathbf{i}} + \frac{\partial}{\partial x_2}\hat{\mathbf{j}} + \frac{\partial}{\partial x_3}\hat{\mathbf{k}}. \tag{4.2}$$

This equation implies that the fluid velocity field can be written as

$$\mathbf{V} = -\nabla\phi, \tag{4.3}$$

where $\phi$ is a scalar velocity potential. Furthermore, when the fluid is incompressible, the velocity field must satisfy the continuity equation.

$$\mathbf{V}.\mathbf{V} = 0. \tag{4.4}$$

Substituting Eq. (4.3) into Eq. (4.4) results in

$$\nabla^2\phi = 0, \tag{4.5}$$

where $\nabla^2 = \nabla.\nabla$ is the Laplacian operator. Equation (4.5) is called the Laplace or potential equation and its solutions are called harmonic or potential functions. In the real world, many physical problems are described by the Laplace equation. An example is the incompressible fluid irrotational flow mentioned above.

### *4.2.1 Properties of Harmonic functions*

The properties of harmonic functions related to local minimum are described in the following.

1. *Superposition Property*. This property of the potential functions are related to the linearity of the Laplace equation. If $\phi_1$ and $\phi_2$ are harmonic functions (they satisfy the Laplace equation), then any linear combination of $\phi_1$ and $\phi_2$ is also a harmonic function and a solution of Laplace equation.
2. *Mean-value Property*. A 2D potential function $\phi(x_1, x_2)$ that is harmonic in a circle with center at $(x_{o_1}, x_{o_2})$, there exists the mean-value of property of $\phi$

$$\phi(x_{o_1}, x_{o_2}) = \frac{1}{2\pi} \oint \phi(x_{o_1} + r\cos\theta, x_{o_2} + r\sin\theta) d\theta \tag{4.6}$$

In words, the value of the potential at the center of any arbitrary circle is equal to the average of the potential integrated over the circumference of the circle. This property is independent of the radius $r$ of the circle only if the function is harmonic inside the circle. A similar result holds for an arbitrary number of dimensions. For example, in three dimensions, the potential at $(x_{o_1}, x_{o_2}, x_{o_3})$ can be obtained by integration over the whole surface of a sphere $S$.

$$\phi(x_{o_1}, x_{o_2}, x_{o_3}) = \frac{1}{4\pi} \int \int \phi(S) dS \tag{4.7}$$

The converse of this Property is also true. If $\phi(x_1, x_2)$ is continuous and has the mean-value property for every circle in a domain, then $\phi$ is harmonic. This property can be used to prove the maximum and minimum principle of harmonic functions.

3. *The Maximum Potential Property*. The maximum of a nonconstant harmonic function occurs on singular boundaries where the potential tends to infinity.
4. *The Minimum Potential Property*. The minimum of a nonconstant harmonic function also occurs on singular boundaries where the potential tends to infinity. The above properties of a harmonic function are very useful in building an artificial potential field for obstacle avoidance problem. Because the harmonic function completely eliminates local minima, which eliminates the possibility of generating a stationary point in the velocity field except the goal point.

## 4.3 Two-Dimensional Harmonic Potential Functions

In this section, examples of harmonic functions that will be used to build an artificial potential are introduced. First, harmonic functions with spherical symmetry are introduced. These functions should be expressed in an spherical coordinate system. For a spherical symmetry, the potential function must not depend on angular terms. It must only depend on $r$ (distance from the origin), e.g., $\phi = \phi(r)$. The general $n$-dimensional expression of the Laplace equation in spherical coordinate system can be written as

$$\nabla^2 \phi = \phi_{rr} + \frac{n-1}{r} \phi_r + \text{angular terms}, \tag{4.8}$$

where $\phi_r$ is the first partial derivative of $\phi$ with respect to $r$ and $\phi_{rr}$ is the second partial derivative of $\phi$ with respect to $r$. Since $\phi = \phi(r)$, the angular terms in Eq. (4.8) vanish and the Laplace equation becomes

$$\phi_{rr} + \frac{n-1}{r} \phi_r = 0, \tag{4.9}$$

or

$$\frac{\phi_{rr}}{\phi_r} + \frac{n-1}{r} = 0. \tag{4.10}$$

Integration of Eq. (4.10) once results

$$\phi_r = \frac{C_1}{r^{n-1}}. \tag{4.11}$$

For $n = 2$, the integration of Eq. (4.11) results in

$$\phi = C_1 \ln r + C_2. \tag{4.12}$$

For $n > 2$, the solution to Eq. (4.11) for $\phi$ becomes

$$\phi = \frac{C_3}{r^{n-2}} + C_4. \tag{4.13}$$

In Eqs. (4.12) and (4.13), $C_i$'s are constant. From Eqs. (4.12) and (4.13), it is observed that every harmonic function with spherical symmetry has its singularity at origin (e.g., at $r = 0$) and is not harmonic at this singular point. According to the Property 3 and 4 for harmonic functions, the maximum or minimum of a potential function with spherical symmetry occurs at the origin (the singularity point). Since the origin can be placed anywhere (the Laplace equation is invariant under translation), one can always choose the location of the origin outside the free space for a manipulator or a mobile robot. That is, by locating the origins of the harmonic functions on the surface of the obstacles or inside obstacles. We can build an artificial potential field with no local minimum and only one global minimum in free space.

### *4.3.1 Potential of a Point Source or a Point Sink*

In hydrodynamics, a harmonic function with spherical symmetry, for Example (4.12) or (4.13), is called a source or a sink, depending on the sign of $C_1$ in Eq. (4.12) and $C_3$ in (4.13). A sink is similar to a drain in a bathtub and a source is like a faucet. In a 2D space, a source/sink at the origin can be represented by

$$\phi = \frac{\lambda}{2\pi} \ln(r). \tag{4.14}$$

The magnitude of $\lambda$ is the strength of the source ($\lambda < 0$) or sink ($\lambda > 0$). Since both a source and a sink are singular at the origin, they are called singularity potentials.

*Example 4.1.* Using Eq. (4.14), plot the potential of a point sink located at the origin with a strength of $\lambda = 1$ in a square area of 2 by 2 m around the origin.

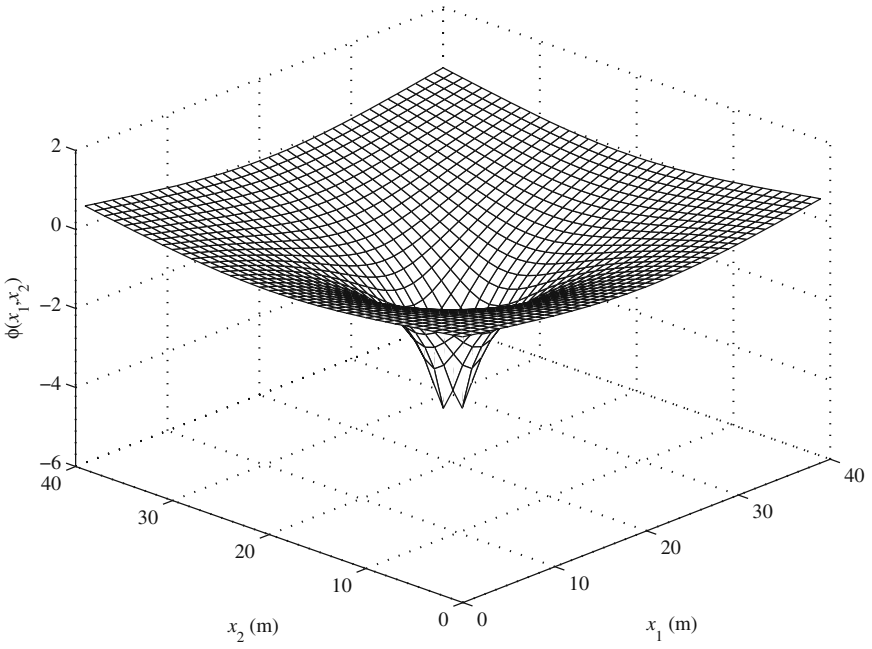


**Fig. 4.1** Potential of a point sink

*Solution.* For a point source at the origin, the distance variable $r$ in terms of the Cartesian coordinate components $(x_1, x_2)$ become

$$r = \sqrt{x_1^2 + x_2^2}. \tag{4.15}$$

The potential is plotted in Fig. 4.1. Note that at the origin the potential is infinite. One can imagine the direction of the motion of a fluid particle (or a mobile robot) caused by this potential field by assuming a bead that is released from a point on this surface. If one imagines a bead on the surface shown in Fig. 4.1, they can see that the bead (or a mobile robot) will roll toward the point sink (representative of the goal position). The other important observation that can be made using Fig. 4.1 is that the singularity point is the global maximum of the potential field. This is a property of a harmonic potential function. There is no local minimum. Therefore, a fluid particle (or a mobile robot) moves toward the sink (or the goal position) and will never be trapped in a local minimum.

### *4.3.2 Potential of a Uniform Flow*

Another harmonic function useful for building artificial potentials is the uniform flow potential, which varies linearly along the direction of the uniform flow it represents. In a 2D space, when the fluid flows in a direction that makes an angle $\alpha$ with the $x_1$-axis, the potential function for this uniform flow is

$$\phi = -U(x_1 \cos\alpha + x_2 \sin\alpha). \tag{4.16}$$

The magnitude of the coefficient $U$ is called the strength of uniform flow. The source/sink singularities in Eq. (4.14) is used to derive the repulsive force (high potential) of the obstacles and the attractive force (low potential) of goal position. The uniform flow in (4.16) is used to derive a more effective artificial potential field from a starting point to the goal position. This uniform flow provides a linearly decreasing potential in the direction from a starting point to a goal position for the unbounded environment.

*Example 4.2.* Using Eq. (4.16) plot the potential of a uniform flow with a direction of 45° with respect to the $x_1$-axis and with a strength of $U = -1$ in a square area of 2 by 2 m around the origin.

*Solution.* For a uniform flow along a directions that makes a 45° angle with the $x_1$-axis, the direction variable $\alpha$ becomes

$$\alpha = \frac{\pi}{4} \tag{4.17}$$

The potential is plotted in Fig. 4.2. If one imagines a bead on the surface shown in Fig. 4.2, they can see that the bead (or a mobile robot) will roll along a 45° line toward the third Cartesian quadrant (because $U$ is negative). The other important observation that can be made using Fig. 4.2 is that there is no singularity. Therefore, there is no global optimum for the potential field and the potential is decreasing

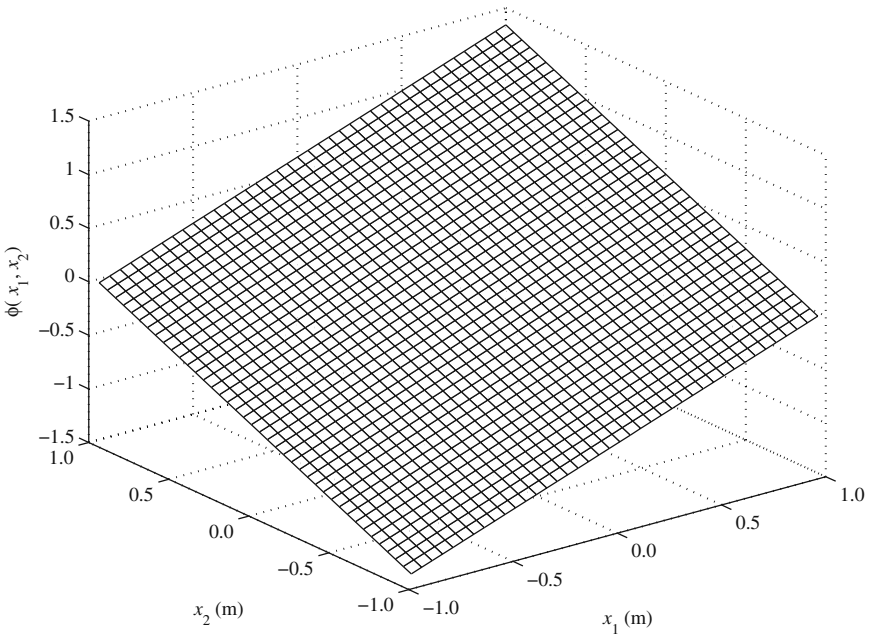


**Fig. 4.2** Potential of a uniform flow

monotonically in one direction. This is a property of a harmonic potential function. There is no local minimum. Therefore, a fluid particle (or a mobile robot) moves indefinitely and will never be trapped in a local minimum.

### 4.3.3 Potential of a Line Segment (a Panel)

Since the boundary of the obstacles in 2D spaces will be approximated by line segments and define repulsive potential for them, the potential of a line segment, also known as a panel in fluid mechanics, must be defined. The single panel in Fig. 4.3 is distributed with uniform sources, with strength per unit length $\lambda$. The potential at any point $(x_1, x_2)$ induced by the sources contained within the small element $dl$ of the panel at $(0, l)$ is

$$d\phi = \frac{\lambda dl}{2\pi} \ln r = \frac{-\lambda}{2\pi} \ln \sqrt{x_1^2 + (x_2 - l)^2} dl. \tag{4.18}$$

The induced potential function by the whole panel is

$$\phi(x_1, x_2) = \frac{\lambda}{4\pi} \int_{-L}^{L} \ln(x_1^2 + (x_2 - l)^2) dl. \tag{4.19}$$

*Example 4.3.* Using Eq. (4.19), plot the potential of a 1-m-long source line segment ($L = 0.5\,\text{m}$) as shown in Fig. 4.3 with a strength per unit length of $\lambda = -1$ in a square area of 2 by 2 m around the origin.

*Solution.* To calculate the potential of a source panel, Eq. (4.19) must be integrated numerically for different values of $x_1$ and $x_2$. The potential is plotted in Fig. 4.4. If one imagines a bead on the surface shown in Fig. 4.4, they can see that the bead

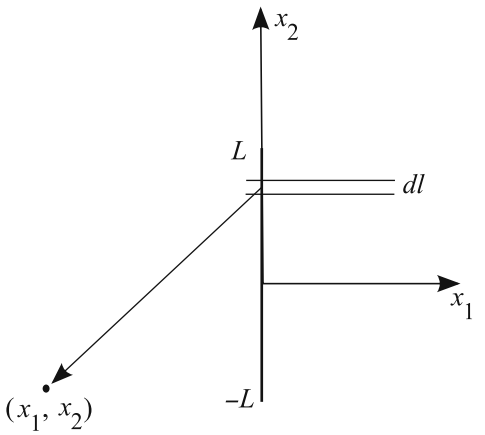


**Fig. 4.3** A single line source (panel)

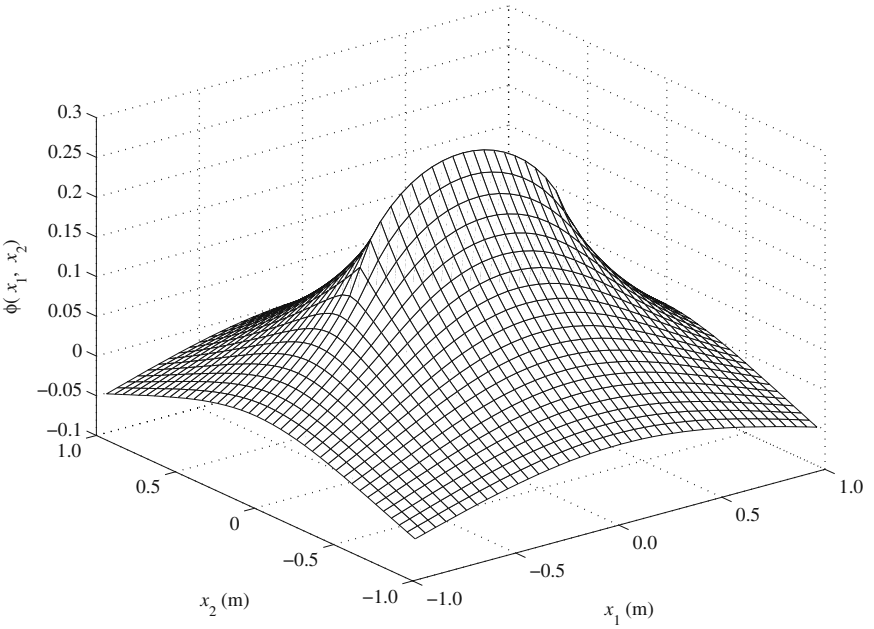


**Fig. 4.4** Potential of a source line segment (or source panel)

(or a mobile robot) will roll away from the source panel (representative of a line obstacle). The other important observation that can be made using Fig. 4.4 is that the global maximum occurs on the panel itself (since a singularity is distributed along the panel). This is a property of a harmonic potential function. There is no local minimum. Therefore, a fluid particle (or a mobile robot) moves away from the obstacle indefinitely and will never be trapped in a local minimum.

The velocity field generated by a line panel can be found by partial differentiation of the potential field function. Differentiation with respect to $x_1$ and $x_2$ gives the expressions for the velocity components $u_1$ and $u_2$ corresponding to the Cartesian coordinates $x_1$ and $x_2$, respectively.

$$u_1(x_1, x_2) = -\frac{\partial \phi}{\partial x_1} = \frac{-\lambda}{2\pi} \int_{-L}^{L} \frac{x_1}{x_1^2 + (x_2 - l)^2} dl, \tag{4.20}$$

$$u_2(x_1, x_2) = -\frac{\partial \phi}{\partial x_2} = \frac{-\lambda}{2\pi} \int_{-L}^{L} \frac{x_2 - l}{x_1^2 + (x_2 - l)^2} dl, \tag{4.21}$$

which result in

$$u_1(x_1, x_2) = \frac{-\lambda}{2\pi} \left( \arctan \frac{x_2 + L}{x_1} - \arctan \frac{x_2 - L}{x_1} \right), \tag{4.22}$$

$$u_2(x_1, x_2) = \frac{-\lambda}{4\pi} \ln\left(\frac{x_1^2 + (x_2 + L)^2}{x_1^2 + (x_2 - L)^2}\right). \tag{4.23}$$

The limiting value of normal velocity $u_1(x_1, x_2)$ in Eq. (4.22) is $u_1(0^-, x_2) = -\lambda/2$ on the left face of the panel. This limit is $u_1(0^+, x_2) = \lambda/2$ on the right face of the panel. This shows that a source panel with a strength per unit length $\lambda$ creates a uniform normal outward velocity of magnitude $\lambda/2$ at the surface. The tangential velocity starts at zero at the center of the panel and increases along the panel to the edges, where the normal velocity is not defined and the tangential velocity becomes infinite. The single panel has a singular point at each edge.

### *4.3.4 Superposition of Potentials*

Any number of potential fields can be added to form a complicated potential field. Since the summation of any number of harmonic potential fields still satisfy the Laplace equation, the summation itself is a harmonic potential function.

Here, two potential functions are added as an example. A uniform flow of strength $U$ that flows in the direction of positive $x_1$-axis ($\alpha = 0$) is considered as

$$\phi(x_1, x_2) = -Ux_1. \tag{4.24}$$

The velocity components of this uniform flow are

$$u_1(x_1, x_2) = U, \tag{4.25}$$
$$u_2(x_1, x_2) = 0. \tag{4.26}$$

Note that these velocity components correspond to a flow in the positive $x_1$ direction. A simple superposition of the two harmonic functions in (4.19) and (4.24) results in the uniform flow deflected by the source panel. The total $x_1$ direction velocity component of the source panel and the uniform flow at the left face of panel $-L < x_2 < L$ is

$$u_1(0^-, x_2) = U + \frac{\lambda}{2}. \tag{4.27}$$

This normal velocity on the left face of the panel is important, because the uniform flow, which flows to the right, is deflected by the left side of the panel. If $U$ equals $\lambda/2$, then $u_1(0^-, x_2)$ becomes zero for $-L < x_2 < L$. That is, the induced normal velocity from the source panel exactly cancels the velocity of the uniform flow on the left face. Therefore, the resulting flow becomes tangential to the surface. Both normal and tangential velocities at the origin $(0^-, 0)$ are zero. This is a stagnation point that the velocity of a fluid particle becomes zero instantaneously and changes its direction to $-x_2$ or $+x_2$ direction.

In hydrodynamics, $u_1(0^-, x_2)$ is set to zero to satisfy the requirement that the oncoming flow must be tangent to the panel, which represents the boundary of a

solid. However, for our problem of obstacle avoidance, this requirement must be modified. That is, the normal velocity must be greater than or equal to zero. This requirement can be represented by

$$V_n = -u_1(0^-, x_2) \geq 0 \tag{4.28}$$

Three examples of different $V_n$'s are shown in Figs. 4.5, 4.6, and 4.7. Here $U = 1$, $L = 1$, and $\lambda$ is determined by the given $V_n$:

$$\lambda = -2(U + V_n) \tag{4.29}$$

Figures 4.5, 4.6, and 4.7 correspond to $V_n = 0$, $V_n = 0.5$, and $V_n = 2.0$, respectively. The corresponding strengths are $\lambda = -2$, $-3$, and $-6$ from Eq. (4.29). These figures show the trajectories of some fluid particles starting at different positions. Again, note that the motion of a fluid particle can be thought of as the navigation of a point mobile robot that avoids a line obstacle. We can observe that the fluid particle (or mobile robot) moves further away from the panel as the strength of the panel increases. This is a matter of economy or safety. When we increase the strength, the path is longer, however, the trajectory is safer.

Note that Fig. 4.5 represents the solution used for actual fluid flow simulations, in which $V_n = 0$ indicates that the velocity of any fluid particle that touches the panel (as the boundary of a solid body) is zero. However, since $V_n = 0$ allows a fluid particle (a mobile robot) to touch the panel, it is not safe for use for obstacle avoidance. The two main differences between hydrodynamics and our obstacle avoidance problem are as follows.

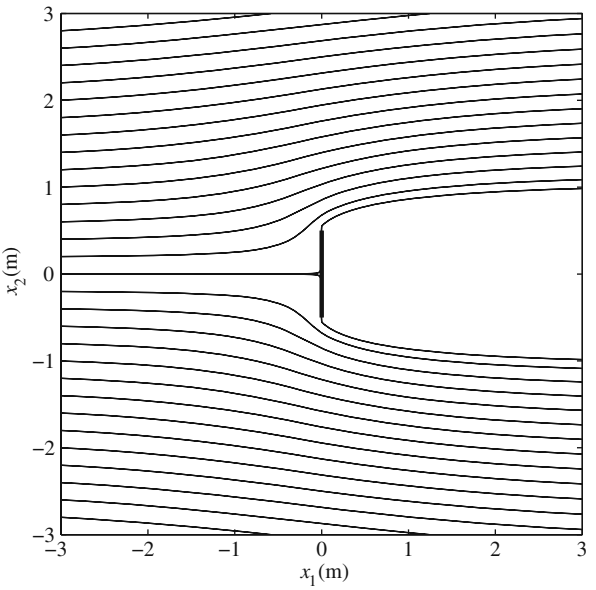


**Fig. 4.5** Single panel with strength, $\lambda = -2$, for $V_n = 0.0$

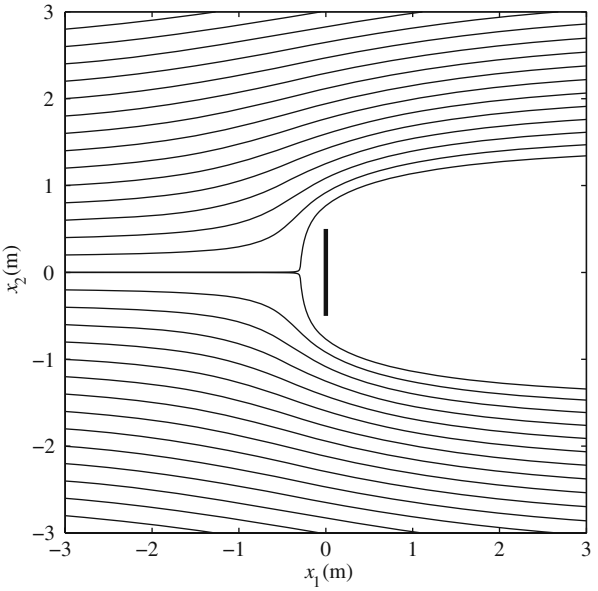


**Fig. 4.6** Single panel with strength, $\lambda = -3$, for $V_n = 0.5$

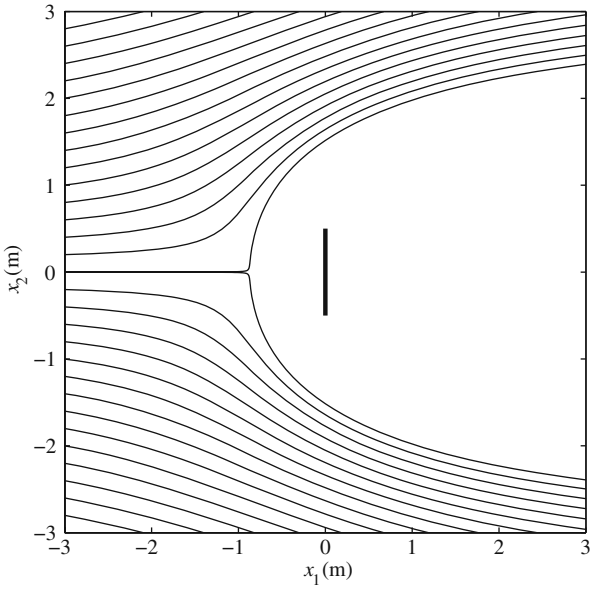


**Fig. 4.7** Single panel with strength, $\lambda = -6$, for $V_n = 2.0$

1. For obstacle avoidance, the normal velocity on the left face of a panel is recommended to be greater than zero to avoid a path very close to the obstacle.
2. The obstacle avoidance problem has a goal point that robot must reach. Thus, the potential for the obstacle avoidance will be composed of a uniform flow, distributed singularities on the panels (obstacle boundaries), and a sink (a goal singularity). In the next section, we consider the use of multiple panels to represent complex obstacles.

Until now, it was shown how a particle moving in a potential flow is analogous to a mobile platform avoiding obstacles. Then, the equations of potential flow for a single line-shaped obstacle were derived. In the next section, the potential field method is extended to the case of arbitrary shaped obstacles. The arbitrary-shaped obstacles are represented by polygons. The polygons are modeled using multiple line obstacles.

### *4.3.5 Multiple Line Obstacles*

Figure 4.8 illustrates the use of a set of source/sink line obstacles for representing an arbitrarily shaped obstacle in two dimensions. The obstacle is approximated by a set of line obstacles, which are numbered clockwise. Figure 4.9 shows the details of the line obstacle geometry. The desired outward normal velocity generated by each line obstacle at its center point is considered as an input variable. The boundary points are the intersections of neighboring line obstacles. The angle between line obstacle $i$ and the $x_1$-axis is denoted by $\theta_i$, and the angle between the outward normal unit

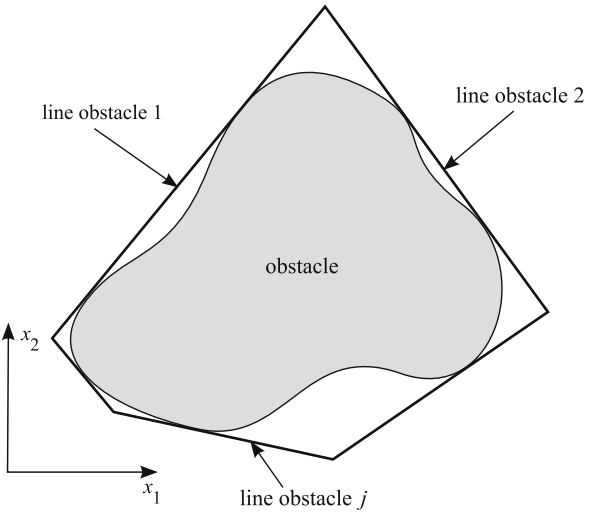


**Fig. 4.8** Approximating the geometry of an obstacle by a number of line obstacles

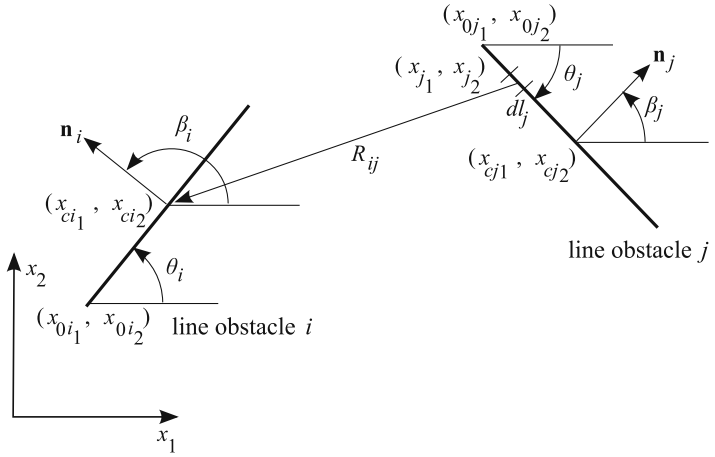


**Fig. 4.9** Two line obstacles $i$ and $j$. The effect of the source/sink singularity distributed along the length of line obstacle $j$ is integrated at the center point of line obstacle $i$

vector $\hat{\mathbf{n}}$ of line obstacle $i$ and the $x_1$-axis is denoted by $\beta_i$. One can see that $\beta_i = \theta_i + \pi/2$.

$m$ is the total number of line obstacles, whose lengths are usually not equal. Sources/sinks of uniform density are distributed on each of the $m$ line obstacles and $\lambda_1$ to $\lambda_m$ represent source/sink strengths per unit length on these line obstacles. Line obstacle $j$ produces the following velocity potential at any point $(x_1, x_2)$ in the 2D space

$$\phi_j = \frac{\lambda_j}{2\pi} \int_j \ln R_j dl_j, \tag{4.30}$$

where

$$R_j = \sqrt{(x_1 - x_{j_1})^2 + (x_2 - x_{j_2})^2}. \tag{4.31}$$

Since it is assumed that the robot should reach a goal, an attractive potential is needed at this goal, where the potential has only one global minimum. This attractive goal can be represented by a singular point sink. This sink works similar to a point drain in a bathtub. It is assumed that the goal sink have a strength of $\lambda_j > 0$. Then its potential is

$$\phi_g = \frac{\lambda_g}{2\pi} \ln R_g, \tag{4.32}$$

where

$$R_g = \sqrt{(x_1 - x_{g_1})^2 + (x_2 - x_{g_2})^2} \tag{4.33}$$

is the distance between the point $(x_1, x_2)$ and the goal point $(x_{g_1}, x_{g_2})$. The potential of uniform flow, which tends to push the robot to the goal, is rewritten as

$$\phi_u = -U(x_1 \cos\alpha + x_2 \sin\alpha), \tag{4.34}$$

where $\alpha$ is the angle between the $x_1$-axis and direction of the uniform flow. The obstacles, goal, and uniform flow generate a total potential as follows

$$\phi(x_1, x_2) = \phi_u + \phi_g + \sum_{j=1}^{m} \phi_j. \tag{4.35}$$

If the strength of the uniform flow $U$ and the strength of the goal sink $\lambda_g$ are specified, our objective is to derive the strengths of the $m$ line obstacles. At least $m$ independent equations are needed to solve this problem. If the outward normal velocities generated at the center of the $m$ line obstacles are specified, these $m$ equations can be derived. In Eq. (4.29), the relationship between the normal velocity on a line obstacle and the line obstacle strength for a single line obstacle was derived. A similar expression for the general case can be derived with given desired outward normal velocity on each line obstacle. A $V_i > 0$ is assumed for the desired outward normal velocity at the center point $(x_{c_1}, x_{c_2})$ of line obstacle $i$. Note that this outward normal velocity is satisfied only at the center point of each line obstacle. The resulting $m$ equations are

$$\frac{\partial}{\partial n_i}\phi(x_1, x_2) = -V_i \quad i = 1, \ldots, m \tag{4.36}$$

These are $m$ linearly independent equations with $m$ unknowns $\lambda_1$ to $\lambda_m$. The potential at the center point $(x_{c_1}, x_{c_2})$ is as follows

$$\phi(x_{ci_1}, x_{ci_2}) = -U(x_{ci_1}\cos\alpha + x_{ci_2}\sin\alpha) + \frac{\lambda_g}{2\pi}\ln R_{gi} + \sum_{j=1}^{m}\frac{\lambda_j}{2\pi}\int_j \ln R_{ij}\, dl_j, \tag{4.37}$$

where $R_{ij}$ is the distance between the goal and the center point of line obstacle $i$, $(x_{ci_1}, x_{ci_2})$, and $R_{ij}$ is the distance between $(x_{ci_1}, x_{ci_2})$ and a point on line obstacle $j$ as shown in Fig. 4.9.

Equation (4.37) is substituted into Eq. (4.36). Note that the contribution to the normal velocity on line obstacle $i$ by itself is $\lambda_i/2$ (as shown with a single line obstacle in the previous section). Therefore, Eq. (4.36) becomes

$$\frac{\lambda_i}{2} + \sum_{j\neq i}^{m}\frac{\lambda_j}{2\pi} I_{ij} = -V_i + U\frac{\partial}{\partial n_i}(x_{ci_1}\cos\alpha + x_{ci_2}\sin\alpha) - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial n_i}\ln R_{gi}, \quad i = 1, \ldots, m, \tag{4.38}$$

where

$$I_{ij} = \int_j \frac{\partial}{\partial n_i} \ln R_{ij} dl_j, \tag{4.39}$$

and

$$R_{ij} = \sqrt{(x_{j_1} - x_{ci_1})^2 + (x_{j_2} - x_{ci_2})^2}. \tag{4.40}$$

Using the geometric relations

$$x_{j_1} = x_{0j_1} + l_j \cos\theta_j, \tag{4.41}$$
$$x_{j_2} = x_{0j_2} + l_j \sin\theta_j, \tag{4.42}$$

where $\theta_j$ is the angle between line obstacle $j$ and $x_1$-axis, and is positive in counter-clockwise direction, $I_{ij}$ can be integrated as

$$I_{ij} = \frac{1}{2} C \ln\left(1 + \frac{L_j^2 + 2L_j A}{B}\right) - \left(\arctan\frac{L_j + A}{E} - \arctan\frac{A}{E}\right)\cos(\theta_i - \theta_j) \tag{4.43}$$

for $E \neq 0$, and

$$I_{ij} = C\left(\ln\left|\frac{L_j + A}{B}\right| - \frac{L_j}{L_j + A}\right) + D\left(\frac{1}{A} - \frac{1}{L_j + A}\right) \tag{4.44}$$

for $E = 0$, where

$$A = -(x_{ci_1} - x_{0j_1})\cos\theta_j - (x_{ci_2} - x_{0j_2})\sin\theta_j, \tag{4.45}$$
$$B = (x_{ci_1} - x_{0j_1})^2 + (x_{ci_2} - x_{0j_2})^2, \tag{4.46}$$
$$C = \sin(\theta_i - \theta_j), \tag{4.47}$$
$$D = -(x_{ci_1} - x_{0j_1})\sin\theta_i - (x_{ci_2} - x_{0j_2})\cos\theta_i, \tag{4.48}$$
$$E = (x_{ci_1} - x_{0j_1})\sin\theta_j - (x_{ci_2} - x_{0j_2})\cos\theta_j. \tag{4.49}$$

The parameter $E$ becomes zero when the center point of line obstacle $i$ is on the extension of line obstacle $j$. Other terms of Eq. (4.38) are as follows

$$U\frac{\partial}{\partial n_i}(x_{ci_1}\cos\alpha + x_{ci_2}\sin\alpha) = U\sin(\alpha - \theta_i), \tag{4.50}$$

$$\begin{aligned}\frac{\lambda_g}{2\pi}\frac{\partial}{\partial n_i}R_{gi} &= \frac{\lambda_g}{4\pi}\frac{\partial}{\partial n_i}\ln((x_{ci_1} - x_{g_1})^2 + (x_{ci_2} - x_{g_2})^2) \\ &= \frac{\lambda_g}{2\pi}\frac{-(x_{ci_1} - x_{g_1})\sin\theta_i + (x_{ci_2} - x_{g_2})\cos\theta_i}{(x_{ci_1} - x_{g_1})^2 + (x_{ci_2} - x_{g_2})^2}.\end{aligned} \tag{4.51}$$

Now, Eq. (4.38) can be written as

$$\mathbf{P}\Lambda = \mathbf{q}, \tag{4.52}$$

where

$$P_{ij} = \begin{cases} 1/2, & \text{if } i = j \\ I_{ij}/2\pi, & \text{if } i \neq j \end{cases}, \tag{4.53}$$

$$q_i = -V_i + U\frac{\partial}{\partial n_i}(x_{ci_1}\cos\alpha + x_{ci_2}\sin\alpha) - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial n_i}\ln R_{gi}, \tag{4.54}$$

and

$$\Lambda = [\lambda_1, \ldots, \lambda_m]^T. \tag{4.55}$$

Now, Eq. (4.52) is solved to determine the strengths per unit length of the line obstacles. Once the strengths are obtained, the following velocity equations are used to derive a trajectory for mobile robot

$$u_1(x_1, x_2) = -\frac{\partial\phi}{\partial x_1} = U\cos\alpha - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial x_1}\ln R_{gi} - \sum_{j=1}^{m}\frac{\lambda_j}{2\pi}\int_j \frac{\partial}{\partial x_1}\ln R_{ij}dl_j, \tag{4.56}$$

$$u_2(x_1, x_2) = -\frac{\partial\phi}{\partial x_2} = U\sin\alpha - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial x_2}\ln R_{gi} - \sum_{j=1}^{m}\frac{\lambda_j}{2\pi}, \int_j \frac{\partial}{\partial x_2}\ln R_{ij}dl_j \tag{4.57}$$

where

$$\frac{\partial}{\partial x_1}\ln R_{gi} = \frac{x_1 - x_{g_1}}{(x_1 - x_{g_1})^2 + (x_2 - x_{g_2})^2}, \tag{4.58}$$

$$\frac{\partial}{\partial x_2}\ln R_{gi} = \frac{x_2 - x_{g_2}}{(x_1 - x_{g_1})^2 + (x_2 - x_{g_2})^2}, \tag{4.59}$$

$$\begin{aligned}\int_j \frac{\partial}{\partial x_1}\ln R_{ij}dl_j = &-\frac{1}{2}\ln(1 + \frac{L_j^2 + 2L_jA_1}{B_1})\cos\theta_j \\ &+ (\arctan\frac{L_j + A_1}{E_1} - \arctan\frac{A_1}{E_1})\sin\theta_j,\end{aligned} \tag{4.60}$$

$$\int_j \frac{\partial}{\partial x_2} \ln R_{ij} dl_j = -\frac{1}{2} \ln(1 + \frac{L_j^2 + 2L_j A_1}{B_1}) \sin\theta_j - (\arctan \frac{L_j + A_1}{E_1} - \arctan \frac{A_1}{E_1}) \cos\theta_j, \quad (4.61)$$

and

$$A_1 = -(x_1 - x_{0j_1}) \cos\theta_j - (x_2 - x_{0j_2}) \sin\theta_j, \quad (4.62)$$

$$B_1 = (x_1 - x_{0j_1})^2 + (x_2 - x_{0j_2})^2, \quad (4.63)$$

$$E_1 = (x_1 - x_{0j_1}) \sin\theta_j - (x_2 - x_{0j_2}) \cos\theta_j. \quad (4.64)$$

### 4.3.6 Uniform Flow

The uniform flow, which is added to the potential field, generates a more effective potential field from a starting position to a goal position. This acts as an intuition for the robot to persue the goal. The direction of the uniform flow can be determined as

$$\alpha = \arctan \frac{x_{g_2} - x_{s_2}}{x_{g_1} - x_{s_1}}, \quad (4.65)$$

where $(x_{s_1}, x_{s_2})$ is the coordinate of the start point for the robot. The uniform flow is directed along a straight line connecting the start and goal positions. The relationship between the strength of a uniform flow and the strength of a single source line obstacle was presented in the previous section. Increasing the strength of a uniform flow affects the resulting trajectory the same as decreasing the strength of a source line obstacle. Again, the strength of a uniform flow is assumed, but the strengths of line obstacles are determined by Eq. (4.52). If the strength of a uniform flow is increased, the strengths of line obstacles, obtained from Eq. (4.52), also increase to satisfy the given normal velocity $V_j$.

### 4.3.7 Goal Sink

The purpose of using the goal sink is to provide a global minimum. That is, the potential function of (4.35) has only one global minimum at the location of this goal sink. The strength of this goal sink must be large enough such that it can attract the robot. If not, a robot may follow the uniform flow and can miss the goal and pass by it. To minimize the possibility of collision of the robot with obstacles and the possibility of missing the goal, the strength of the goal sink and the source/sink line obstacles of the obstacle must satisfy the following inequality

$$-\lambda_g < \lambda_o < 0, \quad (4.66)$$

where the obstacle strength $\lambda_o$ is defined as

$$\lambda_o = \sum_{i=1}^{m} \lambda_i L_i. \tag{4.67}$$

If the obstacle strength $\lambda_o$ is positive, one can conclude that there is more sink than source in the line obstacles and the net effect of obstacle is attractive. Thus, with the resulting potential function, the robot may collide with the obstacles. This positive obstacle strength can be derived, if velocities $V_i$'s in Eq. (4.38) are too small. For an obstacle to provide a repulsive potential, line obstacles must have more source than sink. On the contrary, if velocities $V_i$'s are very large, it is possible that $-\lambda_g > \lambda_o$. Then, the velocity field created by the obstacles may prevent the particles in the flow from going into the goal sink. That is, a mobile robot cannot reach the goal and will move to infinity following the uniform flow. In conclusion, one can say that the inequality (4.66) indirectly gives the bounds of $V_i$'s, which we have to specify to derive the strengths of line obstacles. Large $V_i$'s imply a safer but less economical (longer) trajectory away from obstacles. However, there is a limit of how high $V_i$'s can be. This limit is indirectly set by the inequality (4.66). The following example illustrates this concept.

*Example 4.4.* Consider an equilateral triangle as an obstacle. The vertices of the triangle are located at $(-1, 0)$ m, $(0, 1.73)$ m, and $(1, 0)$ m. A mobile robot is initially located at the start point $\mathbf{x}_s = (-1, -4)$ m and must go to the goal point $\mathbf{x}_g = (1, 4)$ m. Using the potential field method with harmonic potentials, plan a path from the start point to the goal point. Assume that the uniform flow has a unit strength ($U = 1$) and the strength of the goal is 30 ($\lambda_g = 30$). Use equal $V_i$'s for all the sides of the triangle (line obstacles). Test $V_i$ equal to 0.0, 1.0, 2.0, 3.0, 4.0, and 5.0 and compare the corresponding calculated paths.

*Solution.* The parameters that define the line obstacles of the triangular obstacle are determined. Figure 4.9 is used as a template. It is crucial that the line obstacle parameters are determined such that the normal vector of all line obstacle, $\mathbf{n}_i$'s, point toward outside of the obstacle. The parameters for line obstacles 1–3 are shown in Table 4.1.

The uniform flow direction is calculated such that the flow directed from the robot's start point to its goal point. Using Eq. (4.65) results in

$$\alpha = \arctan \frac{4 - (-4)}{1 - (-1)} = 1.3258 \text{ rad}. \tag{4.68}$$

**Table 4.1** Parameters of line obstacles forming the triangular obstacle

| Line obstacle $j$ | $x_{0j_1}$ (m) | $x_{0j_2}$ (m) | $L_j$ (m) | $\theta_j$ (rad) |
|---|---|---|---|---|
| 1 | −1.00 | 0.00 | 2.00 | $\pi/3$ |
| 2 | 0.00 | 1.73 | 2.00 | $-\pi/3$ |
| 3 | 1.00 | 0.00 | 2.00 | $\pi$ |

All the parameters required to calculate the terms in Eq. (4.52) are at hand at this stage, and the matrices needed for solving for the line obstacle strengths can be determined.

For $V_i$'s equal to 0.0, the following matrices can be derived.

$$\mathbf{P} = \begin{bmatrix} 0.5000 & 0.2007 & 0.2007 \\ 0.2007 & 0.5000 & 0.2007 \\ 0.2007 & 0.2007 & 0.5000 \end{bmatrix} \quad \mathbf{q} = \begin{bmatrix} 0.3810 \\ 1.6432 \\ -2.0936 \end{bmatrix} \tag{4.69}$$

These matrices result in the following strengths per unit length for the line obstacles.

$$\Lambda = \begin{bmatrix} 1.3247 \\ 5.5421 \\ -6.9437 \end{bmatrix} \tag{4.70}$$

The strengths corresponding to the other values of $V_i$ can be determined by using a similar procedure as described above. The results are listed in Table 4.2.

Once the strengths are determined, one can use Eqs. (4.56) and (4.57) to find the flow particle velocities at any point $(x_1, x_2)$. These velocities are numerically integrated to obtain a path from the robot's start point to its goal point. The path is defined as an array of points. The coordinates of each point $k + 1$ of this array is calculated based on the coordinates of the previous point $k$ and the particle velocities at that point $k$.

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{\mathbf{u}_k}{|\mathbf{u}_k|}\Delta s, \tag{4.71}$$

where $\Delta s$ is a small path step for integration, and $\mathbf{u} = [u_1, u_2]$ is the particle velocity vector, whose components are calculated using Eqs. (4.56) and (4.57). The start point's coordinates are used to initialize the integration. That is,

$$\mathbf{x}_1 = \mathbf{x}_s \tag{4.72}$$

The numerical integration procedure is terminated when an $\mathbf{x}_k$ is found that is closer to the robot's goal point than a predefined tolerance.

**Table 4.2** Obstacle's total strength for different $V_i$'s

| $V_i$ | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\lambda_o$ | $\lambda_g$ | Eq. (4.66) true? |
|---|---|---|---|---|---|---|
| 0.0 | 1.3247 | 5.5421 | −6.9437 | −0.1537 | 30 | Yes |
| 1.0 | 0.2154 | 4.4328 | −8.0530 | −6.8099 | 30 | Yes |
| 2.0 | −0.8940 | 3.3234 | −9.1624 | −13.4660 | 30 | Yes |
| 3.0 | −2.0033 | 2.2141 | −10.2718 | −20.1221 | 30 | Yes |
| 4.0 | −3.1127 | 1.1047 | −11.3811 | −26.7782 | 30 | Yes |
| 5.0 | −4.2220 | −0.0046 | −12.4905 | −33.4343 | 30 | No |

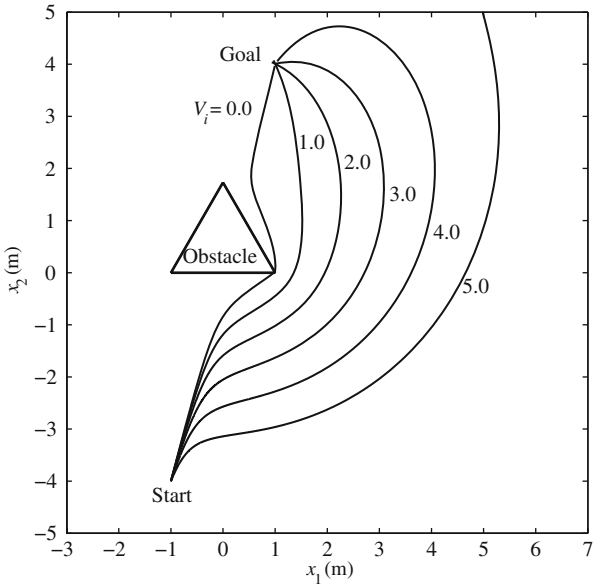


**Fig. 4.10** Paths for a robot avoiding a *triangular obstacle* for different $V_i$'s

The above integration procedure is used to find the paths for different $V_i$'s. These paths are shown in Fig. 4.10. The path corresponding to $V_i = 0.0$ is the closest to the obstacle. (This is actually an approximation of the streamline of a fluid flow around the triangle as a solid body.) Note that this path is not practical for a robot with a finite size. As it was anticipated, larger $V_i$'s result in a longer path farther from the obstacle. Depending on the size of the robot, any of the paths for $V_i = 1.0$–$4.0$ are acceptable solutions.

Note that although higher values for $V_i$ result in safer paths, there is a limit for valid $V_i$'s. As shown in Table 4.2, the obstacle's total strength increases as the value for $V_i$'s increases. At some $V_i$ value, the total repulsive source strength of the obstacle becomes stronger than the attractive sink strength of the goal and the inequality (4.66) no longer holds (see Table 4.2). This situation has occurred for $V_i = 5.0$ for this example. As can be seen in Fig. 4.10, the goal cannot attract the robot and the robot misses the goal.

The above example shows how the effectiveness of the potential field is affected by the value selected for $V_i$'s. The correct value for $V_i$'s depends on the size and shape of the obstacle and the relative position of the robot's start point and the goal point with respect to the obstacle. Determining this value by trial and error in real world situations is not practical. A method is required to determine the safe value automatically. The next section discusses this method.

## 4.4 Two-Dimensional Robust Harmonic Potential Field

In the previous section, for generating an artificial potential field, the velocity of particles at the center point of each panel had to be specified to be able to solve a path planning problem. The value of this velocity must have been selected such that the total obstacle repulsive strength becomes less than the goal attractive strength.

This is required to guarantee that the robot does not miss the goal and the goal is the global minimum of the potential. The value of these velocities highly depend on the size and shape of the obstacles. Since the sizes of the obstacles are different for different path planning problems, it is impossible to decide the value of normal velocities by trial and error. Therefore, a method for automatically adjusting the potential field parameters based on the obstacle sizes is needed. This section presents the innovative method. A robust artificial potential field can be generated for any sizes of obstacles by using the method to be presented in this section. This potential field is used for obstacle avoidance.

In the previous section, the artificial potential field was generated using the superposition of a number of harmonic potential functions.

$$\phi(x_1, x_2) = \phi_u + \phi_g + \sum_{j=1}^{m} \phi_j \tag{4.73}$$

In the above equation, which is a repeat of Eq. (4.35), $U$ and $\lambda_g$ must be specified. Then, to compute the $m$ panel strengths, $m$ normal outward velocities $V_i$ on each panel must be specified and the following system of linear equations must be solved in terms of $m$ panel strengths per unit length $\lambda_j$. This equation is the repeat of Eq. (4.38).

$$\frac{\lambda_i}{2} + \sum_{j \neq i}^{m} \frac{\lambda_j}{2\pi} I_{ij} = -V_i + U \frac{\partial}{\partial n_i}(x_{ci_1} \cos\alpha + x_{ci_2} \sin\alpha) - \frac{\lambda_g}{2\pi} \frac{\partial}{\partial n_i} \ln R_{gi} \quad i = 1, \ldots, m \tag{4.74}$$

The $m$ normal outward velocities $V_i$ must be chosen such that, after solving (4.74), the following condition is satisfied.

$$-\lambda_g > \lambda_o > 0, \tag{4.75}$$

where the obstacle strength $\lambda_o$ was defined as

$$\lambda_o = \sum_{i=1}^{m} \lambda_i L_i. \tag{4.76}$$

Equation (4.75) is called the convergence condition because it indicates that the summation of obstacle repulsive strengths is less than the goal attractive strength. If the convergence condition is true, it is guaranteed that the robot does not miss

the goal and the goal is the global minimum of the potential. But whether or not this condition is true highly depends on how the $m$ normal outward velocities $V_i$ on each panel are specified. In the case of known obstacles, it may be enough to specify them once with trial and error and the potential field could be useful. However, in the case of unknown obstacles, a method is needed to compute them automatically such that the inequality (4.75) is satisfied.

Equation (4.74) is rewritten as

$$\sum_{j=1}^{m} P_{ij}\lambda_j = -V_i + W_i, \tag{4.77}$$

where

$$P_{ij} = \begin{cases} 1/2 & \text{if } i = j \\ I_{ij}/2\pi & \text{if } i \neq j \end{cases}, \tag{4.78}$$

and

$$W_i = U\frac{\partial}{\partial n_i}(x_{ci_1}\cos\alpha + x_{ci_2}\sin\alpha) - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial n_i}\ln R_{gi} \quad i = 1, \ldots, m. \tag{4.79}$$

If $\mathbf{J} = \mathbf{P}^{-1}$, one can write

$$\lambda_i = \sum_{j=1}^{m} J_{ij}(-V_j + W_j) \quad i = 1, \ldots, m. \tag{4.80}$$

Substituting Eq. (4.80) into (4.76) and (4.75) results in

$$\sum_{i=1}^{m} L_i \sum_{j=1}^{m} J_{ij}V_j < \lambda_g + \sum_{i=1}^{m} L_i \sum_{j=1}^{m} J_{ij}W_j. \tag{4.81}$$

If it is assumed that the outward normal velocity of a panel is proportional to its length, that is,

$$V_j = aL_j, \tag{4.82}$$

then Eq. (4.81) reduces to

$$a < a_{\max}, \tag{4.83}$$

where

$$a_{\max} = \frac{\lambda_g + \sum_{i=1}^{m} L_i \sum_{j=1}^{m} J_{ij}W_j}{\sum_{i=1}^{m} L_i \sum_{j=1}^{m} J_{ij}L_j}. \tag{4.84}$$

Since all the parameters on the right hand side of Eq. (4.84) are known, one can simply calculate $a_{\max}$ and choose a value for $a$ that satisfies the inequality (4.83). Then, the outward normal velocities that are guaranteed to satisfy Eq. (4.66) are computed from Eq. (4.82).

The parameter $a$ is called the safety parameter. Larger $a$'s result in larger $V_i$'s, which in turn result in a safer path far from the obstacle. $a_{\max}$ is the maximum allowable safety parameter, based on which one can determine the maximum allowable $V_i$'s. If $a$ is greater than $a_{\max}$, the repulsive strength of the obstacle will be too much and the robot will miss the goal point. The safety ratio is defined as

$$r_a = \frac{a}{a_{\max}}, \quad 0 < r_a < 1. \tag{4.85}$$

For practical purposes, the safety ratio $r_a$ is selected by the robot's user. Then, the maximum allowable safety parameter is determined using Eq. (4.84) and $a$ is determined using Eq. (4.85). When $a$ is at hand, the robust panel strengths are computed as

$$\lambda_i = \sum_{j=1}^{m} J_{ij}(-aL_j + W_j) \quad i = 1, \ldots, m. \tag{4.86}$$

These panel strengths are used for forming the total potential field, calculating the fluid particle velocities. The corresponding velocity field, $\mathbf{u} = (u_1, u_2)$, is then derived from $\mathbf{u} = -\nabla\phi$ and using Eq. (4.73).

$$u_1(x_1, x_2) = U\cos\alpha - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial x_1}\ln R_g - \sum_{j=1}^{m}\frac{\lambda_j}{2\pi}\int_j \frac{\partial}{\partial x}\ln R_j dl_j \tag{4.87}$$

$$u_2(x_1, x_2) = U\sin\alpha - \frac{\lambda_g}{2\pi}\frac{\partial}{\partial x_2}\ln R_g - \sum_{j=1}^{m}\frac{\lambda_j}{2\pi}\int_j \frac{\partial}{\partial y}\ln R_j dl_j \tag{4.88}$$

Note that these velocities correspond to the artificial potential field built for obstacle avoidance. Once again, one may use the analogy of this artificial potential to that of an irrotational incompressible fluid flow, and note that these velocities can be thought of as the velocity of fluid particles, which are not the real velocity of the robots. In fact, it may not be possible to use these velocities directly to control the velocity of the robots because they could be too high for the robots to achieve. In this situation, a control strategy is needed to use the velocity information of the potential field and plan a trajectory for the robots. In the following section, these issues are discussed and methods to use the velocity field information for obstacle avoidance are proposed.

## 4.5 Path Planning for a Single Mobile Robot

The following steps summarize the use of the developed theories for path planning.

### *4.5.1 Algorithm for a Single Robot*

1. The uniform flow strength $U$, the goal strength $\lambda_g$, and the safety ratio $r_a$ are selected. It is recommended that the goal strength should be much larger than the uniform flow strength for a more effective potential field.
2. The start position $\mathbf{x}_s = (x_{s1}, x_{s2})$ and the goal position $\mathbf{x}_g = (x_{g1}, x_{g2})$ are defined.
3. The obstacle line segments' parameters according to the notation introduced for line obstacle $i$ in Fig. 4.9 are determined. Extra care must be taken in defining the reference point $(x_{0i1}, x_{0i1})$ and $\theta_i$ such that the normal unit vector $\mathbf{n}_i$ points to the outside of the obstacle. If this condition is not met, the generated paths may interfere with the obstacles.
4. The direction of the uniform flow is obtained from Eq. (4.65).
5. The elements $P_{ij}$ are calculated using Eq. (4.53) [or (4.78)], and $W_i$ from Eq. (4.79).
6. The parameter $a_{\max}$ is evaluated using Eq. (4.84) and $a$ is found from Eq. (4.85).
7. The strength per unit length for the $m$ panels are calculated using Eq. (4.86).
8. The velocity components given by Eqs. (4.87) and (4.88) are used to find the direction of the local tangent to the path. The path is generated numerically. The path starts at $\mathbf{x}_1 = \mathbf{x}_s$. At any step $k$, the direction of the instantaneous velocity (the direction of local tangent to the path) and the new position are calculated as follows.

$$\mathbf{u}_k = \mathbf{u}(\mathbf{x}_k), \tag{4.89}$$

$$\hat{\mathbf{u}}_k = \frac{\mathbf{u}_k}{|\mathbf{u}_k|}, \tag{4.90}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta s \hat{\mathbf{u}}_k, \tag{4.91}$$

where $\Delta s$ is an arbitrary small distance. The iteration in $k$ continues until the point found for the path is closer than the defined small distance $\Delta s$.

$$||\mathbf{x}_{k+1} - \mathbf{x}_g|| \leq \Delta s \tag{4.92}$$

9. The array of points $\mathbf{x}_k$ is the planned path.

In the following, a path planning example is presented.

*Example 4.5.* Consider the obstacle avoidance scenario laid out in Example 4.4. Calculate the maximum allowable safety parameter and the maximum allowable $V_i$'s for all the line obstacles. Plan the path of the robot from its start position to its goal position for the safety ratio values of 0.0–1.0 with increments of 0.2.

**Table 4.3** Obstacle's total strength for different $r_a$'s

| $r_a$ | $a$ | $V_i$ | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\lambda_o$ | $\lambda_g$ |
|---|---|---|---|---|---|---|---|
| 0.0 | 0.0000 | 0.0000 | 1.3247 | 5.5421 | −6.9437 | −0.1537 | 30 |
| 0.2 | 0.4484 | 0.8968 | 0.3298 | 4.5472 | −7.9386 | −6.1230 | 30 |
| 0.4 | 0.8968 | 1.7936 | −0.6650 | 3.5524 | −8.9335 | −12.0922 | 30 |
| 0.6 | 1.3452 | 2.6904 | −1.6599 | 2.5575 | −9.9283 | −18.0615 | 30 |
| 0.8 | 1.7936 | 3.5872 | −2.6548 | 1.5626 | −10.9232 | −24.0307 | 30 |
| 1.0 | 2.2420 | 4.4840 | −3.6496 | 0.5677 | −11.9181 | −30.0000 | 30 |

*Solution.* For the robot's start and goal positions and the obstacle given in Example 4.4, one can calculate the matrices **J** and **W**.

$$\mathbf{J} = \begin{bmatrix} 2.5973 & -0.7440 & -0.7440 \\ -0.7440 & 2.5973 & -0.7440 \\ -0.7440 & -0.7440 & 2.5973 \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} 0.3810 \\ 1.6432 \\ -2.0936 \end{bmatrix} \tag{4.93}$$

Substituting these values into Eq. (4.84) results in

$$a_{\max} = 2.2420 \tag{4.94}$$

Since all the line obstacles have the same length of $L_i = 2.0\,\text{m}$ ($i = 1, \ldots, 3$), the maximum allowable $V_i$ for all the line obstacles are equal to

$$V_{i_{\max}} = a_{\max} L_i = 4.4840 \tag{4.95}$$

This confirms the result of Example 4.4, in which a $V_i = 5.0 > 4.4840$ caused the robot to miss its goal point.

After $a_{\max}$ is at hand, one can use the safety ratio $r_a$ to find $a$ and the allowable $V_i$'s, calculate the line obstacle strengths using Eq. (4.86), and plan a path for the robot. The line obstacle strengths for different values of the safety ratio are listed in Table 4.3. As a conclusion from Table 4.3, note that as long as one selects a safety ratio less than (or equal to) one, the obstacle strength is smaller than (or equal to) the goal's strength, and the robot does not miss its goal point. The robot's path corresponding to different safety ratios are shown in Fig. 4.11.

## 4.6 Path Planning for Multiple Mobile Robots

In this section, the case in which more than one robot is working in an environment with obstacles is considered. The method discussed in this section also applies to a single robot in an environment with moving obstacles.[1]

[1] Some of the material of this section has been adapted from the article titled: "Real-time obstacle avoidance for multiple mobile robots," By Farbod Fahimi, C. Nataraj and Hashem Ashrafiuon, Robotica, Copyright 2008 Cambridge University Press, UK.

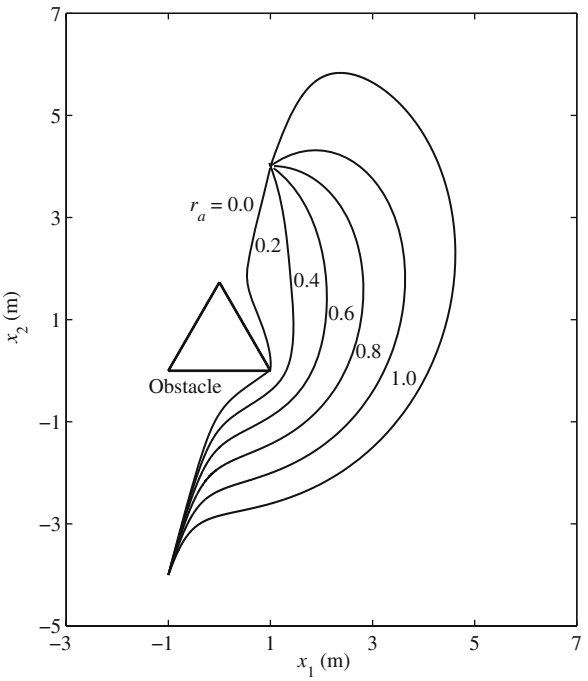


**Fig. 4.11** Paths for a robot avoiding a *triangular obstacle* for different $r_a$'s

In the case of path planning for a single robot in an environment with stationary obstacles, the potential field does not change; hence, it is sufficient to specify the potential gradients once such that the convergence condition is satisfied.

However, in the case of a group of robots, where each robot must consider the other robots of the group as moving obstacles, the obstacles' positions are constantly changing. When the obstacles' positions are constantly changing, it is necessary to update the potential field, Eq. (4.73), as the robots move. In other words, the changing potential field has to satisfy the convergence condition (4.75) at all times.

The implementation of the potential method for trajectory planning of a group of $n$ mobile robots follows. It is assumed that a typical robot in the group considers the other robots of the group as moving obstacles. This means that each robot of the group has a different artificial potential field, which is time-dependent. Let $\mathbf{u}^{(i)}(t) = -\nabla\phi^{(i)}(t)$ be the potential gradient that is computed for the robot $i$. Also, without loss of generality, it can be assumed that all members of the group have a maximum velocity of $v_{\max}$. Then, the normalized instantaneous velocity of each robot is set as

$$\mathbf{v}^{(i)}(t) = \frac{v_{\max}}{u_{\max}(t)}\mathbf{u}^{(i)}(t), \tag{4.96}$$

where $u_{\max}(t)$ is the magnitude of the largest potential gradient of those computed for the robots at time $t$.

$$u_{\max}(t) = \max(|\mathbf{u}^{(i)}(t)|, i = 1, \ldots, n), \tag{4.97}$$

where $n$ is the number of robots. Given the initial position vector of the robot $i$, $\mathbf{P}_0^{(i)}$, one can compute the trajectory of each robot by integrating the instantaneous velocity.

$$\mathbf{P}^{(i)}(t) = \mathbf{P}_0^{(i)} + \int_0^t \mathbf{v}^{(i)}(\tau) d\tau. \tag{4.98}$$

Once the trajectory has been determined, it can be fed to a trajectory-tracking controller, which controls the robots such that they follow the planned trajectory. The trajectory-tracking controllers are discussed in the next chapter.

### *4.6.1 Algorithm for Multiple Robots*

A group of $n$ mobile robots are considered in a field of moving and stationary obstacles. It is assumed that the starting position and geometry of robots and obstacles are known. The goal position of all the robots, the uniform flow strength, $U$, the goal strength, $\Lambda_g$, the safety ratio $r_a = a/a_{\max}$, and the update time interval $\Delta t$ are preselected. Also, the trajectories of the moving obstacles are assumed to be known. A ground station, which plans the trajectories, is in communication with the robots. The ground station and the robots execute the following algorithm.

1. At time $t$, the robots sequentially communicate their positions to the ground station. The ground station sets up an obstacle field for robot $i$ based on the panels corresponding to the stationary and moving obstacles plus $4(n-1)$ panels representing the other robots of the group. The goal sink is defined at the corresponding goal position of the robot $i$, and the uniform flow potential is directed from the current position of this robot to its goal position.
2. For robot $i$, the ground station uses the above artificial potential setup along with Eqs. (4.53), (4.79), and (4.84) and computes the robot $i$'s corresponding $a_{\max}$.
3. Then, it obtains $a$ by using the specified value for the safety ratio $r_a$ and calculates the corresponding panel strengths, $\lambda_j$'s from Eq (4.86).
4. For robot $i$, the ground station determines the potential gradient, $\mathbf{u}^{(i)}(t)$, corresponding to the robot $i$'s current location from Eqs. (4.87) and (4.88). The ground station executes Steps 2 to 4 for all the robots.
5. The ground station searches for equal magnitudes of the potential gradients among all the robots. If it detects a subgroup of robots that see equal potential gradients, it modifies the potential gradients of that subgroup. The potential gradient of the robot with the lowest number in the subgroup remains unchanged, and the potential gradient for the robot with the highest number in the subgroup is set to zero. Then, the ground station determines the potential gradient of the rest of the robots in the subgroup by linear interpolation between the equal

potential gradient and zero. This step is necessary, or members of the subgroup may collide with each other.
6. The ground station searches for the maximum potential gradients that are calculated for the group of robots, Eq. (4.97), and plans the robots' instantaneous velocities at current time $t$ using the control strategy defined in Eq. (4.96). Then, it sequentially sends the calculated instantaneous velocities to all the robots.
7. Robot $i$ computes its desired position at time $t + \Delta t$ from Eq. (4.98) and tracks this trajectory using a trajectory-tracking controller.

At time $t + \Delta t$ the robots have moved to new positions. The algorithm starts from the first step and repeats for the new positions. This procedure continues until all the robots are at their goal positions.

Note that although the robots are numbered from 1 to $n$, the numbers are not priority numbers by any means and are only for reference. In general, the numbering does not affect the result of the algorithm because the velocity of the robots are determined only based on the potential gradients. The only minor exception is when a subgroup of robots see equal potential gradients. In this case, the robot numbers affect the calculation of the velocity of the robots in the subgroup as explained above.

*Example 4.6.* Consider three robots 1, 2, and 3 that are initially located at start positions $S_1(0.6, 1.5)$, $S_2(-0.6, 1.5)$, and $S_3(0.0, 1.5)$ m, respectively. The robots' goal points are defined at $G_1(-0.6, -1.2)$, $G_2(0.6, -1.5)$, and $G_3(0.0, -1.2)$ m for robots 1, 2, and 3, respectively. All the robots have the same size of 0.20 m by 0.15 m and a maximum speed of 0.1 m/s. Two long obstacles with a depth of 0.4 m have left an opening of 0.45 m. The opening is centered around the $y$ axis. Apply the procedure described in Section 4.6.1 to plan the trajectory of the three robots.

*Solution.* The values $U = 1$ m/s and $\lambda_g = 30\,\text{m}^2/\text{s}$ are used for uniform flow strength and goal attractive strength, respectively, for all robots. Also, safety ratio $r_a$ is set to 0.99 to minimize the possibility of collision. The robots' geometry are approximated by rectangles with dimensions twice as large as the robots' dimensions. This is required because when the potential field is generated for a typical robot in the group, the robot itself is treated as a point. By assuming larger dimensions for the other robots, it is guaranteed that the robots do not collide. The same argument is valid for the dimensions of the obstacles. Therefore, the stationary and moving obstacles are approximated by polygons that are larger than the actual obstacles by half of the largest dimension of the largest robot. The procedure described in Section 4.6.1 is applied to plan the trajectory of the three robots.

Figure 4.12 shows six snapshots of the robot group motion. The start and goal points are marked on the figure. The two obstacles forming a narrow passage are shown by thick lines. Robots are shown by rectangles with arrows indicating the direction of their movement. The traces of the robots are also shown.

It is seen that due to the width of the narrow passage, the robots cannot pass the opening at the same time and are on a collision course. The only way is to go through the opening one at a time. Also, note that robots 1 and 2 are symmetrical

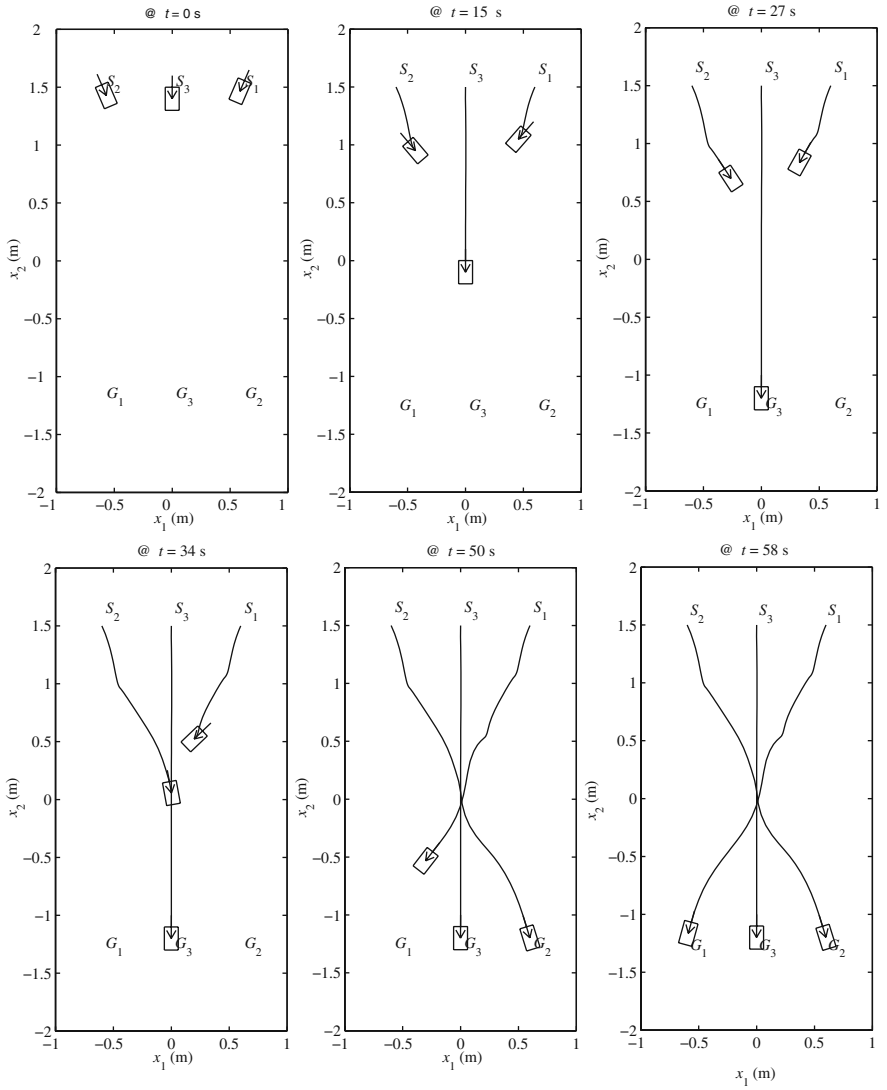


**Fig. 4.12** Snapshots of the motion of three robots avoiding obstacles

with respect to obstacles, which means that they are at points with equal potentials. The symmetry detecting algorithm, described in Section 4.6.1, slows down robot 1 to avoid collision. As seen in Fig. 4.12, robots 1 and 2 slow down while robot 3 is passing the opening, after which robot 2 speeds up and robot 1 follows. The robots manage to arrive at their corresponding goal points without colliding with each other and the obstacles. Robots 1, 2, and 3 are at their goals after 58, 50, and 27 s, respectively.

## 4.7 Structural Local Minimum and Stagnation Points

Maybe the most common disadvantage of the potential field methods is the problem of local minima. Fortunately, this problem can be addressed by using harmonic potentials and the panel method. However, attention must be paid to the general limitations of the potential field methods and the particular issues of the panel method (or harmonic potential fields).

Generally, in the potential field methods, the robot is considered as a point without physical dimensions. With this assumption, a planned trajectory among obstacles that is valid for a point robot may not be valid for a robot with real physical dimensions. This is known as the *structural local minimum*. To avoid the structural local minimum, the size of the mobile robot whose path is being planned must be considered. This can be done by extending the size of the obstacles and other robots in the workspace by at least half of the size of the mobile robot. When the size of the obstacles are extended as described above, an opening between two obstacles that has a dimension just equal to the size of the robot appears to be blocked. Therefore, no path will be planned through that opening, which is impossible to be passed by the robot.

Another issue of harmonic potential fields is the existence of stagnation points (local maxima) at which the potential gradient is zero. In the vicinity of these points, the planned velocity for a robot becomes very close to zero and there is a possibility that a robot will be trapped at these points. However, these points are not considered as stable equilibrium points in the potential because the gradient of the potential in the vicinity of these points (local maxima) is negative. This means that if the robot is disturbed from a stagnation point, it will continue its way toward the global minimum of the potential. A step can be easily incorporated into the path planning algorithm, which will assign a small velocity away from the obstacle to the robot if it detects a stagnation point.

## 4.8 Three-Dimensional Harmonic Potential Functions

In the previous section, the 2D path planning problem was considered. A 2D path can be used for vehicles that work in 2D environments, for example, on the ground or in the sea. However, to plan a path for the end-effector of a spatial manipulator or for an aerial robot moving in a 3D space, a spatial method is required. This section presents such a method.

In this section, the basic idea of using a harmonic potential field, as discussed in the previous section, is used. The formulations, however, are designed for the case of a spatial motion. Here, a systematic method can be found to plan a spatial path for a single aerial robot or a manipulator's end-effector to avoid known 3D obstacles.

### *4.8.1 Uniform Flow*

Similar to the 2D case, a uniform flow from a starting position to a goal position is used to setup the potential field to generate a more effective field. When the start

point is far from the goal point, the goal point's potential is too weak to attract the robot effectively. The uniform flow drives the robot toward the goal point in such a situation, resulting to a more effective potential. The 3D potential of uniform flow, $\phi_u$, is written as

$$\phi_u = -(a_1 x_1 + a_2 x_2 + a_3 x_3)U, \tag{4.99}$$

where $a_1$, $a_2$, and $a_3$ are the direction cosines of a line connecting the start point to the goal point and $U$ is the strength of the potential. Note that the potential function defined for $\phi_u$ is harmonic, i.e., it satisfies the 3D form of the Laplace equation (4.5).

*Example 4.7.* Plot the potential generated by a uniform potential of

$$\phi_u = (x_1 + x_2 + x_3)U, \qquad U = 1 \text{ m/s}, \tag{4.100}$$

on a 2 by 2 m flat surface parallel to the $x_1 - x_2$ plane passing through $x_3 = 0$ m and centered at the origin

*Solution.* The target plane can be described by

$$-2 \leq x_1 \leq 2 \text{ m}, \qquad -2 \leq x_2 \leq 2 \text{ m}, \qquad x_3 = 0 \text{ m}. \tag{4.101}$$

The $x_1$ and $x_2$ intervals are divided into 30 equal increments, summing to a total of 961 points. The potential at these points are calculated and plotted as a mesh. Figure 4.13 shows the resulting mesh. The slope of this potential in the $x_1$ and $x_2$ directions, observed from the figure, are $-1$ (m$^2$/s)/m. The velocity field that this potential generates is negative of the gradient of the potential field. Therefore, the components of the generated velocity field along the $x_1$ and $x_2$ directions are $+1$ m/s. The component of the velocity field in the $x_3$ direction cannot be observed from Fig. 4.13.

### *4.8.2 Goal Sink*

Since the robot must reach the goal, an attractive harmonic potential is needed at the goal, where the superposed potential has only one global minimum. This attractive goal can be represented by a singular point sink. The 3D harmonic potential generated by the goal sink is expressed as

$$\phi_g = -\frac{\lambda_g}{R_g}, \tag{4.102}$$

where $R_g$ is the distance between the point $(x_1, x_2, x_3)$ (the current location of the robot) and the goal point $G(x_{g1}, x_{g2}, x_{g3})$ and $\lambda_g$ is the goal sink strength. It can be shown that the goal potential, as defined in Eq. (4.102), satisfies the Laplace equation (4.5).

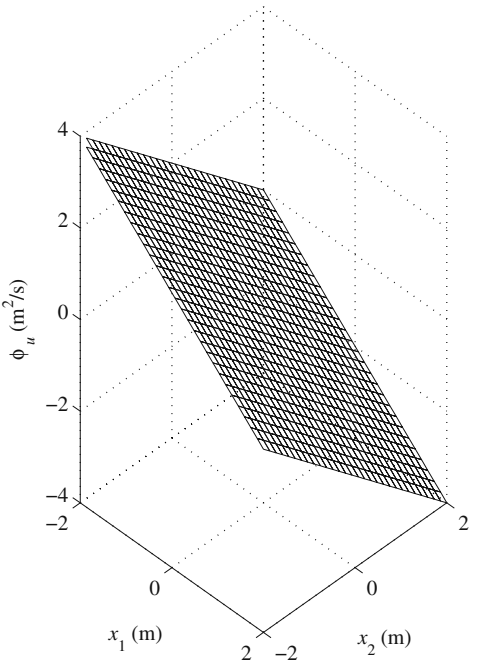


**Fig. 4.13** The potential generated on a 2 by 2 m horizontal plane by a uniform flow

*Example 4.8.* Plot the potential generated by a goal sink of

$$\phi_g = -\frac{\lambda_g}{R_g}, \qquad \lambda_g = 1 \text{ m}^3/\text{s}, \tag{4.103}$$

located at the origin on two 2 by 2 m flat surfaces parallel to the $x_1$–$x_2$ plane passing through $x_3 = 0.1$ and 1.0 m and centered at the origin

*Solution.* The target planes can be described by

$$-2 \le x_1 \le 2 \text{ m}, \qquad -2 \le x_2 \le 2 \text{ m}, \qquad x_3 = 0.1 \text{ m}, \tag{4.104}$$

and

$$-2 \le x_1 \le 2 \text{ m}, \qquad -2 \le x_2 \le 2 \text{ m}, \qquad x_3 = 1.0 \text{ m}. \tag{4.105}$$

The $x_1$ and $x_2$ intervals are divided into 30 equal increments, summing to a total of 961 points. The potential at these points are calculated and plotted as a mesh. Figure 4.14 shows the resulting mesh. As can be seen in the figure, the potential is maximum at the center of the 2 by 2 plane because this point is the closest point on the plane to the goal position. As the distance of the 2 by 2 plane with the goal position increases from 0.1 to 1.0 m, the maximum potential decreases. This can be seen by comparing Fig. 4.14a and b.

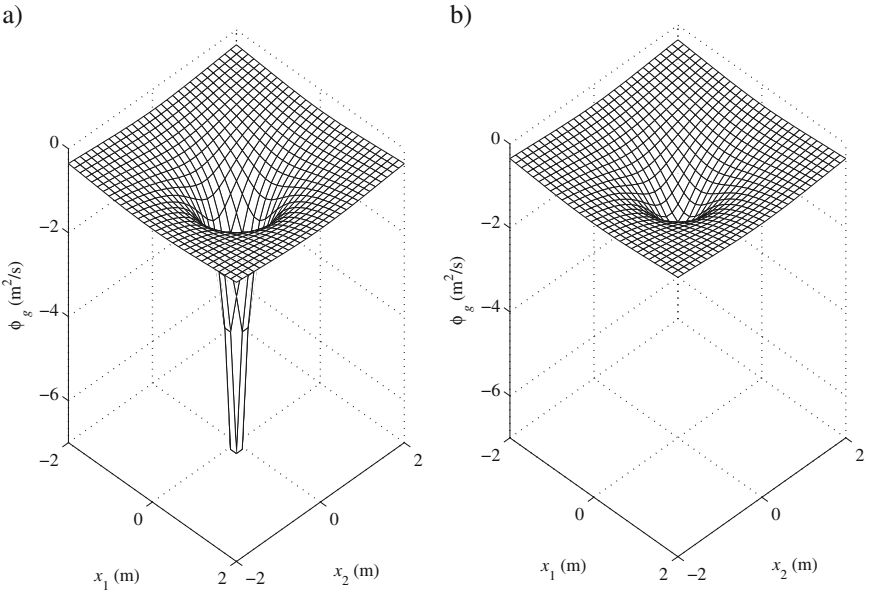


**Fig. 4.14** The potential generated by a goal sink on a 2 by 2 m horizontal plane located at a distance of (**a**) 0.1 m, (**b**) 1.0 m, from the goal sink

### *4.8.3 Spatial Panel*

Any 3D obstacle can be approximated by a number of spatial panels as shown in Fig. 4.15. The potential due to the obstacles can be obtained by calculating the potential for each polygonal panel and superposing the results. Harmonic sources or sinks of uniform density, similar to the functions defined for the goal potential, are distributed on each panel. A single panel is a planar on which uniform sources or sinks, with the strength per unit area $\lambda_i$, are distributed. The total potential resulting from this distribution must be calculated for each 3D panel. The calculation is done by integration.

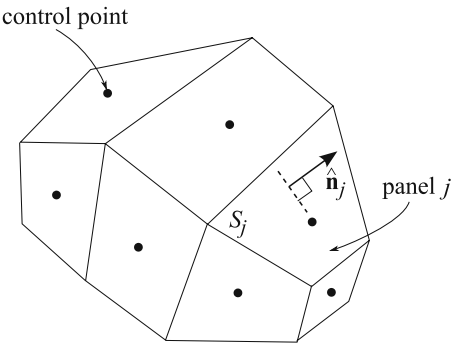


**Fig. 4.15** Representation of a three-dimensional obstacle by multiple spatial panels

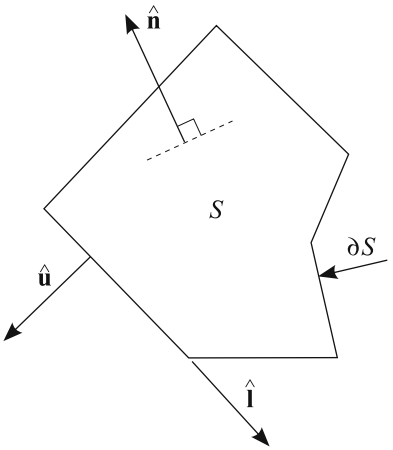


**Fig. 4.16** A single spatial panel

Figure 4.16 shows a single panel, which is a spatial polygon in general. The unit vector $\hat{\mathbf{n}}$ is perpendicular to the surface of the panel and is directed outward of the obstacle volume. The unit vector $\hat{\mathbf{l}}$ is directed along a leg of the panel, generally shown by $\partial S$. The unit vector $\hat{\mathbf{u}}$ is in the plane of the panel and perpendicular to the unit vector $\hat{\mathbf{l}}$. Panel $j$, with surface $S_j$, produces the following potential at a point $C(x_1, x_2, x_3)$.

$$\phi_j(x_1, x_2, x_3) = \lambda_j \int_{S_j} \frac{dS_j}{R_j} \tag{4.106}$$

where $R_j$ is the distance of the point $C(x_1, x_2, x_3)$ to an arbitrary point on the spatial panel $j$. The origin of the coordinate system is located at point $O$ (Fig. 4.17). The vector representing point $C$, at which the potential is being calculated, is denoted by $\mathbf{r}_C$. If $\mathbf{r}'$ is the vector pointing from the origin to an arbitrary point on leg $\partial S_k$ of the polygon $S$, then $R_k$ can be expressed as

$$\frac{1}{R_k} = \frac{1}{|\mathbf{r}_C - \mathbf{r}'|}. \tag{4.107}$$

The integral (4.106) can be converted to a summation of integrals over the boundary of the panel $S$, in which the position of an arbitrary point on the edges of the panel, Eq. (4.107), appear. The details of integration are beyond this text, therefore, only the integration result is presented here. More information about the derivation of the integral (4.106) can be found in [79].

$$\begin{aligned}\phi_j = \ & \lambda_j \sum_k \hat{\mathbf{P}}_k^0 . \hat{\mathbf{u}}_k [P_k^0 \ln \frac{R_k^+ + l_k^+}{R_k^- + l_k^-} \\ & - |d| \left( \arctan \frac{P_k^0 l_k^+}{(R_k^0)^2 + |d| R_k^+} - \arctan \frac{P_k^0 l_k^-}{(R_k^0)^2 + |d| R_k^-} \right)]\end{aligned} \tag{4.108}$$

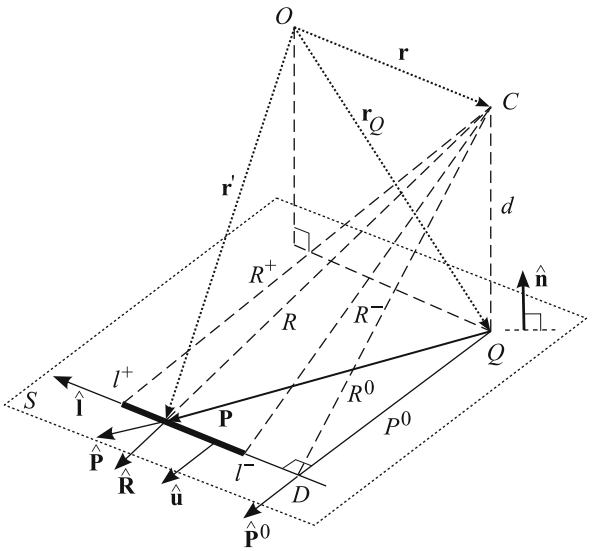


**Fig. 4.17** Geometrical quantities associated with a line segment $k$ as a leg of the panel $S_i$ shown by a thick line. The point $C$, with coordinates $(x_1, x_2, x_3)$, at which the potential is being observed, is located by the position vector $\mathbf{r}_C$ with respect to the coordinate origin $O$. Subscript $k$, indicating the line segment $k$, is dropped for simplicity

The geometric parameters (lengths) used in Eq. (4.108) are shown in Fig. 4.17 (note that the subscript $k$ is dropped for the parameters in this figure for simplicity). These parameters must be calculated before the potential at point $C$ can be determined.

All these parameters can be evaluated when the edges of the panel $S_i$ are defined. An edge $k$ of the panel $S_i$ is defined as a line segment by specifying the position of the two ends of the edge $k$ with respect to the origin $O$ as vectors $\mathbf{r}_k^+$ and $\mathbf{r}_k^-$, respectively. In addition, a unit vector $\hat{\mathbf{u}}_k$ is defined such that it is perpendicular to the edge $k$ and lies in the plane of the polygon $S_i$.

With the position of the two ends of the edge $k$ given, the distances $R_k^0$, $R_k^-$, and $R_k^+$ (Fig. 4.17) can be found.

$$R_k^0 = \frac{|(\mathbf{r}_k^+ - \mathbf{r}_k^-) \times (\mathbf{r}_C - \mathbf{r}_k^-)|}{|\mathbf{r}_k^+ - \mathbf{r}_k^-|}, \tag{4.109}$$

$$R_k^- = |\mathbf{r}_k^- - \mathbf{r}_C|, \tag{4.110}$$

$$R_k^+ = |\mathbf{r}_k^+ - \mathbf{r}_C|. \tag{4.111}$$

The distance $d$ is the height of the observation point $C$ above the plane of $S_k$, measured positively in the direction of $\hat{\mathbf{n}}$, may be calculated as

$$d = \mathbf{n}.(\mathbf{r} - \mathbf{r}_k^+) = \mathbf{n}.(\mathbf{r} - \mathbf{r}_k^-). \tag{4.112}$$

The three distances $R_k^0$, $d$, and $P_k^0$ form a right triangle, which results in

$$P_k^0 = \sqrt{(R_k^0)^2 - d^2}. \tag{4.113}$$

The signed distances $l_k^-$ and $l_k^+$ from point $D$ on the extension of the edge $k$ to the two ends of the edge $k$ must be calculated. The sign of these distances is positive if the vector pointing from $D$ to the corresponding end of the line segment has the same direction as $\hat{\mathbf{l}}_k$'s. The signed distances $l_k^-$ and $l_k^+$ can be determined after the position of point $D$ is calculated. The position of point $D$ is obtained by following the procedure discussed below. First, the unit vector of the edge $k$ is defined as

$$\hat{\mathbf{l}}_k = \frac{\mathbf{r}_k^+ - \mathbf{r}_k^-}{|\mathbf{r}_k^+ - \mathbf{r}_k^-|}. \tag{4.114}$$

Then, the position of point $D$ can be found as

$$\mathbf{r}_{Dk} = \left[\hat{\mathbf{l}}.(\mathbf{r}_C - \mathbf{r}_k^-)\right]\hat{\mathbf{l}} + \mathbf{r}_k^-. \tag{4.115}$$

Using the position of point $D$ from Eq. (4.115), one can obtain the two signed distances $l_k^-$ and $l_k^+$ as

$$l_k^- = (\mathbf{r}_k^- - \mathbf{r}_{Dk}).\hat{\mathbf{l}}_k, \tag{4.116}$$

$$l_k^+ = (\mathbf{r}_k^+ - \mathbf{r}_{Dk}).\hat{\mathbf{l}}_k. \tag{4.117}$$

Finally, the unit vector $\hat{\mathbf{P}}_k^0$ can be written by observing Fig. 4.17 as

$$\hat{\mathbf{P}}_k^0 = \left[\frac{(\mathbf{r}_{Dk} - \mathbf{r}_C).\hat{\mathbf{u}}_k}{|(\mathbf{r}_{Dk} - \mathbf{r}_C).\hat{\mathbf{u}}_k|}\right]\hat{\mathbf{u}}_k, \tag{4.118}$$

where

$$\hat{\mathbf{u}}_k = \hat{\mathbf{n}}_k \times \hat{\mathbf{l}}_k. \tag{4.119}$$

Note that in Eq. (4.118), the vectors $\hat{\mathbf{P}}_k^0$ and $\hat{\mathbf{u}}_k$ are always parallel. The term in the brackets determines if the two vectors have the same direction or have opposite directions. This fact can also be seen in Fig. 4.17. At configuration shown for the position of point $C$, the two vectors $\hat{\mathbf{P}}_k^0$ and $\hat{\mathbf{u}}_k$ have the same direction. If the projection of point $C$ on the panel $S_i$ (point $Q$) were on the other side of the edge $k$, the two vectors $\hat{\mathbf{P}}_k^0$ and $\hat{\mathbf{u}}_k$ would have opposite directions.

*Example 4.9.* Consider a square panel in a 3D space with sides of 2 m long. The square panel lies in the $x_1$–$x_2$ plane and its center of area is located at the origin of a fixed Cartesian coordinate system. The strength per unit area of the panel is $\lambda = -1$ m/s. Plot the potential $\phi(x_1, x_2, x_3)$ generated by this panel on 4 by 4 m$^2$ surfaces with the centers of area at (0, 0, 0.1), (0, 0, 0.5), (0, 0, 1.0), and (0, 0, 1.5) m.

*Solution.* First the geometry of the panel is defined by setting up the correct $\mathbf{r}_j^-$ and $\mathbf{r}_j^+$ vectors ($j = 1, \ldots, 4$ for the four sides of the square panel), which point to the vertices of the polygon. Since the direction of integration of the potential field along all the edges of the polygon must be consistant, care must be taken in defining the order of polygon vertices. A good practice is to number the vertices starting from an arbitrary vertex in the direction a of right hand rotation about the vector $\hat{\mathbf{n}}_j$, which is perpendicular to the panel and pointing outward of an obstacle volume. This is done for the square panel.The results for the vectors $\mathbf{r}_j^-$ and $\mathbf{r}_j^+$ are shown in Table 4.4.

After the panel geometry is defined, the geometrical parameters defined in Fig. 4.17 can be calculated using Eqs. (4.111), (4.112), (4.113), (4.114), (4.115), (4.116), (4.117), (4.118), and (4.119). These parameters are related to the coordinate of the point $C$ at which the potential is being calculated. The result of these calculation for point $C(0, 0, 1)$ m for this example are shown in Table 4.5. These numbers along with the position of point $C$ are substituted into Eq. (4.108) to calculate the potential generated by the square panel at point $C$.

$$\phi(0, 0, 1) = \sum_{i=1}^{4} \phi_i(0, 0, 1) = 3.173 \text{ m}^2/\text{s} \tag{4.120}$$

The potential generated by the square panel on 4 by 4 m surfaces parallel to the panel and with different distances are shown as 3D meshes is Figs. 4.18 and 4.19. The scale and the range of the $\phi$-axis is fixed for all the figures for easier comparison. For all the figures, the maximum potential happens at the center of the panel due to its symmetry. As expected, the maximum potential diminishes as the target surface is positioned farther from the square panel. When the target surface is at a distance of 1 m from the square panel (Fig. 4.19a), the peak of the potential is 3.172 $\text{m}^2/\text{s}$ as confirmed by Eq. (4.120).

**Table 4.4** Parameters of the three-dimensional square panel

| Edge $j$ | $\mathbf{r}_j^-$ (m) | $\mathbf{r}_j^-$ (m) |
|---|---|---|
| 1 | [0, 1, 1] | [0, −1, 1] |
| 2 | [0, −1, 1] | [0, −1, −1] |
| 3 | [0, −1−, 1] | [0, 1, −1] |
| 4 | [0, 1, −1] | [0, 1, 1] |

**Table 4.5** Geometrical parameters, defined in Fig. 4.17, for the three-dimensional square panel at point $C(0, 0, 1)$ m

| Side $j$ | $R_j^-$ | $R_j^+$ | $R_j^0$ | $d$ | $P_j^0$ | $\hat{\mathbf{l}}_j$ | $\mathbf{r}_{Dj}$ | $l_j^-$ | $l_j^+$ | $\hat{\mathbf{P}}_j^0$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.73 | 1.73 | 1.41 | 1.00 | 1.00 | [−1, 0, 0] | [0, 1, 0] | −1.00 | 1.00 | [−0, 1, 0] |
| 2 | 1.73 | 1.73 | 1.41 | 1.00 | 1.00 | [0, −1, 0] | [−1, 0, 0] | −1.00 | 1.00 | [−1, 0, 0] |
| 3 | 1.73 | 1.73 | 1.41 | 1.00 | 1.00 | [1, 0, 0] | [0, −1, 0] | −1.00 | 1.00 | [−0, −1, 0] |
| 4 | 1.73 | 1.73 | 1.41 | 1.00 | 1.00 | [0, 1, 0] | [1, 0, 0] | −1.00 | 1.00 | [1, 0, 0] |

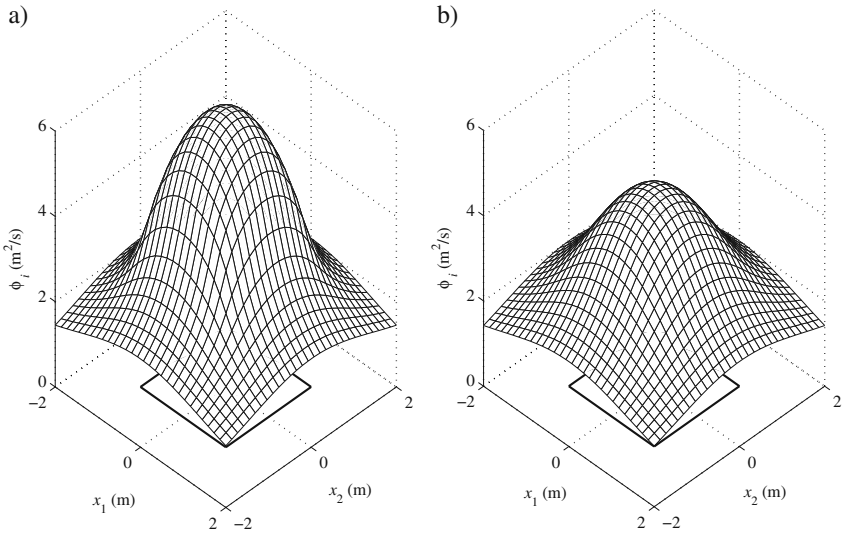


**Fig. 4.18** Potential generated by a *square polygon* on a surface parallel to the polygon at a distance (**a**) 0.1 m, and (**b**) 0.5 m

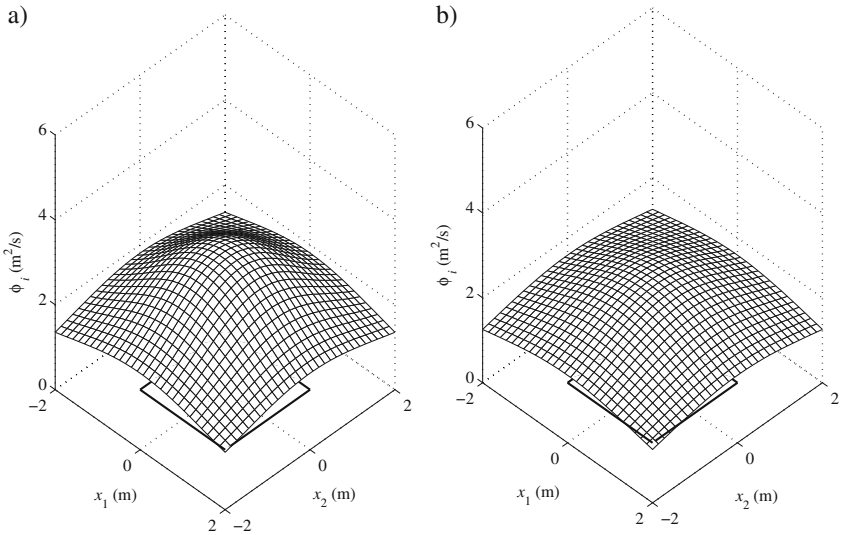


**Fig. 4.19** Potential generated by a *square polygon* on a surface parallel to the polygon at a distance (**a**) 1.0 m, and (**b**) 1.5 m

## 4.9 Three-Dimensional Robust Harmonic Potential Field

Similar to the 2D potential field for path planning, a 3D potential field for obstacle avoidance consists of a uniform potential pointing from the start to the goal of the desired path, a goal sink potential at the goal of the desired path, and several spatial

panels surrounding and representing the obstacles. To obtain an effective potential, the potential gradients at the center of the faces of the spatial panels must point outward of the obstacles volume. The strength of the spatial panels must be determined such that the conditions on the total potential gradients at the center of the panels are satisfied.

$$V_i = -\frac{\partial \phi}{\partial n_i} > 0 \qquad i = 1, .., m \tag{4.121}$$

The notation $n_i$ in Eq. (4.121) means the gradient in the $\mathbf{n}_i$ direction (unit vector perpendicular to the spatial panel) and $m$ is the number of spatial panels. Normally, after $V_i$'s are specified, the $m$ equations (4.121) are solved for the unknown panel strengths per unit area.

The other condition on the panel strengths (per unit area) is the 3D convergence condition, which is an extention of the 2D convergence condition described by Eq. (4.75).

$$-\lambda_g > \lambda_o > 0 \tag{4.122}$$

In Eq. (4.122), $\lambda_o$ is the total strength of all the spatial panels.

$$\lambda_o = \sum_{i=1}^{m} A_i \lambda_i, \tag{4.123}$$

where $A_i$ is the area of panel $i$. Note that not all the arbitrarily specified $V_i$'s guarantee the satisfaction of the condition (4.122). It is more effective to determine the range of $V_i$'s for which the resulting panel strengths per unit area satisfy Eq. (4.122) by a systematic approach rather than trying a guess to solve Eq. (4.121) for the $i$ spatial panel strengths and checking to see if Eq. (4.122) is satisfied. In the following, this systematic procedure, which is somehow similar to that of the 2D case, is presented.

The total potential field at point $C(x_1, x_2, x_3)$ is

$$\phi(x_1, x_2, x_3) = \phi_u + \phi_g + \sum_{j=1}^{m} \phi_j, \tag{4.124}$$

where $\phi_u$ (Eq. 4.99), $\phi_g$ (Eq. 4.102), and $\phi_j$ (Eq. 4.108) are the potential fields due to the uniform, goal, and panel $j$'s strengths. If $\phi'_j$ defines the potential of the spatial panel $j$ per unit strength per unit area (e.g., $\lambda_j = 1$), Eq. (4.124) can be rewritten as

$$\phi(x_1, x_2, x_3) = \phi_u + \phi_g + \lambda_j \sum_{j=1}^{m} \phi'_j. \tag{4.125}$$

Substituting Eq. (4.125) into Eq. (4.121) results in

$$V_i = -\frac{\partial \phi_u}{\partial n_i}|_{ci} - \frac{\partial \phi_g}{\partial n_i}|_{ci} - \lambda_j \sum_{j=1}^{m} \frac{\partial \phi'_j}{\partial n_i}|_{ci} \quad i = 1, \ldots, m, \tag{4.126}$$

where

$$\frac{\partial \phi_u}{\partial n_i}|_{ci} = -\hat{\mathbf{n}}_i.(a_1\hat{\mathbf{i}} + a_2\hat{\mathbf{j}} + a_3\hat{\mathbf{k}}), \tag{4.127}$$

and

$$\frac{\partial \phi_g}{\partial n_i}|_{ci} = -\hat{\mathbf{n}}_i.\frac{\lambda_g(\mathbf{x}_{ci} - \mathbf{x}_g)}{|\mathbf{x}_{ci} - \mathbf{x}_g|^3}, \tag{4.128}$$

and

$$\frac{\partial \phi'_j}{\partial n_i}|_{ci} = -\hat{\mathbf{n}}_i.(\frac{\partial \phi'_j}{\partial x_1}|_{ci}\hat{\mathbf{i}} + \frac{\partial \phi'_j}{\partial x_2}|_{ci}\hat{\mathbf{j}} + \frac{\partial \phi'_j}{\partial x_3}|_{ci}\hat{\mathbf{k}}). \tag{4.129}$$

The partial derivatives in Eq. (4.129) can be calculated by differentiating the summation term in Eq. (4.108). However, since this differentiation is cumbersome, a numerical approximation is considered.

$$\frac{\partial \phi'_j}{\partial x_1}|_{ci} \approx \frac{\phi'_j(x_{ci1} + \Delta x_{ci1}, x_{ci2}, x_{ci3}) - \phi'_j(x_{ci1}, x_{ci2}, x_{ci3})}{\Delta x_{ci1}} \tag{4.130}$$

The other derivative terms in Eq. (4.129) are derived similarly by taking an increment in the correct direction.

Now that the terms in Eq. (4.126) are introduced, it can be rearranged for the unknown panel strengths per unit area $\lambda_j$.

$$\left(\sum_{j=1}^{m} \frac{\partial \phi'_j}{\partial n_i}|_{ci}\right)\lambda_j = -V_i - \frac{\partial \phi_u}{\partial n_i}|_{ci} - \frac{\partial \phi_g}{\partial n_i}|_{ci} \quad i = 1, \ldots, m \tag{4.131}$$

Equation (4.131) is written in the matrix form

$$\mathbf{P}\Lambda = -\mathbf{V} + \mathbf{W}, \tag{4.132}$$

where

$$P_{ij} = \sum_{j=1}^{m} \frac{\partial \phi'_j}{\partial n_i}|_{ci} \quad i, j = 1, \ldots, m, \tag{4.133}$$

$$\Lambda = [\lambda_1, \lambda_2, \ldots, \lambda_m]^T, \tag{4.134}$$

$$\mathbf{V} = [V_1, V_2, \ldots, V_m]^T, \tag{4.135}$$

and

$$W_i = -\frac{\partial \phi_u}{\partial n_i}|_{ci} - \frac{\partial \phi_g}{\partial n_i}|_{ci} \quad i = 1, \ldots, m. \tag{4.136}$$

Equation (4.132) can be solved for the unknown panel strengths.

$$\Lambda = \mathbf{P}^{-1}(-\mathbf{V} + \mathbf{W}). \tag{4.137}$$

The panel strengths found from Eq. (4.137) must satisfy the convergence condition (4.122), in which the total strength of the spatial panels $\lambda_o$ appears. The relation for $\lambda_o$ (Eq. 4.123) accepts the following matrix form

$$\lambda_o = \mathbf{A}^T \Lambda, \tag{4.138}$$

where

$$\mathbf{A} = [A_1, A_2, \ldots, A_m]^T. \tag{4.139}$$

Substituting Eqs. (4.138) and (4.137) into the convergence condition (4.122) results in

$$\mathbf{A}^T \mathbf{P}^{-1}(-\mathbf{V} + \mathbf{W}) < -\lambda_g. \tag{4.140}$$

The desired outward normal velocities at the center of the spatial panels are assumed to be proportional to the panel areas. This assumption is logical. It helps the planned trajectory to be diverted at a farther distance away from the larger panels. This assumption can be formulated as

$$\mathbf{V} = a\mathbf{A}, \tag{4.141}$$

where $a$ is called the safety parameter. Substituting $\mathbf{V}$ into Eq. (4.140) yields

$$a < \frac{\lambda_g + \mathbf{A}^T \mathbf{P}^{-1} \mathbf{W}}{\mathbf{A}^T \mathbf{P}^{-1} \mathbf{A}}. \tag{4.142}$$

Equation (4.142) limits the maximum value of the proportional parameter $a$, also known as the safety factor. The following notation is adapted

$$a_{\max} = \frac{\lambda_g + \mathbf{A}^T \mathbf{P}^{-1} \mathbf{W}}{\mathbf{A}^T \mathbf{P}^{-1} \mathbf{A}}, \tag{4.143}$$

and the safety ratio is defined as

$$r_a = \frac{a}{a_{\max}}. \tag{4.144}$$

Now, the safe $\lambda_j$'s that satisfy the convergence condition (4.122) can be found from

$$\Lambda = \mathbf{P}^{-1}(-r_a a_{\max}\mathbf{A} + \mathbf{W}), \tag{4.145}$$

which is derived by substituting Eqs. (4.141) and (4.144) into Eq. (4.137).

Once the panel strengths are obtained from Eq. (4.145), the components of the velocity field are determined by taking the gradient of the potential field (4.125) as follows

$$\begin{aligned}
u_1(x_1, x_2, x_3) &= -\frac{\partial \phi_u}{\partial x_1} - \frac{\partial \phi_g}{\partial x_1} - \lambda_j \sum_{j=1}^{m} \frac{\partial \phi'_j}{\partial x_1}, \\
u_2(x_1, x_2, x_3) &= -\frac{\partial \phi_u}{\partial x_2} - \frac{\partial \phi_g}{\partial x_2} - \lambda_j \sum_{j=1}^{m} \frac{\partial \phi'_j}{\partial x_2}, \\
u_3(x_1, x_2, x_3) &= -\frac{\partial \phi_u}{\partial x_3} - \frac{\partial \phi_g}{\partial x_3} - \lambda_j \sum_{j=1}^{m} \frac{\partial \phi'_j}{\partial x_3}.
\end{aligned} \tag{4.146}$$

## 4.10 Path Planning for Aerial Robots or Hyper-Redundant Manipulators

The robust spatial potential field introduced in the previous section can be used as a tool for path planning for aerial robots with high maneuvering capabilities, such as autonomous helicopters or other rotary wing aircraft. It is assumed that the obstacles in the environment are known or can be recognized by a vision system. A start point for the aerial robot is assumed and and a destination point is defined. The procedure with which a 3D path can be planned is based on the formulation introduced in the previous section. The algorithm is summarized in the following section.

### *4.10.1 Algorithm for an Aerial Robot*

1. The uniform flow strength $U$, the goal strength $\lambda_g$, and the safety ratio $r_a$ are selected. Experience shows that a goal strength much larger than the uniform flow strength results in a more effective potential field.
2. The start position $\mathbf{x}_s = (x_{s1}, x_{s2}, x_{s3})$ and the goal position $\mathbf{x}_g = (x_{g1}, x_{g2}, x_{g3})$ are defined.
3. The 3D obstacles in the environment must be approximated by volumes that can be contained in a number of 3D polygons. The parameters of the polygons are

determined with a procedure similar to the one presented in Example 4.9. Since the direction of integration of the potential field along all the edges of the polygon must be consistant, care must be taken in defining the order of polygon vertices. A good practice is to number the vertices starting from an arbitrary vertex in the direction a of right hand rotation about the vector $\hat{\mathbf{n}}_j$, which is perpendicular to the panel and pointing outward of an obstacle volume.

4. The direction of the uniform flow is calculated such that the uniform flow points from the start point to the goal point.

$$a_1\hat{\mathbf{i}} + a_2\hat{\mathbf{j}} + a_3\hat{\mathbf{k}} = \frac{\mathbf{x}_g - \mathbf{x}_s}{|\mathbf{x}_g - \mathbf{x}_s|} \tag{4.147}$$

5. The matrix $\mathbf{P}$ is evaluated using Eq. (4.133) and $W_i$ is obtained from Eq. (4.136).
6. The parameter $a_{\max}$ is calculated using Eq. (4.143).
7. A safety ratio ($0 < r_a < 1$) is selected. Larger safety ratios yield to longer paths farther from the obstacle. Too small safety factors should be avoided. They may result in a path colliding with the obstacles, especially larger panels, because the outward normal velocity condition ($V_i > 0$) is only satisfied for the center point of each panel.
8. The strength per unit area for the $m$ panels are calculated using Eq. (4.145).
9. Equations (4.146) are used to determine the direction of the local tangent to the path. The path is generated by stepping a small constant length along the calculated local tangent. This procedure is done numerically. The path starts at $\mathbf{x}_1 = \mathbf{x}_s$. At any integration step $k$, the direction of the local tangent to the path (the direction of the instantaneous velocity) and the new position are calculated as

$$\mathbf{u}_k = \mathbf{u}(\mathbf{x}_k), \tag{4.148}$$

$$\hat{\mathbf{u}}_k = \frac{\mathbf{u}_k}{|\mathbf{u}_k|}, \tag{4.149}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \hat{\mathbf{u}}_k(\Delta s). \tag{4.150}$$

where $\Delta s$ is an arbitrary small distance. The iteration in $k$ continues until the point found for the path is closer than the defined small distance $\Delta s$.

$$||\mathbf{x}_{k+1} - \mathbf{x}_g|| \leq \Delta s \tag{4.151}$$

10. The array of points $\mathbf{x}_k$ is the 3D planned path.

In the following a path planning example is presented.

*Example 4.10.* Consider a spatial environment with two obstacles represented by a cubic and a rectangular box, which are shown in the front and top view in Figs. 4.20a and b. The start position and the goal position are $\mathbf{x}_s = (-0.5, -0.5, 1.5)$ m and $\mathbf{x}_g = (1.5, 3.5, -2.5)$ m, respectively. Using a robust 3D potential field, plan a spatial path from the start position to the goal position. Assume a safety ratio of

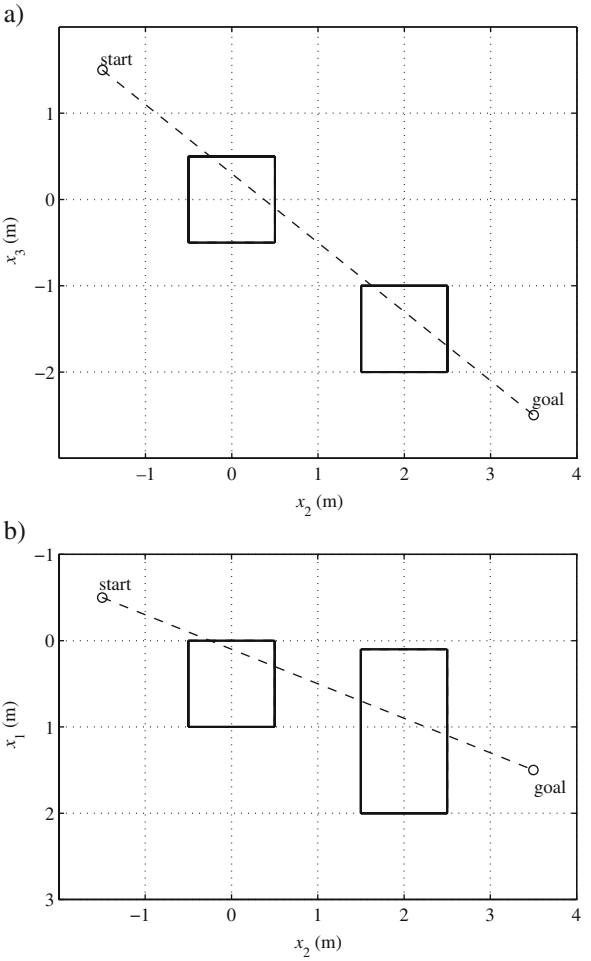


**Fig. 4.20** *One cubic* and *one rectangular box* confining obstacles with arbitrary shapes (not shown); (**a**) front view, (**b**) top view

$r_a = 0.3$, a uniform flow strength of $U = 1\ \mathrm{m}^2/\mathrm{s}$, and a goal strength of $\lambda_g = 30\ \mathrm{m}^2/\mathrm{s}$.

*Solution.* Each face of the cubic and the rectangular obstacles are considered a polygon with four sides. Therefore, a total of 12 polygons must be defined. Each of these rectangular polygons are parameterized with a procedure similar to that demonstrated in Example 4.9 for a square panel. For each polygon, the global coordinates of the four vertices are determined and a table similar to Table 4.4 is formed.

To calculate the strengths per unit area for the panels using Eq. (4.133), the components of the matrix **P** (Eq. 4.133) must be calculated. For this calculation, the coordinates of the center of each panel should be used.

**Table 4.6** Panel strength per unit area (1/s)

| $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\lambda_4$ | $\lambda_5$ | $\lambda_6$ | $\lambda_7$ | $\lambda_8$ | $\lambda_9$ | $\lambda_{10}$ | $\lambda_{11}$ | $\lambda_{12}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.028 | −0.787 | −1.009 | −0.058 | −0.536 | −0.221 | 0.357 | −1.602 | −3.021 | 0.281 | 0.512 | 0.836 |

$$\mathbf{x}_{ci} = \frac{1}{n_v} \sum_{k=1}^{n_v} \mathbf{r}_i^+, \tag{4.152}$$

where $n_v$ is the number of vertices of panel $i$. However, since the potential $\phi'_j$ generated by some of the panels at the center of other panels is singular, the modified center of panel is used. This modified center is shifted 0.1 m outward of the panel $i$.

$$\mathbf{x}_{ci} = \frac{1}{n_v} \sum_{k=1}^{n_v} \mathbf{r}_i^+ + 0.1\hat{\mathbf{n}}_i \tag{4.153}$$

The above modified centers of area for the panels are used for calculating $\mathbf{P}$ and $\mathbf{W}$ from Eqs. (4.133) and (4.136), respectively. The parameter $a_{\max} = 14.144$ is calculated using Eq. (4.143). Finally, the strength per unit areas for the panels are found by using Eq. (4.145). The results for $\lambda_j$'s are shown in Table 4.6. The total obstacle strength becomes $\lambda_o = -8.805\ \mathrm{m}^2/\mathrm{s}$, which is less than the goal's strength. Therefore, the convergence condition (4.122) is satisfied.

The planned path is shown in Fig. 4.21. The path is diverted by the obstacles. It ends at the goal position.

## 4.11 Summary

The path planning problem for single mobile robots, multiple mobile robots, and single aerial vehicles in the presence of known obstacles was addressed. Harmonic potential functions and the panel method were adapted to solve the obstacle avoidance problem. This chapter also introduced a complement to the traditional panel method to generate a more effective harmonic potential field for obstacle avoidance, which rendered the traditional harmonic potential method more suitable for dynamically changing environments. An algorithm for a group of mobile robots working in an environment with stationary and moving obstacles was developed. In this algorithm, robots move from one position to another without maintaining a formation during the motion. Every robot considered the other robots of the group as moving obstacles. Hence, the physical dimensions of the robots were considered in path planning. The path of each robot was planned based on the changing position of the other robots and moving obstacles, and the position of stationary obstacles. Finally, the effectiveness of the scheme was shown through examples and simulations.

Although the obstacles are assumed to be known in this chapter, the methods introduced here are not limited to the cases with known obstacles. In the case of unknown obstacles, a vision system along with an obstacle identification algorithm

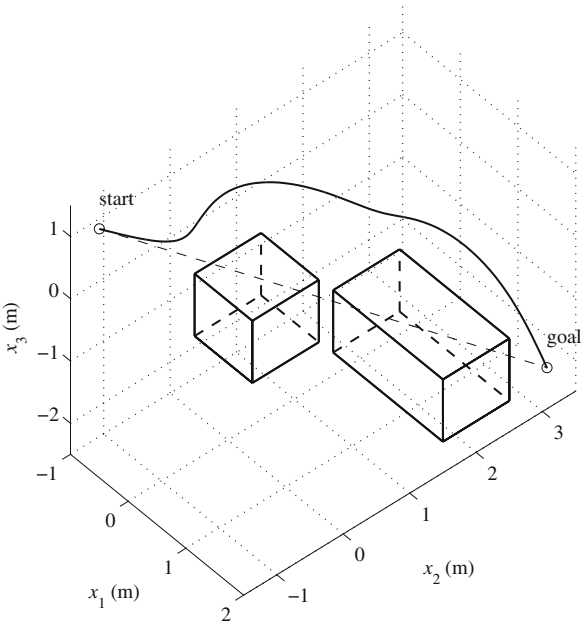


**Fig. 4.21** The planned path for an aerial vehicle that avoids *one cubic* and *one rectangular box* obstacles

can provide the geometry of the obstacles to a planner algorithm written based on the methods presented in this chapter. When the obstacle identification algorithm can detect new obstacles as they enter the range of the vision system while the vehicle moves, the potential field is recalculated and a new path is generated for the new situation.

The methods proposed in this chapter are simple, efficient, accurate, and scalable, and are suited for practical situations as shown by the numerical simulations.

## Problems

**Problem 4.1.** Show that the 2D goal potential defined in Eq. (4.14) is a harmonic function.

**Problem 4.2.** Show that the 2D uniform flow potential defined in Eq. (4.16) is a harmonic function.

**Problem 4.3.** Show that the potential of a line obstacle defined in Eq. (4.19) is a harmonic function in a 2D space.

**Problem 4.4.** Derive the contribution of a vertical line panel (Fig. 4.3) to the normal velocity at its own center point using Eq. (4.22).

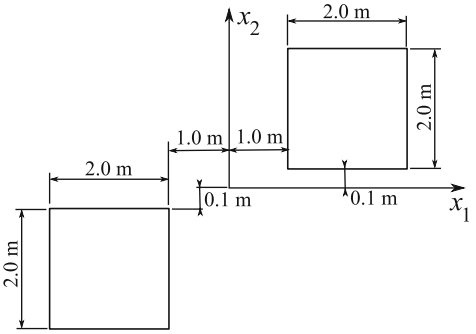


**Fig. 4.22** Two square obstacles

**Problem 4.5.** Using Eqs. (4.16) and (4.19), plot the potential of the superposition of a uniform flow with a direction of 45° with respect to the $x_1$-axis and with a strength of $U = -1$ and a 1-m-long source line segment ($L = 0.5$ m) as shown in Fig. 4.3 with a strength per unit length of $\lambda = -1$ in an square area of 2 by 2 m around the origin.

**Problem 4.6.** Consider a mobile robot that has to work in an environment containing obstacles as shown in Fig. 4.22. The robot has to reach the goal position $\mathbf{x}_g = (x_{g1}, x_{g2}) = (0.0, 5.0)$ m from different start points listed in Table 4.7.

(a) Determine the correct line obstacle parameters for the the eight line segments in the obstacle field according to the standard notation in Fig. 4.9. Make sure that the angles $\theta_i$ for the panels are defined such that the normal vector to the line obstacles are outward for all the panels.
(b) Determine the direction of the uniform flow for each start point.
(c) Form a robust harmonic potential field for each start point and find the maximum safety factor, $a_{\max}$.
(d) Derive safe paths for the robot for each start point using two different safety ratios of $r_a = 0.5$ and $r_a = 0.99$.
(e) Plot the paths for the two safety ratios for each start point and compare and discuss the resulting plots.

**Problem 4.7.** Assume two identical robots, with 0.25 by 0.25 m dimensions, working simultaneously in an environment with obstacles as shown in Fig. 4.22. The two robots are initially at the start points $\mathbf{x}_{s1} = [-2.0, -5.0]^T$ m and $\mathbf{x}_{s2} =$

**Table 4.7** List of desired start points (in meters)

| | |
|---|---|
| $\mathbf{x}_{s1} = [-4.0, -5.0]$ | $\mathbf{x}_{s2} = [-3.0, -5.0]$ |
| $\mathbf{x}_{s3} = [-2.0, -5.0]$ | $\mathbf{x}_{s4} = [-1.0, -5.0]$ |
| $\mathbf{x}_{s5} = [0.0, -5.0]$ | $\mathbf{x}_{s6} = [1.0, -5.0]$ |
| $\mathbf{x}_{s7} = [2.0, -5.0]$ | $\mathbf{x}_{s8} = [3.0, -5.0]$ |
| $\mathbf{x}_{s9} = [4.0, -5.0]$ | |

$[2.0, -5.0]^T$ m. The goal points for the two robots are $\mathbf{x}_{g1} = [2.0, 5.0]^T$ m and $\mathbf{x}_{g2} = [-2.0, 5.0]^T$ m. The maximum velocity of the robots is 0.25 m/s. Using the algorithm presented in Section 4.6.1, plan the trajectory for the robots such that they do not collide with each other or the obstacles.

**Problem 4.8.** Show that the 3D uniform flow potential defined in Eq. (4.99) is a harmonic function.

**Problem 4.9.** Show that the 3D goal potential defined in Eq. (4.102) is a harmonic function.

**Problem 4.10.** Show that the 2D potential defined in Eq. (4.106) for a spatial flat polygon is a harmonic function.

**Problem 4.11.** Consider an aerial robot that has to work in an environment containing a cubic obstacle with 1-m sides centered at the origin of the coordinate system. The aerial robot has to reach the goal position $\mathbf{x}_g = (3, 0, 0)$ m from its starting point at $\mathbf{x}_s = (-3, 0.1, 0.1)$ m.

(a) Consider the sides of the cube as six independent squares. Determine the correct parameters for the edges of the squares using a procedure similar to that introduced in Example 4.9.
(b) Determine the direction of the 3D uniform flow.
(c) Form a robust harmonic potential field for each start point and find the maximum safety factor, $a_{\max}$.
(d) Derive safe paths for the robot for each start point using two different safety ratios of $r_a = 0.5$ and $r_a = 0.99$.
(e) Plot the paths for the two safety ratios for each start point and compare and discuss the resulting plots.

# Chapter 5
# Control of Manipulators

## 5.1 Introduction

For general multi-input nonlinear systems, including robotic manipulators, feedback control and, especially, robustness issues are still research topics.

In this chapter, we discuss the application of different control methods for controlling manipulators and present the advantages and disadvantages of each method. We try to explain the physical interpretation of the mathematical methods used for controller development.

In the following, we first consider simpler control methods to build the ground for physical interpretations and justify the motivation for using more advanced methods. Later in this chapter, we use the Lyapunov theory for designing advanced robust controllers for manipulators. The Lyapunov theory itself is based on physical understanding of mechanical systems.

## 5.2 Evolving Control Requirements

Today's robotic manipulators are expected to work faster and more accurately. Most of the time, to accomplish a certain job, a manipulator's end-effector must follow a trajectory precisely. This in turn requires robust trajectory-tracking for the joints variables. Since manipulators have non linear dynamics, their controller design presents a challenging problem.

The traditional linear control approaches used to lead to a controller with an acceptable performance for the past generation of manipulators. The reason was that those manipulators were highly geared, which reduced the strong interactive dynamic effects between links. The typically used gear ratios of about 100 caused the torque transmission ratios of 1/100 between the links. In the next generation manipulators, these gears had to be eliminated to achieve greater position and force control accuracy. The joint drives were replaced by gear-free direct-drive motors, which reduced friction and avoided gear backlash all together. This came at the cost of high transmission of the driving torque between the links, causing the higher non linear dynamic interaction between the links. In this situation, explicit account of

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_5

the nonlinear dynamic effects became critical in order to exploit the full dynamic potential of the new high-performance manipulator arms.

Since the non linear dynamic of a manipulator plays an important role in the controller design, the first step of any successful controller design is formulating the mathematical dynamic model of the manipulator. In the following, the dynamic modeling is discussed.

## 5.3 General Dynamic Model

### *5.3.1 Standard Second-Order Form*

Consider, for example, a Prismatic-Revolute (PR) planar manipulator (Fig. 5.1). The joint positions of the manipulator are organized in a 2 by 1 vector $\mathbf{q}$. The actuator input forces/torques applied at the manipulator joints are set in a 2 by 1 vector $\boldsymbol{\tau}$. The general form of the nonlinear dynamic equations of this manipulator can be presented as

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau}. \tag{5.1}$$

In the above equation, $\mathbf{H}(\mathbf{q})$ is the symmetric positive-definite 2 by 2 manipulator inertia matrix. $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ contains the terms representing the centripetal and Coriolis torques. It can be easily verified that $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is a $2 \times 2$ matrix. $\mathbf{G}(\mathbf{q})$ is the 2 by 1 vector of gravitational torques. Now, a controller must be designed that determines the required actuator inputs such that the joint positions $\mathbf{q}$ and velocities $\dot{\mathbf{q}}$ follow a desired trajectory. The vector $\mathbf{q}$ of joint angles and the vector $\dot{\mathbf{q}}$ of joint velocities are called the states of the manipulator.

Note that the inertia matrix $\mathbf{H}$ is a function of the joint position vector $\mathbf{q}$. This mathematical dependency emphasizes the physical fact that a manipulator arm has a different inertia when it is folded compared to when it is extended. The centripetal torque components are a function of the square of each joint velocity. Also, the products of velocities at two different joints appear in the Coriolis torque terms.

The kinetic energy of the manipulator can be found based on the mass matrix as

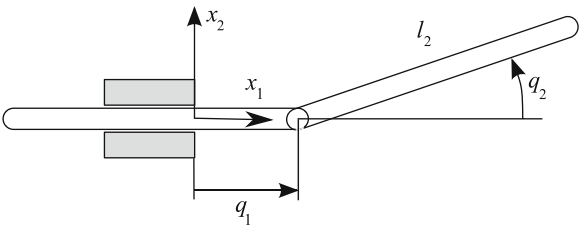


**Fig. 5.1** A PR manipulator

$$\frac{1}{2}\dot{\mathbf{q}}^T \mathbf{H}(\mathbf{q})\dot{\mathbf{q}}. \tag{5.2}$$

The above equation can be seen as the vectorial version of the familiar one-dimensional form of kinetic energy $\frac{1}{2}mv^2$. Note that the kinetic energy must be strictly positive independent of the value of $\mathbf{q}$, if the joint velocity $\dot{\mathbf{q}}$ is nonzero. This physical fact translates to a mathematical requirement for the mass matrix $\mathbf{H}(\mathbf{q})$, i.e., the mass matrix must be strictly positive-definite. In other words, the eigenvalues of the mass matrix must be uniformly positive for any value of $\mathbf{q}$. If there existed a position $\mathbf{q}$ in the workspace where the inertia matrix has a zero eigenvalue, the robot arm could have moved with a nonzero velocity but with zero kinetic energy, which is physically impossible. Thus, $\mathbf{H}(\mathbf{q})$ is indeed uniformly positive-definite.

*Example 5.1.* PR manipulator shown in Fig. 5.1. Using the Lagrange method, derive the dynamic equations of motion for the manipulator. Write the equations of motion in the standard form (5.1). Neglect the moment of inertia of the second link.

*Solution.* To use the Lagrange method for deriving the equations of motion, first, we write the Lagrangian $L = T - V$ for the manipulator, where $T$ is the total kinetic energy and $V$ is the total potential energy of the manipulator at an arbitrary state:

$$L = \frac{1}{2}m_1\dot{q}_1^2 + \frac{1}{2}m_2((\dot{q}_1 - l_2\dot{q}_2\sin q_2)^2 + (l_2\dot{q}_2\cos q_2)^2) - m_2 g l_2 \sin q_2. \tag{5.3}$$

Once the Lagrangian is determined, one can derive the equations of motion by applying the following general form of the Lagrange equations of motion:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \tau_i, \quad i = 1, \ldots, n, \tag{5.4}$$

where $n$ is the number of joints. Note that right hand side of the above form has been simplified for manipulators and it is only correct when the the driving force/torque $\tau_i$ is directly acting on the joint $q_i$, allowing us to write the work done by $\tau_i$ as $\tau_i q_i$. For other situations, a more general form must be used for the right hand side of Eq. (5.4).

Substituting Eq. (5.3) into (5.4) results in the following dynamic equations of motion for the PR manipulator:

$$(m_1 + m_2)\ddot{q}_1 - m_2 l_2 \ddot{q}_2 \sin q_2 - m_2 l_2 \dot{q}_2^2 \cos q_2 = \tau_1, \tag{5.5}$$

$$m_2 l_2^2 \ddot{q}_2 - m_2 l_2 \ddot{q}_1 \sin q_2 + m_2 g l_2 \cos q_2 = \tau_2. \tag{5.6}$$

These equations can be written in the standard form (5.1) as

$$\mathbf{H}(\mathbf{q})\begin{bmatrix}\ddot{q}_1 \\ \ddot{q}_2\end{bmatrix} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\begin{bmatrix}\dot{q}_1 \\ \dot{q}_2\end{bmatrix} + \mathbf{G}(\mathbf{q}) = \begin{bmatrix}\tau_1 \\ \tau_2\end{bmatrix}, \tag{5.7}$$

where

$$\mathbf{H}(\mathbf{q}) = \begin{bmatrix} m_1 + m_2 & -m_2 l_2 \sin q_2 \\ -m_2 l_2 \sin q_2 & m_2 l_2^2 \end{bmatrix}, \tag{5.8}$$

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} 0 & -m_2 l_2 \dot{q}_2 \cos q_2 \\ 0 & 0 \end{bmatrix}, \tag{5.9}$$

$$\mathbf{G}(\mathbf{q}) = \begin{bmatrix} 0 \\ m_2 g l_2 \cos q_2 \end{bmatrix}. \tag{5.10}$$

This completes the solution to this example.

### 5.3.2 Standard First-Order Form

Although the second-order standard form is sufficient for controller design derivations, there is always a need for a first-order form of the dynamic model for simulation purposes. The dynamic state of the manipulator not only depends on the joint positions, as a representative of the geometrical configuration of the manipulator, but also depends on the joint speeds, which reflect the motion of the manipulator. Therefore, to uniquely define the dynamic state of the manipulator, one should group the joint positions and speeds in a single vector, known as the state vector of the manipulator:

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix}. \tag{5.11}$$

With the above definition, the general form of the first-order equations of motion for a manipulator can be written as

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{q}} \\ \mathbf{H}^{-1}(\boldsymbol{\tau} - \mathbf{C}\dot{\mathbf{q}} - \mathbf{G}) \end{bmatrix}. \tag{5.12}$$

*Example 5.2.* Consider the PR manipulator shown in Fig. 5.1. Using the Lagrange method, derive the dynamic equations of motion for the manipulator. Write the first-oder form of the equations of motion.

*Solution.* The state vector for the PR manipulator is assembled as

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}. \tag{5.13}$$

The first-order equations are derived by differentiating the above definition of the state vector and substituting for $\ddot{\mathbf{q}}$ from the second-order form of the dynamic model in the result.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \ddot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \mathbf{H}^{-1}(x_1, x_2)(\boldsymbol{\tau} - \mathbf{C}(\mathbf{x})\dot{\mathbf{q}}) - \mathbf{G}(x_1, x_2) \end{bmatrix}, \tag{5.14}$$

where

$$\mathbf{H}(x_1, x_2) = \begin{bmatrix} m_1 + m_2 & -m_2 l_2 \sin x_2 \\ -m_2 l_2 \sin x_2 & m_2 l_2^2 \end{bmatrix}, \tag{5.15}$$

$$\mathbf{C}(\mathbf{x}) = \begin{bmatrix} 0 & -m_2 l_2 x_4 \cos x_2 \\ 0 & 0 \end{bmatrix}, \tag{5.16}$$

$$\mathbf{G}(x_1, x_2) = \begin{bmatrix} 0 \\ m_2 g l_2 \cos x_2 \end{bmatrix}. \tag{5.17}$$

This completes the solution to this example.

## 5.4 Position Control

In the position control problem, the manipulator task is to reach a final desired joint position vector, specified by a constant vector $\mathbf{q}^d$. For position control, we assume that each joint is controlled independently. The independent controller is assumed to be a joint proportional-derivative (PD) controller. The PD feedback control law selects each actuator input based on the local measurements of position errors $\tilde{q}_j = q_j - q_j^d$ and joint velocities $\dot{q}_j$ $(j = 1, 2)$. The PD control law is formulated as

$$\tau_j = -k_{Pj}\tilde{q}_j - k_{Dj}\dot{q}_j, \tag{5.18}$$

where $k_{Pj}$ and $k_{Dj}$ are strictly positive constants. Note that this control law can be physically interpreted as a spring-damper system mounted on the joint. The first term in the control law (5.18) acts like a spring with constant $k_{Pj}$ with the desired rest position of $q_j^d$, where the spring force vanishes. The second term of this control law acts like a damper that resists the joint velocities, causing the motion to eventually diminish at the equilibrium point of the spring. The resulting passive physical system would simply show damped oscillations toward the rest position $\mathbf{q}^d$.

The behavior of a spring-damper system attached to a mass is easier to understand when the energy of the system is studied. Consider the manipulator at an initial position different than the desired position. At such a position, $\tilde{q}_j$ is not zero. Thus, there is some energy stored in the virtual spring with constant $k_{Pj}$. If the damper with constant $k_{Dj}$ can drain the system out of this energy, the system will eventually come to a stop at the spring equilibrium point, which happens to represent the desired joint position.

In order to formulate the above discussion, we rewrite the system's dynamics in a Hamiltonian form, which shows the energy transfer:

$$\frac{1}{2}\frac{d}{dt}(\dot{\mathbf{q}}^T\mathbf{H}\dot{\mathbf{q}}) = \dot{\mathbf{q}}^T\boldsymbol{\tau}. \tag{5.19}$$

This equation implies that the rate of the kinetic energy of the manipulator (the left-hand side of Eq. 5.19) is equal to the work of the external forces/torques acting on the manipulator (the right-hand side of Eq. 5.19). Note that taking the derivative in the left-hand side term will of course result in the Coriolis and centripetal terms appearing in Eq. (5.1). This rate of energy and work balance equation can be used to derive stability and convergence proof for the PD controller. Let us continue with writing the vectorial version of Eq. (5.18).

$$\boldsymbol{\tau} = -\mathbf{k}_P\tilde{\mathbf{q}} - \mathbf{k}_D\dot{\mathbf{q}}, \tag{5.20}$$

where $\mathbf{k}_P$ and $\mathbf{k}_D$ are constant symmetric positive-definite matrices. One valid choice for these matrices could be forming diagonal matrices with $k_{Pj}$'s and $k_{Dj}$'s of Eq. (5.18) as diagonal components. If the control law (5.20) were implemented using physical springs and dampers, one could write the total mechanical energy of the manipulator and spring-dampers as

$$V = \frac{1}{2}(\dot{\mathbf{q}}^T\mathbf{H}\dot{\mathbf{q}} + \tilde{\mathbf{q}}^T\mathbf{k}_P\tilde{\mathbf{q}}). \tag{5.21}$$

We can use this virtual mechanical energy $V$ as a Lyapunov function and analyze the closed-loop behavior of the controlled system. Given Eq. (5.19), the time-derivative of $V$ can be written as

$$\dot{V} = \dot{\mathbf{q}}^T(\boldsymbol{\tau} + \mathbf{k}_P\tilde{\mathbf{q}}). \tag{5.22}$$

One can use the control law (5.20), and simplify the above equation as

$$\dot{V} = -\dot{\mathbf{q}}^T\mathbf{k}_D\dot{\mathbf{q}} \leq 0. \tag{5.23}$$

The above equation implies that the rate of the total energy of the manipulator and the virtual spring-dampers as the controller, represented by $\dot{V}$, is negative. This means that $V$ decreases as time is passed until $\dot{\mathbf{q}}$ becomes zero and the manipulator stops at an equilibrium position, which verifies the stability of such a controller. However, we have to make sure that the equilibrium position is actually the desired position and the manipulator does not stop at a position other than the desired position, where $V$ equal 0 while $\mathbf{q}$ does not equal $\mathbf{q}^d$.

To further discuss this concept, let us assume a planar manipulator that works in a horizontal plane, which imples that the gravitational term $\mathbf{G}(\mathbf{q})$ in Eq. (5.1) vanishes. Since $\dot{V} = 0$ implies that $\dot{\mathbf{q}} = \mathbf{0}$, the dynamics at the equilibrium reduces to

$$\ddot{\mathbf{q}} = -\mathbf{H}^{-1}\mathbf{k}_P\tilde{\mathbf{q}}. \tag{5.24}$$

The above equation implies that $\dot{V}$ is identically 0 only if $\tilde{\mathbf{q}}$ equals 0, since at the equilibrium $\ddot{\mathbf{q}}$ is also zero. This means that the equilibrium point is in fact $\tilde{\mathbf{q}} = 0$ and the system does converge to the desired state.

On the other hand, if the manipulator is working in the vertical plane, $\mathbf{G}(\mathbf{q})$ is not zero. In this case, the dynamics at the equilibrium reduces to

$$\ddot{\mathbf{q}} = \mathbf{H}^{-1}(\mathbf{G}(\mathbf{q}) - \mathbf{k}_P \tilde{\mathbf{q}}). \tag{5.25}$$

This implies that the equilibrium point (or the steady-state error) is

$$\tilde{\mathbf{q}}_{ss} = -\mathbf{k}_P^{-1}\mathbf{G}(\mathbf{q}). \tag{5.26}$$

In this case, depending on how large the components of $\mathbf{k}_P$ are, there will be some offset with the desired position at the equilibrium point. The following example illustrates these concepts further.

*Example 5.3.* Consider the PR manipulator of Example 5.1. Design a PD controller that can bring the manipulator to any given desired position. Assume the following for the properties of the manipulator.

$$m_1 = 1 \text{ kg}, \quad m_2 = 1 \text{ kg}, \quad l_2 = 1 \text{ m}. \tag{5.27}$$

Simulate the manipulator's response under the control if it starts from the initial posture $\mathbf{q} = [0.0 \text{ m}, -\pi/2 \text{ rad}]^T$ and is trying to reach a desired posture of $\mathbf{q}^d = [0.1 \text{ m}, -\pi/3 \text{ rad}]^T$.

*Solution.* The PD feedback control law (5.20) is used here to derive the controller.

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = -\begin{bmatrix} k_{P1} & 0 \\ 0 & k_{P2} \end{bmatrix} \begin{bmatrix} q_1 - q_1^d \\ q_2 - q_2^d \end{bmatrix} - \begin{bmatrix} k_{D1} & 0 \\ 0 & k_{D2} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}, \tag{5.28}$$

where $k_{Pi}$ and $k_{Di}$ $(i = 1, 2)$ are selected as

$$k_{Pi} = 300, \quad k_{Di} = 100, \quad i = 1, 2. \tag{5.29}$$

These numbers are selected by trial and error such that the time required for the joints to reach their desired values are small, while the driving force and torque are reasonable, the response is not oscillatory and the final errors are acceptable.

To simulate the motion of the joints, one substitutes the control law (5.28) in the first-order equations of motion (5.14) for the manipulator and the set of differential equations are numerically integrated. The results of the numerical simulation are discussed in the following.

Figure 5.2 shows the trajectory of the joint variables. Recall the analogy of the control law (5.28) to a set of springs and dampers controlling the joint positions. Since the weight of the manipulator does not disturb the position of the first joint,

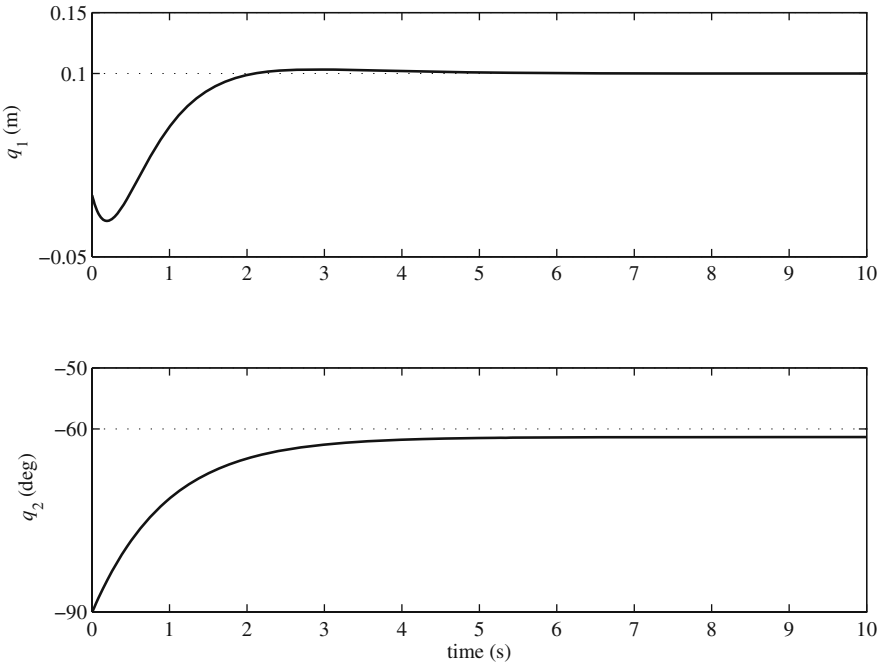


**Fig. 5.2** Trajectory of the joints for the PR manipulator under PD position control

the first proportional term of the controller (the spring $k_{P1}$) does not have to compensate any weights, and the controller has been able to bring that joint to the desired position $q_1^d = 0.1$ m. This is not true for the final position of the second joint, which has stopped at an angle less than the desired angle of $-\pi/3$ (or 60°). A steady-state error (offset) in the position of the second joint exists. This offset, $\tilde{q}_{2ss}$, must exist so that the steady state value of the second joint torque, $\tau_{2ss}$, can balance the weight of the second link. That is,

$$-k_{P2}\tilde{q}_{2ss} = \tau_{2ss}, \tag{5.30}$$

where

$$\tau_{2ss} \approx m_2 g l_2 cos\frac{\pi}{3} = 4.905 \text{ Nm}. \tag{5.31}$$

Note that the approximate sign is used because $q_2$ is not exactly $\pi/3$. One can calculate the approximate steady-state error as

$$\tilde{q}_{2ss} = -\frac{\tau_{2ss}}{k_{P2}} \approx -\frac{4.905}{300} = -0.01635 \text{ rad} \approx -0.94 \text{ deg}. \tag{5.32}$$

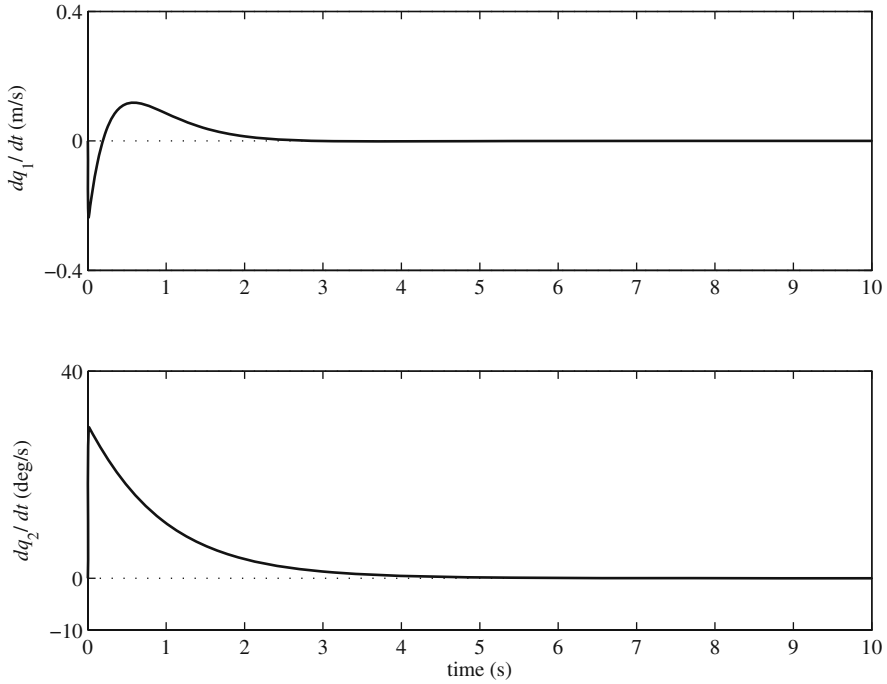


**Fig. 5.3** Joints speeds for the PR manipulator under PD position control

This offset is easily seen in Fig. 5.2. Note that this offset can be eliminated by adding a weighted integral of joint errors to the control law. Figure 5.3 shows that the joint speeds vanish at the steady state, indicating that the manipulator remains at an equilibrium state. Figure 5.4 shows the time response of the joints' force and moment. These responses confirm that the first joint can stay at its equilibrium position without a steady-state force, while the second joint needs a constant torque at the steady state to balance the weight of the second joint. The amount of this steady-state force matches the value that was previously calculated in this example.

## 5.5 Trajectory-Tracking Control

In many practical cases, position control is not enough and the manipulator's joints actually have to follow a time dependent desired trajectory to generate a specified time dependent path at the end-effector. Examples of these practical situations are when the end-effector has to move on a prescribed path with a prescribed velocity, e.g., when welding 3D parts, avoiding obstacles, spraying paint, etc. In these situations, the problem is how close the manipulator can track a given trajectory. In other words, the tracking accuracy becomes important during the whole period of the motion.

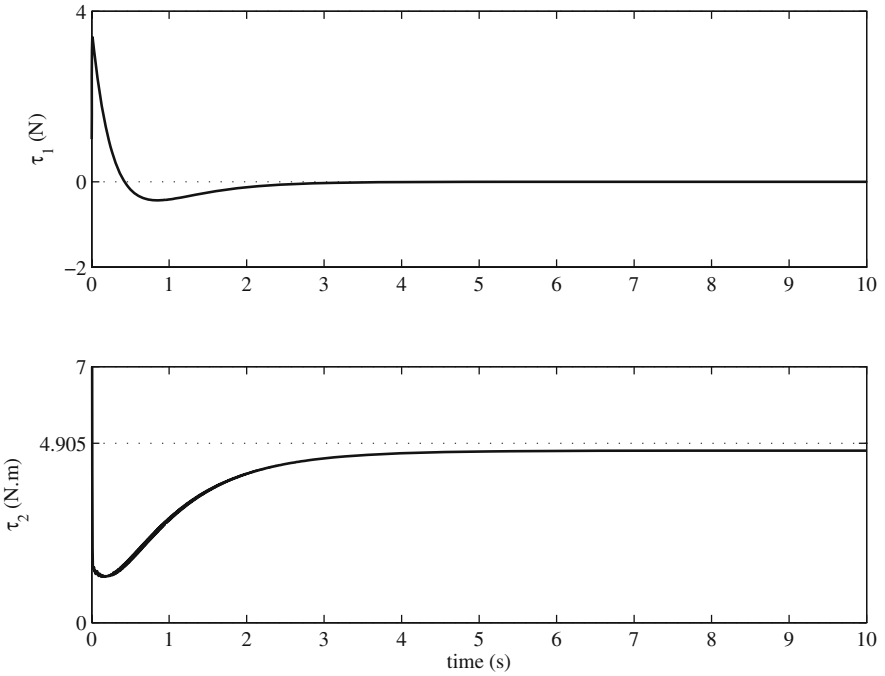


**Fig. 5.4** Driving force and torque for the PR manipulator under PD position control

The dynamic demands of trajectory tracking cannot be met by the simple PD controller. Consider the physical analogy of the PD controller with a spring-damper system. In such a system, setting the desired point could be seen as changing the position of the base of the spring, in which case, the spring would adjust the system with the new base position. We can see that by specifying a time dependent motion for the spring base, excessive vibrations may happen, which is not desireable. Therefore, the simple PD cannot handle the dynamic demands of trajectory tracking effectively. More advanced controllers are needed for trajectory tracking.

In the following, first, we present the feedback linearization method for controller design. We then discuss the sliding mode control method for a robust controller design, for which external disturbances have a minimal effect on the performance of the manipulator. All the formulations presented here are applicable to manipulators with an arbitrary number of DOFs and having revolute or translational joints (or both).

### *5.5.1 Feedback Linearization*

In the feedback linearization method, a different control input representation is defined for the system such that the dynamic equation seems similar to that of a linear

system. Then, a controller is designed for the linear system using any classical control method. The control law, then, is derived by transforming the equations for the input back to the initial input representation.

We start with the general dynamic model of a manipulator presented in Eq. (5.1). Here, let us define the new control input $\mathbf{v}$ as

$$\mathbf{v} = \mathbf{H}^{-1}(\mathbf{q})[\boldsymbol{\tau} - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{G}(\mathbf{q})]. \tag{5.33}$$

Substituting this new control input in the general equation of motion (5.1) results in the following simple system, which looks similar to a linear system:

$$\ddot{\mathbf{q}} = \mathbf{v}. \tag{5.34}$$

Now, we need to design a control law for this simple system. If we define $\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}^d$ as the tracking error, we can show that the control law

$$\mathbf{v} = \ddot{\mathbf{q}}^d - 2\lambda\dot{\tilde{\mathbf{q}}} - \lambda^2\tilde{\mathbf{q}} \tag{5.35}$$

with $\lambda > 0$, leads to an exponentially stable closed-loop dynamics. The closed-loop error dynamics can be derived by substituting Eq. (5.35) into Eq. (5.34), which results in

$$\ddot{\tilde{\mathbf{q}}} + 2\lambda\dot{\tilde{\mathbf{q}}} + \lambda^2\tilde{\mathbf{q}} = \mathbf{0}. \tag{5.36}$$

After the control law is designed for the new input of the linear-looking system (5.34), we can convert it to the initial control input by using Eq. (5.33):

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})\mathbf{v} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}). \tag{5.37}$$

Expression (5.37) is known as the "computed torque" in the robotics literature. This assumes, of course, that the dynamic model, used in Eq. (5.37), is exact. If there are uncertainties in determining the dynamic parameters of the manipulator, or the dynamic parameters of the manipulator changes (e.g. due to handling different external loads), the performance of the controller is adversely affected.

*Example 5.4.* Consider the PR manipulator of Example 5.1. Derive a tracking control law for the manipulator using the feedback linearization method. Investigate the performance of the manipulator for an uncertain parameter $m_2$. This uncertainty represents the variations in loads carried by the second link. Assume $\pm 20\%$ bounds for uncertainty in $m_2$. For the performance investigations, use the following scenarios.

(a) Position control performance: The manipulator is at rest at an initial posture of (0.0 m, $-\pi/2$ rad). It should reach the desired posture of (0.1 m, $-\pi/3$ rad).
(b) Trajectory-tracking performance: The manipulator is at rest at an initial posture of (0.1 m, $-\pi/36$ rad). Both the manipulator's joints must follow the following desired time dependent trajectory.

$$q_i^d(t) = \begin{cases} 0.2t & \text{if } t \leq 3.333 \\ 0.667 & \text{if } 3.333 < t \leq 6.667 \\ 0.667 - 0.2(t - 6.667) & \text{if } 6.667 < t \leq 10.000 \\ 0.0 & \text{if } t > 10.00 \end{cases} \qquad i = 1, 2, \tag{5.38}$$

where $q_1^d(t)$ is in meters and $q_2^d(t)$ is in radians.

*Solution.* The dynamic model of the PR manipulator that was derived in Example 5.1 is used here. When the dynamic model is at hand, the feedback control law can be determined by simply substituting Eq. (5.35) into Eq. (5.37):

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})(\ddot{\mathbf{q}}^d - 2\lambda\dot{\tilde{\mathbf{q}}} - \lambda^2\tilde{\mathbf{q}}) + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}). \tag{5.39}$$

This control law is applied to the first-order form of the dynamic equation for the PR manipulator, Eq. (5.14), for simulations.

(a) For the position control scenario, the desired joint positions are

$$\mathbf{q}^d = \begin{bmatrix} 0.1 \\ -\frac{\pi}{3} \end{bmatrix} \begin{matrix} \text{m} \\ \text{rad} \end{matrix}. \tag{5.40}$$

The desired joint positions, $q_i^d$, are not a function of time. Therefore, one can write

$$\dot{\mathbf{q}}^d = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{matrix} \text{m/s} \\ \text{rad/s} \end{matrix} \quad \ddot{\mathbf{q}}^d = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \begin{matrix} \text{m/s}^2 \\ \text{rad/s}^2 \end{matrix}. \tag{5.41}$$

These information are used to calculate $\ddot{\mathbf{q}}^d$, $\dot{\tilde{\mathbf{q}}}$, and $\tilde{\mathbf{q}}$ used in Eq. (5.39).

To simulate the performance of a controller under the uncertainty of the dynamic parameters, one must "always" use the nominal dynamic parameters (i.e., $m_2 = 1.0\,\text{kg}$) for calculating the control law (5.39), while using an assumed uncertain parameter (e.g., $m_2 = 1.2\,\text{kg}$) in the dynamic model (5.14). To investigate the performance of the position control via feedback linearization for the PR manipulator with $\pm 20\%$ uncertainty in parameter $m_2$, one must consider at least three cases to simulate.

One simulation must be done with $m_2$ set to 1.0 kg (i.e., using the nominal parameters) in the dynamic model, representing the nominal situation. The result of this simulation case must be satisfactory before the uncertainty is introduced to the dynamic model. Therefore, any controller gain that must be set by trial and error for a good performance must be determined here. For this example, the controller gain $\lambda = 8$ in Eq. (5.39) showed a satisfactory performance for the nominal case.

Once the controller gains are tuned for a satisfactory performance in the nominal case, two simulations must be done, each for the uncertain parameter set at the bounds of the uncertainty. In this example, $m_2$ is set at 0.8 and 1.2 kg for the two simulations.

The trajectory of the joints for the three simulated cases are shown in Fig. 5.5. As can be seen in the figure, the controller is able to accurately position the joints for the nominal case. There are no steady-state errors in this case, which indicates a better performance compared to that of the PD position controller of the previous section. However, this is only true when the exact dynamic parameters of the manipulator are known. The two simulations for the bounds of the uncertain $m_2$ show a steady-state error in the joint positions. These errors may or may not be acceptable depending on the application. Higher values for the controller gain $\lambda$ can result in a better performance, however, care must be taken in increasing $\lambda$ such that the driving forces/torques do not exceed the limits of the joint motors. Figures 5.6 and 5.7 show the joint speeds and the driving force and torque for the PR manipulator in three simulation cases. The difference in joint speeds for the three cases are negligible. This is because the convergence rate of the joint positions mostly depends on the value of $\lambda$, which remains the same for all the cases. The driving forces for the three cases are very close, while the driving torque shows a rather larger difference between the three cases. The

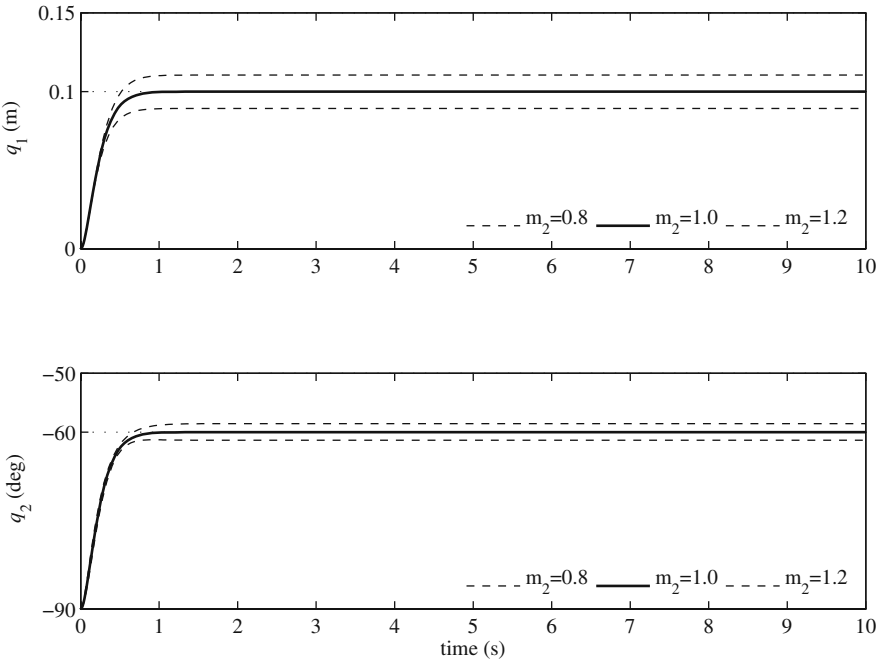


**Fig. 5.5** Trajectory of the joints for the PR manipulator under feedback linearized control

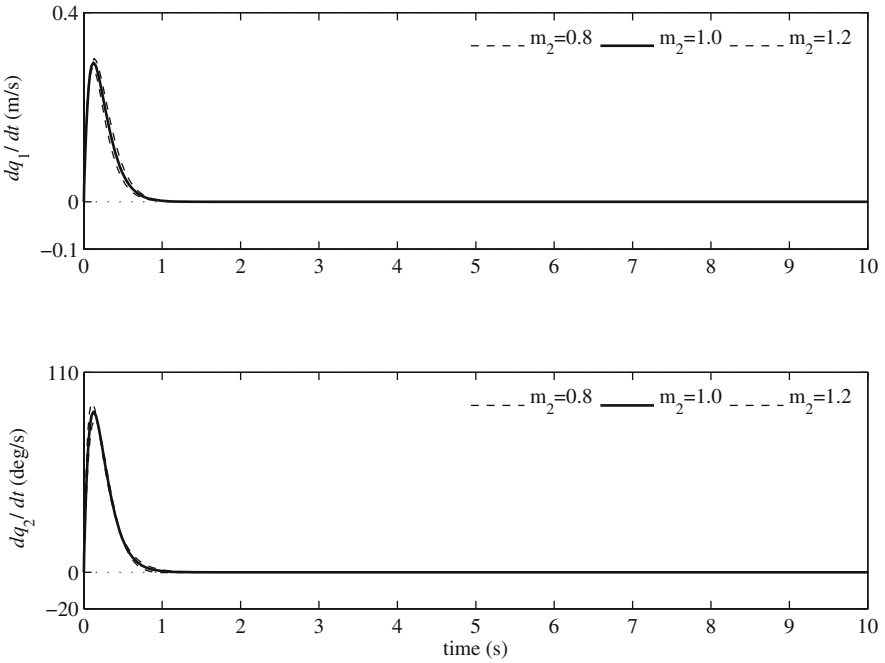


**Fig. 5.6** Joint speeds for the PR manipulator under feedback linearized control

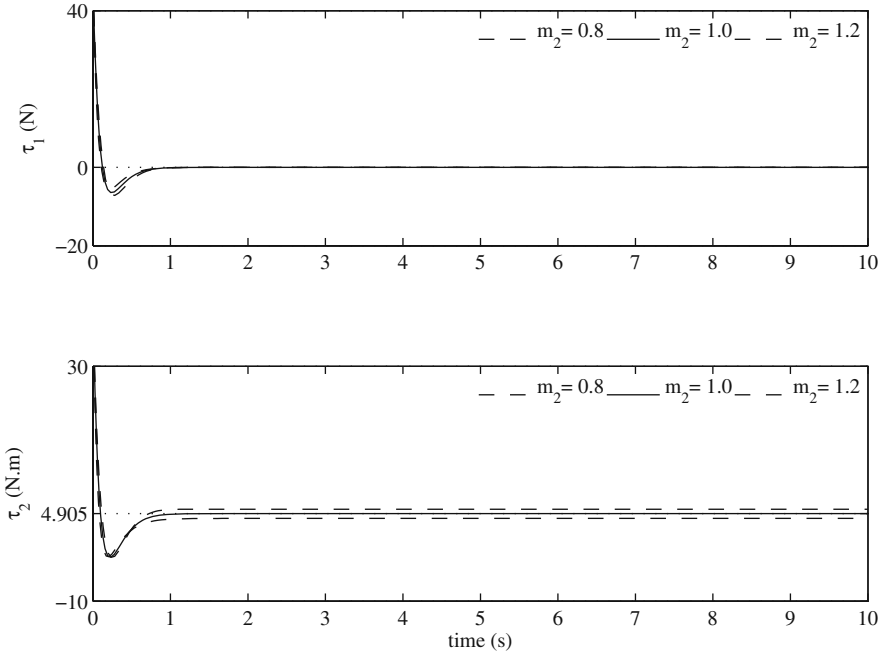


**Fig. 5.7** Driving force and torque for the PR manipulator under feedback linearized control

reason for this larger difference is that the second joint is more affected by the uncertainty in the second link's mass than the first joint.

(b) For the trajectory-tracking scenario, the desired joint speeds and acceleration are derived from the given desired joint position trajectories.

$$\dot{q}_i^d(t) = \begin{cases} 0.2 & \text{if } t \leq 3.333 \\ 0.0 & \text{if } 3.333 < t \leq 6.667 \\ -0.2 & \text{if } 6.667 < t \leq 10.000 \\ 0.0 & \text{if } t > 10.00 \end{cases} \qquad i = 1, 2, \tag{5.42}$$

and

$$\ddot{q}_i^d(t) = 0, \qquad i = 1, 2, \tag{5.43}$$

where $\dot{q}_1^d(t)$ is in m/s; $\dot{q}_2^d(t)$ is in rad/s; $\ddot{q}_1^d(t)$ is in m/s$^2$; and $\ddot{q}_2^d(t)$ is in rad/$s^2$.

The three cases of uncertainty for $m_2$ as described in part (a) of this example are simulated with the above desired values and the same controller gain $\lambda = 8$. Figure 5.8 shows the trajectory of the joint positions for the three cases of uncertainty along with the desired trajectories. The uncertainty in the mass of the second link has more strongly affected the position of the second joint compared

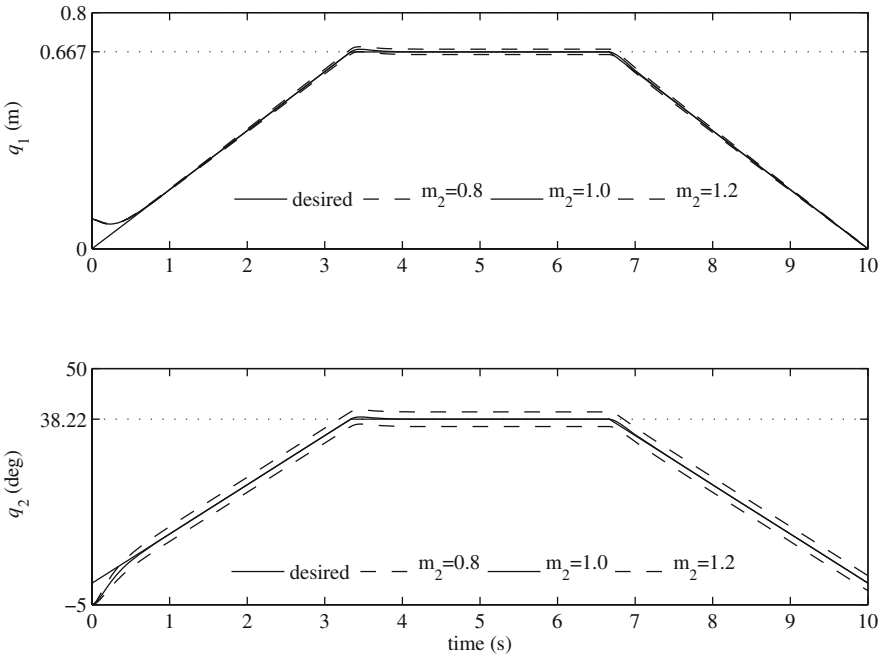


**Fig. 5.8** Trajectory of the joints for the PR manipulator under feedback linearized control

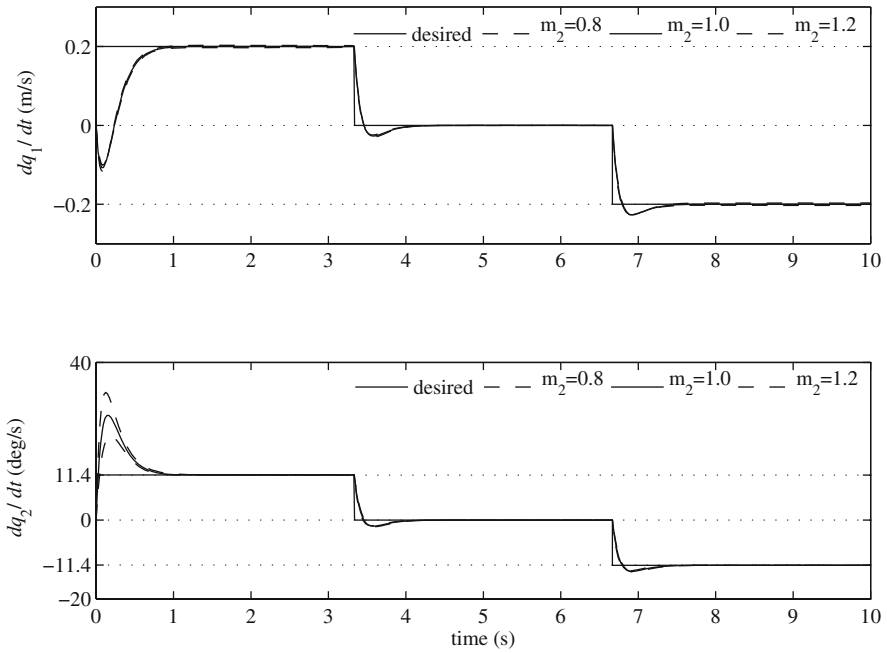


**Fig. 5.9** Joint speeds for the PR manipulator under feedback linearized control

to that of the first joint. However, as shown in Fig. 5.9, the joint speeds for the three cases are almost identical, except for the initial transient response, when the controller is trying to accelerate the manipulator from rest to the desired nonzero joint speeds. Finally, as seen in Fig. 5.10, the driving forces for the three cases are very close, while the driving torques show a rather larger difference between the three cases. This is consistant with the simulation results for the position control scenario. Here, the control force curve has rather large spikes. This may concern a control engineer regarding whether or not the joint actuator can generate such a fast response. If the actuator's response is not fast enough, the designed controller may not perform as well as the simulation shows in real application.

### *5.5.2 Robust Control*

In Example 5.4, it was shown that although the feedback linearization method has a good performance when the manipulator's model is very accurate, its performance deteriorates when there are uncertainty in the manipulator's model. In practical situations, there is not only a high possibility that the nominal dynamic parameters of a manipulator are not very accurate, but also they will even change by adding or reducing loads at the end-effector. Therefore, the feedback linearization method

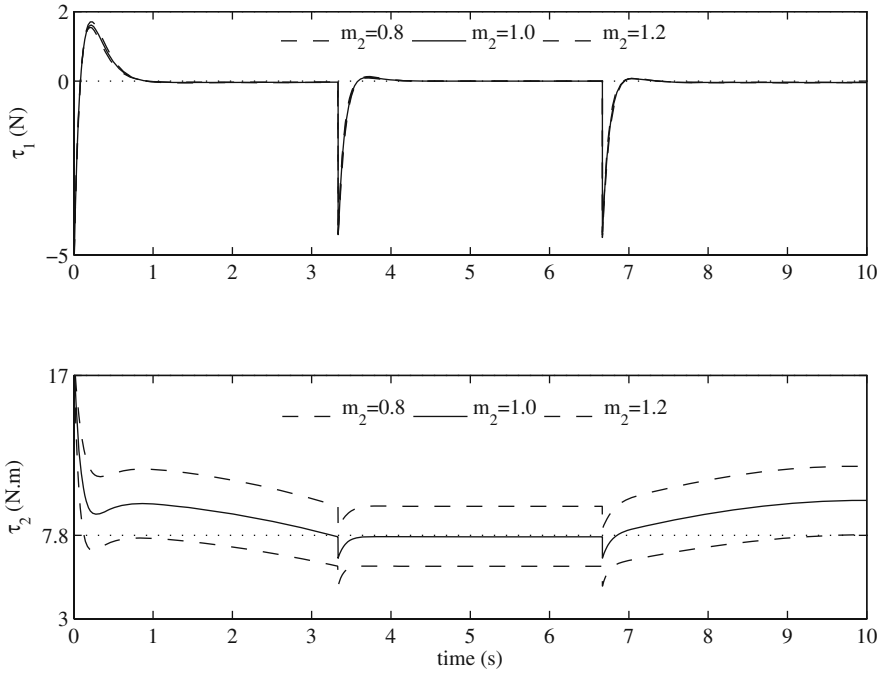


**Fig. 5.10** Driving force and torque for the PR manipulator under feedback linearized control

can only have limited control performance for manipulators. There is a need for a controller that performs well even when there are uncertainties in the dynamic model. A controller that performs well in presence of uncertainties in the dynamic model is called a robust controller.

In this section, we present a robust controller suitable for manipulators based on the sliding mode control method. The sliding mode control method, also known as the variable structure control method, is a general robust control method that can be applied to any dynamic system with equal number of inputs and outputs.

#### 5.5.2.1 The Sliding Mode Control Method

Since the sliding mode control is a model based control method, we need the dynamic model of the manipulator. The general form of the equations of motion for a manipulator, Eq. (5.1), is repeated here for convenience:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau}. \tag{5.44}$$

In sliding mode control, the required behavior of the closed-loop system (plant plus controller) from an initial state $\mathbf{q}$ to the desired dynamic equilibrium state $\mathbf{q}^d$ is defined as

$$\mathbf{s} = \dot{\tilde{\mathbf{q}}} + \Lambda\tilde{\mathbf{q}}, \tag{5.45}$$

where

$$\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}^d, \tag{5.46}$$

and $\Lambda$ is a diagonal gain matrix with positive diagonal components.

Now, a controller must be designed to guarantee the generation of the desired behavior of the plant states as defined in Eq. (5.45). The idea behind the sliding mode control design is as follows. If a controller can be derived such that it can stabilize the variable $\mathbf{s}$ at zero despite of uncertainties of the plant model, the states of the plant will follow the desired behavior

$$\dot{\tilde{\mathbf{q}}} + \Lambda\tilde{\mathbf{q}} = \mathbf{0}, \tag{5.47}$$

which asymptotically approaches $\tilde{\mathbf{q}} = 0$, because the diagonal gain matrix $\Lambda$ has positive diagonal components. The required behavior of the closed-loop system for $\mathbf{s} = \mathbf{0}$, i.e., Eq. (5.47), is called the surface. When $\tilde{\mathbf{q}}$ is zero, $\mathbf{q} = \mathbf{q}^d$, i.e., the system follows the desired trajectory.

With this in mind, one can define the sliding mode control design problem as finding a control law that guarantees the stabilization of $\mathbf{s}$ at zero despite any bounded uncertainty in the plant's dynamic model, and ensures the states of the system follow the surface $\mathbf{s} = \mathbf{0}$ to the dynamic equilibrium states.

#### 5.5.2.2 Sliding Model Controller Design for Manipulators

As noted above, a sliding mode controller has two major parts. One part, known as the equivalent control, ensures that the states of the system slide on the surface to the dynamic equilibrium states. The other part, which brings robustness to the controller, guarantees that the states of the system reach the surface from any initial condition and remain on the surface. These two parts are designed rather separately. In the following, first, some notations are defined. Next, the first part of the controller is derived. And finally, the derivation of the second part is discussed.

*Definitions.* One of the properties of the sliding mode control method is its robustness to model parameter uncertainty. When designing a sliding mode controller, one should be able to distinguish between the nominal dynamic model (with nominal system parameters) and the real dynamic model (with uncertain dynamic parameters). Let us assume that the general form (5.44) represents the real dynamic model. The nominal model has the same form as the real model, except it uses the nominal system parameters. This model is described by

$$\hat{\mathbf{H}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \hat{\mathbf{G}}(\mathbf{q}) = \boldsymbol{\tau}. \tag{5.48}$$

The matrices in this equation are the same as the ones defined for Eq. (5.1). The "hat" for these terms emphesis the fact that these terms are calculated using

the nominal parameters of the manipulator. Also, to simplify the notation of the derivations, let us rewrite the definition of the required behavior of the closed-loop system, Eq. (5.45), in the following form:

$$\mathbf{s} = \dot{\mathbf{q}} - \mathbf{s}_r, \tag{5.49}$$

where

$$\mathbf{s}_r = \dot{\mathbf{q}}^d - \Lambda\tilde{\mathbf{q}}. \tag{5.50}$$

With this definition, the equation defining the surface becomes

$$\dot{\mathbf{q}} - \mathbf{s}_r = \mathbf{0}. \tag{5.51}$$

*Equivalent control.* The equivalent control part of a sliding mode control law must ensure that the states of the closed-loop system follow the surface to the dynamic equilibrium of the system. We use the definition of the surface, Eq. (5.51), and the nominal model of the manipulator to derive the equivalent control. We calculate $\ddot{\mathbf{q}}$ by differentiating Eq. (5.51) with respect to time and substitute the result into the nominal dynamic model (5.48).

$$\hat{\boldsymbol{\tau}} = \hat{\mathbf{H}}\dot{\mathbf{s}}_r + \hat{\mathbf{C}}\mathbf{s}_r + \hat{\mathbf{G}}, \tag{5.52}$$

where Eq. (5.51) has also been used to replace $\dot{\mathbf{q}}$ with $\mathbf{s}_r$. The control command, $\hat{\boldsymbol{\tau}}$, that is defined by the above equation is the equivalent control.

*Robust control law.* The equivalent control has been developed with the assumption that the states of the systems are already on the surface, which is not generally true for an arbitrary initial state conditions. Furthermore, since the equivalent control is calculated based on the nominal parameters of the the manipulator, if any amount of uncertainty exists in the dynamic model, the trajectory of the states will depart from the surface, which means that the manipulator will not reach the dynamic equilibrium states. The second part of a sliding mode controller must guarantee that $\mathbf{s}$ becomes zero and stays zero even when there are model parameter uncertainties.

The second part of a sliding mode controller is a discontinuous function of $\mathbf{s}$ that is added to the equivalent control. A complete sliding mode control law is written as

$$\boldsymbol{\tau} = \hat{\boldsymbol{\tau}} - \mathbf{K}\,\mathrm{sgn}(\mathbf{s}), \tag{5.53}$$

or

$$\boldsymbol{\tau} = (\hat{\mathbf{H}}\dot{\mathbf{s}}_r + \hat{\mathbf{C}}\mathbf{s}_r + \hat{\mathbf{G}}) - \mathbf{K}\,\mathrm{sgn}(\mathbf{s}), \tag{5.54}$$

where $\mathbf{K}$ is a diagonal controller discontinuity gain matrix whose diagonal components must be properly determined, and the "sgn" function returns a vector with the sign of the components of $\mathbf{s}$.

Although the control law (5.53) seems complete, the diagonal components of the controller discontinuity gain **K** must still be determined such that the control law is robust to a given bounded parameter uncertainty. To determine the discontinuity gains such that **s** approaches zero even in the presence of uncertainties, one must use the Lyapunov stability method. In this method, a function of the variables that have to approach zero as time passes is defined. This function, known as the Lyapunov function, must be positive for all values of the variables and must be zero only when the variables are zero. Now, if one can find the conditions under which the time derivative of this function is always negative, it is guaranteed that, for those conditions, the variables approach zero as time passes independent of the variables' initial value at time zero. The conditions give information about the required controller gains. The following simple example illustrates the use of a Lyapunov function for controller design.

*Example 5.5.* Consider a 1 DOF spring-mass system with the following dynamic model.

$$m\ddot{x} + kx = f, \tag{5.55}$$

where $m$, $k$, and $f$ are the body mass, the spring stiffness, and the force applied to the body. Assume that the position of the mass has to be controlled using a force. The force is generated according the following control law:

$$f = -c\dot{x}. \tag{5.56}$$

Define a proper Lyapunov function for the spring-mass system. Find the conditions under which the assumed control law can stabilize the system at $x = 0$.

*Solution.* A proper Lyapunov function must contain all the states of the system, must be positive for all values of the system states, and must be zero only when the states of the system are at the desired state (here, $x = 0$ and $\dot{x} = 0$). One can verify that the following function possesses all these properties:

$$V = \frac{1}{2}m\dot{x}^2 + \frac{1}{2}kx^2. \tag{5.57}$$

For the controlled system (close-loop system) to stabilize at the desired state, the time rate of the Lyapunov function must be negative for all values of the states:

$$\dot{V} < 0. \tag{5.58}$$

For the defined Lyapunov function, the rate is obtained as

$$\dot{V} = m\dot{x}\ddot{x} + k\dot{x}x. \tag{5.59}$$

Substituting the dynamics of the system and the control law into the above equation results in

$$\dot{V} = f\dot{x} = -c\dot{x}^2. \tag{5.60}$$

The rate of the Lyapunov function must be negative, therefore,

$$-c\dot{x}^2 < 0, \tag{5.61}$$

which results in the following condition on the controller gain $c$:

$$c > 0. \tag{5.62}$$

This means that as long as a positive number is selected as the controller gain, the controlled system will be stabilized at the desired states regardless of the initial states.

The same procedure as the solution procedure of Example 5.5 is followed for determining the discontinuity gain $\mathbf{K}$ of the sliding mode robust controller for a manipulator. First, a proper Lyapunov function must be defined. Since the goal of the second part of a sliding mode controller is to stabilize $\mathbf{s}$, the Lyapunov function must contain $\mathbf{s}$.

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{H}\mathbf{s}. \tag{5.63}$$

Note that the matrix $\mathbf{H}$, appearing in the dynamic model of the manipulator is positive-definite. Therefore, the above Lyapunov function is positive for all values of $\mathbf{s}$ and is zero only when $\mathbf{s}$ is equal to zero. Now, the rate of this Lyapunov function must be investigated.

$$\dot{V} = \frac{1}{2}\dot{\mathbf{s}}^T\mathbf{H}\mathbf{s} + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{H}}\mathbf{s} + \frac{1}{2}\mathbf{s}^T\mathbf{H}\dot{\mathbf{s}}. \tag{5.64}$$

Since $\mathbf{H}$ is always symmetric for serial manipulators, the first and the last term of the right hand side of the above expression are equal.

$$\dot{V} = \mathbf{s}^T\mathbf{H}\dot{\mathbf{s}} + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{H}}\mathbf{s}. \tag{5.65}$$

Substituting for $\dot{\mathbf{s}}$ from Eq. (5.49) results in

$$\dot{V} = \mathbf{s}^T\mathbf{H}(\ddot{\mathbf{q}} - \dot{\mathbf{s}}_r) + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{H}}\mathbf{s}. \tag{5.66}$$

Substituting for $\ddot{\mathbf{q}}$ from the dynamic model of the manipulator, Eq. (5.44), yields

$$\dot{V} = \mathbf{s}^T(\boldsymbol{\tau} - \mathbf{C}\dot{\mathbf{q}} - \mathbf{G} - \mathbf{H}\dot{\mathbf{s}}_r) + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{H}}\mathbf{s}. \tag{5.67}$$

Using $\dot{\mathbf{q}} = \mathbf{s} + \mathbf{s}_r$ (from Eq. (5.49)) reduces the above equation to

$$\dot{V} = \mathbf{s}^T(\boldsymbol{\tau} - \mathbf{C}\mathbf{s}_r - \mathbf{G} - \mathbf{H}\dot{\mathbf{s}}_r) + \frac{1}{2}\mathbf{s}^T(\dot{\mathbf{H}} - 2\mathbf{C})\mathbf{s}. \tag{5.68}$$

At this stage, let us assume that the matrix $\dot{\mathbf{H}} - 2\mathbf{C}$ is skew-symmetric, which causes the last term of the right hand side of the above equation to vanish. Later, it will be shown that indeed one can always find the matrix $\mathbf{C}$ in the dynamic model of the manipulator such that this assumption is correct. With this assumption, the rate of the Lyapunov function becomes

$$\dot{V} = \mathbf{s}^T(\boldsymbol{\tau} - \mathbf{H}\dot{\mathbf{s}}_r - \mathbf{C}\mathbf{s}_r - \mathbf{G}). \tag{5.69}$$

Now, the derived control law (5.54) must be substituted into the above relation so that the controller discontinuity gains appear in the expression for the rate of the Lyapunov function. After some simplification, this results in the following:

$$\dot{V} = \mathbf{s}^T[(\hat{\mathbf{H}} - \mathbf{H})\dot{\mathbf{s}}_r + (\hat{\mathbf{C}} - \mathbf{C})\mathbf{s}_r + (\hat{\mathbf{G}} - \mathbf{G}) - \mathbf{K}\,\mathrm{sgn}(\mathbf{s})]. \tag{5.70}$$

Interestingly enough, the differences between the major terms in the nominal dynamic model and the real (uncertain) dynamic model have appeared in the expression for the rate of the Lyapunov function. This fortunate event allows us to determine the discontinuity gain of the controller based on some estimated bounds on these differences. The differences in these major terms of the dynamic model need some new notation as follows:

$$\tilde{\mathbf{H}} = \hat{\mathbf{H}} - \mathbf{H}, \qquad \tilde{\mathbf{C}} = \hat{\mathbf{C}} - \mathbf{C}, \qquad \tilde{\mathbf{G}} = \hat{\mathbf{G}} - \mathbf{G}. \tag{5.71}$$

With this notation, $\dot{V}$ becomes

$$\dot{V} = \mathbf{s}^T(\tilde{\mathbf{H}}\dot{\mathbf{s}}_r + \tilde{\mathbf{C}}\mathbf{s}_r + \tilde{\mathbf{G}}) - \mathbf{s}^T\mathbf{K}\,\mathrm{sgn}(\mathbf{s}). \tag{5.72}$$

The matrix multiplications in the above equations are converted to summations so that the components of the discontinuity gain appear in the expression.

$$\dot{V} = \sum_{i=1}^{n} s_i(\tilde{\mathbf{H}}\dot{\mathbf{s}}_r + \tilde{\mathbf{C}}\mathbf{s}_r + \tilde{\mathbf{G}})_i - K_i|s_i|. \tag{5.73}$$

To simplify the derivations, the terms being multiplied in the first term of the right hand side of the above equation are replaced by their absolute values. This increases the total summation of the right hand side, which forces us to also replace the equal sign with a "less than or equal" sign.

$$\dot{V} \leq \sum_{i=1}^{n} |s_i|.|(\tilde{\mathbf{H}}\dot{\mathbf{s}}_r + \tilde{\mathbf{C}}\mathbf{s}_r + \tilde{\mathbf{G}})_i| - |s_i|K_i. \tag{5.74}$$

The above expression is further simplified.

$$\dot{V} \leq -\sum_{i=1}^{n} |s_i|.(K_i - |(\tilde{\mathbf{H}}\dot{\mathbf{s}}_r + \tilde{\mathbf{C}}\mathbf{s}_r + \tilde{\mathbf{G}})_i|). \tag{5.75}$$

This expression implies that if the discontinuity gains of the sliding mode controller are selected such that

$$K_i \geq |(\tilde{\mathbf{H}}\dot{\mathbf{s}}_r + \tilde{\mathbf{C}}\mathbf{s}_r + \tilde{\mathbf{G}})_i| + \eta_i, \tag{5.76}$$

where $\eta_i$'s are arbitrary positive constants, then

$$\dot{V} \leq -\sum_{i=1}^{n} |s_i|.\eta_i. \tag{5.77}$$

This means that the rate of the Lyapunov function $V$ defined in Eq. (5.63) is always negative if $K_i$'s are selected such that they satisfy Eq. (5.76). Therefore, $\mathbf{s}$ approaches zero as time passes regardless of its initial value and regardless of an uncertainty in the dynamic model whose bounds are defined in Eq. (5.71).

*Chattering.* The discontinuity of the sliding mode control law can not only cause problems for numerical differential equation solvers during the simulations, but also can lead to chattering of the system (high-frequency actuation and vibration) in practical applications. The reason for chattering is that the surface parameters $s_i$ are never exactly zero to the last precision digit during the computer control calculations. Therefore, the discontinuity term keeps switching from a small positive $s_i$ to a small negative $s_i$. These switchings are magnified by the multiplier $K_i$ in the control law. This magnification may cause significant fluctuations in $\tau_i$, which in turn may cause the manipulator to vibrate.

To avoid chattering, a saturation function replaces the sign function in the sliding mode control law (5.54). The saturation function is continuous around the surface $s_i = 0$, which allows $s_i$ to smoothly converge to zero. The saturation function is defined as follows:

$$\text{sat}(x) = \begin{cases} \text{sgn}(x), & \text{if abs}(x) \geq 1 \\ x, & \text{if } x < 1 \end{cases}. \tag{5.78}$$

The saturation function must be only activated inside a boundary around the surface. The thickness of this boundary is represented by $\phi_i$. After replacing the sign function with the saturation function, the control law (5.54) becomes

$$\boldsymbol{\tau} = (\hat{\mathbf{H}}\dot{\mathbf{s}}_r + \hat{\mathbf{C}}\mathbf{s}_r + \hat{\mathbf{G}}) - \mathbf{K}\,\text{sat}(\mathbf{s}/\boldsymbol{\phi}), \tag{5.79}$$

where $\text{sat}(\mathbf{s}/\boldsymbol{\phi})$ is a $n \times 1$ column vector with components

$$\text{sat}(s_i/\phi_i). \tag{5.80}$$

*Condition on the dynamic model.* The reader should be reminded that the condition on the discontinuity gains, Eq. (5.76), has been derived with the assumption that the matrix $\dot{\mathbf{H}} - 2\mathbf{C}$ is skew-symmetric. In fact, when the dynamic model is being written in the standard form (5.44), there are many correct options for the matrix $\mathbf{C}$. However, if the dynamic model is to be used for designing a robust controller, the matrix $\mathbf{C}$ must be selected such that the term $\dot{\mathbf{H}} - 2\mathbf{C}$ is skew-symmetric. It can be shown that if the components of $\mathbf{C}$ are derived using the following relation, the matrix $\dot{\mathbf{H}} - 2\mathbf{C}$ is skew-symmetric.

$$C_{ij} = \frac{1}{2}\dot{H}_{ij} + \frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial H_{ik}}{\partial q_j} - \frac{\partial H_{jk}}{\partial q_i}\right)\dot{q}_k. \tag{5.81}$$

The following example shows the usage of Eq. (5.81).

*Example 5.6.* Consider the PR manipulator of Example 5.1. The term $\mathbf{H}$ in the standard form of the PR manipulator's dynamic model was derived as

$$\mathbf{H}(\mathbf{q}) = \begin{bmatrix} m_1 + m_2 & -m_2 l_2 \sin q_2 \\ -m_2 l_2 \sin q_2 & m_2 l_2^2 \end{bmatrix}. \tag{5.82}$$

Derive the $\mathbf{C}$ matrix suitable for sliding mode robust controller design.

*Solution.* A $\mathbf{C}$ matrix suitable for sliding mode robust controller design must ensure that the term $\dot{\mathbf{H}} - 2\mathbf{C}$ is a skew-symmetric matrix. Equation (5.81) is used to calculate such a $\mathbf{C}$ matrix. Since the PR-manipulator has two joint variables, $n$ is equal to 2 and $\mathbf{C}$ is a $2 \times 2$ matrix. The components of $\mathbf{C}$ are calculated as follows:

$$C_{11} = 0, \tag{5.83}$$
$$C_{12} = -m_2 l_2 \dot{q}_2 \cos q_2, \tag{5.84}$$
$$C_{21} = 0, \tag{5.85}$$
$$C_{22} = 0. \tag{5.86}$$

Therefore,

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} 0 & -m_2 l_2 \dot{q}_2 \cos q_2 \\ 0 & 0 \end{bmatrix}. \tag{5.87}$$

The term $\dot{\mathbf{H}} - 2\mathbf{C}$ is calculated as

$$\dot{\mathbf{H}} - 2\mathbf{C} = \begin{bmatrix} 0 & m_2 l_2 \dot{q}_2 \cos q_2 \\ -m_2 l_2 \dot{q}_2 \cos q_2 & 0 \end{bmatrix}, \tag{5.88}$$

which is skew-symmetric.

*Example 5.7.* Consider the PR manipulator of Example 5.1. Derive a sliding mode robust control law for the manipulator. Design the discontinuity gain of the controller such that the controller is robust to up to ±25% uncertainty in the manipulator parameters $m_1$, $m_2$, and $l_2$. Assume the desired trajectory introduced in Example 5.4. Investigate the performance of the manipulator for ±20% uncertainty in parameter $m_2$. Also, verify that the required response of the states of the system under the sliding mode control is defined by Eq. (5.45) as expected.

*Solution.* For the 2DOF PR manipulator, the required response of the states of the system under the sliding mode control is defined based on Eq. (5.45) as follows:

$$\begin{bmatrix} s_1 \\ s_2 \end{bmatrix} = \begin{bmatrix} \dot{q}_1 - \dot{q}_1^d \\ \dot{q}_2 - \dot{q}_2^d \end{bmatrix} + \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} q_1 - q_1^d \\ q_2 - q_2^d \end{bmatrix}. \tag{5.89}$$

The auxiliary terms $\mathbf{s}_r$ and $\dot{\mathbf{s}}_r$ are derived as follows:

$$\begin{bmatrix} s_{r1} \\ s_{r2} \end{bmatrix} = \begin{bmatrix} \dot{q}_1^d \\ \dot{q}_2^d \end{bmatrix} - \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} q_1 - q_1^d \\ q_2 - q_2^d \end{bmatrix}, \tag{5.90}$$

$$\begin{bmatrix} \dot{s}_{r1} \\ \dot{s}_{r2} \end{bmatrix} = \begin{bmatrix} \ddot{q}_1^d \\ \ddot{q}_2^d \end{bmatrix} - \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \dot{q}_1 - \dot{q}_1^d \\ \dot{q}_2 - \dot{q}_2^d \end{bmatrix}. \tag{5.91}$$

The required response of the states of the system under the sliding mode control when $\mathbf{s}$ is zero can be set by selecting the gain matrix $\Lambda$ components. In this example, this gain matrix is selected as

$$\Lambda = \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix}. \tag{5.92}$$

The equivalent control is derived based on the nominal dynamic model, which uses the nominal parameters of the manipulator. For the PR manipulator of this example, these nominal parameters are

$$\hat{m}_1 = 1.0\ \text{kg}, \qquad \hat{m}_2 = 1.0\ \text{kg}, \qquad \hat{l}_2 = 1.0\ \text{m}. \tag{5.93}$$

The terms of the manipulator's nominal dynamic model are

$$\hat{\mathbf{H}} = \begin{bmatrix} \hat{m}_1 + \hat{m}_2 & -\hat{m}_2 \hat{l}_2 \sin q_2 \\ -\hat{m}_2 \hat{l}_2 \sin q_2 & \hat{m}_2 \hat{l}_2^2 \end{bmatrix}, \tag{5.94}$$

$$\hat{\mathbf{C}} = \begin{bmatrix} 0 & -\hat{m}_2 \hat{l}_2 \dot{q}_2 \cos q_2 \\ 0 & 0 \end{bmatrix}, \tag{5.95}$$

$$\hat{\mathbf{G}} = \begin{bmatrix} 0 \\ \hat{m}_2 g \hat{l}_2 \cos q_2 \end{bmatrix}. \tag{5.96}$$

To design the discontinuity gains such that the controller is robust to up to ±25% uncertainty in the manipulator's dynamic model parameters, the following uncertain parameters are assumed as the worst case.

$$m_1 = 1.25 \text{ kg}, \qquad m_2 = 1.25 \text{ kg}, \qquad l_2 = 1.25 \text{ m}. \tag{5.97}$$

The terms of the manipulator's uncertain dynamic model are

$$\mathbf{H} = \begin{bmatrix} m_1 + m_2 & -m_2 l_2 \sin q_2 \\ -m_2 l_2 \sin q_2 & m_2 l_2^2 \end{bmatrix}, \tag{5.98}$$

$$\mathbf{C} = \begin{bmatrix} 0 & -m_2 l_2 \dot{q}_2 \cos q_2 \\ 0 & 0 \end{bmatrix}, \tag{5.99}$$

$$\mathbf{G} = \begin{bmatrix} 0 \\ m_2 g l_2 \cos q_2 \end{bmatrix}. \tag{5.100}$$

To calculate the discontinuity gain of the controller, first, the derived matrices $\hat{\mathbf{H}}$, $\hat{\mathbf{C}}$, $\hat{\mathbf{G}}$, $\mathbf{H}$, $\mathbf{C}$, and $\mathbf{G}$ are substituted into Eq. (5.71). Then, the results are used in Eq. (5.76). For this example, $\eta_i$'s are selected to be 1. With the components of $\mathbf{K}$ calculated, Eq. (5.79) is used to control the PR manipulator. The boundary layer widths for the saturation function in Eq. (5.79) are selected as

$$\phi_1 = 0.01 \text{ m/s}, \qquad \phi_2 = 0.05 \text{ rad/s}. \tag{5.101}$$

Three scenarios similar to the ones discussed in Example 5.4 are considered for investigating the performance of the controller under uncertain dynamic model when tracking the desired trajectory. Figure 5.11 shows the trajectory of the joint positions for the three scenarios. Since the robust controller's discontinuity gains have been calculated based on a maximum of ±25% uncertainty, the controller's performance does not deteriorate for $m_2 = 0.8$ and 1.2 kg (i.e., ±20% uncertainty).

Figure 5.12 shows the speed of the joints while the manipulator is tracking the desired trajectory. Once again, all of the responses for the nominal and the uncertain dynamics of the system are coincident, and the desired speeds are tracked accurately despite of uncertainty in the dynamic model.

Figure 5.13 illustrates the history of the driving force and torque for the joints. The uncertainty in the mass of the second link has very limited effect on the force of the first joint. However, the second joint actuator, which is under the direct effect of the mass of the second link, has been automatically adjusted by the controller to provide the appropriate torque for the lower or the higher mass conditions.

The trajectory of the manipulator's states errors in the phase plane (a diagram of $\dot{\tilde{q}}_i$ versus $\tilde{q}_i$) shows how the two parts of the robust sliding mode controller operate. Figure 5.14 illustrates such a trajectory for both of the joint variables.

The top diagrams shows the error in the state variables of the first joint. The initial position and speed of the first joint are $q_1(0) = 1.0$ and 0.0 m/s, respectively. Since

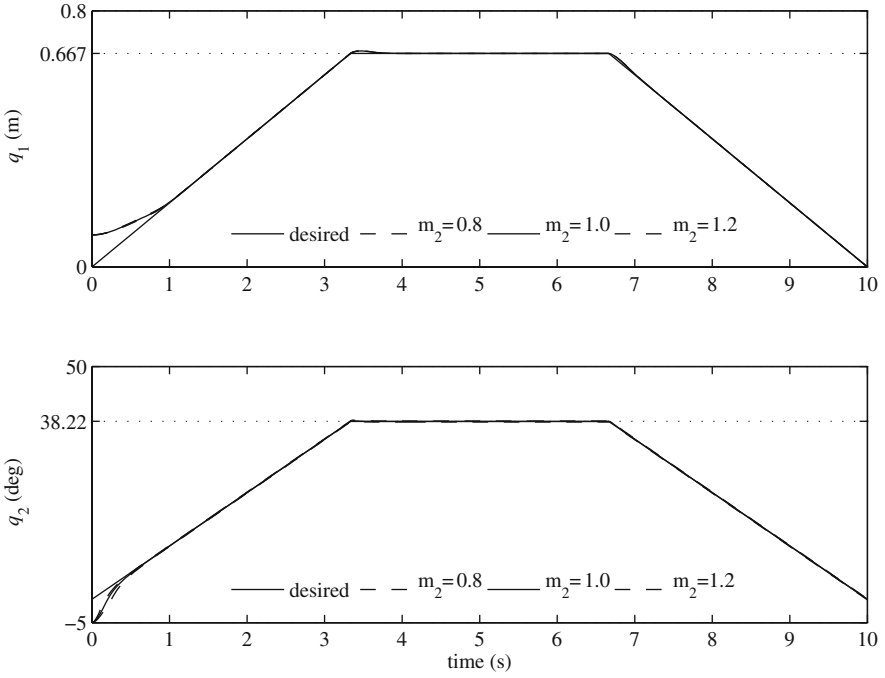


**Fig. 5.11** Trajectory of the joints for the PR manipulator under robust control

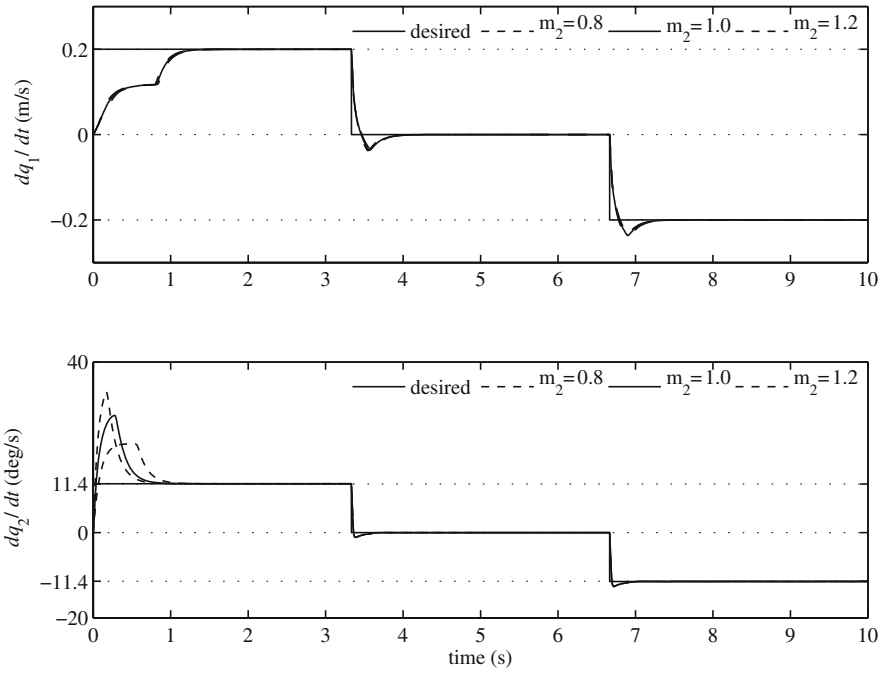


**Fig. 5.12** Joint speeds for the PR manipulator under robust control

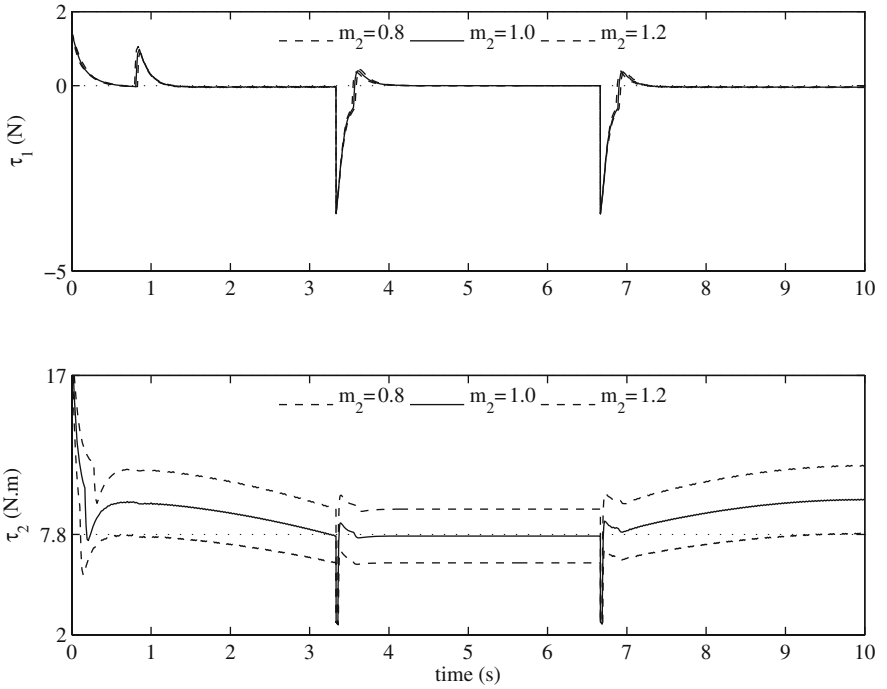


**Fig. 5.13** Driving force and torque for the PR manipulator under robust control

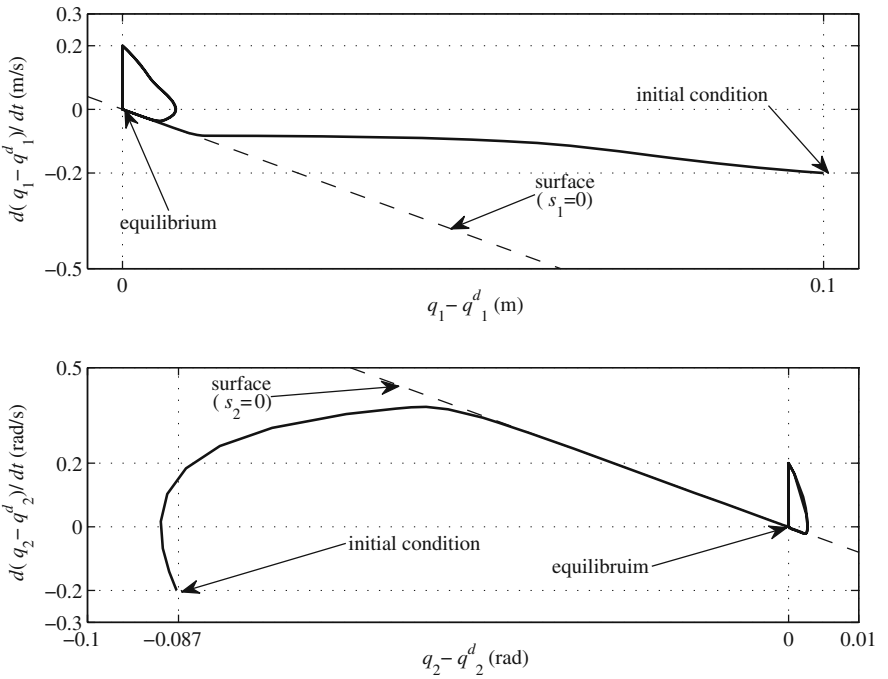


**Fig. 5.14** Phase plane for the PR manipulator under robust control

the desired position and velocity of the first joint at time zero are $q_1^d(0) = 0.0$ m and $\dot{q}_1(0) = 0.2$ m/s, respectively, the initial error of the states of the first joint is

$$\tilde{q}_1(0) = q_1(0) - q_1^d(0) = 0.1 \text{ m} \qquad \dot{\tilde{q}}_1(0) = \dot{q}_1(0) - \dot{q}_1^d(0) = -0.2 \text{ m/s} \quad (5.102)$$

This is the point where the trajectory of the error in the phase plane starts. As seen in the top diagram of Fig. 5.14, this initial point is not on the defined surface ($\mathbf{s} = \mathbf{0}$), which is the required bahavior of the system that leads to equilibrium. The second part of the sliding mode controller, the discontinuity term, pushes the error trajectory from the initial condition onto the surface and assures that the trajectory remains on the surface. This can be seen in Fig. 5.14. Now, note that, for the first joint, the equation describing the surface, Eq. (5.47), becomes

$$0 = \dot{\tilde{q}}_1 + \lambda_1 \tilde{q}_1, \quad (5.103)$$

which has an asymptotically stable equilibrium point at $\dot{\tilde{q}}_1 = 0$ and $\tilde{q}_1 = 0$. The equivalent control is responsible for this behavior of the system. This means that when the trajectory of the error reaches the surface, it will be pushed on the surface to the equilibrium point by the equivalent control. This has happened to the trajectory of the error for the first joint, as can be seen in the top plot of Fig. 5.14. However, the trajectory will not remain at the equilibrium point for our simulation because, at time $t = 0.333$ s, the desired value for the joint speed changes abruptly. At this moment, suddenly, there is an error of $\dot{\tilde{q}} = 0.2$ m/s in the speed of the first joint, while the position has no error. Therefore, the trajectory of the error jumps to the point (0.0 m, 0.2 m/s) on the phase plane. The controller acts rapidly and pushes the trajectory back onto the surface and slides the trajectory on the surface to the equilibrium once again. This procedure repeats when the desired speed jumps to $-0.2$ m/s once again at time $t = 6.667$ s.

The same argument can be repeated for the trajectory of the error for the second joint.

## Problems

**Problem 5.1.** Consider the Revolute-Prismatic (RP) manipulator shown in Fig. 5.15. The manipulator is working in the vertical plane.

(a) Using the Lagrange method, derive the dynamic equations of motion for the manipulator.
(b) Write the equations of motion in the standard form (Eq. 5.1).
(c) Write the first-oder form of the equations of motion (Eq. 5.12).

**Problem 5.2.** Consider the RP manipulator of Problem 5.1. Design a PD controller that can bring the manipulator to any given desired position. Assume a mass of 2 kg and a moment of inertia of 0.04 kg.m$^2$ about the center of mass for each link.

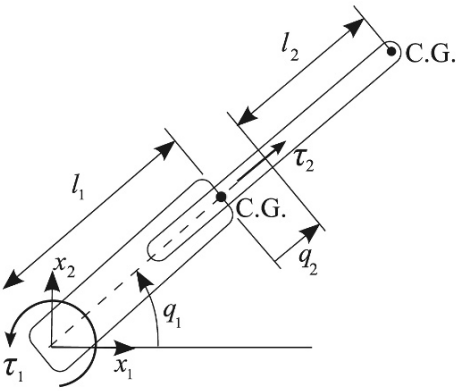


**Fig. 5.15** A 2-DOF Revolute-Prismatic (RP) manipulator

The link lengths as shown in Fig. 5.15 are 0.5 m each. Simulate the manipulator's response under the control if it starts from the initial posture $\mathbf{q} = [0 \text{ rad}, 0 \text{ m}]^T$ and is trying to reach a desired posture of $\mathbf{q}^d = [\pi/3 \text{ rad}, 0.3 \text{ m}]^T$.

**Problem 5.3.** Consider the PR manipulator of Problem 5.2. Derive a tracking control law for the manipulator using the feedback linearization method. Investigate the performance of the manipulator for an uncertain mass for the second link, $m_2$. This uncertainty represents the variations in loads carried by the second link. Assume $\pm 20\%$ bounds for uncertainty in $m_2$. For the performance investigations, use the following scenarios.

(a) Position control performance: the manipulator is at rest at an initial posture of (0 rad, 0 m). It should reach the desired posture of ($\pi/3$ rad, 0.3 m).
(b) Trajectory-tracking performance: the manipulator is at rest at an initial posture of (0 rad, 0 m). The manipulator's joints must follow the following desired time dependent trajectory.

$$q_1^d(t) = \begin{cases} \frac{3\pi}{50}t^2 - \frac{\pi}{125}t^3 & \text{if} \quad 0 < t \leq 5 \\ \frac{\pi}{2} & \text{if} \quad 5 < t \leq 10 \\ \frac{\pi}{2} - \frac{3\pi}{50}(t-10)^2 + \frac{\pi}{125}(t-10)^3 & \text{if} \quad 10 < t \leq 15 \\ 0 & \text{if} \quad t > 15 \end{cases},$$

$$q_2^d(t) = \begin{cases} \frac{3}{100}t^2 - \frac{1}{250}t^3 & \text{if} \quad 0 < t \leq 5 \\ \frac{1}{4} & \text{if} \quad 5 < t \leq 10 \\ \frac{1}{4} - \frac{3}{100}(t-10)^2 + \frac{1}{250}(t-10)^3 & \text{if} \quad 10 < t \leq 15 \\ 0 & \text{if} \quad t > 15 \end{cases},$$

where $q_1^d(t)$ is in radians and $q_2^d(t)$ is in meters.

**Problem 5.4.** Consider the RP manipulator of Problem 5.2. The robot must be able to handle different loads with masses of up to 0.3 kg, with negligible moments of inertia, without loss of tacking accuracy. The load's center of mass is assumed to be coincident with the center of mass of the second link. Design a sliding mode robust controller that can control the manipulator on any given trajectory. Investigate the performance of the robust controller for three cases:

(a) no load at the end-effector;
(b) a 0.15-kg load at the end-effector; and
(c) a 0.5-kg load at the end-effector.

Use the desired joint trajectory defined in Problem 5.3.

**Problem 5.5.** Consider the spatial (3-Revolute) manipulator shown in Fig. 5.16. Link one of the manipulator lies in the horizontal $x_1$–$x_2$ plane, while links two and three are working in the vertical plane.

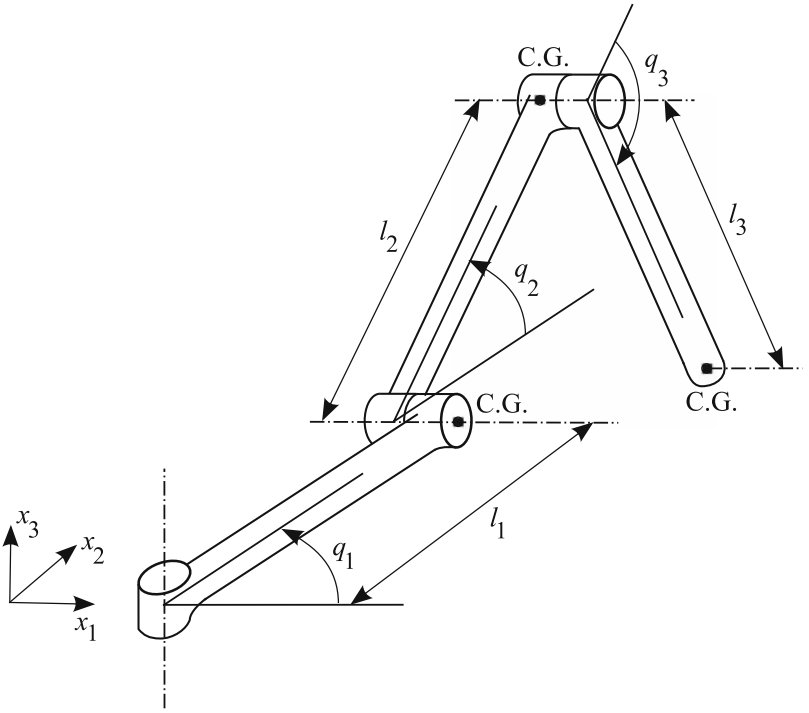


**Fig. 5.16** A 3-DOF 3-Revolute (3R) manipulator. The joint angle $q_3$ is shown in a negative configuration

(a) Using the Lagrange method, derive the dynamic equations of motion for the manipulator.
(b) Write the equations of motion in the standard form (Eq. 5.1).
(c) Write the first-oder form of the equations of motion (Eq. 5.12).

**Problem 5.6.** Consider the 3-R manipulator of Problem 5.5. Design a PD controller that can bring the manipulator to any given desired position. Assume a mass of 3 kg and a moment of inertia of 0.06 kg.m$^2$ about the center of mass for each link. The link lengths as shown in Fig. 5.16 are 0.5 m each. Simulate the manipulator's response under the control if it starts from the initial posture $\mathbf{q} = [0, 0, 0]^T$ rad and is trying to reach a desired posture of $\mathbf{q}^d = [\pi/3, \pi/4, -\pi/12]^T$ rad.

**Problem 5.7.** Consider the PR manipulator of Problem 5.5. Derive a tracking control law for the manipulator using the feedback linearization method. Investigate the performance of the manipulator for an uncertain mass for the third link, $m_3$. This uncertainty represents the variations in loads carried by the second link. Assume $\pm 20\%$ bounds for uncertainty in $m_3$. For the performance investigations, use the following scenarios.

(a) Position control performance: the manipulator is at rest at an initial posture of (0, 0, 0) rad. It should reach the desired posture of $(\pi/3, \pi/4, -\pi/12)$.
(b) Trajectory-tracking performance: the manipulator is at rest at an initial posture of (0, 0, 0) rad. All the manipulator's joints must follow the following desired time dependent trajectory.

$$q_i^d(t) = \begin{cases} \frac{3\pi}{50}t^2 - \frac{\pi}{125}t^3 & \text{if} \quad 0 < t \leq 5 \\ \frac{\pi}{2} & \text{if} \quad 5 < t \leq 10 \\ \frac{\pi}{2} - \frac{3\pi}{50}(t-10)^2 + \frac{\pi}{125}(t-10)^3 & \text{if} \quad 10 < t \leq 15 \\ 0 & \text{if} \quad t > 15 \end{cases} \qquad i = 1, 2, 3,$$

where $q_i^d(t)$ is in radians.

**Problem 5.8.** Consider the RP manipulator of Problem 5.5. The robot must be able to handle different loads with masses of up to 0.5 kg, with negligible moments of inertia, without loss of tacking accuracy. The load's center of mass is assumed to be coincident with the center of mass of the second link. Design a sliding mode robust controller that can control the manipulator on any given trajectory. Investigate the performance of the robust controller for three cases:

(a) no load at the end-effector;
(b) a 0.25-kg load at the end-effector; and
(c) a 0.6-kg load at the end-effector.

Use the desired joint trajectory defined in Problem 5.7.

# Chapter 6
# Mobile Robots

## 6.1 Introduction

In this chapter, the mechanics and controls for two types of mobile robots is studied. The two mobile robot types are Hilare-type and car-like mobile robots. Hilare-type robots have two independently driven wheels as the drive mechanism and are usually balanced by a passive caster wheel. They have good maneuvering abilities, e.g., a zero minimum turn radius, and are easier to control. They are also easier to build due to their simple drive mechanism. Car-like mobile robots, as their name implies, have a drive mechanism similar to cars. They are driven by a single motor that powers a differential, which in turn distributes the motor's torque to the rear wheels. They have a steering mechanism at the front wheel(s), which is driven by a motor to generate steering angles to steer the robot. Car-like robots have a nonzero minimum turn radius. The nonzero minimum turn radius limits the manovering ability of car-like robots. Care must be taken when defining a desired path such that the minimum radius of curvature of the path is not less than the minimum turn radius of the car-like robot. Otherwise, the robot will not be able to follow that path correctly independent of the controller that is being used for the trajectory tracking.

The rest of this chapter is organized as follows. Since the kinematic model of mobile robots are common for simulating this system and are usually used for controller design, we present the kinematics models of Hilare and car-like robots. Then, we use these kinematic models to derive trajectory-tracking control laws for the robots. Finally, the dynamic model of the mobile robots are presented at the end of this chapter.

## 6.2 Kinematic Models of Mobile Robots

### *6.2.1 Hilare Mobile Robots*

A schematic figure of a Hilare mobile robot is shown in Fig. 6.1. This type of robot is mostly used for indoor applications. The drive mechanism of a Hilare-type robot has two independent motors. Each of these motors power one of the robot's

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_6

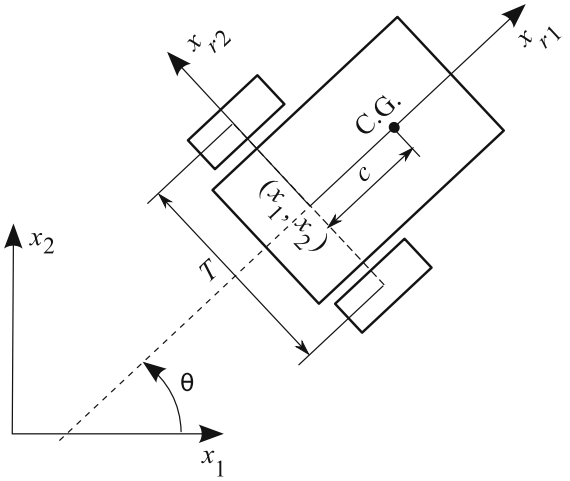


**Fig. 6.1** A Hilare mobile robot

wheels. Thus, the actual kinematic inputs that drive the robot and affect its speed and direction of motion are the two wheel speeds. With this in mind, at first glance it seems intuitive to write the kinematic equations of motion of a Hilare mobile robot in terms of these speeds. However, on most commercial mobile robots, there exists a low-level controller that controls the linear and angular velocity of the robot. Therefore, for application purposes, it is more convenient to choose the linear and angular velocity of the mobile robot as the inputs of the kinematic model. When a control law is found later based on this model, it can be more easily applied using the development packages available for commercial robots.

Now, consider the Hilare-type mobile robot shown in Fig. 6.1. Assume that the robot motion is reasonably slow such that the longitudinal traction and lateral force exerted on the robot's tires do not exceed the maximum static friction between the tires and the floor in the longitudinal and lateral directions. In other words, assume that no-slip happens between the robot's tire and the floor during the whole motion of the robot.

The first direct result of this assumption is that the velocities of the center of the robot's wheels do not have any lateral components. As a consequence, one can assume that the velocity of point $(x_1, x_2)$, the midpoint of the line attaching the center of the wheels, does not have any lateral component and is parallel with the wheel planes. The second result of the no-slip assumption is that one can relate the velocity of point $(x_1, x_2)$, the midpoint of the line attaching the center of the wheels, to the rotational velocity of the wheels.

Before writing the kinematic equations of motion for the robot, one has to define the configuration variables of the robot. Let the coordinates of point $(x_1, x_2)$ define the global position of the robot with respect to the inertial coordinate system $x_1 - x_2$. Consider a line that is perpendicular to the wheel axis and goes through the point $(x_1, x_2)$ as an orientation reference for the robot. The angle that this line makes with

the positive $x_1$ axis, $\theta$, represents the orientation of the robot. The three variables that define the geometrical configuration of the robot at any given time are

$$\mathbf{q} = \begin{bmatrix} x_1 \\ x_2 \\ \theta \end{bmatrix}. \tag{6.1}$$

Assume that the point $(x_1, x_2)$ on the robot moves with a linear speed of $v$, while the robot has an angular velocity of $\omega$. Now, one can use the first direct result of the no-slip assumption and write the velocity components of the point $(x_1, x_2)$ in the inertial frame as

$$\begin{aligned} \dot{x}_1 &= v\cos\theta, \\ \dot{x}_2 &= v\sin\theta. \end{aligned} \tag{6.2}$$

Also, the rate of change of the robot's orientation is

$$\dot{\theta} = \omega. \tag{6.3}$$

Combining Eqs. (6.2) and (6.3) results in the kinematic equations of motion of the robot, which can be written in the following matrix form:

$$\dot{\mathbf{q}} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}, \tag{6.4}$$

where $\mathbf{u} = \begin{bmatrix} v & \omega \end{bmatrix}^T$. Once the input vector $\mathbf{u}$ is known as a function of time, Eq. (6.4) can be numerically integrated to predict the motion of the robot. Note that one can choose inputs different than the ones that are used here. Example of other sets of inputs are the rotational velocity of the wheels, the linear velocity of the point $(x_1, x_2)$ and the difference of the linear velocity of the wheels, etc.

*Example 6.1.* Consider the Hilare-type mobile robot shown in Fig. 6.1. Assume the inputs are selected to be the linear velocity of the robot $v_1$ and the difference of the velocity of the two wheels $v_2 = v_r - v_l$, where $v_r$ and $v_l$ are the velocities of the right and the left wheel centers, respectively. Revise the kinematic equations of motion of the robot and write them in terms of the new inputs.

*Solution.* The linear velocity of the robot is assumed to be $v_1$, therefore $v = v_1$. Also, the angular velocity of the robot can be calculated as $\omega = (v_r - v_l)/T = v_2/T$, where $T$ is the robot's track (see Fig. 6.1). With these in mind, one can write the revised kinematic equation of motion as

$$\dot{\mathbf{q}} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & \frac{1}{T} \end{bmatrix} \mathbf{u}, \tag{6.5}$$

where $\mathbf{u} = \begin{bmatrix} v_1 & v_2 \end{bmatrix}^T$. This completes the solution to this example.

### 6.2.2 Car-Like Mobile Robots

Consider the car-like robot shown in Fig. 6.2. The body coordinate system of the robot has an origin at the midpoint of the rear axle of the robot. The longitudinal axis $x_{r1}$ points toward the front of the robot and the transversal axis $x_{r2}$ point toward the left wheel. The geometrical configuration of the robot at any given time can be defined by knowing four variables, the two components of the global position of the origin of the body frame, $(x_1, x_2)$, the angle between the robot's longitudinal axis and the inertial $x_1$ axis, $\theta$ and the steering angle, $\phi$, (the angle between the plane of the front wheel and the body axis $x_{r1}$). The configuration variables are grouped togother as

$$\mathbf{q} = \begin{bmatrix} x_1 \\ x_2 \\ \theta \\ \phi \end{bmatrix}^T . \tag{6.6}$$

The drive mechanism of a car-like robot, as its name implies, is similar to that of a car. The driving inputs for such a robot are the linear velocity of the origin of the body coordinate system of the robot, $v_1$, and the time rate of the steering angle of the front wheels, $v_2$. These inputs are grouped as

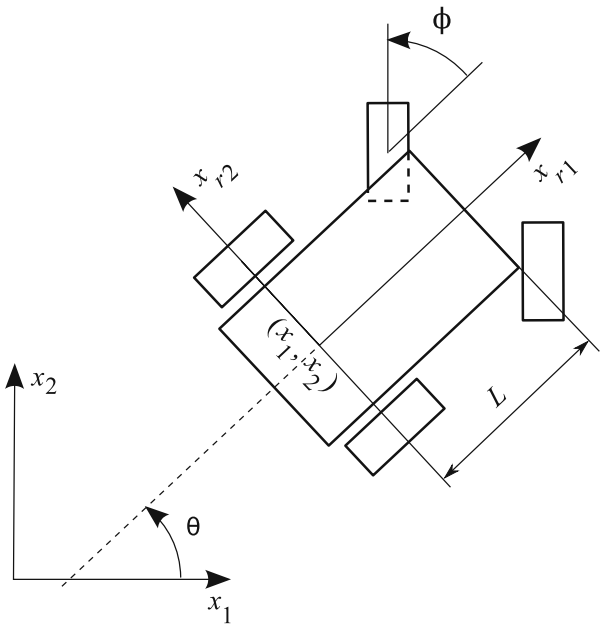


**Fig. 6.2** A car-like mobile robot

$$\mathbf{u} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}^T. \tag{6.7}$$

Here, we are going to derive the differential relations between the configuration variables and the driving inputs, known as the kinematic model of the robot. For deriving these relations, we, once again, assume the no-slip condition, i.e., the acceleration of the motion of the car-like robot is such that the interaction forces between the tires and floor (ground) do not exceed their maximum allowable static friction. One of the consequences of this assumption is that the robot wheels have no lateral velocity component. Based on this, we can also assume that the lateral component of the velocity of the midpoint of the front and the rear axle are zero. We must use the mathematical representation of the no-slip condition to derive the kinematic equations of the robot. The lateral no-slip condition for the midpoint of the front and the rear wheel axles can be expressed as

$$\dot{x}_1 \sin\theta - \dot{x}_2 \cos\theta = 0, \tag{6.8}$$

$$\dot{x}_{f1} \sin(\theta + \phi) - \dot{x}_{f2} \cos(\theta + \phi) = 0, \tag{6.9}$$

where $(x_{f1}, x_{f2})$ is the Cartesian position of the front wheel center. A more useful form of the lateral no-slip condition for the front wheel, Eq. (6.9), would solely be expressed in terms of the configuration variables and their derivate. This form can be obtained by using the following relations

$$\begin{aligned} x_{f1} &= x_1 + L\cos\theta, \\ x_{f2} &= x_2 + L\sin\theta, \end{aligned} \tag{6.10}$$

which indicate that the robot is a rigid body, and one can calculate the position of the midpoint of the front axle by knowing the position of the midpoint of the rear axle, the orientation of the robot, and the wheel-base $L$ of the robot. The derivatives of Eqs. (6.10) are substituted in Eq. (6.9) to obtain the more useful form of

$$\dot{x}_1 \sin(\theta + \phi) - \dot{x}_2 \cos(\theta + \phi) - L\dot{\theta}\cos\phi = 0 \tag{6.11}$$

for the front wheel no-slip condition.

Now, we are ready to write the kinematic model of the car-like robot. The velocity components of the midpoint of the rear axle, i.e., the origin of the robot's body frame, expressed in the inertial coordinate system are

$$\begin{aligned} \dot{x}_1 &= \dot{x}_{r1}\cos\theta - \dot{x}_{r2}\sin\theta, \\ \dot{x}_2 &= \dot{x}_{r1}\sin\theta + \dot{x}_{r2}\cos\theta, \end{aligned} \tag{6.12}$$

where $\dot{x}_{r1}$ and $\dot{x}_{r2}$ are the longitudinal and lateral velocity components expressed in the robot's body frame. The no-slip condition dictates that $\dot{x}_{r2} = 0$. Also, the linear velocity of the robot was assumed to be a driving input, i.e., $\dot{x}_{r1} = v_1$ is a driving input. Using these information, one can write the first two kinematic equations of motion for the robot based on Eqs. (6.12):

$$\begin{aligned} \dot{x}_1 &= v_1 \cos\theta, \\ \dot{x}_2 &= v_1 \sin\theta. \end{aligned} \tag{6.13}$$

The third kinematic equation is obtained by substituting Eqs. (6.13) in the front wheel no-slip condition Eq. (6.11), which results in

$$(\cos\theta \sin(\theta + \phi) - \sin\theta \cos(\theta + \phi))v_1 - L\dot{\theta} \cos\phi = 0, \tag{6.14}$$

which after simplification yields to

$$\dot{\theta} = \frac{\tan\phi}{L} v_1. \tag{6.15}$$

The last kinematic equation is obtained by considering the fact that the rate of the steering angle has been selected as the second driving input, i.e.,

$$\dot{\phi} = v_2. \tag{6.16}$$

The whole kinematic model, in matrix form, is derived by gathering Eqs. (6.13), (6.15), and (6.16).

$$\dot{\mathbf{q}} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ \frac{\tan\phi}{L} & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}. \tag{6.17}$$

By specifying the driving input, $\mathbf{u} = [v_1, v_2]^T$, one can predict the motion of the car-like robot using the kinematic model (6.17).

## 6.3 Trajectory-Tracking Control Based on Kinematic Models

### *6.3.1 Hilare-Type Mobile Robots*

A trajectory-tracking controller is needed for the robots to be able to follow the planned trajectory [57]. In this section, we present the controller based on the kinematic model of Hilare nonholonomic robots, as shown in Fig. 6.1. The control inputs for the mobile robot are assumed to be $v$ and $\omega$ as defined previously. The kinematics equations of a Hilare robot are written as

$$\begin{aligned} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= \omega. \end{aligned} \tag{6.18}$$

Assume that the desired trajectory for the mobile robot in the inertial coordinate system is defined by selecting the two position components of the origin of the robot's body frame as functions of time.

$$\begin{aligned} x_1^d &= x_1^d(t), \\ x_2^d &= x_2^d(t). \end{aligned} \tag{6.19}$$

The desired velocity components of the robot can be derived by differentiating Eqs. (6.19).

$$\begin{aligned} \dot{x}_1^d &= \dot{x}_1^d(t), \\ \dot{x}_2^d &= \dot{x}_2^d(t). \end{aligned} \tag{6.20}$$

Note that these velocities are in general a function of time, which implies the robot must follow a desired user-defined speed. This user-defined speed is also determinted when the user is defining the desired positions as a function of time. In fact, the desired path of the robot, as a function of $x_1$ and $x_2$ only, can be found by eliminating the time $t$ between the desired position components listed in Eq. (6.19). Therefore, the desired postions as defined in Eq. (6.19) contain the information about the desired speed as well as the geometry of the desired path. A user must keep this in mind while determining these functions.

Once the desired velocity components are derived from Eq. (6.20), the desired orientation of the robot must be derived. Since the Hilare-type mobile robot has a nonholonomic constraint, any desired orientation can not be selected by the user. The desired orientation must conform to the robot's kinematic constraint, i.e., it has to obey the lateral no-slip condition. The lateral no-slip condition states that there is no lateral slip, i.e., the lateral velocity of the robot is zero at all times. The no-slip condition can mathematically be expressed as follows:

$$\dot{x}_{r2} = -\dot{x}_1\sin\theta + \dot{x}_2\cos\theta = 0. \tag{6.21}$$

The above relation dictates that once the desired velocities are determined from Eq. (6.20), the desired orientation can be derived as follows:

$$\theta^d(t) = \arctan\left(\frac{\dot{x}_2^d(t)}{\dot{x}_1^d(t)}\right). \tag{6.22}$$

By observing the kinematic model, one can find a new set of configuration variables that can simplify the kinematic model. These new variables are listed in the following:

$$\begin{aligned} z_1 &= x_1, \\ z_2 &= \tan\theta, \\ z_3 &= x_2. \end{aligned} \tag{6.23}$$

One can show that the new configuration variables, along with the following new control inputs can further simplify the form of the robot's kinematic equations. The new control inputs are listed below:

$$\begin{aligned} u_1 &= v\cos\theta, \\ u_2 &= \omega(1+\tan^2\theta). \end{aligned} \tag{6.24}$$

Applying these new variables to Eqs. (6.2) and (6.3) results in a new form for the kinematic equations of motion of the robot.

$$\begin{aligned} \dot{z}_1 &= u_1, \\ \dot{z}_2 &= u_2, \\ \dot{z}_3 &= z_2 u_1. \end{aligned} \tag{6.25}$$

This equation is known as the "chain form" for kinematic equations of the robot. In fact, this form can be extended for mobile robots with other types of drive mechanism (e.g., car-like robots) that have different number of DOFs (configuration variables). The general formula of the chain form is

$$\begin{aligned} \dot{z}_1 &= u_1, \\ \dot{z}_2 &= u_2, \\ \dot{z}_k &= z_{k-1}u_1, \quad (k = 3, \ldots, n). \end{aligned} \tag{6.26}$$

The desired trajectory for the new configuration variables can be found by substituting Eqs. (6.22) into Eqs. (6.23):

$$\begin{aligned} z_1^d(t) &= x_1^d(t), \\ z_2^d(t) &= \frac{\dot{x}_2^d(t)}{\dot{x}_1^d(t)}, \\ z_3^d(t) &= x_2^d(t). \end{aligned} \tag{6.27}$$

One can determine the new inputs corresponding to the new desired configuration variables by substituting Eqs. (6.27) into Eqs. (6.25):

$$
\begin{aligned}
u_1^d &= \dot{x}_1^d(t), \\
u_2^d &= \frac{d}{dt}\left(\frac{\dot{x}_2^d(t)}{\dot{x}_1^d(t)}\right), \\
&= \frac{\ddot{x}_2^d(t)\dot{x}_1^d(t) - \ddot{x}_1^d(t)\dot{x}_2^d(t)}{(\dot{x}_1^d(t))^2}.
\end{aligned} \tag{6.28}
$$

Now, our goal is to control the new kinematic equations (6.25) to follow the desired new trajectory (6.27). We write the nonlinear error equations as

$$
\begin{aligned}
\dot{\tilde{z}}_1 &= \tilde{u}_1, \\
\dot{\tilde{z}}_2 &= \tilde{u}_2, \\
\dot{\tilde{z}}_3 &= z_2^d \tilde{u}_1 + \tilde{z}_2 u_1^d + \tilde{z}_2 \tilde{u}_1,
\end{aligned} \tag{6.29}
$$

where

$$\tilde{z}_i = z_i - z_i^d, \qquad i = 1, 2, 3, \tag{6.30}$$

$$\tilde{u}_i = u_i - u_i^d, \qquad i = 1, 2. \tag{6.31}$$

Now, we linearize the nonlinear system (6.29) by neglecting the term $\tilde{z}_2\tilde{u}_1$ and assume the following time-variant feedback control law,

$$
\begin{bmatrix} \tilde{u}_1 \\ \tilde{u}_2 \end{bmatrix} = \begin{bmatrix} k_1 & 0 & 0 \\ 0 & k_2 & k_3/u_1^d \end{bmatrix} \begin{bmatrix} \tilde{z}_1 \\ \tilde{z}_2 \\ \tilde{z}_3 \end{bmatrix}, \tag{6.32}
$$

in which $k_1$, $k_2$, and $k_3$ are constant controller gains. One can show that by applying the control law (6.32) on the linearized form of state error equations (6.29) the following linear time-variant closed loop system is obtained:

$$
\begin{bmatrix} \dot{\tilde{z}}_1 \\ \dot{\tilde{z}}_2 \\ \dot{\tilde{z}}_3 \end{bmatrix} = \begin{bmatrix} k_1 & 0 & 0 \\ 0 & k_2 & k_3/u_1^d \\ k_1 z_1^d & u_1^d & 0 \end{bmatrix} \begin{bmatrix} \tilde{z}_1 \\ \tilde{z}_2 \\ \tilde{z}_3 \end{bmatrix}. \tag{6.33}
$$

Note that although the linear system (6.33) is time dependent, its characteristic equation is time independent. Therefore, if one selects the controller gains as

$$k_1 = -\lambda_1, \qquad k_2 = -2\lambda_2, \qquad k_3 = -(\lambda_2^2 + \lambda_3^2), \tag{6.34}$$

where $\lambda_1$ and $\lambda_2$ are positive constants. Then, the closed loop system (6.33) will be asymptotically stable with poles at $-\lambda_1$, $-\lambda_2 + i\lambda_3$, and $-\lambda_2 - i\lambda_3$.

Finally, the new inputs, $\tilde{u}_1$ and $\tilde{u}_2$, can be computed based on configuration errors with respect to the trajectory, $\tilde{z}_1$ to $\tilde{z}_3$, from the feedback control law (6.32). By using

Eqs. (6.31) and (6.24), one can obtain the control laws for the original inputs, the robot linear and angular velocity, $v$ and $\omega$.

$$v = \frac{\tilde{u}_1 + u_1^d}{\cos\theta},$$
$$\omega = \frac{\tilde{u}_2 + u_2^d}{1 + \tan^2\theta}. \tag{6.35}$$

*Example 6.2.* Consider a Hilare-type mobile robot. Assume that the robot is initially located at (0.0, 0.5) m. Use the derived control laws to control the robot on the following two paths.

(a). A piecewise-linear path defined by

$$x_2 = f(x_1) = \begin{cases} x_1 & \text{if} \quad 0 < x_1 \leq 3\frac{1}{3} \\ 3\frac{1}{3} & \text{if} \quad 1\frac{1}{3} < x_1 \leq 6\frac{2}{3} \\ 10 - x_1 & \text{if} \quad 6\frac{2}{3} < x_1 \leq 10 \\ 0 & \text{if} \quad x_1 > 10 \end{cases}, \tag{6.36}$$

where all the distances are in meters. Assume that the robot must move with a desired velocity of $v^d = \sqrt{2}/2$ m/s on the first and the third segment, while it must have a velocity of $v^d = 1/2$ m/s on the second and the fourth segment.

(b). A sinusoidal path with an amplitude of 2 m defined by

$$x_2 = f(x_1) = 2\sin(x_1) \quad 0 < x_1 < 13\frac{1}{3}, \tag{6.37}$$

where all the distances are in meters. Assume that the robot has to keep a constant velocity component of 1/2 m/s in the $x_1$ direction.

*Solution.* First, the desired position, velocity, and acceleration, as functions of time, required by Eqs. (6.27) and (6.28), must be derived based on the assumptions given in the problem definition. Then, the desired values along with the current values of the configuration variables are used in Eqs. (6.30) and (6.31) to calculate the errors at any given time. These errors in turn are used in Eq. (6.32) to find the control commands $\tilde{u}_1$ and $\tilde{u}_2$. These control commands are transformed into the actual linear and angular velocity control commands $v$ and $\omega$ by using Eq. (6.35). The actual control commands are applied to the kinematic equations (6.18) to obtain the simulated control behavior of the mobile robot. Here, the results of each part are discussed separately.

(a). The desired position components as functions of time become

$$x_1^d(t) = \begin{cases} 0.5t & \text{if} \quad 0 < t \leq 6\frac{2}{3} \\ 0.5t & \text{if} \quad 6\frac{2}{3} < t \leq 13\frac{1}{3} \\ 0.5t & \text{if} \quad 13\frac{1}{3} < t \leq 20 \\ 0.5t & \text{if} \quad t > 20 \end{cases}, \tag{6.38}$$

and

$$x_2^d(t) = \begin{cases} 0.5t & \text{if} \quad 0 < t \leq 6\frac{2}{3} \\ 3\frac{1}{3} & \text{if} \quad 6\frac{2}{3} < t \leq 13\frac{1}{3} \\ 3\frac{1}{3} - 0.5(t - 13\frac{1}{3}) & \text{if} \quad 13\frac{1}{3} < t \leq 20 \\ 0 & \text{if} \quad t > 20 \end{cases}. \tag{6.39}$$

The desired velocity components as functions of time become

$$\dot{x}_1^d(t) = \begin{cases} 0.5 & \text{if} \quad 0 < t \leq 6\frac{2}{3} \\ 0.5 & \text{if} \quad 6\frac{2}{3} < t \leq 13\frac{1}{3} \\ 0.5 & \text{if} \quad 13\frac{1}{3} < t \leq 20 \\ 0.5 & \text{if} \quad t > 20 \end{cases}, \tag{6.40}$$

and

$$\dot{x}_2^d(t) = \begin{cases} 0.5 & \text{if} \quad 0 < t \leq 6\frac{2}{3} \\ 0 & \text{if} \quad 6\frac{2}{3} < t \leq 13\frac{1}{3} \\ -0.5 & \text{if} \quad 13\frac{1}{3} < t \leq 20 \\ 0 & \text{if} \quad t > 20 \end{cases}. \tag{6.41}$$

The desired acceleration components as functions of time become

$$\ddot{x}_1^d(t) = 0, \quad \ddot{x}_2^d(t) = 0, \quad \text{for all } t. \tag{6.42}$$

By using this desired trajectory and applying the control laws as discussed in this section, one can simulate the response of the mobile robot. For these simulations, the values for $\lambda_1$ and $\lambda_2$, which determine the convergence rate of the robot to the path, are selected to be 3. The simulation time, 26 2/3 s, was selected such that the robot can complete the path with specified velocity.

The path that the robot follows is shown in Fig. 6.3. Although the desired path has sharp corners, the robot has been able to follow it rather closely. At the sharp corners, the robot has a transient response, however, it aligns itself with the linear path after a while.

The orientation of the robot is shown in Fig. 6.4. Note that the desired orientation is not directly fedback in the control system. However, it is enforced indirectly by defining the configuration variable $z_2^d$ in Eq. (6.27), which is

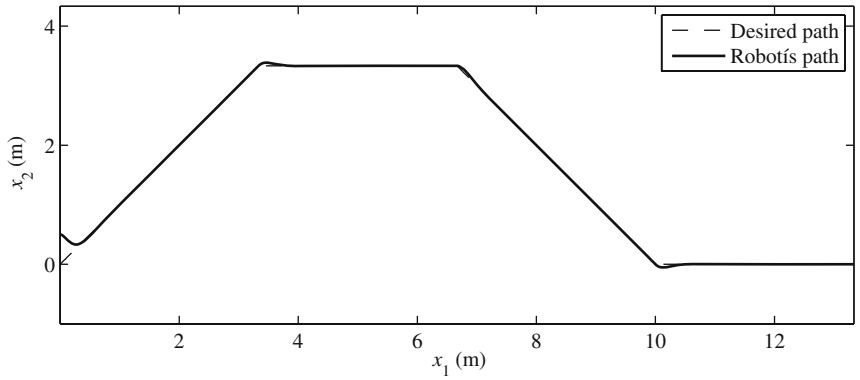


**Fig. 6.3** A Hilare mobile robot tracking a piecewise-linear path

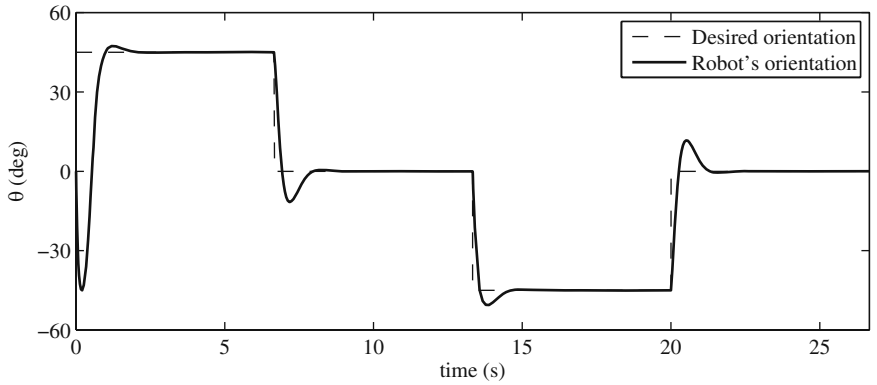


**Fig. 6.4** The orientation of a Hilare mobile robot tracking a piecewise-linear path

related to the orientation $\theta$ via Eq. (6.23). It can be seen from Fig. 6.4 that the robot aligns its orientation with the desired path rapidly.

(b). The desired position components as functions of time become

$$x_1^d(t) = 0.5t, \tag{6.43}$$

and

$$x_2^d(t) = 2\sin(0.5t). \tag{6.44}$$

The desired velocity components as functions of time become

$$\dot{x}_1^d(t) = 0.5, \tag{6.45}$$

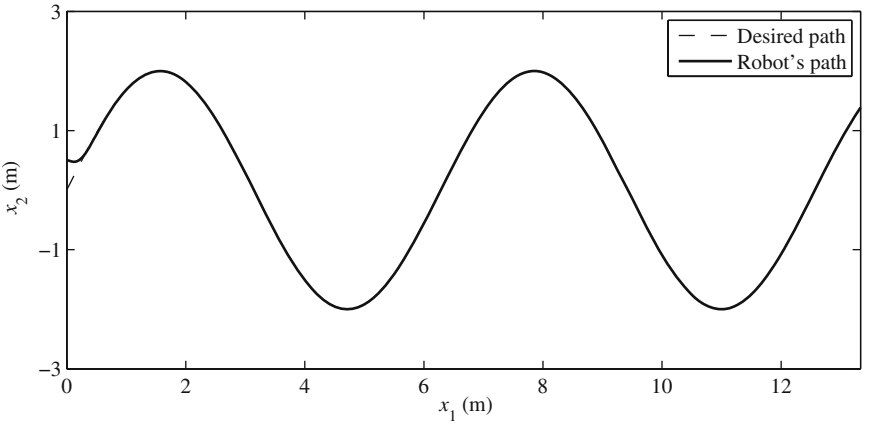


**Fig. 6.5** A Hilare mobile robot tracking a sinusoidal path

and

$$\dot{x}_2^d(t) = 2(0.5)\cos(0.5t). \tag{6.46}$$

The desired acceleration components as functions of time become

$$\ddot{x}_1^d(t) = 0, \tag{6.47}$$

and

$$\ddot{x}_2^d(t) = -2(0.5)^2 \sin(0.5t). \tag{6.48}$$

The path that the robot follows is shown in Fig. 6.5. Although the desired path's curvature dynamically changes, the robot has been able to follow it closely. Even at the sharper corners, the robot does not have any transient response, because the path is smooth and no sudden errors are imposed on the system, unlike the case with the piecewise-linear path.

The orientation of the robot is shown in Fig. 6.6. This figure shows that, after a transient response, the robot keeps its orientation aligned with the desired path at all times. The transient response exists because the robot is facing toward the positive $x_1$-axis at the beginning of the motion.

### *6.3.2 Car-Like Mobile Robots*

The trajectory-tracking controller design for a car-like robot is very similar to that for a Hilare-type robot. The difference is the number of configuration variables and the driving inputs, which will be used as control inputs. The kinematic equations of motion for a car-like robot are

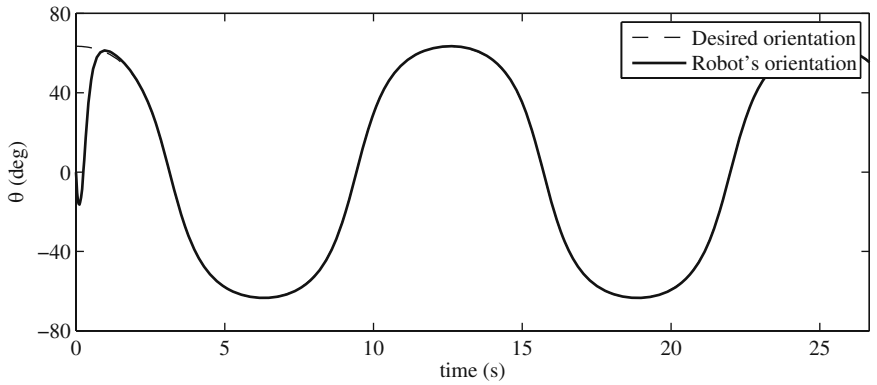


**Fig. 6.6** The orientation of a Hilare mobile robot tracking a sinusoidal path

$$\begin{aligned}
\dot{x}_1 &= v_1 \cos\theta, \\
\dot{x}_2 &= v_1 \sin\theta, \\
\dot{\theta} &= \frac{\tan\phi}{L} v_1, \\
\dot{\phi} &= v_2.
\end{aligned} \tag{6.49}$$

Here, we must define the desired trajectory for the car-like robot. The desired position components for the midpoint of the rear axle are

$$\begin{aligned}
x_1^d &= x_1^d(t), \\
x_2^d &= x_2^d(t).
\end{aligned} \tag{6.50}$$

Any desired velocity, acceleration, and jerk (the time derivative of acceleration) can be derived from this desire position as functions of time. A user must be careful defining these desired position components as functions of time because they not only contain information about the desired position, but also contain information regarding the desired velocity, acceleration, and even jerk.

For the purpose of controller development, let us define a new set of configuration variables similar to what was defined for the Hilare mobile robot. These new sets of configuration variables are

$$\begin{aligned}
z_1 &= x_1, \\
z_2 &= \frac{\tan\phi}{L\cos^3\theta}, \\
z_3 &= \tan\theta, \\
z_4 &= x_2.
\end{aligned} \tag{6.51}$$

By substituting the new configuration variables (6.51) into the kinematic model, Eq. (6.49), one can see that the following new control inputs are relevant.

$$u_1 = v_1 \cos\theta,$$
$$u_2 = \frac{(3\sin\theta \sin^2\phi)v_1 + (L\cos\theta)v_2}{L\cos^2\theta\cos^2\phi}. \tag{6.52}$$

Using the new driving inputs (6.52), one can reduce the kinematic model of the car-like robot to the "chain form," as follows.

$$\dot{z}_1 = u_1,$$
$$\dot{z}_2 = u_2,$$
$$\dot{z}_3 = z_2 u_1, \tag{6.53}$$
$$\dot{z}_4 = z_3 u_1. \tag{6.54}$$

Note that these equations conform to the general chain form template given by Eq. (6.26) with $n = 4$. The desired driving inputs $u_1^d$ and $u_2^d$ as functions of time are derived by combining Eq. (6.50) and its derivatives and the new configuration variables defined in Eq. (6.51).

$$z_1^d(t) = x_1^d(t),$$
$$z_2^d(t) = \frac{\ddot{x}_2^d(t)\dot{x}_1^d(t) - \ddot{x}_1^d(t)\dot{x}_2^d(t)}{(\dot{x}_1^d(t))^3},$$
$$z_3^d(t) = \frac{\dot{x}_2^d(t)}{\dot{x}_1^d(t)}, \tag{6.55}$$
$$z_4^d(t) = x_2^d(t).$$

One can determine the new inputs corresponding to the new desired configuration variables by substituting Eqs. (6.56) into Eqs. (6.54).

$$u_1^d(t) = \dot{x}_1^d(t),$$
$$u_2^d(t) = \frac{\dddot{x}_2^d(t)(\dot{x}_1^d(t))^2 - \dddot{x}_1^d(t)\dot{x}_1^d(t)\dot{x}_2^d(t) - 3\ddot{x}_2^d(t)\dot{x}_1^d(t)\ddot{x}_1^d(t) + 3\dot{x}_2^d(t)(\ddot{x}_1^d(t))^2}{(\dot{x}_1^d(t))^4}. \tag{6.56}$$

After deriving the new desired configuration variables, one can define the error in the new configuration variables as

$$\tilde{z}_i = z_i - z_i^d, \quad i = 1, 2, 3, 4, \tag{6.57}$$
$$\tilde{u}_i = u_i - u_i^d, \quad i = 1, 2. \tag{6.58}$$

The kinematic equations in terms of the new configuration variables and the defined errors are obtained by combining the error Eqs. (6.58) with the kinematic Eqs. (6.54). The result is

$$\begin{aligned}
\dot{\tilde{z}}_1 &= \tilde{u}_1,\\
\dot{\tilde{z}}_2 &= \tilde{u}_2,\\
\dot{\tilde{z}}_3 &= z_2^d\tilde{u}_1 + \tilde{z}_2 u_1^d + \tilde{z}_2\tilde{u}_1,\\
\dot{\tilde{z}}_4 &= z_3^d\tilde{u}_1 + \tilde{z}_3 u_1^d + \tilde{z}_3\tilde{u}_1.
\end{aligned} \tag{6.59}$$

Let us assume that the terms $\tilde{z}_2\tilde{u}_1$ and $\tilde{z}_3\tilde{u}_1$ are small compared to the other terms in their corresponding equations. Then, the error system (6.59) can be written as a time dependent linear system as

$$\dot{\tilde{\mathbf{z}}} = \begin{bmatrix} 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & u_1^d(t) & 0 & 0\\ 0 & 0 & u_1^d(t) & 0 \end{bmatrix}\tilde{\mathbf{z}} + \begin{bmatrix} 1 & 0\\ 0 & 1\\ z_2^d(t) & 0\\ z_3^d(t) & 0 \end{bmatrix}\tilde{\mathbf{u}}. \tag{6.60}$$

We select the following time varying linear feedback control law, which can be proved to guarantee asymptotic stability for linear time varying systems.

$$\tilde{\mathbf{u}} = \begin{bmatrix} k_1 & 0 & 0 & 0\\ 0 & k_2 & \frac{k_3}{u_1^d(t)} & \frac{k_4}{(u_1^d(t))^2} \end{bmatrix}\tilde{\mathbf{z}}. \tag{6.61}$$

The parameters $k_1$, $k_2$, $k_3$, and $k_4$ are the control gains. To obtain a fast convergence to zero in the trajectory tracking, the gains should be chosen in such a way that the closed-loop eigenvalues are located in the left half plane. For example, if we would like to have two negative real eigenvalues $\lambda_1$ and $\lambda_2$ and two eigenvalues with negative real part, with modulus $\omega_n$ and damping coefficient $\zeta$, the gains should be

$$\begin{aligned}
k_1 &= -\lambda_1,\\
k_2 &= -(\lambda_2 + 2\zeta\omega_n),\\
k_3 &= -(\omega_n^2 + 2\zeta\omega_n\lambda_2), &(6.62)\\
k_4 &= -(\omega_n^2\lambda_2). &(6.63)
\end{aligned}$$

Finally, the physical control inputs can be calculated based on the control inputs derived in Eq. (6.61). Using Eq. (6.52), one can find the physical control inputs as

$$\begin{aligned}
v_1 &= \frac{1}{\cos\theta}(\tilde{u}_1 + u_1^d),\\
v_2 &= \frac{-3\sin\theta\sin^2\phi}{L\cos^2\theta}(\tilde{u}_1 + u_1^d) + L\cos^3\theta\cos^2\phi(\tilde{u}_2 + u_2^d).
\end{aligned} \tag{6.64}$$

By applying the control law (6.64) to the car-like robot modeled by kinematic equations (6.49), one can make the robot drive on the desired trajectory defined by Eq. (6.50).

*Example 6.3.* Consider a car-like robot with a wheel-base of $L = 0.2\,\text{m}$. Assume that the robot is initially located at (0, 0.5) meter. Use the derived control laws (6.64) to control the robot on the two paths defined in parts (a) and (b) of Example 6.2.

*Solution.* Deriving the desired position, velocity, and acceleration of the robot as functions of time for this example is similar to what was done in Example 6.2. However, for a car-like robot, the desired jerk (first time derivative of the acceleration) must also be determined. This is simply done by differentiating the desired acceleration. One can simulate the response of the car-like robot by using the above desired trajectory and the control laws derived in Eq. (6.64) and applying them to the car-like kinematic model given by Eqs. (6.49). The following controller gains have been used for this simulation.

$$\lambda_1 = \lambda_2 = -10, \quad \omega_n = 1.0, \quad \zeta = 1.0. \tag{6.65}$$

The simulation time is 26 2/3 s, so that the robot can complete the paths with its specified velocity. The simulations results for part (a) and (b) follow.

(a) Piecewise-linear path: The desired jerk is

$$\dddot{x}_1^d(t) = 0, \quad \dddot{x}_2^d(t) = 0, \quad \text{for all } t. \tag{6.66}$$

The path that the robot follows under the closed-loop control is shown in Fig. 6.7. The piecewise linear desired path has sharp corners that are not possible for a car-like robot to follow. This is because the minimum turn radius of a car-like robot is limited. This fact is reflected in Fig. 6.7. The robot departs from the path at the sharp corners no matter how high the control gains are. Too high control gains result in a very jerky steering angle output form the controller. Hilare-type mobile robots can deal better with sharp corners because their minimum turn radius is zero, i.e., they can turn around themselves at a fixed position. Comparison of Fig. 6.7 with Fig. 6.3 confirms this claim.

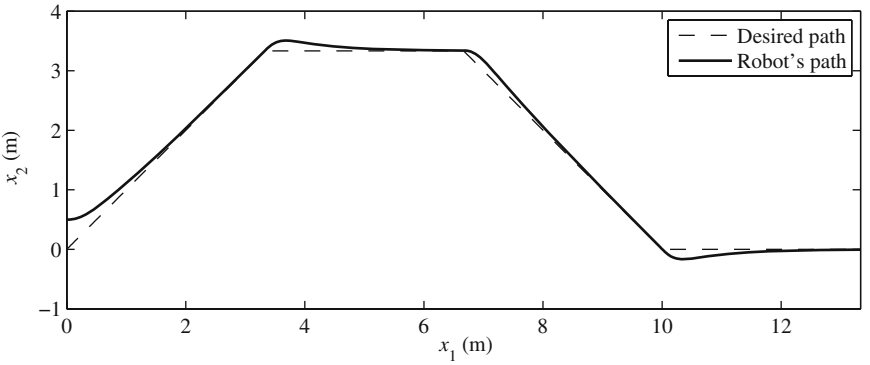


**Fig. 6.7** A car-like mobile robot tracking a piecewise linear path

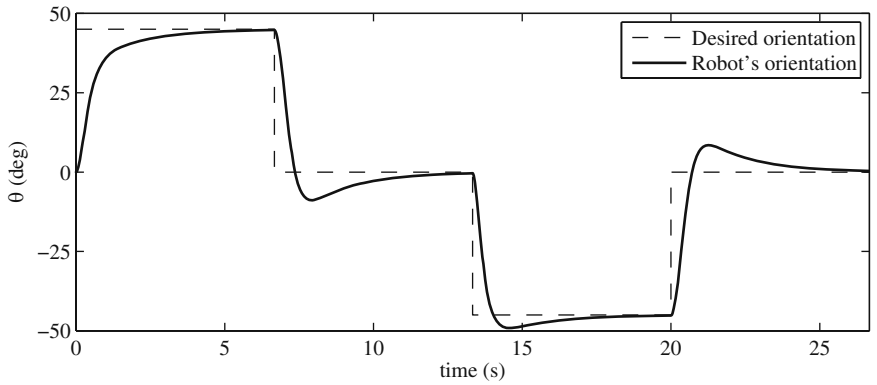


**Fig. 6.8** The orientation of a car-like mobile robot tracking a piecewise linear path

The orientation of the car-like robot is shown in Fig. 6.8. It can be seen that the orientation response of the car-like robot is also slower than that of a Hilare mobile robot (Fig. 6.4). This is once again because of the kinematic property of a car-like robot, which has a nonzero minimum turn radius.

Figure 6.9 shows the steering angle of the car-like robot, which is one of the control inputs. At the sharp corners, the steering angle changes suddenly. The peak of this steering angle depends on the controller gains. Larger controller gains cause higher peaks for the steer angle input. The physical limit of the steer angle must be considered when tuning the controller.

(b) Sinusoidal path: the desire jerk is

$$\dddot{x}_1^d(t) = 0, \quad \dddot{x}_2^d(t) = -2(0.5)^3 \cos(0.5t). \tag{6.67}$$

The path that the robot follows under the closed-loop control is shown in Fig. 6.10. The sinusoidal path is smooth and has no sharp corners and it is easy

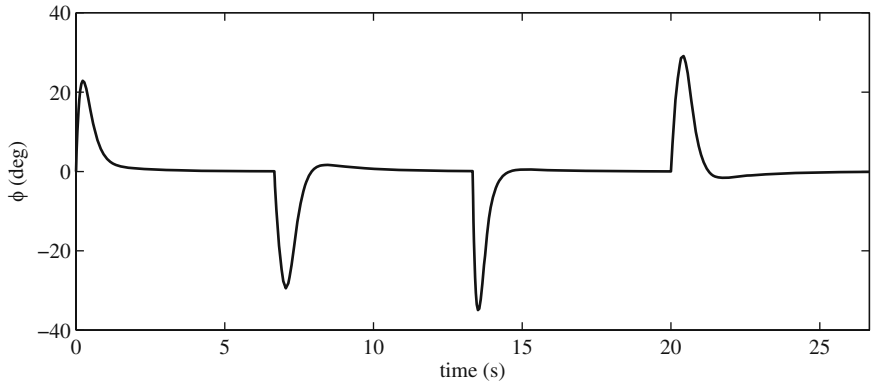


**Fig. 6.9** The steering input for a car-like mobile robot tracking a piecewise-linear path

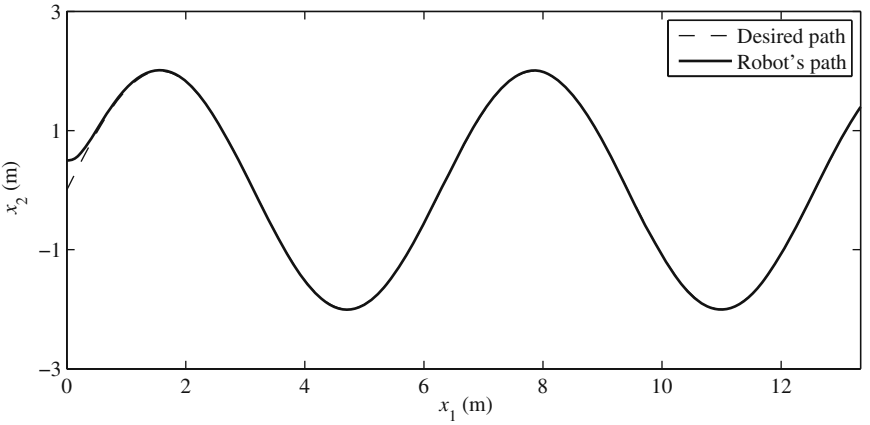


**Fig. 6.10** A car-like mobile robot tracking a sinusoidal path

for the robot to follow, as long as the minimum radius of curvature of the path is not less than the minimum turn radius of the robot. The performance of the car-like robot for this smooth path is rather similar to that of the Hilare robot (Fig. 6.5).

The orientation of the car-like robot is shown in Fig. 6.11. The orientation response of the car-like robot for a smooth path is much better than that for the piecewise linear path. The orientation performance of the car-like robot resembles that of the Hilare robot (Fig. 6.6). One of the control inputs, the steering angle of the car-like robot is shown in Fig. 6.12. At the corners, the steering angle changes faster. Once again, the peak of this steering angle depends on the controller gains. Larger controller gains cause higher peaks for the steer angle input. The physical limit of the steer angle must be considered when tuning the controller. It should be noted that the steering angle is in phase with the

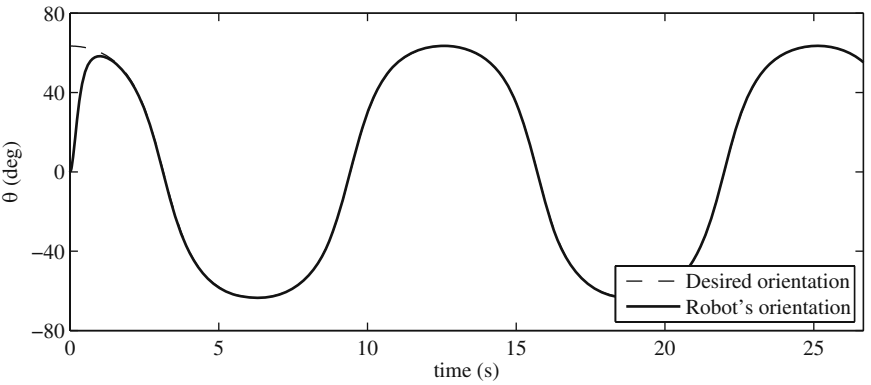


**Fig. 6.11** The orientation of a car-like mobile robot tracking a sinusoidal path

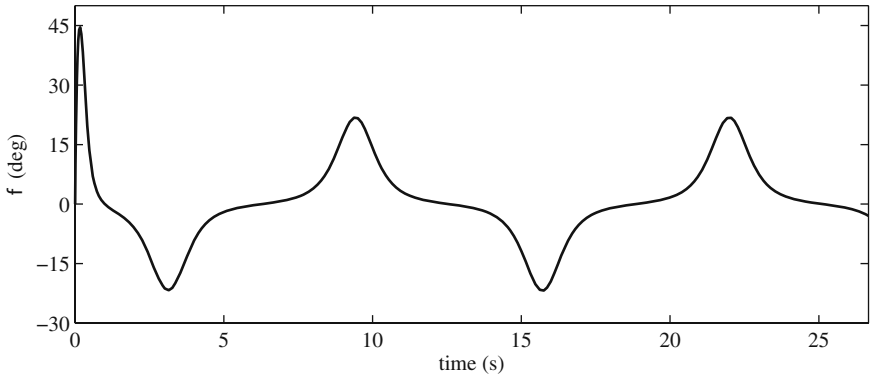


**Fig. 6.12** The steering input for a car-like mobile robot tracking a sinusoidal path

sinusoidal path being tracked. The reason for this is that the steering angle is a representative of the path's curvature. Where the radius of curvature of the path is smaller, the steering angle is larger. This fact can be confirmed by comparing the path (Fig. 6.10) and the steering angle input curves (Fig. 6.12).

## 6.4 Formation Control for Hilare Mobile Robots

This section focuses on control and coordination for multiple robots that have to move as a group with user-defined relative positions, i.e., in formations. To address the control issues of mobile robots moving in formations, several approaches exist. The most common approach is the behavior-based approach [7]. In this approach, emergent or complex behaviors from organizations of many small, individual, and often heterogeneous robots are defined. Generally, this approach either relies on supervisory (centralized) control and process planning, or utilizes theory of evolution to generate a pre-specified system behavior. Although this approach is effective if the behaviors are defined well, the mathematical proof of their performance qualities is very difficult to achieve. This limits the application of the behavior-based approach to situations where uncertainty in the robots behavior can be tolerated, e.g., for soccer robots.

In an industrial setting, where multiple robots may be used for handling a large load, the behavior of the robots must be completely predictable. This predictability must be achieved by using control algorithms with provable performance and derivable limitations. Therefore, the goal being pursued here is to generate rigorously provable measures/bounds on the performance of the multiple robot system.

Here, the problem of control and coordination for many robots moving in formation is considered using decentralized controllers. It is assumed that the overall motion for the formation as a group is given by an external path planner, for example, the potential field obstacle avoidance method presented in Chapter 4. This motion plan is then used to control a single lead robot.

It is also assumed that each robot is aware of the relative position of other robots that are immediately adjacent to it via communication or some kind of vision. While the lead robot follows the given trajectory, the rest of the robots in the formation are controlled by local (decentralized) control laws. The control law for each robot in the formation is derived based on the dynamics of its relative postion with respect to the neighboring robots in the formation. This type of decentralized control law has the advantage of providing easily computable, real-time feedback control, with provable performance for the entire system.

### 6.4.1 Geometrical Leader-Follower Formation Schemes

Based on the methodology introduced in [22], in this section, two schemes for feedback control of the relative distance of the robots within a formation are described. The first scheme is a controller that controls the relative distance and view angle of one robot (the local follower) relative to one neighboring robot (the local leader). This control scheme is used in situations where each robot has one leader, for example, where robots march in a single file or at an edge of a 2D formation geometry. The second scheme is a controller that allows a robot (a local follower) to maintain its position in the formation by keeping specified distances with two neighboring robots (local leaders).

### 6.4.2 Design of the l – α Controller

Two neighboring robots in the formation are shown in Fig. 6.13. The distance of the center of axle of robot 1 and the control point, $p$, of robot 2 is $l_{12}$. The control point is on the longitudinal axis of robot 2 and has a distance $d$ from the center of the axle

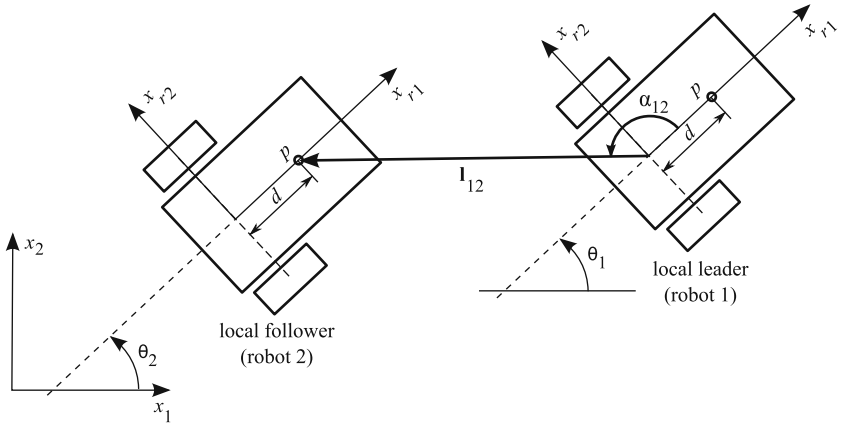


**Fig. 6.13** The $l - \alpha$ control scheme

of robot 2. To maintain the formation, robot 2 must keep a desired distance of $l_{12}^d$ and view angle $\alpha_{12}^d$ with robot 1. A control law for the driving inputs of robot 2, $v_2$ and $\omega_2$, must be determined such robot 2 maintains the desired distance and view angle.

To design such a control law, explicit relations between the control inputs ($v_2$ and $\omega_2$) and the control outputs ($l_{12}$ and $\alpha_{12}$) are required. These relations are known as the input–output relations. In the following, first, these input–output relations are found, then, the control laws are derived. The explicit relations between the input and the output are found by a kinematic analysis of the relative motion of the robots 1 and 2.

#### 6.4.2.1 Kinematic Analysis

A moving coordinate system is assumed with an origin at the center of axle of robot 1. This coordinate system is rotating with the vector $\mathbf{l}_{12}$, which is the line of sight of robot 2 from robot 1's center of axle (Fig. 6.13). Also, two coincident points are assumed. The first one is $p_1$, and is attached to this moving coordinate system. The second one is $p_2$, which is attached to robot 2. Both points are coincident with the instantaneous location of the control point $p$. One can see that, in the eye of an observer attached to the defined moving coordinate system standing at $p_1$, point $p_2$ moves along the vector $\mathbf{l}_{12}$. Mathematically, this can be expressed by the following kinematic equation for the inertial velocity of $p_2$:

$$\begin{aligned}\mathbf{v}_{p2} &= \mathbf{v}_{p1} + \mathbf{v}_{p2/1},\\ &= (\mathbf{v}_1 + \dot{\alpha}_0\hat{\mathbf{k}} \times \mathbf{l}_{12}) + \dot{\mathbf{l}}_{12},\end{aligned} \tag{6.68}$$

where $\mathbf{v}_1$ is the velocity of the center of axle of robot 1, and

$$\alpha_0 = \theta_1 + \alpha_{12}, \tag{6.69}$$

$$\dot{\alpha}_0 = \omega_1 + \dot{\alpha}_{12}. \tag{6.70}$$

Now, the the velocity of point $p_2$ on robot 2 can also be written in terms the velocity of the center of axle of the robot 2, $\mathbf{v}_2$:

$$\mathbf{v}_{p2} = \mathbf{v}_2 + \omega_2\hat{\mathbf{k}} \times \mathbf{d}. \tag{6.71}$$

The vectorial Eqs. (6.68) and (6.71) are combined, expanded, and solved for the highest derivatives of the outputs, $\dot{l}_{12}$ and $\dot{\alpha}_{12}$, to result in

$$\dot{l}_{12} = v_2 \cos\gamma_1 - v_1 \cos\alpha_{12} + d\omega_2 \sin\gamma_1, \tag{6.72}$$

$$\dot{\alpha}_{12} = \tfrac{1}{l_{12}}(v_1 \sin\alpha_{12} - v_2 \sin\gamma_1 + d\omega_2 \cos\gamma_1 - l_{12}\omega_1), \tag{6.73}$$

where

$$\gamma_1 = \theta_1 + \alpha_{12} - \theta_2. \tag{6.74}$$

#### 6.4.2.2 Input–Output Equations

The kinematic equations (6.72) and (6.73) already relate the controller inputs $[v_2, \omega_2]$ of the follower robot to the formation scheme outputs $[l_{12}, \alpha_{12}]$, and provide the input–output equations. Here, the input–output relations are written in the matrix form:

$$\dot{\mathbf{z}} = \mathbf{f} + \mathbf{b}\mathbf{u}, \tag{6.75}$$

where

$$\mathbf{z} = \begin{bmatrix} l_{12} \\ \alpha_{12} \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} -v_1 \cos\alpha_{12} \\ \frac{1}{l_{12}}(v_1 \sin\alpha_{12} - l_{12}\omega_1) \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \cos\gamma_1 & d\sin\gamma_1 \\ -\frac{\sin\gamma_1}{l_{12}} & \frac{d\cos\gamma_1}{l_{12}} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} v_2 \\ \omega_2 \end{bmatrix}. \tag{6.76}$$

#### 6.4.2.3 Control Laws

With the input–output equations at hand, nonlinear control laws can be proposed. Here, a nonlinear controller is designed based on the input–output equation (6.75). A stable first-order nonlinear error dynamics is introduced ($k_i > 0, i = 1, 2$):

$$\dot{\tilde{z}}_i + k_i \tilde{z}_i = 0, \qquad i = 1, 2, \tag{6.77}$$

where $\tilde{z}_1$ and $\tilde{z}_2$ are the output errors defined as

$$\tilde{z}_1 = l_{12} - l_{12}^d \qquad \tilde{z}_2 = \alpha_{12} - \alpha_{12}^d. \tag{6.78}$$

For a constant $k_i > 0$, the error dynamics (6.77) results in a damped linear first-order closed-loop input-output system. The desired error behavior (6.77) is written in the matrix form for convenience.

$$\dot{\tilde{\mathbf{z}}} + \mathbf{K}\tilde{\mathbf{z}} = \mathbf{0}, \tag{6.79}$$

where

$$\mathbf{K} = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}. \tag{6.80}$$

Equation (6.79) can be solved for $\dot{\mathbf{z}}$.

$$\dot{\mathbf{z}} = \dot{\mathbf{z}}^d - \mathbf{K}\tilde{\mathbf{z}}. \tag{6.81}$$

Substituting for $\dot{\mathbf{z}}$ from Eq. (6.81) into Eq. (6.75) and solving for the control input $\mathbf{u}$ results in

$$\mathbf{u} = \mathbf{b}^{-1}(\dot{\mathbf{z}}^d - \mathbf{K}\tilde{\mathbf{z}} - \mathbf{f}). \tag{6.82}$$

Note that the matrix $\mathbf{b}$ must be inverted for calculating the control law. Therefore, the determinant of this matrix must be nonzero at all times. This determinant is obtained as

$$\det(\mathbf{b}) = \frac{d}{l_{12}}. \tag{6.83}$$

As can be seen, this determinant is nonzero as long as the parameter $d$, defining the location of the control point, is not zero. This emphasizes the importance of the distance between the control point of the robot 2 and its center of axle, $d$. If the control point was chosen to be coincident with the center of axle of the robot 2 ($d = 0$), the formation scheme would have become uncontrollable.

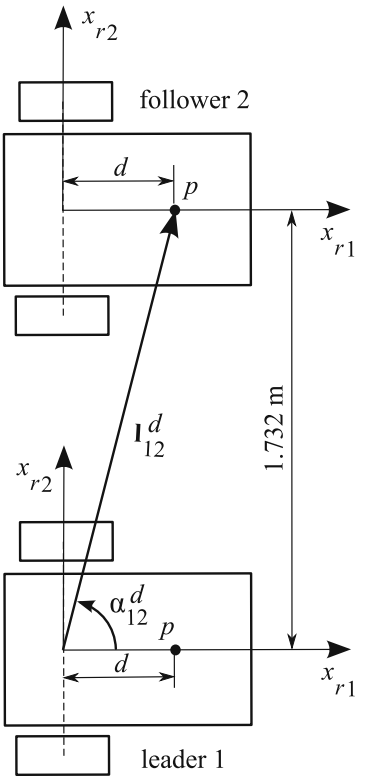


**Fig. 6.14** Two robots moving side by side using the $l - \alpha$ control scheme

*Example 6.4.* Two Hilare-type mobile robots have to move side by side with a distance of 1.732 m between their centers of axle (Fig. 6.14).

(a) Calculate the desired formation parameters $[l_{12}^d, \alpha_{12}^d]$. Assume $d = 1$ m.
(b) Assume that one of the robots moves on a desired trajectory defined as follows:

$$x_1(t) = 0.5t,$$
$$x_2(t) = 0.2\sin(0.5t).$$

Apply the $l - \alpha$ control scheme to control the motion of the second robot, which starts it motion form rest at point (0, 1) m.

*Solution.*

(a) The desired formation parameters are shown in Fig. 6.14. They can be calculated based on the geometry of the formation as follows.

$$l_{12}^d = \sqrt{1.732^2 + 1^2} = 2.000 \text{ m}$$
$$\alpha_{12}^d = \arctan(\frac{2}{1}) = 63.43^\circ$$

(b) The $l - \alpha$ control law (6.82) is used and applied to the kinematic equations (6.4) of the Hilare-type robot. Note that the controller needs the outputs, the formation parameters, at any instant as feedback. The formation parameters can be calculated at any instant in time by knowing the instantaneous pose (position and orientation) of the leader and the follower robots. The follower robot that knows its own pose must receive the pose data from the leader robot to be able to carry out the control calculations in a real application situation. However, for the simulations, the given path of the leader provides the necessary information for determining the instantaneous formation parameters.

The diagonal elements of the gain matrix **K** has been selected to be 1. The response of the follower robot is simulated. Figure 6.15 shows the path of the leader and the follower robots. The follower robot is initially closer to the leader than what is specified by the desired formation parameter, however, after some time, the follower keeps the desired distance with the leader robot. One can see the side by side relative postion of the two robots by looking at their positions at the end of their paths. The leader robot is moving on a sinusoidal curve and the follower is able to capture that motion.

The formation parameters are shown in Fig. 6.16. This figure indicates that the follower robot reaches its correct position in the formation in about 5 s. Despite the variable velocity and orientation of the leader throughout the motion, the follower has been able to keep the formation parameters at the constant desired values.

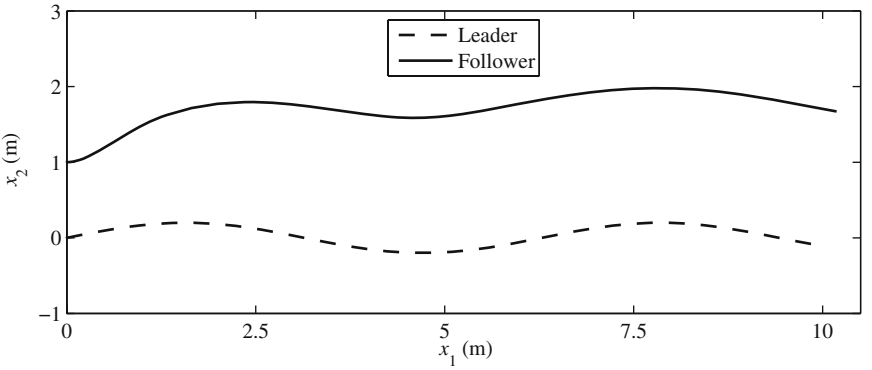


**Fig. 6.15** The path of the leader and the follower robot using the $l - \alpha$ control scheme

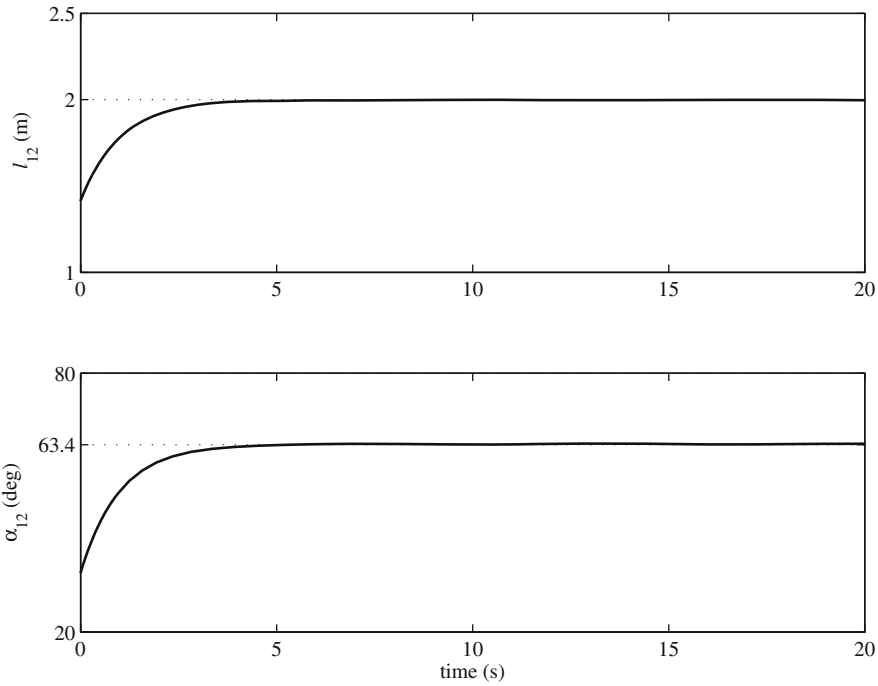


**Fig. 6.16** The formation parameters for the leader and the follower robot using the $l - \alpha$ control scheme

### *6.4.3 Design of the l – l Controller*

Three neighboring robots in the formation structure are shown in Fig. 6.17. Assume that the formation structure is defined such that robot 3 is designated to keep specified distances $l_{13}^d$ and $l_{23}^d$, with robots 1 and 2, respectively. A controller must be

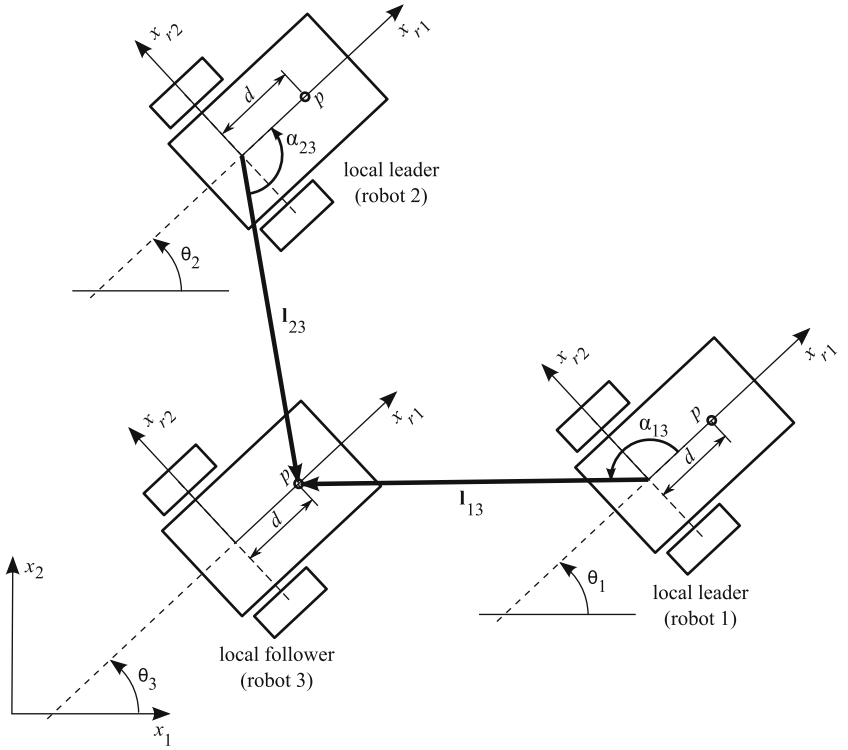


**Fig. 6.17** The $l - l$ control scheme

designed to stabilize these distances, which are measured from the centres of the axle of robot 1 and 2 to a control point on robot 3. The driving velocity and the angular rate of robot 3, $v_3$ and $\omega_3$, are the control inputs, whereas the distances $l_{13}$ and $l_{23}$ are the control outputs. The equations relating the control inputs to the output, known as the input–output relations, must be obtained for designing the controller.

#### 6.4.3.1 Kinematic Analysis

Similar to the $l - \alpha$ case, the relative velocity equations for the control point $p$ on the follower robot 3 with respect to the center of the axle of the leader robot 1 is written. Similar relative velocity relations are written for the follower robot 3 and the leader robot 2. These velocity equations are, then, solved for ($\dot{l}_{13}$, $\dot{\alpha}_{13}$), and ($\dot{l}_{23}$, $\dot{\alpha}_{23}$), respectively. However, since the distances are the controller outputs, only the following resultant kinematic equations are relevant:

$$\dot{l}_{13} = v_3 \cos\gamma_2 - v_1 \cos\alpha_{13} + d\omega_3 \sin\gamma_2, \tag{6.84}$$

$$\dot{l}_{23} = v_3 \cos\gamma_3 - v_1 \cos\alpha_{23} + d\omega_3 \sin\gamma_3, \tag{6.85}$$

where

$$\alpha_1 = \theta_1 + \alpha_{13}, \qquad \gamma_2 = \theta_1 + \alpha_{13} - \theta_3,$$
$$\alpha_2 = \theta_2 + \alpha_{23}, \qquad \gamma_3 = \theta_2 + \alpha_{23} - \theta_3.$$

#### 6.4.3.2 Input–Output Equations

The kinematic equations (6.84) and (6.85) relate the formation scheme outputs $[l_{13}, l_{23}]$ to the control inputs of the follower robot 3. Therefore, they can be used as the input–output equations. They are written in the matrix form for simplicity of the control development notations.

$$\dot{\mathbf{z}} = \mathbf{f} + \mathbf{b}\mathbf{u}, \tag{6.86}$$

where

$$\mathbf{z} = \begin{bmatrix} l_{13} \\ l_{23} \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} -v_1 \cos\alpha_{13} \\ -v_1 \cos\alpha_{23} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \cos\gamma_2 & d\sin\gamma_2 \\ \cos\gamma_3 & d\sin\gamma_3 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} v_3 \\ \omega_3 \end{bmatrix}. \tag{6.87}$$

#### 6.4.3.3 Control Law

The controller is proposed based on the input–output equation (6.86). The output errors are defined as:

$$\tilde{z}_1 = l_{13} - l_{13}^d, \qquad \tilde{z}_2 = l_{23} - l_{23}^d, \tag{6.88}$$

and a stable first-order nonlinear error dynamics is assumed ($k_i > 0, i = 1, 2$):

$$\dot{\tilde{z}}_i + k_i \tilde{z}_i = 0, \qquad i = 1, 2. \tag{6.89}$$

The desired error behavior (6.89) is written in the matrix form for convenience.

$$\dot{\tilde{\mathbf{z}}} + \mathbf{K}\tilde{\mathbf{z}} = \mathbf{0}, \tag{6.90}$$

where

$$\mathbf{K} = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}. \tag{6.91}$$

Equation (6.90) can be solved for $\dot{\mathbf{z}}$.

$$\ddot{\mathbf{z}} = \ddot{\mathbf{z}}^d - \mathbf{K}\tilde{\mathbf{z}}. \tag{6.92}$$

Substituting for $\dot{\mathbf{z}}$ from Eq. (6.92) into Eq. (6.86) and solving for the control input $\mathbf{u}$ results in

$$\mathbf{u} = \mathbf{b}^{-1}(\dot{\mathbf{z}}^d - \mathbf{K}\tilde{\mathbf{z}} - \mathbf{f}). \tag{6.93}$$

Note that the matrix $\mathbf{b}$ must be inverted for calculating the control law. Therefore, the determinant of this matrix must be nonzero at all times. This determinant is obtained as

$$\begin{aligned} \det(\mathbf{b}) &= d\sin(\gamma_3 - \gamma_2), \\ &= d\sin(\alpha_3 - \alpha_2). \end{aligned} \tag{6.94}$$

By observing this determinant, once again, one can see the importance of the choice of the control point such that it is not coincident with the center of the axle of the robot ($d \neq 0$). However, by observing this determinant, one can recognize another condition under which the system becomes uncontrollable, that is if $\sin(\alpha_2 - \alpha_1)$ becomes zero. This is equivalent to $(\theta_2 + \alpha_{23}) - (\theta_1 + \alpha_{13}) = n\pi$ and happens when the origin of the coordinate systems of robot 1 and 2 and the control point of robot 3 are collinear (Fig. 6.17). This situation must be avoided by defining proper desired formation structures. However, if such a configuration is approached during a change in formation, the robot 3 temporarily switches to the $l - \alpha$ controller with one of the robots 1 or 2 until the singularity is cleared. Then, robot 3 switches back to the $l - l$ controller. This switch can be triggered if $|\alpha_2 - \alpha_1|$ becomes less than a predefined threshold.

It is also important to note that the above control scheme has two distinct equilibrium points for a given set of desired values $[l_{13}, l_{23}]$. The two points have different corresponding angle sets $[\alpha_{13}, \alpha_{23}]$. One of these equilibrium points is desirable depending on the desired formation. Therefore, in practical applications, these angles should be monitored by the robot's computer. If the robot detects that it is approaching the wrong equilibrium point, it temporarily switches to using the $l - \alpha$ scheme with one of the neighbors until it is close to the correct equilibrium position. Then, it switches back to using the $l - l$ scheme with both its neighbors.

*Example 6.5.* Three Hilare-type mobile robots have to move in an equilateral triangular formation with 2-m-long sides (Fig. 6.18).

(a) Calculate the desired formation parameters $[l_{13}^d, l_{23}^d]$. Assume $d = 1$ m.
(b) Assume that two leader robots move on a desired trajectory defined as follows:

$$\text{Leader 1} \begin{cases} x_1(t) = 0.5t \\ x_2(t) = 1 + 0.2\sin(0.5t) \end{cases},$$

$$\text{Leader 2} \begin{cases} x_1(t) = 0.5t \\ x_2(t) = -1 + 0.2\sin(0.5t) \end{cases}.$$

Apply the $l - l$ control scheme to control the motion of the third robot, which starts its motion form rest at point $(-2, 1)$ m.

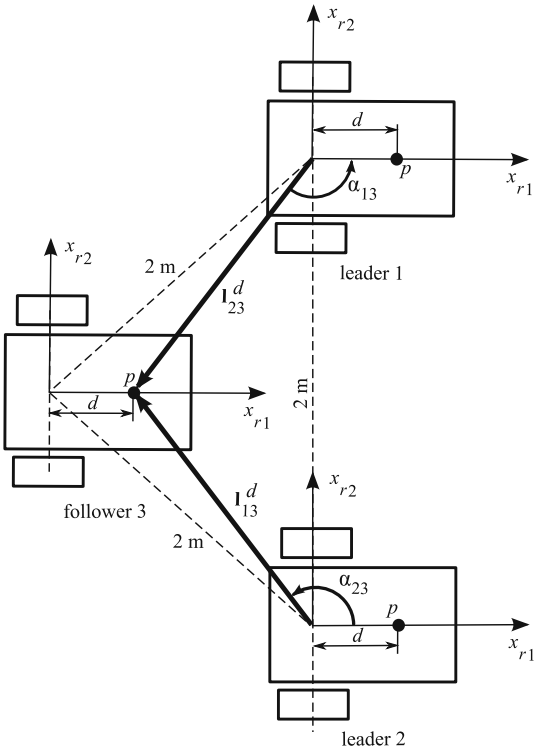


**Fig. 6.18** Three robots in a triangular formation using the $l - l$ control scheme

*Solution*.

(a) The desired formation parameters are shown in Fig. 6.18. They can be calculated based on the geometry of the formation as follows:

$$l_{13}^d = l_{23}^d = \sqrt{(2\cos 30^\circ - 1)^2 + 1^2} = 1.239 \text{ m}.$$

(b) The $l - l$ control law (6.93) is used and applied to the kinematic Eq. (6.4) of the Hilare-type robot. For the simulations, the given path of the leaders provide the necessary information for determining the instantaneous formation parameters.

The diagonal elements of the gain matrix $\mathbf{K}$ has been selected to be 1. The response of the follower robot is simulated. Figure 6.19 shows the path of the two leader robots and the follower robot. The follower robot is initially far from the desired formation, however, after some time, it gains the desired distances with the leader robots. One can see the triangular formation of the three robot by

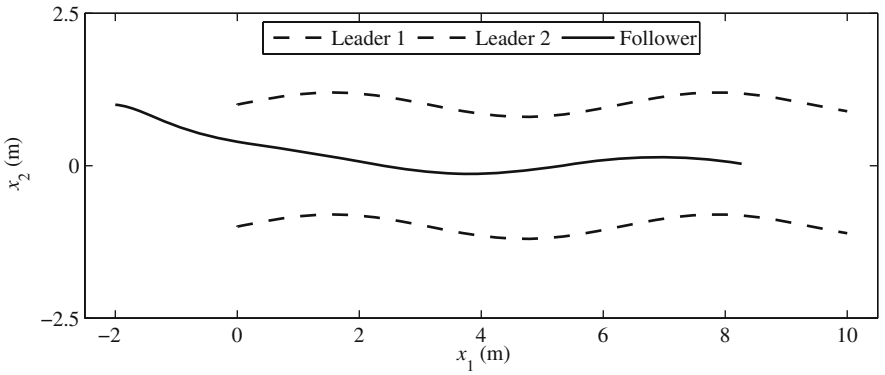


**Fig. 6.19** The path of the leaders and the follower robot using the $l - l$ control scheme

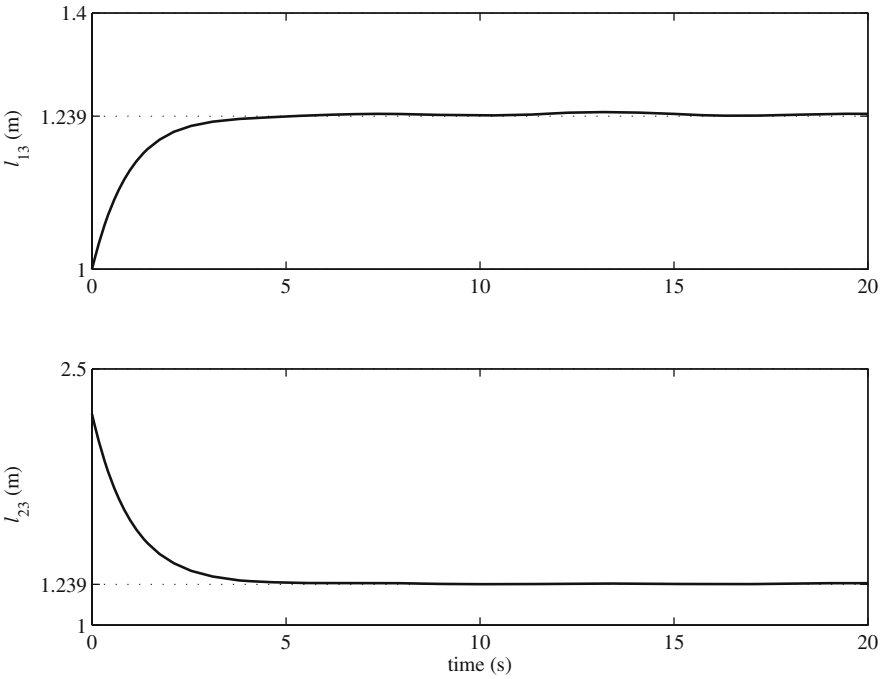


**Fig. 6.20** The formation parameters for the leader and the follower robot using the $l - \alpha$ control scheme

looking at their positions at the end of their paths. The leader robots are moving on a sinusoidal curve and the follower is able to capture that motion.

The formation parameters are shown in Fig. 6.20. This figure indicates that the follower robot reaches its correct position in the formation in about 5 s. Despite the variable velocity and orientation of the leader robots throughout the motion, the follower has been able to keep the formation parameters at the constant desired values.

## 6.5 Dynamics of Mobile Robots

In this section, the dynamic equations of motion of some mobile robots are derived and presented.

### *6.5.1 Hilare-Type Mobile Robots*

Consider the Hilare-type mobile robot shown in Fig. 6.1. The body coordinate system $x_{r1} - x_{r2}$ is attached to the robot's body at the middle of the line connecting the center of the two wheels. Note that the center of gravity of the robot is assumed to be on the body longitudinal axis $x_{r1}$ with a distance $c$ from the origin of the body coordinate system. For dynamic modeling of a mobile robot, one must assume the torques exerted by the motors to the wheels as the input. The mass of the robot wheels usually have the same order of magnitude as that of the robot's body. Therefore, neglecting the wheels rotation and assuming the whole robot as one rigid body may result in an inaccurate dynamic model. It is best to include the effect of the wheel rotations in the dynamic model.

There are several methods available for deriving the dynamic equations of motion for mobile robots. Here, we are using the Newton-Euler equations of motion. We will use a more systematic approach. The steps of this systematic approach follow:

1. The generalized coordinates and the generalized speeds that define the dynamic state of the system are defined.
2. The acceleration of the centers of mass of the robot and the wheels are expressed in terms of the derivative of the generalized speeds.
3. The free body diagram of the robot and the wheels are drawn and all the involved forces are shown on the diagram. Since the robot body and the wheels have different motions, each must be drawn separately in the free body diagram. Also, the kinetic diagram of the robot and the wheels are drawn and the components of all the inertial forces and moments caused by acceleration are shown.
4. The equations of motion are written using the balance of all the forces and the moments with the inertia forces and moments. This usually results in a set of second-order equations. As we will see later, these equations are not integrable.

5. The non integrable (nonholonomic) constraint equation, which is derived from the no-slip condition, is used to complete the dynamic equations of motion.
6. Finally, the equations of motion are converted to the first-order form to allow for simpler numerical simulations.

#### 6.5.1.1 Generalized Coordinates and Generalized Speeds

The generalized coordinates for a dynamic system are the variables that uniquely define the geometrical configuration of the system at any given time. These coordinates must be independent, in which case the number of them is equal to the number of DOFs of the dynamic system. If the generalized coordinates are not chosen to be independent, there must be geometrical constraint equations that define the relation between the ones that are dependent. For a mobile robot that works in a 2D space, there exists three DOFs. Therefore, three independent variables are needed to uniquely define the geometrical configuration of the robot. These variables are the two components of the inertial position vector of the midpoint of the transverse line attaching the wheel centers, $(x_1, x_2)$, and the angle between the longitudinal axis of the robot with the inertial $x_1$ axis, $\theta$ (see Fig. 6.1).

The generalized coordinates uniquely define the geometrical configuration (i.e., position and orientation) of the dynamic system, however, to completely define the dynamic state of a system, the time rate of change of the generalized coordinates must also be involved. These rates are included in the dynamic model by defining the generalized speeds. By definition, the generalized speeds are independent functions of the first-order derivatives of the generalized coordinates. These speeds are selected such that the form of the dynamic equations of motion becomes simple. In general, the number of generalized speeds are equal to the number of degrees of freedom of the dynamic system. If some of the generalized speeds are dependent, velocity constraint equations must be written. For a mobile robot, two of the generalized speeds are selected to be the components of the velocity of the point $(x_1, x_2)$ defined in terms of the body axis, $(\dot{x}_{r1}, \dot{x}_{r2})$. The angular velocity of the robot body, $\dot{\theta}$, is selected as the third generalized speed. Note that because of the no-slip condition, a velocity constraint equation can be written in terms of $\dot{x}_{r1}$ and $\dot{x}_{r2}$.

The above defined generalized coordinates and speeds are grouped together in a single vector, $\mathbf{q}$, known as the state vector of the mobile robot.

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \theta \\ \dot{x}_{r1} \\ \dot{x}_{r2} \\ \dot{\theta} \end{bmatrix} \tag{6.95}$$

The no slip condition for the lateral motion results in the following kinematic constraint.

$$\dot{x}_{r2} = -\dot{x}_1 \sin\theta + \dot{x}_2 \cos\theta = 0 \tag{6.96}$$

We shall use this kinematic constraint later when writing the equations of motion. Note that this constraint equation defines a condition on the robot's velocity components. This equation cannot be integrated with respect to time because $\theta$ is not known as a function of time before any dynamic simulation. Therefore, this constraint cannot be converted into a constraint between the generalized coordinates by integration. Such a non integrable constraint on velocity components is called a nonholonomic constraint. A nonholonomic constraint equation must be integrated during a dynamic simulation because $\theta$ as a function of time only becomes known during the simulation. In fact, as we will see later, the dynamic equations cannot be integrated without using this kinematic constraint. Therefore, this kinematic equation is an inseparable part of the dynamic model. In general, any dynamic system that has a non integrable constraint equation is called a nonholonomic dynamic system.

#### 6.5.1.2 The Acceleration of the Centers of Mass of the Robot and the Wheels

The second step of the systematic approach to finding the equations of motion using the Newton-Euler method is finding the accelerations of the centers of mass for all the bodies. Here, the robot as a dynamic system consists of three rigid bodies: the robot body and two wheels. The form of the equations will be simpler if one uses the body coordinate system to express the inertial accelerations of the three bodies.

Let us assume that the inertial acceleration of the origin of the body coordinate system is

$$\mathbf{a}_o = \ddot{x}_{r1}\hat{\mathbf{i}}_r + \ddot{x}_{r2}\hat{\mathbf{j}}_r, \tag{6.97}$$

where $\hat{\mathbf{i}}_r$ and $\hat{\mathbf{j}}_r$ are the unit vectors of the body coordinate system. The angular velocity and acceleration of the body coordinate system are

$$\boldsymbol{\omega} = \dot{\theta}\hat{\mathbf{k}}_r, \quad \boldsymbol{\alpha} = \ddot{\theta}\hat{\mathbf{k}}_r. \tag{6.98}$$

If the distance of the center of mass of the robot's body with the origin of the robot's body frame is denote by $\mathbf{c} = c\hat{\mathbf{j}}_r$, then the inertial acceleration of the center of mass of the robot body expressed in the body coordinate system is

$$\begin{aligned} \mathbf{a}_r &= \mathbf{a}_o + \boldsymbol{\alpha} \times \mathbf{c} - \omega^2 \mathbf{c}, \\ a_{r1}\hat{\mathbf{i}}_r + a_{r2}\hat{\mathbf{j}}_r &= (\ddot{x}_{r1} - c\dot{\theta}^2)\hat{\mathbf{i}}_r + (\ddot{x}_{r2} + c\ddot{\theta})\hat{\mathbf{j}}_r. \end{aligned} \tag{6.99}$$

If the distance of the center of mass of the robot's right wheel with the origin of the robot's body frame is denote by $\mathbf{d}_r = -(T/2)\hat{\mathbf{i}}_r$, then the inertial acceleration of the center of mass of the robot right wheel expressed in the body frame is

$$\mathbf{a}_{wr} = \mathbf{a}_o + \boldsymbol{\alpha} \times \mathbf{d}_r - \omega^2 \mathbf{d}_r,$$
$$a_{wr1}\hat{\mathbf{i}}_r + a_{wr2}\hat{\mathbf{j}}_r = (\ddot{x}_{r1} + \frac{T}{2}\dot{\theta}^2)\hat{\mathbf{i}}_r + (\ddot{x}_{r2} + \frac{T}{2}\ddot{\theta})\hat{\mathbf{j}}_r. \tag{6.100}$$

If the distance of the center of mass of the robot's left wheel with the origin of the robot's body frame is denote by $\mathbf{d}_l = (T/2)\hat{\mathbf{i}}_r$, then the inertial acceleration of the center of mass of the robot left wheel expressed in the body frame is

$$\mathbf{a}_{wl} = \mathbf{a}_o + \boldsymbol{\alpha} \times \mathbf{d}_l - \omega^2 \mathbf{d}_l,$$
$$a_{wl1}\hat{\mathbf{i}}_r + a_{wl2}\hat{\mathbf{j}}_r = (\ddot{x}_{r1} - \frac{T}{2}\dot{\theta}^2)\hat{\mathbf{i}}_r + (\ddot{x}_{r2} - \frac{T}{2}\ddot{\theta})\hat{\mathbf{j}}_r. \tag{6.101}$$

#### 6.5.1.3 Free Body and Kinetic Diagrams

Since in this modeling the dynamic effect of the wheels' rotation is being considered in deriving the equations of motion, the wheels and the body must be separated in the free body diagram. The free body diagram of the mobile robot body and the wheels is shown in Fig. 6.21. This free body diagram helps us better visualize the forces on the wheels. Now, the 2D version of the free body and kinetic diagrams for the whole robot and the wheels as separate parts can be drawn. These diagrams are shown in Figs. 6.22 and 6.23, respectively.

#### 6.5.1.4 The Equations of Motion

The free body and kinetic diagrams shown in Figs. 6.22 and 6.23 simplify the writing of the equations of motion. By observing Fig. 6.22, one can write the external force and the inertia force balance in the $x_{r1}$ direction.

$$\begin{aligned} F_r + F_l &= m_w a_{wr1} + m_w a_{wl1} + m_r a_{r1}, \\ &= m_w(\ddot{x}_{r1} + \frac{T}{2}\ddot{\theta}) + m_w(\ddot{x}_{r1} - \frac{T}{2}\ddot{\theta}) + m_r(\ddot{x}_{r1} - c\dot{\theta}^2), \\ &= (m_r + 2m_w)\ddot{x}_{r1} - m_r c\dot{\theta}^2. \end{aligned} \tag{6.102}$$

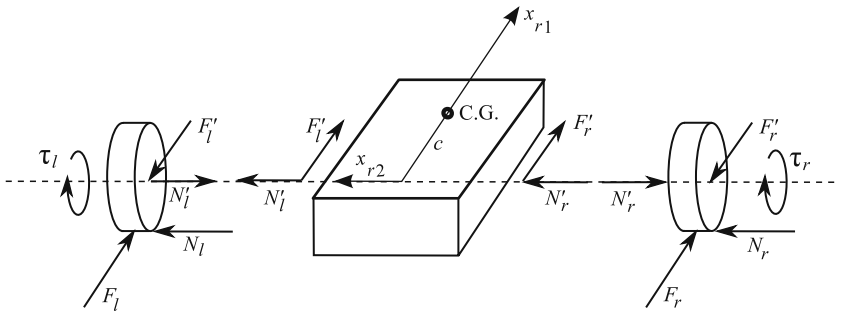


**Fig. 6.21** The detailed free body diagram for a Hilare mobile robot

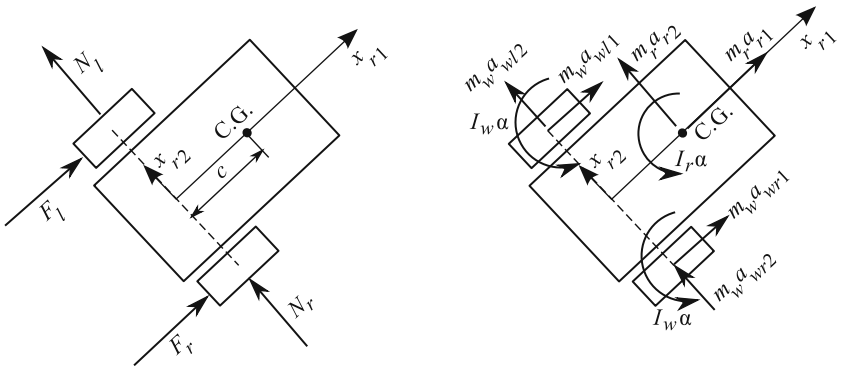


**Fig. 6.22** The free body and kinetic diagram for a Hilare mobile robot as a single system

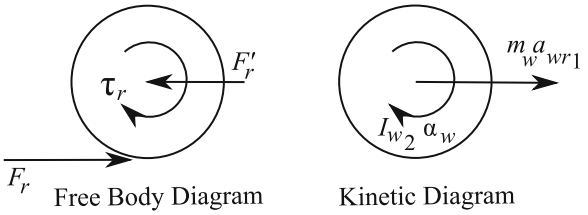


**Fig. 6.23** The free body and the kinetic diagram for the right wheel

By observing Fig. 6.22, one can write the external force and the inertia force balance in the $x_{r2}$ direction.

$$\begin{aligned} N_r + N_l &= m_w a_{wr2} + m_w a_{wl2} + m_r a_{r2}, \\ &= m_w(\ddot{x}_{r2} - \frac{T}{2}\dot{\theta}^2) + m_w(\ddot{x}_{r2} + \frac{T}{2}\dot{\theta}^2) + m_r(\ddot{x}_{r2} + c\ddot{\theta}), \\ &= (m_r + 2m_w)\ddot{x}_{r2} + m_r c\ddot{\theta}. \end{aligned} \tag{6.103}$$

By observing Fig. 6.22, one can write the external moment and the inertia moment balance in the $x_{r3}$ direction.

$$\begin{aligned} (F_r - F_l)\frac{T}{2} &= (I_r + 2I_w)\alpha + m_w a_{wr1}\frac{T}{2} - m_w a_{wl1}\frac{T}{2} + m_r a_{r2} c, \\ &= (I_r + 2I_w)\ddot{\theta} + m_w(\ddot{x}_{r1} + \frac{T}{2}\ddot{\theta})\frac{T}{2} - m_w(\ddot{x}_{r1} - \frac{T}{2}\ddot{\theta})\frac{T}{2} + m_r(\ddot{x}_{r2} + c\ddot{\theta})c, \\ &= (I_r + 2I_w + m_w\frac{T^2}{2})\ddot{\theta} + m_r c(\ddot{x}_{r2} + c\ddot{\theta}). \end{aligned} \tag{6.104}$$

By observing Fig. 6.23, one can write the external moment and the inertia moment balance in the $x_{r2}$ direction.

$$
\begin{aligned}
-F_r r + \tau_r &= I_{w2}\alpha_w, \\
&= I_{w2}\frac{1}{r}\left(\ddot{x}_{r1} + \frac{T}{2}\ddot{\theta}\right).
\end{aligned}
$$

This results in the following for the traction force, $F_r$, on the right wheel:

$$
F_r = \frac{1}{r}\left(\tau_r - \frac{I_{w2}}{r}(\ddot{x}_{r1} + \frac{T}{2}\ddot{\theta})\right). \tag{6.105}
$$

A similar result for the traction force of the left wheel, $F_l$, can be achieved.

$$
F_l = \frac{1}{r}\left(\tau_l - \frac{I_{w2}}{r}(\ddot{x}_{r1} - \frac{T}{2}\ddot{\theta})\right). \tag{6.106}
$$

In Eqs. (6.105) and (6.106), $\tau_r$ and $\tau_l$ are the right and the left wheel's driving torque, respectively. Since these torques are the input to the system, we shall write the equations of motion in terms of these torques, rather than in terms of the wheel tractions. This goal is achieved by substituting Eqs. (6.105) and (6.106) into Eqs. (6.102), (6.103), and (6.104). The resulting dynamic equations of motion are

$$
\begin{aligned}
&(m_r + 2m_w + \frac{2}{r^2}I_{w2})\ddot{x}_{r1} - m_r c\dot{\theta}^2 = \frac{1}{r}(\tau_r + \tau_l), \\
&(m_r + 2m_w)\ddot{x}_{r2} + m_r c\ddot{\theta} = N_r + N_l, \\
&(I_r + 2I_w + m_w\frac{T^2}{2} + I_{w2}\frac{T^2}{2r^2} + m_r c)\ddot{\theta} + m_r\ddot{x}_{r2} = \frac{T}{2r}(\tau_r - \tau_l).
\end{aligned} \tag{6.107}
$$

Although the dynamic equations (6.107) are complete and accurate, they cannot be used in this form to simulate the motion of the robot. The reason for this fact is that the lateral tire force $N_r + N_l$ is unknown. This lateral tire force is a function of the lateral and angular acceleration of the robot and it changes dynamically. An extra equation is needed to for us to be able to determine this lateral force as a function of time. By observing the dynamic equations (6.107) more closely, once can see that if one can find a kinematic relation for the lateral acceleration, they can calculate this unknown lateral force during the simulation of the dynamic equations (6.107). Finding this kinematic relation is discussed in the following.

#### 6.5.1.5 The Role of the Nonholonomic Constraint Equation

Knowing the lateral acceleration of the robot, $\ddot{x}_{r2}$, is one of the requirements for the integration of the dynamic equations (6.107). The nonholonomic constraint defined in Eq. (6.96) can be used for determining the lateral acceleration component $\ddot{x}_{r2}$. The nonholonomic constraint is repeated here.

$$
\dot{x}_{r2} = -\dot{x}_1 \sin\theta + \dot{x}_2 \cos\theta = 0. \tag{6.108}
$$

Note that although the lateral velocity $\dot{x}_{r2}$ is zero due to the no-slip condition, $\ddot{x}_{r2}$ could be nonzero under certain conditions. Here, we shall investigate whether $\ddot{x}_{r2}$ is in fact zero. The lateral acceleration can be found by differentiating Eq. (6.108).

$$\begin{aligned}\ddot{x}_{r2} &= -\ddot{x}_1 \sin\theta + \ddot{x}_2 \cos\theta - \dot{\theta}(\dot{x}_1 \cos\theta + \dot{x}_2 \sin\theta),\\ &= -\ddot{x}_1 \sin\theta + \ddot{x}_2 \cos\theta - \dot{\theta}\dot{x}_{r1}.\end{aligned} \tag{6.109}$$

To complete the calculation of $\ddot{x}_{r2}$, let us try to determine the robot's acceleration components expressed in the inertial coordinate system, $\ddot{x}_1$ and $\ddot{x}_2$. Equation (6.108) implies that

$$\dot{x}_2 = \dot{x}_1 \tan\theta. \tag{6.110}$$

Differentiating the above equation results in

$$\ddot{x}_2 = \ddot{x}_1 \tan\theta + \dot{x}_1\dot{\theta}(1 + \tan^2\theta). \tag{6.111}$$

Substituting the above relation into Eq. (6.109) yields

$$\ddot{x}_{r2} = \dot{\theta}(\dot{x}_1(1 + \tan^2\theta)\cos\theta - \dot{x}_{r1}). \tag{6.112}$$

However, we know that $\dot{x}_1 = \dot{x}_{r1}\cos\theta - \dot{x}_{r2}\sin\theta$ and $\dot{x}_{r2} = 0$. Using these relations with the above equation shows that the lateral acceleration component of the robot becomes

$$\ddot{x}_{r2} = 0. \tag{6.113}$$

Equation (6.113) is an important result. It allows us to neglect the second dynamic equation in (6.107) and simplify the third dynamic equation in (6.107). Finally, the dynamic equations of motion for the robot reduce to the following:

$$\begin{aligned}&(m_r + 2m_w + \frac{2}{r^2}I_{w2})\ddot{x}_{r1} - m_r c\dot{\theta}^2 = \frac{1}{r}(\tau_r + \tau_l),\\ &(I_r + 2I_w + m_w\frac{T^2}{2} + I_{w2}\frac{T^2}{2r^2} + m_r c)\ddot{\theta} = \frac{T}{2r}(\tau_r - \tau_l).\end{aligned} \tag{6.114}$$

#### 6.5.1.6 The First-Order Form of the Dynamic Equations

For simulation purposes, it is desirable to write the equations of motion of the mobile robot in the first-order form. Let us define a new set of state variable to exclude $\dot{x}_{r2}$, which is zero.

$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \theta \\ \dot{x}_{r1} \\ \dot{\theta} \end{bmatrix}. \tag{6.115}$$

Using the definition in Eq. (6.115) for the dynamic states of the robot, the equations of motion are written as follows:

$$\begin{aligned} \dot{q}_1 &= q_4 \cos q_3, \\ \dot{q}_2 &= q_4 \sin q_3, \\ \dot{q}_3 &= q_5, \\ \dot{q}_4 &= \frac{1}{m}(\frac{1}{r}(\tau_r + \tau_l) + m_r c q_5^2), \\ \dot{q}_5 &= \frac{T}{2I}\frac{1}{r}(\tau_r - \tau_l), \end{aligned} \tag{6.116}$$

where

$$\begin{aligned} m &= m_r + 2(m_w + \frac{1}{r^2} I_{w2}), \\ I &= I_r + 2I_w + (m_w + \frac{1}{r^2} I_{w2})\frac{T^2}{2} + m_r c. \end{aligned} \tag{6.117}$$

## 6.6 Trajectory-Tracking Control Based on Dynamic Models

Although the control methods discussed in Section 6.3 are successful in driving a mobile robot on a desired trajectory, they have some shortcomings. One of the shortcomings of the methods discussed in Section 6.3 is that their control input are linear and angular velocity. Once the controller calculates these velocities, there is no guarantee that the robot's drive train can generate them. If the desired trajectory is not defined with care such that sudden change in desired velocity or acceleration are not avoided, it is highly possible that the calculated large velocity (and their corresponding acceleration) is too high for the robot's drive train to generate. In this situation, a better control method must be used in which the torques generated by the controller can be limited to a maximum value. Such a method, of course, must be based on the dynamic model of the robot, so that its control inputs are torques (or forces).

Model predictive control, which determines a series of future control commands based on a real-time optimization, is a very good method for controlling systems with constraints. In the following section, the model predictive control method is used for controlling a Hilare-type mobile robot with motors with limited torque capability. The model predictive control method presented here is also applicable to any dynamic system without constrained inputs.

### 6.6.1 Hilare-Type Mobile Robots

A Hilare-type mobile robot is assumed. A control law for two limited control inputs $u_1 = \tau_r$ and $u_2 = \tau_l$ must be found to drive the robot on a desired trajectory. It is assumed that the motors that drive the left and the right wheels of the robot have limited torques such that

$$u_1 \leq u_{1\max}, \qquad u_2 \leq u_{2\max}. \tag{6.118}$$

Inequalities (6.118) must be converted to equalities before they can used as constraints in an optimization-based control method such as the model predictive control. To convert the inequalities (6.118) to equalities, two dummy control inputs $u_3$ and $u_4$ are defined such that

$$\begin{aligned} u_{1\max}^2 - u_1^2 &= u_3^2 \geq 0, \\ u_{2\max}^2 - u_2^2 &= u_4^2 \geq 0. \end{aligned} \tag{6.119}$$

The dummy control parameters are calculated at each step of the control such that they guarantee the satisfaction of Eqs. (6.119). The equalities (6.119) can be written in the form of constraint equations.

$$\begin{aligned} C_1(u_1, u_3) &= u_1^2 + u_3^2 - u_{1\max}^2 = 0, \\ C_2(u_2, u_4) &= u_2^2 + u_4^2 - u_{2\max}^2 = 0. \end{aligned} \tag{6.120}$$

The constraint Eqs. (6.120) are written in the following matrix form:

$$\mathbf{C}(\mathbf{u}) = \mathbf{0}, \tag{6.121}$$

where the augmented control input $\mathbf{u}$ is

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}. \tag{6.122}$$

The dynamic model of a Hilare-type robot, presented in Eq. (6.116), can be written in the following general form:

$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q}, \mathbf{u}), \tag{6.123}$$

where $\mathbf{q}$ is defined in Eq. (6.115).

#### 6.6.1.1 Model Predictive Control Problem

In the model predictive control method, at any time $t$, the future control action is predicted for a time $T$ into the future such that it minimizes a cost function penalizing the state errors and high control actions. In mathematical notation, this can be expressed as finding the control action as a function of time

$$\mathbf{u}(t) \qquad \tau \in [t, t+T], \tag{6.124}$$

such that the cost function

$$J = \phi(\mathbf{q}(t+T)) + \int_{t}^{t+T} L(\mathbf{q}(\tau), \mathbf{u}(\tau))d\tau \tag{6.125}$$

is minimized. The functions $\phi(\mathbf{q}(t+T))$ and $L(\mathbf{q}(\tau), \mathbf{u}(\tau))$ are positive-definite. The dynamic model (6.123) and the equality constraint (6.121) are also imposed over the time interval $T$, also known as the control horizon.

Solving analytically for $\mathbf{u}(\tau)$ is very difficult for most nonlinear models representing real physical systems, including the Hilare-type dynamic model. Therefore, numerical methods are used. To use the numerical methods, the future control action $\mathbf{u}(\tau)$ is approximated by a stepwise function with $N$ steps. The step widths are equal to $t_s = T/N$. The height of step $i$ is represented by the following notation:

$$\mathbf{u}_k = \mathbf{u}(t + kt_s), \qquad k = 0, \ldots, N-1. \tag{6.126}$$

With this assumption, the integral in the cost function (6.125) can be replaced by the summation

$$J = \phi(\mathbf{q}_N) + \sum_{k=0}^{N-1} L_k(\mathbf{q}_k, \mathbf{u}_k)t_s, \tag{6.127}$$

where $\mathbf{q}_k$'s are "predicted" based on the discretized dynamic "model" of the system derived from Eq. (6.123).

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \frac{T}{N}\mathbf{f}_k(\mathbf{q}_k, \mathbf{u}_k), \qquad k = 0, \ldots, N-1, \tag{6.128}$$

where

$$\mathbf{q}_0 = \mathbf{q}(t). \tag{6.129}$$

To solve for the future discretized control action $\mathbf{u}_k$, $k = 0, \ldots, N-1$ at any control time $t$, the discretized cost function (6.127) must be minimized in real-time using numerical calculations. The solution to the minimization problem must be compatible with the dynamics of the system to be controlled and must satisfy the

constraints. To address these requirements, the cost function (6.127) is augmented with (zero) terms representing the dynamic model and constraints.

$$J = \phi(\mathbf{q}_N) + \sum_{k=0}^{N-1} [L_k + \boldsymbol{\lambda}_{k+1}^T(\mathbf{f}_k - \mathbf{q}_{k+1}) + \boldsymbol{\mu}_k^T \mathbf{C}_k] t_s, \tag{6.130}$$

where $\boldsymbol{\lambda}_k$'s are the costates and $\boldsymbol{\mu}_k$'s are the Lagrange multipliers. The augmented cost function (6.130) accepts a simpler form by introducing a notation for Hamiltonian $H$ as follows:

$$H_k = L_k + \boldsymbol{\lambda}_{k+1}^T \mathbf{f}_k + \boldsymbol{\mu}_k^T \mathbf{C}_k. \tag{6.131}$$

Now, the augmented discretized cost function in terms of Hamiltonian becomes

$$J = \phi(\mathbf{q}_N) + \sum_{k=0}^{N-1} (H_k - \boldsymbol{\lambda}_{k+1}^T \mathbf{q}_{k+1}) t_s. \tag{6.132}$$

The indices in the summation term of Eq. (6.132) can be equalized by rearraging the terms of the summation.

$$J = \phi(\mathbf{q}_N) - \boldsymbol{\lambda}_N^T \mathbf{q}_N t_s + H_0 t_s + \sum_{k=1}^{N-1} (H_k - \boldsymbol{\lambda}_k^T \mathbf{q}_k) t_s. \tag{6.133}$$

The minimization is done by equating to zero the complete differential of the cost function $J$, which vanishes at the function's optimum point.

$$dJ = \sum_{k=0}^{N} \frac{\partial J}{\partial \mathbf{q}_k^T} d\mathbf{q}_k + \sum_{k=0}^{N-1} \left( \frac{\partial J}{\partial \mathbf{u}_k^T} d\mathbf{u}_k + \frac{\partial J}{\partial \boldsymbol{\mu}_k^T} d\boldsymbol{\mu}_k \right) = 0. \tag{6.134}$$

For the complete differential to vanish, all partial derivatives must vanish. In the following, these partial derivatives are calculated to provide the system of equations needed for solving for the discretized future control action $\mathbf{u}_k$. First, the partial derivative of $J$ with respect to the states is derived.

$$\frac{\partial J}{\partial \mathbf{q}_0^T} d\mathbf{q}_0 = \frac{\partial H_0}{\partial \mathbf{q}_0^T} t_s d\mathbf{q}_0, \tag{6.135}$$

$$\frac{\partial J}{\partial \mathbf{q}_k^T} d\mathbf{q}_k = \sum_{k=1}^{N-1} \left( \frac{\partial H_k}{\partial \mathbf{q}_k^T} - \boldsymbol{\lambda}_k^T \right) t_s d\mathbf{q}_k, \qquad k = 1, \ldots, N-1 \tag{6.136}$$

$$\frac{\partial J}{\partial \mathbf{q}_N^T} d\mathbf{q}_N = \left( \frac{\partial \phi}{\partial \mathbf{q}_N^T} - \boldsymbol{\lambda}_N^T t_s \right) d\mathbf{q}_N. \tag{6.137}$$

Note that $\mathbf{q}_0 = \mathbf{q}(t)$ is the states of the system at which the control calculations are being done. Since these states are not a function of the future control action, $d\mathbf{q}_0$ is zero. Therefore, the partial derivative in Eq. (6.135) is identically zero. The partial derivatives in Eqs. (6.136) and (6.137) must also vanish. This is done by selecting the right costates. Since the costates $\boldsymbol{\lambda}_k$'s are the coefficient of zero terms in the augmented cost function (6.130), they can be chosen arbitrarily. This fact is exploited and the costates are selected such that Eqs. (6.136) and (6.137) are zero.

$$\boldsymbol{\lambda}_N^T = \frac{\partial \phi}{\partial \mathbf{q}_N^T}, \tag{6.138}$$

$$\begin{aligned} \boldsymbol{\lambda}_k^T &= \frac{\partial H_k}{\partial \mathbf{q}_k^T}, \\ &= \frac{\partial L_k}{\partial \mathbf{q}_k^T} + \boldsymbol{\lambda}_{k+1}^T \frac{\partial \mathbf{f}_k}{\partial \mathbf{q}_k^T} + \boldsymbol{\mu}_k^T \frac{\partial \mathbf{C}_k}{\partial \mathbf{q}_k^T}, \qquad k = N-1, \ldots, 1. \end{aligned} \tag{6.139}$$

The partial derivatives of $J$ with respect to the discretized control input $\mathbf{u}_k$ is derived as follows:

$$\begin{aligned} \frac{\partial J}{\partial \mathbf{u}_k^T} &= \frac{\partial H_k}{\partial \mathbf{u}_k^T} = \mathbf{0}, \\ &= \frac{\partial L_k}{\partial \mathbf{u}_k^T} + \boldsymbol{\lambda}_{k+1}^T \frac{\partial \mathbf{f}_k}{\partial \mathbf{u}_k^T} + \boldsymbol{\mu}_k^T \frac{\partial \mathbf{C}_k}{\partial \mathbf{u}_k^T} = \mathbf{0}, \qquad k = 1, \ldots, N-1. \end{aligned} \tag{6.140}$$

The partial derivatives of $J$ with respect to the Lagrange multipliers $\boldsymbol{\mu}_k$ is derived as follows:

$$\begin{aligned} \frac{\partial J}{\partial \mu_k^T} &= \frac{\partial H_k}{\partial \boldsymbol{\mu}_k^T} = \mathbf{0}, \\ &= \mathbf{C}_k = \mathbf{0}, \qquad k = 1, \ldots, N-1. \end{aligned} \tag{6.141}$$

Once the costates are calculated using Eqs. (6.138) and (6.139), Eqs. (6.140) and (6.141) must be simultaneously solved for the future control command steps $\mathbf{u}_k$ $(k = 0, \ldots, N)$ and the Lagrange multipliers $\boldsymbol{\mu}_k$ $(k = 0, \ldots, N)$. Note that the actual number of scalar unknowns for a Hilare robot are a total of $6N$ components; that is, $4N$ component for the stepwise control $\mathbf{u}_k$ plus $2N$ components for the Lagrange multipliers $\boldsymbol{\mu}_k$. These unknowns are gathered in a single $6N \times 1$ unknown vector $\mathbf{U}$.

$$\mathbf{U} = \left[\mathbf{u}_0^T \ \boldsymbol{\mu}_0^T \ \ldots \ \mathbf{u}_k^T \ \boldsymbol{\mu}_k^T \ \ldots \ \mathbf{u}_{N-1}^T \ \boldsymbol{\mu}_{N-1}^T\right]^T. \tag{6.142}$$

Equations (6.140) and (6.141) are also gathered in the form of a system of $6N$ nonlinear equations in terms of the components of $\mathbf{U}$.

$$\mathbf{F}(\mathbf{U}, \mathbf{q}_0, t) = \begin{bmatrix} (\partial H_0/\partial \mathbf{u}_0^T)^T \\ \mathbf{C}_0 \\ \vdots \\ \vdots \\ (\partial H_k/\partial \mathbf{u}_k^T)^T \\ \mathbf{C}_k \\ \vdots \\ \vdots \\ (\partial H_{N-1}/\partial \mathbf{u}_{N-1}^T)^T \\ \mathbf{C}_{N-1} \end{bmatrix} = \mathbf{0} \tag{6.143}$$

The system of nonlinear Eqs. (6.143) must be solved for the control input and Lagrange multiplier components $\mathbf{U}$ in real-time. There are several ways for solving this nonlinear system, however, a method must be chosen that is fast enough for real-time calculations and has a constant calculation cycle time throughout the real-time control. One of the most efficient methods of solution is called the continuation/GMRES (Generalized Minimum RESidual) method. This method is discussed next.

#### 6.6.1.2 Model Predictive Control Solution Method

The Continuation Method

The algebraic system of nonlinear equations (6.143) is computationally expensive to be solved using classical (e.g., Newton iterative) methods. The time required for such calculations is prohibitive for real-time applications. Instead, the system (6.143) is converted to a differential system of equations whose equilibrium point is $\mathbf{F}(\mathbf{U}, \mathbf{q}_0, t) = \mathbf{0}$. This system of differential equations with an appropriate initial condition is integrated through time in real-time to calculate $\mathbf{U}(t)$ at any control time $t$.

The following system of simultaneous differential equations with a proper initial condition $\mathbf{U}(0) = \mathbf{U}_0$ is introduced:

$$\dot{\mathbf{F}}(\mathbf{U}(t), \mathbf{q}(t), t) = \mathbf{A}_s \mathbf{F}(\mathbf{U}(t), \mathbf{q}(t), t), \tag{6.144}$$

where $\mathbf{U}_0$ is the solution of $\mathbf{F}(\mathbf{U}_0, \mathbf{q}(0)) = \mathbf{0}$, and $\mathbf{A}_s$ is a negative definite matrix such that the system (6.144) is stable. The left hand side of the system (6.144) can be expanded.

$$\dot{\mathbf{F}} = \mathbf{F}_U \dot{\mathbf{U}} + \mathbf{F}_q \dot{\mathbf{q}} + \mathbf{F}_t, \tag{6.145}$$

where

$$\mathbf{F}_U = \frac{\partial \mathbf{F}}{\partial \mathbf{U}^T}, \quad \mathbf{F}_q = \frac{\partial \mathbf{F}}{\partial \mathbf{q}^T}, \quad \mathbf{F}_t = \frac{\partial \mathbf{F}}{\partial t}. \tag{6.146}$$

The time rate of $\mathbf{U}$ can now be determined by combining Eqs. (6.144) and (6.145).

$$\dot{\mathbf{U}} = \mathbf{F}_U^{-1}(\mathbf{A}_s\mathbf{F} - \mathbf{F}_q\dot{\mathbf{q}} - \mathbf{F}_t). \tag{6.147}$$

The differential system (6.147) along with the initial condition $\mathbf{U}(0) = \mathbf{U}_0$ is integrated through time in real-time to find $\mathbf{U}(t)$ at any given time $t$. Since $\mathbf{U}(t)$ includes the steps of the control action $\mathbf{u}_k$, it can determine the required control action at time $t$.

However, calculating the right hand side of the system (6.147) using the conventional way includes the determination of the Jacobian matrices $\mathbf{F}_U$ $(6N \times 6N)$ and $\mathbf{F}_q$ $(6N \times 5N)$. Since these matrices are very difficult to determine analytically, an approximation method should be used. However, approximating the Jacobian matrices directly is computationally cumbersome due to the large sizes of these matrices. A better way is approximating the product of these matrices with their corresponding multipliers, i.e., $\mathbf{F}_U\dot{\mathbf{U}}$ and $\mathbf{F}_q\dot{\mathbf{q}} + \mathbf{F}_t$. This approximation is written using a simple finite difference method.

$$\mathbf{F}_U\dot{\mathbf{U}} \approx \frac{\mathbf{F}(\mathbf{U} + \dot{\mathbf{U}}t_s, \mathbf{q}, t) - \mathbf{F}(\mathbf{U}, \mathbf{q}, t)}{t_s}, \tag{6.148}$$

and

$$\mathbf{F}_q\dot{\mathbf{q}} + \mathbf{F}_t \approx \frac{\mathbf{F}(\mathbf{U}, \mathbf{q} + \dot{\mathbf{q}}t_s, t + t_s) - \mathbf{F}(\mathbf{U}, \mathbf{q}, t)}{t_s}. \tag{6.149}$$

With this approximation, the solution for $\dot{\mathbf{U}}$ from Eq. (6.147) can be found using an efficient and fast numerical method called the GMRES method.

### The Generalized Minimum Residual Method

In mathematics, the GMRES method is an iterative method for the numerical solution of a system of equations. The method approximates the solution of the system by a vector in a Krylov subspace with a minimal residual. The Arnoldi iteration is used to find this vector. This method was developed in 1986 [65].

This method, applied to the current problem, can calculated $\dot{\mathbf{U}}$ at each control time $t$ such that the residual

$$r = ||\mathbf{F}_U\dot{\mathbf{U}} - (\mathbf{A}_s\mathbf{F} - \mathbf{F}_q\dot{\mathbf{q}} - \mathbf{F}_t)|| \tag{6.150}$$

is minimized. The GMRES method converges to the exact solution after $6N$ iterations, however, the idea is that after a small number of iterations (relative to $6N$),

the vector $\dot{\mathbf{U}}$ is already a good approximation to the exact solution.[1] In practice, a fixed low number of iterations are specified for the real-time calculations. This not only helps to speed up the real-time control, but also guarantees that the calculation burden for the real-time control stays constant over time, which is a crucial factor in control applications.

Note that the GMRES method needs an initial guess for $\mathbf{U}$ and $\dot{\mathbf{U}}$ for the first iteration. At the start of control at time zero, the initial guess for $\mathbf{U}$ is provided by the initial condition of the continuous system $\mathbf{U}_0$. At any time $t$ during the control, the initial guess for $\mathbf{U}$ is the control input calculated for the previous control step. The initial guess for $\dot{\mathbf{U}}$ can always be zero.

Once the rate of the stepwise control and the Lagrange multipliers $\dot{\mathbf{U}}$ are determined at a control step at time $t$, the control step heights and the Lagrange multipliers are found by intergrating their corresponding rates. Since the rate is

$$\dot{\mathbf{U}} = \left[\dot{\mathbf{u}}_0^T \ \dot{\boldsymbol{\mu}}_0^T \ \ldots \ \dot{\mathbf{u}}_k^T \ \dot{\boldsymbol{\mu}}_k^T \ \ldots \ \dot{\mathbf{u}}_{N-1}^T \ \dot{\boldsymbol{\mu}}_{N-1}^T\right]^T, \tag{6.151}$$

the step heights of the control commands and the Lagrange multipliers are

$$\mathbf{u}_0 = \mathbf{u}_{0g} + \dot{\mathbf{u}}_0 t_s \tag{6.152}$$

$$\mathbf{u}_k = \mathbf{u}_{k-1} + \dot{\mathbf{u}}_k t_s \quad k = 1, \ldots, N-1 \tag{6.153}$$

$$\boldsymbol{\mu}_0 = \boldsymbol{\mu}_{0g} + \dot{\boldsymbol{\mu}}_0 t_s \tag{6.154}$$

$$\boldsymbol{\mu}_k = \boldsymbol{\mu}_{k-1} + \dot{\boldsymbol{\mu}}_k t_s \quad k = 1, \ldots, N-1 \tag{6.155}$$

where $\mathbf{u}_{0g}$ and $\boldsymbol{\mu}_{0g}$ are extracted from $\mathbf{U}$ calculated at the previous control step.

### Hilare-Type Robot Specific Derivations

The following forms are defined for the terms that appear in the cost function (6.127) for the real-time optimization method:

$$\phi(\mathbf{q}_N) = \frac{1}{2}(\mathbf{q}_N - \mathbf{q}_N^d)^T \mathbf{Q}_0 (\mathbf{q}_N - \mathbf{q}_N^d), \tag{6.156}$$

$$L_k = \frac{1}{2}((\mathbf{q}_k - \mathbf{q}_k^d)^T \mathbf{Q}(\mathbf{q}_k - \mathbf{q}_k^d) + \mathbf{u}_k^T \mathbf{R}\mathbf{u}_k - \mathbf{S}\mathbf{u}_k). \tag{6.157}$$

In Eqs. (6.156) and (6.157), $\mathbf{Q}_0$ and $\mathbf{Q}$ are $5\times5$ positive-definite diagonal weight matrices. $\mathbf{R}$ is a $4 \times 4$ positive-semi definite matrix penalizing the control effort. Since, $u_3$ and $u_4$ are dummy inputs defined to convert the inequality constraint (6.118) to the equality constraint (6.119), they must not be penalized. Therefore, the diagonal matrix $\mathbf{R}$ is defined as

[1] The GMRES method, despite being new, is used extensively in numerical calculations. Some scientific and engineering calculation software such as MATLAB have built in functions that apply this solution method. See the MATLAB help on `GMRES` for more information.

$$\mathbf{R} = \begin{bmatrix} R_{11} & 0 & 0 & 0 \\ 0 & R_{22} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.158}$$

The sign of the dummy inputs $u_3$ and $u_4$ do not affect the optimality, since only their second powers are present in Eq. (6.119). In this situation, the solution for $\mathbf{U}$ can bifurcate when the dummy inputs become zero. To avoid this problem, a small $4 \times 4$ positive-semi definite dummy penalty weight matrix $\mathbf{S}$ has been added to the cost function (6.157). This matrix must not penalize the real control inputs $u_1$ and $u_2$. Therefore, it is selected to be

$$\mathbf{S} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & S_{33} & 0 \\ 0 & 0 & 0 & S_{44} \end{bmatrix}. \tag{6.159}$$

With the definition of the terms in the cost function, their partial derivatives, which appear in Eqs. (6.139) and (6.140) can be derived as

$$\frac{\partial \phi}{\partial \mathbf{q}_N^T} = (\mathbf{q}_N - \mathbf{q}_N^d)^T \mathbf{Q}_0, \tag{6.160}$$

and

$$\frac{\partial L_k}{\partial \mathbf{q}_k^T} = (\mathbf{q}_k - \mathbf{q}_k^d)^T \mathbf{Q}, \tag{6.161}$$

and

$$\frac{\partial L_k}{\partial \mathbf{u}_k^T} = \mathbf{u}_k^T \mathbf{R} - \mathbf{S}. \tag{6.162}$$

In the discretized model (6.128), the matrix $\mathbf{f}_k$ is

$$\mathbf{f}_k = \begin{bmatrix} q_{k4} \cos q_{k3} \\ q_{k4} \sin q_{k3} \\ q_{k5} \\ \frac{1}{m}(\frac{1}{r}(u_{k1} + u_{k2}) + m_r c q_{k5}^2) \\ \frac{T}{2I}\frac{1}{r}(u_{k1} - u_{k2}) \end{bmatrix}. \tag{6.163}$$

The partial derivative in Eqs. (6.139) and (6.140) can be calculated as

$$\frac{\partial \mathbf{f}_k}{\partial \mathbf{q}_k^T} = \begin{bmatrix} 0\ 0 & -q_{k4}\sin q_{k3} & \cos q_{k3} & 0 \\ 0\ 0 & +q_{k4}\cos q_{k3} & \sin q_{k3} & 0 \\ 0\ 0 & 0 & 0 & 1 \\ 0\ 0 & 0 & 0 & 2m_r c q_{k5}/m \\ 0\ 0 & 0 & 0 & 0 \end{bmatrix}, \tag{6.164}$$

and

$$\frac{\partial \mathbf{f}_k}{\partial \mathbf{u}_k^T} = \begin{bmatrix} 0 & 0 & 0\ 0 \\ 0 & 0 & 0\ 0 \\ \frac{1}{mr} & \frac{1}{mr} & 0\ 0 \\ \frac{T}{2Ir} & -\frac{T}{2Ir} & 0\ 0 \end{bmatrix}. \tag{6.165}$$

The partial derivatives of the input constraint relation (6.120) that appear in Eq. (6.140) can also be determined.

$$\frac{\partial \mathbf{C}_k}{\partial \mathbf{u}_k^T} = \begin{bmatrix} u_1 & 0 & u_3 & 0 \\ 0 & u_2 & 0 & u_4 \end{bmatrix}, \tag{6.166}$$

$$\frac{\partial \mathbf{C}_k}{\partial \boldsymbol{\mu}_k^T} = \mathbf{0}. \tag{6.167}$$

*Example 6.6.* Consider a Hilare-type robot with the following geometrical and mass properties.

$$m = 5\ \text{kg},\ m_r = 0.5\ \text{kg},\ c = 0\ \text{m},\ r = 0.1\ \text{m},\ T = 0.5\ \text{m},\ I = 0.05\ \text{kg.m}^2$$

The maximum driving torques at the wheels of the robot are $u_{1\max} = u_{2\max} = 0.35\,\text{Nm}$. Use the model predictive method to control the robot on the following desired trajectories.

(a) On a straight line with equation $x_2 = x_1$ with a linear speed of $\sqrt{2}/2\,\text{m/s}$. Assume that the robot is at rest at (0, 0) m and its longitudinal axis is aligned with the desired path.
(b) A sinusoidal path as defined in part (b) of Example 6.2. Assume that the robot is at (0, 0) m moving with a speed of $\sqrt{5}/2\,\text{m/s}$ with a direction of $\theta = \pi/4$ rad.

*Solution.* First the control horizon $T$ is selected as 0.05 s. This horizon is discretized into the $N = 5$ steps, leading to a time step of $t_s = T/N = 0.01$ s. Then, the controller cost function weight matrices in Eqs. (6.156) and (6.157), and the continuation gain $\mathbf{A}_s$ in Eq. (6.144) must be selected.

$$\begin{aligned}\mathbf{Q}_0 &= \text{diag}_{(5\times5)}[1, 1, 1, 1, 1],\\ \mathbf{Q} &= \text{diag}_{(5\times5)}[1, 1, 1, 1, 1],\\ \mathbf{S} &= \text{diag}_{(4\times4)}[0, 0, 0.1, 0.1],\\ \mathbf{R} &= \text{diag}_{(4\times4)}[0.1, 0.1, 0, 0],\\ \mathbf{A}_s &= \text{diag}_{(6N\times6N)}[-10, -10, -10, -10].\end{aligned}$$

At the first step of control at time zero ($t = 0$), Eq. (6.143) must be solved for $\mathbf{U}_0$ to provide the initial condition for the continuation Eq. (6.144). This is done via the following procedure.

1. The initial condition $\mathbf{q}(0)$ and an initial guess for $\mathbf{U}_0$ is used along with the discretized model of the robot (6.128) to predict the behavior of the robot at discretized times ($\mathbf{q}_{k+1}, k = 0, \ldots, N-1$).
2. The costates are calculated from Eqs. (6.138) to (6.139).
3. The components of $\mathbf{F}(\mathbf{U}_0, \mathbf{q}(0), 0)$ (Eq. (6.143)) are calculated using Eqs. (6.140) and (6.141).
4. The above procedure provides the input function $\mathbf{F}(\mathbf{U}_0, \mathbf{q}(0), 0)$ to a Gauss-Newton iterative algorithm, which finds the solution $\mathbf{U}_0$ through iterations.[2]

   Note that this method is computationally very expensive and should be used only for the first step of control calculations, when the system is being initialized and the calculation time is not crucial.

After the initial control command $\mathbf{U}_0$ is calculated, the continuation Eq. (6.144) is integrated in real-time for $\mathbf{U}$. This is done via the following procedure.

1. As any control time $t$, the current state of the system $\mathbf{q}(t)$ and an initial guess for the future control action $U_g$, which is normally chosen equal to the previous control action for faster convergence, are used along with the discretized model of the robot Eq. (6.128) to predict the behavior of the robot at discretized times ($\mathbf{q}_{k+1}, k = 0, \ldots, N-1$).
2. The costates are calculated from Eqs. (6.138) to (6.139).
3. The matrices $\mathbf{F}_U\dot{\mathbf{U}}$ and $\mathbf{F}_q\dot{\mathbf{q}} + \mathbf{F}_t$ are calculated from Eqs. (6.148) to (6.149).
4. The terms $\mathbf{F}_U\dot{\mathbf{U}}$ and $\mathbf{A}_s\mathbf{F} - \mathbf{F}_q\dot{\mathbf{q}} - \mathbf{F}_t$ from the inputs to a standard GMRES solver, which calculates $\dot{\mathbf{U}}$ using minimum iterations.
5. The control action $\mathbf{U}$ for time $t$ is found by integrating $\dot{\mathbf{U}}$ using Eqs. (6.152), (6.153), (6.154), and (6.155).
6. Only the control action $\mathbf{u}_0$ is applied to the robot at time $t$ because the approximation accuracy of other control actions found for the rest of the control horizon may not be adequate. The calculations resume at step one of this procedure for time $t + t_s$.

[2] The Gauss-Newton algorithm is a standard routine in many scientific and engineering calculation software. For example, the function `lsqnonlin` in the optimization toolbox of MATLAB can be used. For more information, see the MATLAB help.

**Table 6.1** The result of initialization procedure for the straight-line scenario

| $k$ | $\mathbf{u}_k^T = [u_1, u_2, u_3, u_4]_k^T$ (Nm) | $\boldsymbol{\mu}_k^T = [\mu_1, \mu_2]_k^T$ (dimensionless) |
|---|---|---|
| 0 | [0.3494, 0.3494, 0.0253, 0.0253] | [3.9464, 3.9498] |
| 1 | [0.3492, 0.3492, 0.0256, 0.0256] | [3.9088, 3.9093] |
| 2 | [0.3491, 0.3491, 0.0258, 0.0258] | [3.8687, 3.8688] |
| 3 | [0.3494, 0.3494, 0.0262, 0.0262] | [3.8224, 3.8216] |
| 4 | [0.3493, 0.3492, 0.0282, 0.0282] | [3.5484, 3.5502] |

The above procedures for initialization and real-time calculations are applied to the Hilare robot introduced in this example. The results for parts (a) and (b) follow.

(a) The result of the initialization procedure for $\mathbf{U}_0$ for the straight line motion scenario is listed in Table 6.1. The vector $\mathbf{U}_0$ is broken into its components using Eq. (6.142) for easier comprehension. As can be seen in Table 6.1, the $u_1$ and $u_2$ components are saturated at a value close to the maximum allowed control input of 0.35 Nm. The $u_3$ and $u_4$ dummy control input components are close to zero so that the actual inputs are penalized more strongly in the constraint Eqs. (6.120). The value of the Lagrange multipliers are also high compared to that of the cost function weights to emphasize more on the satisfaction of the constraint equations. These are all done automatically because of the way the optimization problem has been set up.

The initialization control action $\mathbf{U}_0$ is used with the real-time control procedure discussed previously. The control action $\mathbf{U}$ at time $t$ is calculated, $\mathbf{u}_0$ is extracted from $\mathbf{U}$, and is applied to the equations of motion of the robot as a constant input from time $t$ to $t + t_s$. The behavior of the robot is recorded and the value of $\mathbf{q}(t + t_s)$ is used for calculation of the control input for the next step in time. The results for the robot behavior are shown and discussed in the following.

The path of the robot is shown in Fig. 6.24. The robot stays on the desired linear path, however, its velocity is not the desired velocity at the beginning of the motion because it starts from rest. Figure 6.25 shows the linear speed of the robot. At the beginning of the motion, the curve of the robot's speed is linear. This is because of the fact that the driving torque of the robot wheels are saturated at a constant value, causing a constant linear acceleration. The robot has to move faster than the desired speed to catch up with its timed position on the path. After a while, the robot moves on the desired trajectory with the desired speed.

The control inputs to the robot are shown in Fig. 6.26. The two driving torques $u_1$ and $u_2$ are saturated at the maximum value at the beginning of the motion. The dummy inputs have low values at the beginning of the motion because the real inputs are high enough to satisfy the constraint equations. After the robot reaches its timed position on the path, the input torques are reduced to zero, which is the equilibrium input due to the absence of friction in the model. At this time, the dummy inputs are increased to 0.35 Nm to satisfy the constraints.

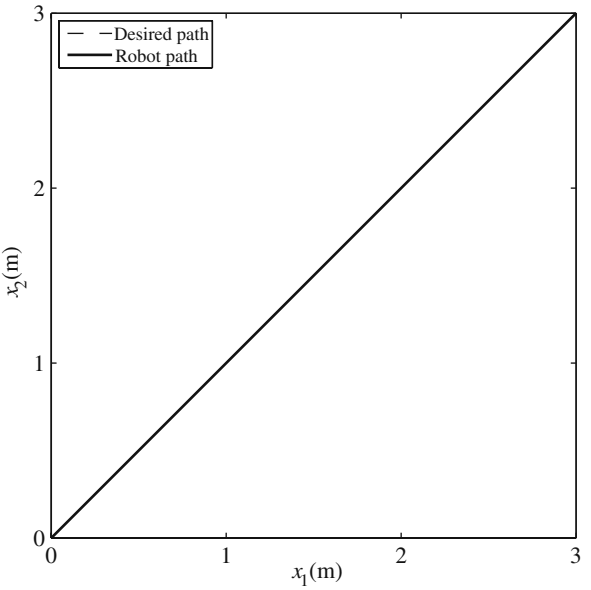


**Fig. 6.24** The path of the Hilare mobile robot

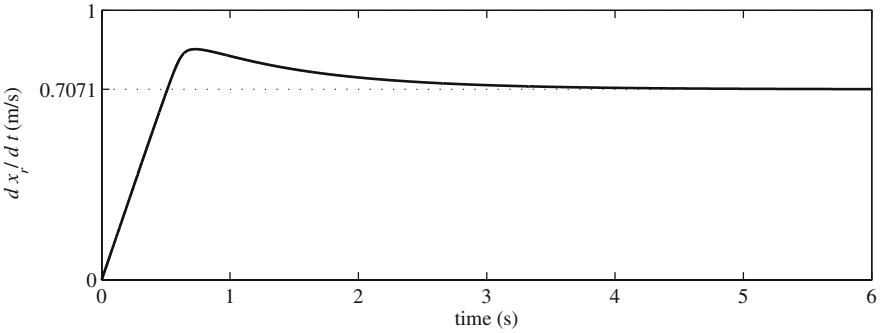


**Fig. 6.25** The speed of the Hilare mobile robot

Lagrange multipliers for the real-time optimization of the control action are plotted in Fig. 6.27. When the inputs are saturated, the multiplier gains a larger value to penalize the constraint equation more strongly. The multipliers reduce when the dummy inputs are saturated.

(b) The result of the initialization procedure for $\mathbf{U}_0$ for the sine path scenario is listed in Table 6.2. It can be seen that the $u_1$ and $u_2$ components are much lower than the maximum value for allowed control input of 0.35 Nm because the robot is already moving with the desired speed at the beginning of the motion. The $u_3$ and $u_4$ dummy control input components are 0.35 Nm to satisfy the constraint

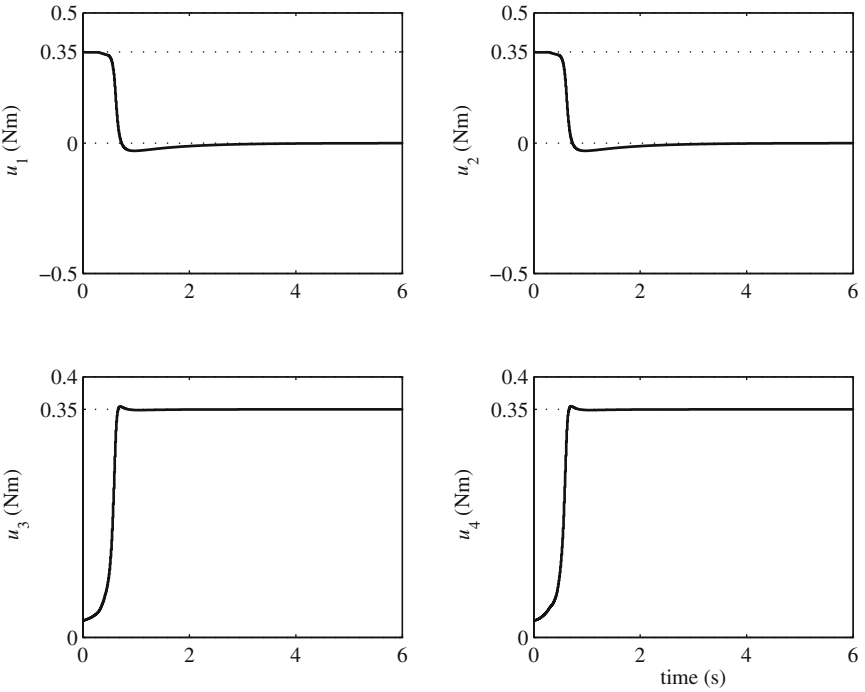


**Fig. 6.26** The driving torques and the dummy control inputs for the Hilare mobile robot

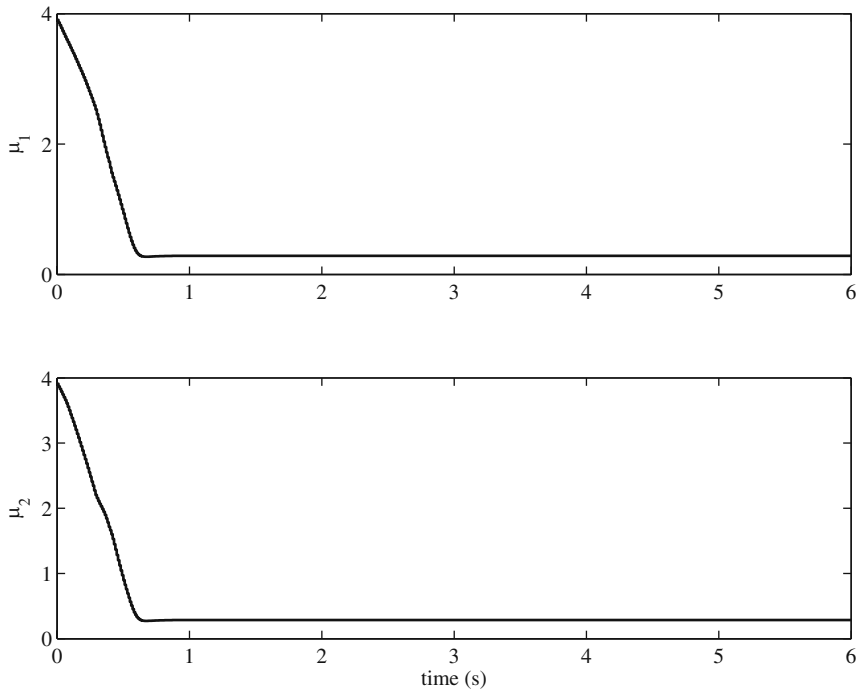


**Fig. 6.27** The Lagrange multiplier for the real-time optimizations for the Hilare mobile robot

**Table 6.2** The result of initialization procedure for the sine path scenario

| $k$ | $\mathbf{u}_k^T = [u_1, u_2, u_3, u_4]_k^T$ (Nm) | $\boldsymbol{\mu}_k^T = [\mu_1, \mu_2]_k^T$ (dimensionless) |
|---|---|---|
| 0 | [−0.0011, 0.0009, 0.3500, 0.3500] | [0.2857, 0.2857] |
| 1 | [−0.0012, 0.0008, 0.3500, 0.3500] | [0.2857, 0.2857] |
| 2 | [−0.0014, 0.0006, 0.3500, 0.3500] | [0.2857, 0.2857] |
| 3 | [−0.0017, 0.0003, 0.3500, 0.3500] | [0.2857, 0.2857] |
| 4 | [−0.0019, 0.0000, 0.3500, 0.3500] | [0.2857, 0.2857] |

Eqs. (6.120). The value of the Lagrange multipliers are also low compared to that of the cost function weights to emphasize more on the satisfaction of the desired states. The results for the robot behavior are shown and discussed in the following.

The path of the robot is shown in Fig. 6.28. The robot stays on the desired sine path, while its velocity is the desired velocity. Figure 6.29 shows the linear speed of the robot. The $x_1$ component of the robot's velocity is constant, while the $x_2$ component changes from a maximum of 1 m/s to a minimum of $-1$ m/s. Therefore, the minimum speed of 0.5 m/s happens when the $x_2$ component of the robot's velocity vanishes. The maximum velocity of the robot is 1.118 m/s ($\sqrt{1.0^2 + 0.5^2}$). These values are confirmed by Fig. 6.29.

The control inputs to the robot are shown in Fig. 6.30. The two driving torques $u_1$ and $u_2$ are below the maximum value at all times. These torques vary with time to provide the required change in the robot's momentum while moving on a sine curve. The dummy inputs remain close to the maximum value of 0.35 Nm to satisfy the control input constraints.

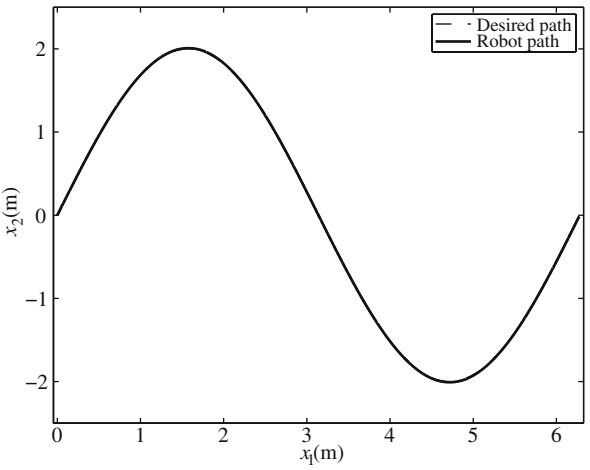


**Fig. 6.28** The path of the Hilare mobile robot

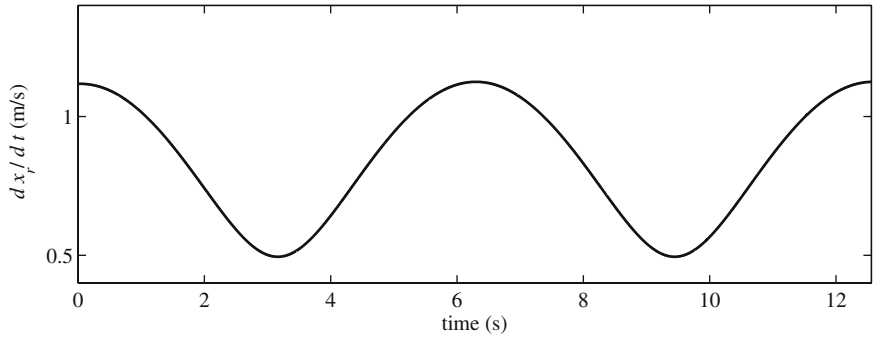


**Fig. 6.29** The speed of the Hilare mobile robot

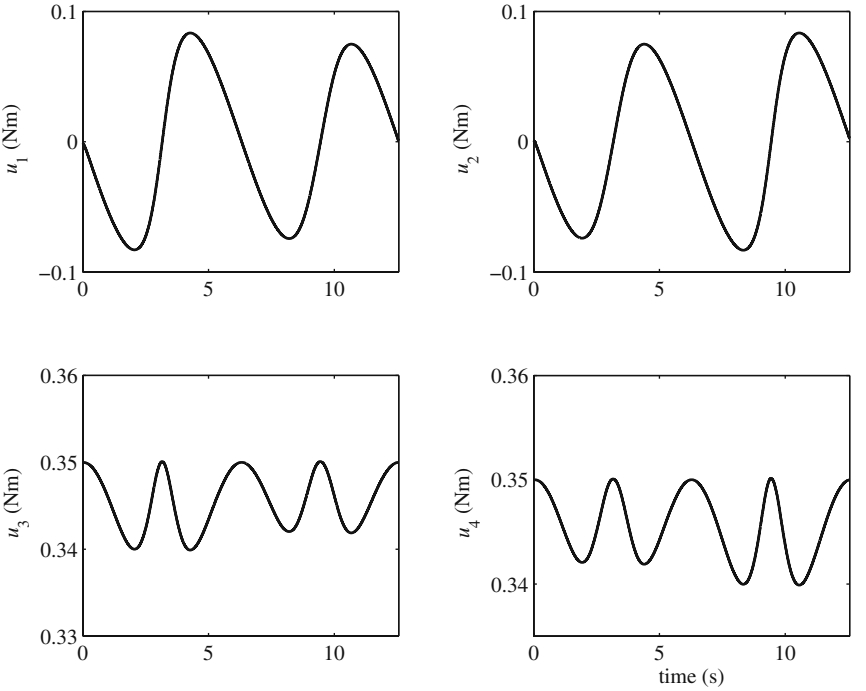


**Fig. 6.30** The driving torques and the dummy control inputs for the Hilare mobile robot

Lagrange multipliers for the real-time optimization of the control action are plotted in Fig. 6.31. The multipliers are small compared to the cost function weights for the states errors, which gives more priority for error reduction rather than the input saturation. The values of the Lagrange multipliers only change marginally through the motion.

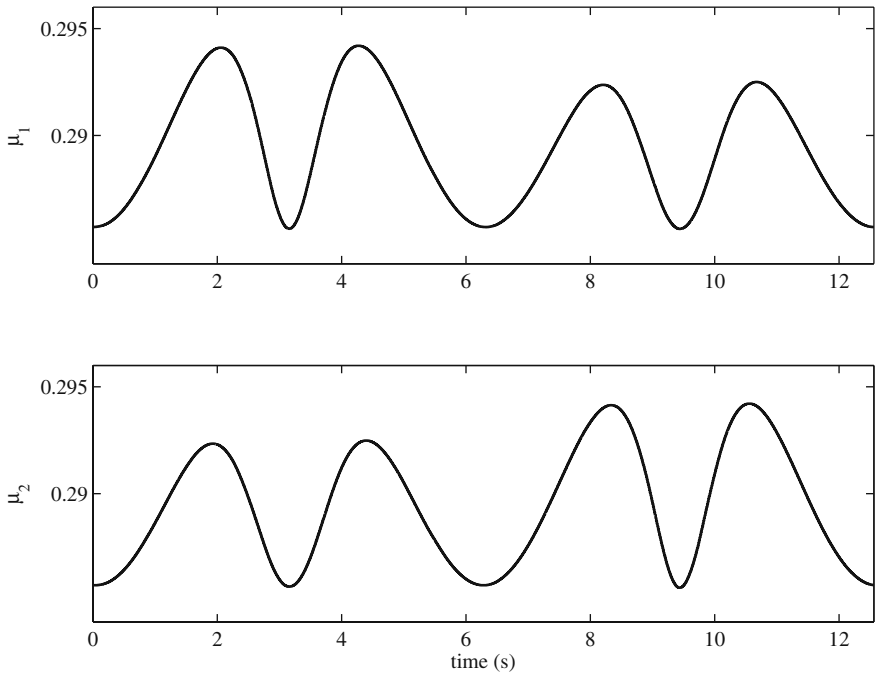


**Fig. 6.31** The Lagrange multiplier for the real-time optimizations for the Hilare mobile robot

## Problems

**Problem 6.1.** Consider the Hilare robot shown in Fig. 6.1 with the following geometrical and mass properties:

$$m = 5 \text{ kg}, m_r = 0.5 \text{ kg}, c = 0 \text{ m}, r = 0.1 \text{ m}, T = 0.5 \text{ m}, I = 0.05 \text{ kg.m}^2.$$

(a) Design a control law based on the chain form of the kinematic equations of the robot such that the eigenvalues of the closed-loop system are

$$\begin{aligned} p_1 &= -\lambda_1, \\ p_2 &= -\lambda_2 + i\lambda_3, \\ p_3 &= -\lambda_2 - i\lambda_3, \end{aligned}$$

where $\lambda_i > 0$.

(b) Apply the control law to the kinematic model of the Hilare robot such that the robot follows the path shown in Fig. 6.32 with a constant linear velocity of 0.25 m/s. The two linear sections have a length of $a = 0.5$ m and the two middle sections are circular arcs with a radius of $r = 2/(2 - \sqrt{2})$ m. Assume that the

robot is initially at the origin and its longitudinal axis makes a 45° angle with the $x_1$ axis.

(c) Using the dynamic model (6.116), calculate the wheel torques required to control the robot on the desired path. Obtain the minimum power of the motors wheels for completing this maneuver. What could happen if the minimum power requirement is not met?

(d) Determine the maximum lateral force that the robot wheel must be able to withstand for the robot to move on this path. What could happen if the minimum lateral force requirement is not met?

(e) Calculate the force that the wheels exert on the ground. Determine the minimum longitudinal coefficient of friction between the wheels and the ground for the robot to be able to successfully complete this maneuver. What could happen if the minimum friction requirement is not met?

(f) Investigate the effect of the controller gains on the maximum power, the maximum lateral force, and the minimum required coefficient of longitudinal friction between the wheels and the ground. Could these be used as criteria for designing the controller gains?

(g) Can the definition for the desired path be improved to allow for less demanding design criteria for the robot? How can the path be improved?

**Problem 6.2.** Consider the car-like robot shown in Fig. 6.2 with a wheelbase of $L = 0.25\,\text{m}$.

(a) Design a control law based on the chain form of the kinematic equations of the robot such that the eigenvalues of the closed-loop system are

$$\begin{aligned} p_1 &= -\lambda_1, \\ p_2 &= -\lambda_2, \\ p_3 &= -\lambda_3 + i\lambda_4, \\ p_4 &= -\lambda_3 - i\lambda_4, \end{aligned}$$

where $\lambda_i > 0$.

(b) Apply the control law to the kinematic model of the car-like robot such that the robot follows the path shown in Fig. 6.32 with a constant linear velocity of 0.25 m/s. The two linear sections have a length of $a = 1.0\,\text{m}$ and the two middle sections are circular arcs with a radius of $r = 1/(2 - \sqrt{2})$ m. Assume that the robot is initially at the origin and its longitudinal axis makes a 45° angle with the $x_1$ axis.

(c) Find the maximum steering rate $\dot{\phi}_{\max}$ that the car-like robot needs for following the desired trajectory. Assume that the robot has a maximum allowed steering rate $\dot{\phi}_{\text{allowed}}$ that is 10% lower than the $\dot{\phi}_{\max}$. Saturate the control input $\dot{\phi}$ in your simulation program by reassigning a value of $\dot{\phi}_{\text{allowed}}$ to it if the calculated value at any step is larger than $\dot{\phi}_{\text{allowed}}$. Investigate the behavior of the car-like robot

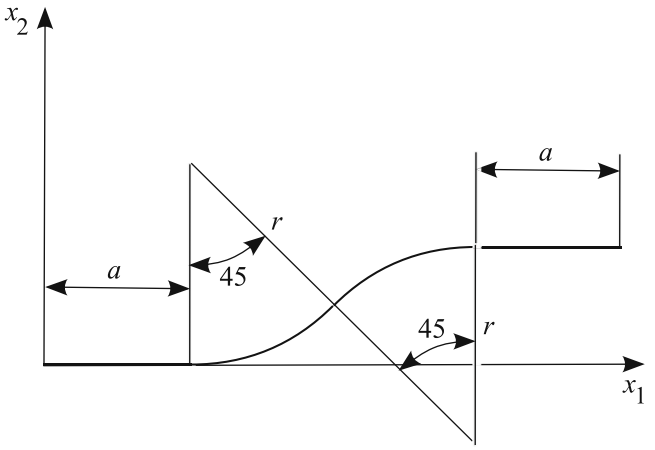


**Fig. 6.32** The desired path for the mobile robot

for this case. Try lower values for $\dot{\phi}_{\text{allowed}}$. How do they affect the response of the car-like robot?

(d) The only model parameter that is used for control design is the wheelbase of the car-like robot, which normally can be measured rather accurately. Assume $\pm 2\%$ uncertainty in this length. Introduce this uncertainty in the kinematic model of your simulation program, while the control calculations are based on the nominal wheelbase given in this problem. Compare the response of the robot with uncertainty in the wheelbase against that of nominal model. Is the loss of performance acceptable? Do the controller gains affect this loss of performance?

**Problem 6.3.** Consider the Hilare robot of Problem 6.1. Use the control law (6.82) derived for the $l - \alpha$ control scheme to make the robot follow a leader robot with a distance of $l_{12}^d = 2$ m and a view angle of $\alpha_{12} = \pi/4$ rad. Assume the leader moves on a straight line coincident with the $x_1$ inertial axis with a constant speed of 1 m/s and starts its motion at time zero at the origin. The follower is initially located at (0, 2) m with $\theta_2(0) = 0$ rad.

**Problem 6.4.** Consider the Hilare robot of Problem 6.1. Use the control law (6.93) derived for the $l - l$ control scheme to make the robot follow two leader robots with a distance of $l_{13}^d = l_{23}^d = 2$ m. Assume the leaders move on straight lines parallel to the $x_1$ inertial axis with a constant speed of 1 m/s and start their motion at time zero at points (0, 1) and (0, −1) m, respectively. The follower is initially located at (0, 2) m with $\theta_3(0) = 0$ rad.

**Problem 6.5.** Design a $l - \alpha$ formation controller for the car-like robot of Problem 6.2 based on a kinematic model using the input–output linearization method. Simulate the motion of the follower car-like robot, if the leader robot moves on a straight line coincident with the $x_1$ inertial axis with a constant speed of 1 m/s and

starts its motion at time zero at the origin. The follower is initially located at (0, 2) m with $\theta_2(0) = 0$ rad.

**Problem 6.6.** Repeat Problem 6.5 for a $l - l$ formation controller.

**Problem 6.7.** Consider the Hilare robot shown in Fig. 6.1 with the following geometrical and mass properties:

$$m = 5 \text{ kg}, m_r = 0.5 \text{ kg}, c = 0 \text{ m}, r = 0.1 \text{ m}, T = 0.5 \text{ m}, I = 0.05 \text{ kg.m}^2.$$

Assume that the wheel torques are limited to 0.35 Nm.

(a) Use the model predictive control method (continuation/GMRES) to design a controller based on the dynamic model of the robot given in Eq. (6.116) that meets the maximum torque constraints. Use this controller and simulate a Hilare robot to follow the desired trajectory shown in Fig. 6.32 with a constant speed of 0.5 m/s. Assume that the robot is at rest at the origin while its longitudinal axis is aligned with the $x_2$ axis.

(1)Plot the path of the robot over the desired path.
(2)Plot the actual and the dummy control inputs.
(3)Plot the Lagrange multipliers.
(4)Plot the speed of the robot.
(5)Interpret and discuss all of the plots.

(b) Assume that there are $\pm 5\%$ uncertainties in the values of the robot equivalent mass $m$, the wheel mass $m_r$, the equivalent moment of inertia of the robot $I$, and the location of the center of gravity $c$. Incorporate these uncertainties in the dynamic model for simulating the response of the robot, while using the nominal parameter for control calculations. Introduce two unequal friction forces at the wheels in the dynamic model. Simulate the response of the robot with uncertainties compare the results to that of the response of the nominal system found in part (a) of this problem. Discuss the differences. Is the model predictive control method robust against unmodeled dynamics and parameter uncertainty?

**Problem 6.8.** Assume that the power at wheels of the Hilare-type robot of Problem 6.7 is limited to $P_{\max}$. Write the appropriate constraint equations to be used with a model predictive control method. What terms in the control derivations are affected by a change of constraint from the constraint on maximum driving torque at the wheels to the constraint on the driving power at the wheels? Recalculate those terms.

**Problem 6.9.** Derive a model predictive control algorithm based on the kinematic model of a car-like robot. Incorporate the maximum speed and the maximum steering rate as constraints. What benefits are anticipated by using a model predictive controller rather than a controller designed based on the chain form?

# Chapter 7
# Autonomous Surface Vessels

## 7.1 Introduction

The control of surface vessels has been investigated by many researchers in the past decade. In most classical literature, the controllers have been designed to maintain the motion of a surface vessel on a linear course with a constant speed. These controllers are known as autopilots. They are meant to assist the vessel's crew in controlling the vessel on long trips, and work similar to an automobile's cruise control. The controllers have been used on large- and medium-size ships. However, they cannot be used for faster smaller surface vessels that must work autonomously, for which controllers capable of trajectory tracking are needed.

Designing controllers for fast surface vessels is challenging. Uncertainty in dynamic models, significant sea disturbances, underactuated dynamics, and lack of nonholonomic kinematic constraints are the issues that a designer must deal with while designing a robust controller for a surface vessel. The more accurate model of a surface vessel has six DOFs. It is common to simplify this model to a 3-DOF model that only reflects surge, sway, and yaw DOFs, the ones that have to be controlled by a trajectory-tracking controller. With this simplification, the model of a surface vessels seems similar to that of a mobile robot. However, there are strong reasons why these two are very different and have to be treated differently.

- The dynamic rather than the kinematic model of the surface vessel must be used for designing the controllers. Because, first, forces and moments are the available physical control inputs. Second, a kinematic model alone cannot determine the lateral motion response of a surface vehicle due to the lack of a nonholonomic lateral motion constraint.
- A surface vessel has a nonlinear dynamic model. The controller development for a surface vessel may be simplified by using lineariazed models and the classical PID control methods. However, with linearized dynamic model and classical control methods, one can only prove the quality of performance of the controller for the maneuvers in which the state of the system is in the vicinity of the linearization state. Using nonlinear control theories, on the other hand, one can conclude about the quality of the system response for the full range of vessel's motion.

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_7

- Since there is no constraint for the lateral motion of surface vessels, they can be categorized as holonomic systems. Also, a surface vessel has more DOFs than actuators. Therefore, surface vessels are considered as underactuated systems. This makes the controller design for such a system even a more challenging task. For a surface vessel as an underactuated holonomic system, due to the absence of the lateral motion constraint, the vessel's orientation during its motion is not necessarily tangent to the motion path. Therefore, the stability of the zero dynamics of the vessel's unactuated DOF needs rigorous proof based on the vessel's dynamic model.

These three problems, which are specific to the control of surface vessels, are addressed in this chapter. First, the 6-DOF dynamic model of a surface vessel is presented. This dynamic model is reduced to a 3-DOF model. Then, a control strategy based on the "control point" is presented, in which the unactuated dynamics of the vessel can be analyzed separately than the actuated DOFs. The inherent stability of the unactuated DOF is investigated. Next, two methods for trajectory-tracking controller design for a surface vehicle are introduced. These methods are the feedback linearization method and the robust sliding mode control. Finally, the problem of controlling multiple surface vessels simultaneously in group formation maneuvers is solved using a decentralized leader-follower approach.

## 7.2 Dynamics of a Surface Vessel

In this section, first, the 6-DOF dynamic model of a surface vessel is presented. The 6-DOF dynamic model is useful for simulating the motion of a surface vessel. However, for control development, a simpler 3-DOF model is more appropriate. Therefore, later in this section, the 6-DOF dynamic model is reduced to derive a 3-DOF model.

The six DOFs for a surface vessel consist of the three global position components of the center of mass of the vessel and three angles that define the orientation of the vessel's body frame with respect to the inertial global frame. These six DOFs, which can uniquely define the configuration of the vessel at any instant in time, are accompanied by six generalized speed components, which define the dynamic state of vessel. These generalized speeds are defined in terms of the vessel's body frame. They are the surge ($u$), sway ($v$), and heave ($w$) linear speeds, and the angular speeds about the longitudinal ($p$), transversal ($q$), and normal ($r$) axes.

For a surface vessel, normally there are two control inputs. The types of inputs depend on the drive train of the vessel. For example, two independent propellers can provide the driving force ($F$) and steering torque ($T$) for the system. If the inertia of the vessel can be assumed to be constant, the vessel has an elliptical body, and the higher-order damping forces can be neglected, the following equations describe the dynamics of the vessel in the local coordinate system [32]:

$$\begin{aligned}
m_{11}\dot{u} - m_{22}vr + m_{33}wq + d_{11}u &= W_u + F,\\
m_{22}\dot{v} - m_{33}wp + m_{11}ur + d_{22}v &= W_v,\\
m_{33}\dot{w} - m_{11}uq + m_{22}vp + d_{33}w &= W_w + mg + Z_w,\\
I_{xx}\dot{p} + (m_{33} - m_{22})wv + (I_{zz} - I_{yy})rq + d_{44}p &= K_p,\\
I_{yy}\dot{q} + (m_{11} - m_{33})uw + (I_{xx} - I_{zz})pr + d_{55}q &= F,\bar{FG} + M_q\\
I_{zz}\dot{r} + (m_{22} - m_{11})vu + (I_{zz} - I_{yy})qp + d_{66}r &= T,
\end{aligned} \tag{7.1}$$

where $m_{ij}$'s denote the added mass of the vessel (including the effect of hydrodynamics), $d_{ij}$'s represent the linear hydrodynamic damping coefficients, $I_{ii}$ denote the moment of inertia, and $W_i$ are the wave force components described in terms of the vessel's local frame. $F\bar{FG}$ is the torque of the thruster's force about the transversal (pitch) axis passing through the vessel's center of mass. $Z_w$, $K_p$, and $M_q$ are the buoyancy force, and roll and pitch restoring torques, respectively. That is

$$K_p = -mg\bar{MT}_p \sin\phi, \quad M_q = -mg\bar{MT}_q \sin\theta, \quad Z_w = -\rho g A_{wp} z. \tag{7.2}$$

In the above relations, the transverse and longitudinal metacentric heights are denoted by $\bar{MT}_p$ and $\bar{MT}_q$, respectively, and the water plane area is presented by $A_{wp}$. The roll and pitch angles of the vessel's local frame are denoted by $\phi$ and $\theta$.

As can be seen in Eq. (7.2), this dynamic model is rather complicated. The derivation of control laws become simpler if this model can be reduced. For this reduction, the inherent stability of some of the vessel's DOFs, which is intentionally built into the vessel's dynamics by a good vessel design, must be used. The buoyancy force of the vessel stabilizes the heave motion. Also, since the vessels is designed to have adequate longitudinal and transversal metacentric heights, the roll and pitch motion of the vessel is stabilized. The stability of the heave ($w$), roll ($p$), and pitch ($q$) motions, and the assumption that the angular DOFs are small, allow us to neglect them for the controller design. With this in mind, one can reduces the 6-DOF dynamic model to a 3-DOF one.

Figure 7.1 shows the reduced 3-DOF model of a surface vessel. Only, $u$, $v$, and $r$ are considered as the three generalized speeds of the vessel expressed in the vessel's body frame. With this assumption, the following equations describe the dynamics of the vessel in the local coordinate system [32].

$$\begin{aligned}
m_{11}\dot{u} - m_{22}vr + d_{11}u &= W_u + F,\\
m_{22}\dot{v} + m_{11}ur + d_{22}v &= W_v,\\
I_{zz}\dot{r} + (m_{22} - m_{11})uv + d_{66}r &= T.
\end{aligned} \tag{7.3}$$

Note that the second equation is not directly affected by the controller inputs. However, this equation determines the lateral motion response of the surface vessel.

When the control inputs are given, the local equations of motion (7.3) can simulate the behavior of the surface vessel. However, since, in the future sections of this chapter, the control of relative distances of multiple surface vessels will be

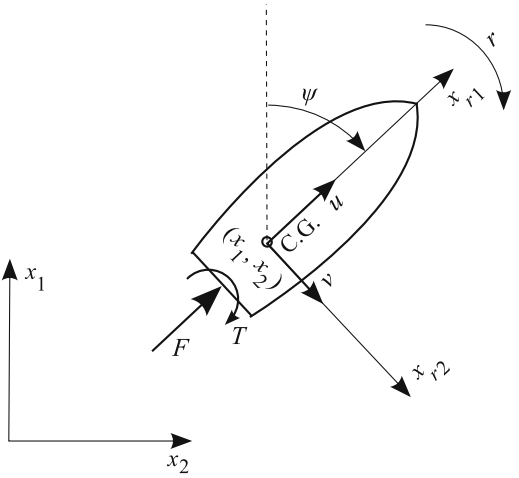


**Fig. 7.1** A 3-DOF dynamic model of a vessel

considered, the equations of motion of the surface vessel in the global coordinate system becomes very useful. Therefore, here, these equations are also presented.

The dynamic equations in terms of the global coordinated can be found by using Eq. (7.3) along with the kinematic relations between the local speed components $(u, v, r)$ and the global speed components $(\dot{x}, \dot{y}, \dot{\psi})$. The result is

$$\begin{aligned} \ddot{x} &= \frac{1}{m_{11}} \left(f_x + F \cos\psi\right), \\ \ddot{y} &= \frac{1}{m_{22}} \left(f_y + F \sin\psi\right), \\ \ddot{\psi} &= \frac{1}{I_{zz}} \left(f_\psi + T\right), \end{aligned} \tag{7.4}$$

where

$$\begin{aligned} f_x &= m_r d_{22} v \sin\psi - d_{11} u \cos\psi + \dot{\psi}(v \cos\psi - m_r u \sin\psi) m_d, \\ f_y &= -m_r d_{22} v \cos\psi - d_{11} u \sin\psi + \dot{\psi}(v \sin\psi + m_r u \cos\psi) m_d, \\ f_\psi &= -m_d u v - d_{33} \dot{\psi}, \\ u &= \dot{x} \cos\psi + \dot{y} \sin\psi, \\ v &= -\dot{x} \sin\psi + \dot{y} \cos\psi, \end{aligned} \tag{7.5}$$

and

$$\begin{aligned} m_d &= m_{22} - m_{11}, \\ m_r &= \frac{m_{11}}{m_{22}}. \end{aligned}$$

Equations (7.4) and (7.5) describe the dynamics of the vessels in terms of the global coordinate components.

## 7.3 The Control Point Concept for Underactuated Vehicles

As discussed previously, the reduced model of the surface vessel has three DOFs, whereas there are only two control inputs available. At the first glance, it may seem impossible to control a 3-DOF system with only two control inputs. In fact, an underactuated system must have some inherent stability to be controllable. For a surface vessel, the two control inputs usually affect the surge and the yaw motions directly. The lateral (sway) motion of the vehicle is not directly actuated. This fact can be observed from the equations of motion (7.3) of a surface vessel. However, one can reflect on their real world experience with surface vessels and remember that humans can successfully control these vessels using only two inputs. This can only mean that the unactuated DOF for a surface vessel is inherently stable. In fact, this stability has been designed into the system. Using this real world experience with the system, a control engineer can confidently choose a representative of two of the three DOFs of the surface vessel as the output of the controller and design a controller to make them track a desired trajectory. However, the control engineer must not only rely on their real world experience about the stability of the unactuated DOF, but also mathematically prove the inherent stability of a representative of the uncontrolled DOF.

### *7.3.1 The Role of the Control Point*

For an underactuated system, one has to select the most important DOFs to control directly. The number of these DOFs must be equal to the number of control inputs. For a surface vessel, if the goal is to make the vessel follow a path (or a trajectory, to be more precise), the two position components are the most important out of the three DOFs. Based on this justification, the simplest choice of the controller outputs seems to be the the two position components of the vessel's center of mass. However, this simple choice is not the best choice. If the components of the vessel's center of gravity are used as the controller output, the controller will not sense any disturbance in the other DOF, the yaw. It is never a good idea to make a controller unaware of any states of the system.

Controlling the position components of a point on the vessel's body other that the center of mass can solve this problem. This point is called the "control point." The postion of this point is a function of all the DOFs of the vessel. When the control point's position components are chosen as the controller output, a disturbance generated by an external source only in the orientation of the vessel will also disturb the controller output, to which the controller can react. This reaction would not happen if the postion of the center of gravity was chosen as the controller output.

Let us pick the control point on the positive longitudinal axis of the surface vessel with a distance $d$ from the vessel's center of gravity. Assuming the control point on the longitudinal axis of the vessel simplifies the relations between the inputs and outputs of the controller. The controller outputs are defined as

$$x_{p1} = x_1 + d\cos\psi,$$
$$x_{p2} = x_2 + d\sin\psi. \tag{7.6}$$

The desired values of these outputs (the trajectory of the surface vessel) is defined by functions of time. That is,

$$x_{p1}^d = x_{p1}^d(t),$$
$$x_{p2}^d = x_{p2}^d(t). \tag{7.7}$$

One can assume that a controller will be designed can stabilize the control outputs (the position of the control point). However, the inherent stability of a representative of the DOF that is not directly controlled must be investigated. This is done in the following subsection.

## 7.4 Zero-Dynamics Stability for a Surface Vessel

Let us assume that the controllers, to be designed, guarantee that the position of the control point follows the desired trajectory. Although the trajectory of the control point is stabilized, there may be a possibility that the vessel oscillates about the control point $p$ during the motion. The oscillation may cause the vessel to become unstable. In such a situation, the vessel's behavior may resemble that of a pendulum oscillating about the moving pivot (the control point). In this case, the position components of the center of mass and the orientation of the vessel may have periodic or unstable trajectories. Therefore, the stability of the zero dynamics of the vessel must be investigated.

Since the oscillations would take place about the control point, the position components of the center of mass and the orientation of the vessel will not be independent. In fact, the trajectory of the center of mass can be determined by knowing the control point's motion, and the orientation of the vessel. Therefore, analyzing the stability of the orientation only can conclude the stability of the vessel as a system. This is consistant with the fact that there is only one underactuated degree of freedom for a 3-DOF vessel.

In the following, the stability of the orientation of the vessel, as a representative of the stability of the zero dynamics of the vessel, is investigated. A general planar motion is assumed for the control point $p$. The velocity and acceleration vectors of point $p$ can be written as

$$\mathbf{v}_p = u_p\hat{\mathbf{t}}, \qquad \mathbf{a}_p = \dot{u}_p\hat{\mathbf{t}} + \frac{u_p^2}{\rho}\hat{\mathbf{n}}, \tag{7.8}$$

where $u_p$ is the linear velocity, $\dot{u}_p$ is the linear acceleration of the motion of the control point, and $\rho$ is the radius of curvature of control point's path. Also, $\hat{\mathbf{t}}$ and $\hat{\mathbf{n}}$ are the unit vectors tangent and perpendicular to the path of $p$, respectively (Fig. 7.2).

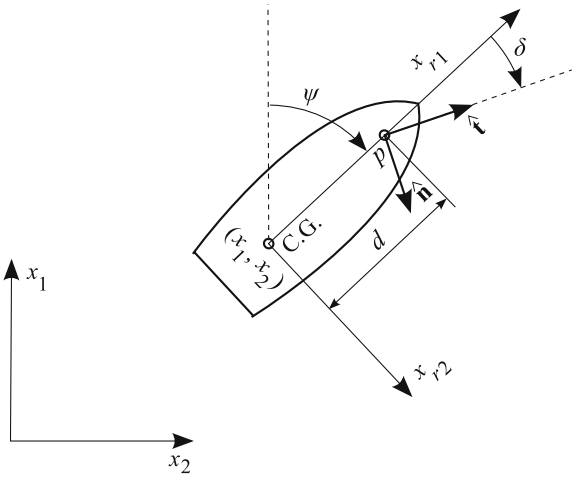


**Fig. 7.2** The velocity of point $p$, which is along $\hat{\mathbf{t}}$, makes an angle $\delta$ with the orientation of the follower

The second equation of motion in Eq. (7.3) is not affected by any of the control inputs. This equation is perfect for zero-dynamic stability analysis. However, this equation must be written in terms of only one variable, representing the orientation of the vessel. An angle $\delta$ is defined as the difference between the orientation of the vessel and the direction of the velocity of point $p$. Then, the velocity and acceleration of the center of mass are calculated in terms of the velocity and acceleration of the control point. These are substituted into the second equation of motion in Eq. (7.3). The result is

$$\ddot{\delta} + \left(\frac{d_{22}d - m_{11}u_p\cos\delta}{m_{22}d}\right)\dot{\delta} + \left(\frac{m_{22}\dot{u}_p + d_{22}u_p}{m_{22}d}\right)\sin\delta - \left(\frac{m_{22}u_p^2/\rho + m_{11}u_p r_p}{m_{22}d}\right)\cos\delta = -\left(\frac{d_{22}r_p + m_{22}\dot{r}_p}{m_{22}}\right), \tag{7.9}$$

where $r_p$ is the rate of change of the direction of the velocity of point $p$. Linearization of this equation about $\delta = 0$ yields

$$\ddot{\delta} + \frac{d_{22}}{m_{22}}\dot{\delta} + \left(\frac{m_{22}\dot{u}_p + d_{22}u_p}{m_{22}d}\right)\delta = -\left(\frac{d_{22}r_p + m_{22}\dot{r}_p}{m_{22}}\right). \tag{7.10}$$

Once again, it should be noted that the trajectory of the control point is dictated by the controller. The variable $\delta$ represents the response of the orientation of the surface vessel to the motion of the control point. Equation (7.10) defined the behavior of the variable $\delta$, as the representative of the stability of the zero dynamics of the vessel, as a function of the motion of the control point. In the following, this behavior is investigated for different motions of the control point as an excitation.

### 7.4.1 Stability in Case of General Motions with Constant Speed

In this section, a general motion with constant speed is considered for the control point and the behavior of the unactuated DOF is investigated. Constant speed implies that

$$u_p = \bar{u}_p, \quad \dot{u}_p = 0. \tag{7.11}$$

The direction of the control point's velocity is variable for a general motion. This orientation is denoted by nonzero $r_p$ and $\dot{r}_p$. When these relations are substituted into Eq. (7.10), the following equation results:

$$\ddot{\delta} + c\dot{\delta} + (\frac{c}{d}\bar{u}_p)\delta = -(cr_p + \dot{r}_p), \tag{7.12}$$

where

$$c = \frac{d_{22}}{m_{22}}. \tag{7.13}$$

The roots of the characteristic equations of this second-order equation determine its stability. These roots are

$$r_{1,2} = -(\frac{c}{2}) \pm \sqrt{\Delta}, \qquad \Delta = (\frac{c}{2})^2 - \frac{c}{d}\bar{u}_p. \tag{7.14}$$

Stability can be guaranteed if one of the following cases is true: either the roots of the characteristic equations must be negative real numbers or it must be complex conjugates with negative real parts. The conditions on $\bar{u}_p$ that satisfy these two cases must be investigated.

1. **Case 1: Two negative real roots.** The following inequalities must be true for the characteristic equation to have two negative real roots:

$$\Delta > 0, \qquad r_1 = -(\frac{c}{2}) - \sqrt{\Delta} < 0, \qquad r_2 = -(\frac{c}{2}) + \sqrt{\Delta} < 0. \tag{7.15}$$

   For the first inequality to be true, $\bar{u}_p < cd/2$. Since both $c$ and $\Delta$ are positive, if the first inequality is true, the second inequality is identically satisfied. Also, since $d > 0$, if

$$\bar{u}_p > 0, \tag{7.16}$$

   the third inequality is satisfied.
2. **Case 2: Two complex conjugate roots with negative real parts.** Another condition under which the zero dynamics is stable is that the characteristic equation

has two complex conjugate roots with negative real parts. This may be the case if $\Delta < 0$, which results in another condition for the linear speed $\bar{u}_p$:

$$\bar{u}_p > \frac{cd}{2} \tag{7.17}$$

It is important to note that the real part of the roots is negative because $c > 0$.

Two conditions have been found for $\bar{u}_p$ based on the mentioned two possible cases. These two conditions must be combined to conclude the zero-dynamic stability of the vessel. It can be concluded that the zero dynamics of the vessel is stable for a forward motion, that is, when $\bar{u}_p > 0$. The second-order zero dynamics of the vessel is over-damped when $0 < \bar{u}_p < \frac{cd}{2}$, and is critically-damped when $\bar{u}_p = \frac{cd}{2}$, and is damped oscillatory when $\bar{u}_p > \frac{cd}{2}$. Note that the distance of the vessel's control point from its center of mass, $d$, determines the quality of the zero-dynamics response. While $c$ depends on the dynamic properties of the vehicle, a larger $d$ can expand the range of operational speeds for which the response of the zero dynamics is over-damped.

### 7.4.2 *Equilibrium Point for Circular and Linear Motions with Constant Speed*

In the previous section, it was shown that the zero-dynamic response of a surface vessel is in fact stable. In this section, the equilibrium point for the orientation of the vessel, when the control point has a circular motion with constant speed, is derived.

The control point velocity components for the circular motion with constant speed become

$$u_p = \bar{u}_p, \dot{u}_p = 0, r_p = \bar{r}_p, \dot{r}_p = 0. \tag{7.18}$$

To find the equilibrium orientation, $\delta^e$, the derivatives $\ddot{\delta}$ and $\dot{\delta}$ in Eq. (7.9) must vanish. This results in

$$d_{22}\bar{u}_p \sin\delta^e - (m_{22}\bar{u}_p^2/\rho + m_{11}\bar{u}_p\bar{r}_p)\cos\delta^e = -d_{22}d\bar{r}_p. \tag{7.19}$$

Solving this equation in terms of $\delta^e$ yields

$$\delta^e = \arccos\left(\frac{d_{22}d\bar{r}_p}{\beta_1}\right) - \beta_2, \tag{7.20}$$

where

$$\beta_1 = \sqrt{(d_{22}\bar{u}_p)^2 + (m_{22}\bar{u}_p^2/\rho + m_{11}\bar{u}_p\bar{r}_p)^2}, \tag{7.21}$$

$$\beta_2 = \arctan\left(\frac{d_{22}\bar{u}_p}{m_{22}\bar{u}_p^2/\rho + m_{11}\bar{u}_p\bar{r}_p}\right). \tag{7.22}$$

Equation (7.20) indicates that when the control point $p$ has a circular motion with constant speed, $\delta$, the difference between the orientation of the surface vessel and the direction of the velocity of the control point $p$ converges to a constant value.

Note that the direction of the desired motion of point $p$ is the desired path's slope. When the vessel's orientation reaches an equilibrium offset $\delta^e$ with the slope of the desired path, it has a constant offset with the tangent to the desired path.

The equilibrium $\delta$ when the motion of the control point is linear with a constant speed can be derived from the result for a circular control point motion. $\rho$ is infinite and $\bar{r}_p$ is zero for a linear motion. If these values are substituted in the above equations, it can be seen that $\beta_2 = \arctan(\infty) = \pi/2$ and

$$\delta^e = \arccos\left(\frac{0}{d_{22}\bar{u}_p}\right) - \pi/2 = 0 \tag{7.23}$$

Note that when the control $p$ moves on a line, the orientation of the motion of point $p$ is equivalent to the slope of the linear path. A zero $\delta$ at the equilibrium as indicated by Eq. (7.23) means that the vessel's longitudinal axis becomes collinear with the linear path.

### *7.4.3 Permissible Practical Motions*

The zero-dynamics stability analysis was done only for control point trajectories with constant speed. Furthermore, linearization was necessary for arriving at a conclusion about the zero-dynamic stability. The result of this stability analysis is only valid in the vicinity of the equilibrium point, around which the linearization was done, and for speeds that are vary very slowly such that they can be considered constant. These limitations must be considered when defining desired trajectories for the control point. It is recommended to define trajectories consisting of lines and circular arcs, for which the equilibrium point of the zero dynamics is known. Also, abrupt changes in desired velocities must be avoided. Hardware experiments are the only means for practical determination of the extents of the zero-dynamic stability.

## 7.5 Trajectory-Tracking Controller Design

The controller design problem can be defined as finding control laws for the driving force and torque appearing in the 3-DOF dynamic equations of the surface vessel such that the control point of the vessels follows a desired trajectory defined in Eq. (7.7). To find such control laws, the dynamic relation between the control inputs

$F$ and $T$ and the control outputs $x_{p1}$ and $x_{p2}$ must be derived. The next section presents the derivations.

### 7.5.1 The Input–Output Relations

The first-order derivative of the control outputs can be written in terms of the speed components of the surface vessel expressed in the vessel's local frame.

$$\begin{aligned}\dot{x}_{p1} &= u\cos\psi - (v + rd)\sin\psi,\\ \dot{x}_{p2} &= u\sin\psi + (v + rd)\cos\psi.\end{aligned} \tag{7.24}$$

The second-order derivative of the controller outputs are computed as

$$\begin{aligned}\ddot{x}_{p1} &= \dot{u}\cos\psi - u\dot{\psi}\sin\psi - (\dot{v} + \dot{r}d)\sin\psi - (v + rd)\dot{\psi}\cos\psi,\\ \ddot{x}_{p2} &= \dot{u}\sin\psi + u\dot{\psi}\cos\psi + (\dot{v} + \dot{r}d)\cos\psi - (v + rd)\dot{\psi}\sin\psi.\end{aligned} \tag{7.25}$$

Substituting for $\dot{u}$, $\dot{v}$, and $\dot{\psi}$ from Eq. (7.3) into the second-order output relations and rearranging in terms of control inputs $F$ and $T$ results in

$$\begin{aligned}\ddot{x}_{p1} &= \left(\frac{\cos\psi}{m_{11}}\right)F - \left(\frac{d\sin\psi}{I_{zz}}\right)T + f_{x1},\\ \ddot{x}_{p2} &= \left(\frac{\sin\psi}{m_{11}}\right)F + \left(\frac{d\cos\psi}{I_{zz}}\right)T + f_{x2},\end{aligned} \tag{7.26}$$

where

$$\begin{aligned}f_{x1} &= (m_{22}vr - d_{11}u)\frac{\cos\psi}{m_{11}} - ur\sin\psi + (m_{11}ur + d_{22}v)\frac{\sin\psi}{m_{22}}\\ &\quad -\frac{d\sin\psi}{I_{zz}}((m_{11} - m_{22})uv - d_{66}r) - (v + rd)r\cos\psi,\\ f_{x2} &= (m_{22}vr - d_{11}u)\frac{\sin\psi}{m_{11}} + ur\sin\psi - (m_{11}ur + d_{22}v)\frac{\cos\psi}{m_{22}}\\ &\quad +\frac{d\cos\psi}{I_{zz}}((m_{11} - m_{22})uv - d_{66}r) - (v + rd)r\sin\psi.\end{aligned} \tag{7.27}$$

Equations (7.26) can be written in the following general matrix form:

$$\begin{bmatrix}\ddot{x}_{p1}\\ \ddot{x}_{p2}\end{bmatrix} = \begin{bmatrix}f_{x1}\\ f_{x2}\end{bmatrix} + \begin{bmatrix}\frac{\cos\psi}{m_{11}} & -\frac{d\sin\psi}{I_{zz}}\\ \frac{\sin\psi}{m_{11}} & \frac{d\cos\psi}{I_{zz}}\end{bmatrix}\begin{bmatrix}F\\ T\end{bmatrix}, \tag{7.28}$$

or

$$\ddot{\mathbf{z}} = \mathbf{f} + \mathbf{b}\mathbf{u}. \tag{7.29}$$

The control input $\mathbf{u}$ must be determined such that the output $\mathbf{z}$ converges to the desired output $\mathbf{z}^d$ as time goes to infinity. This control law can be determined using different methods. In the following, the feedback linearization and the robust sliding mode control method are used.

### 7.5.2 Feedback Linearization

In this method, a new control input is defined such that the form of the input–output equation is linear. The new control input

$$\mathbf{v} = \mathbf{f} + \mathbf{b}\mathbf{u}. \tag{7.30}$$

simplifies the input–output equations in the following linear form:

$$\ddot{\mathbf{z}} = \mathbf{v}. \tag{7.31}$$

Let us assume a second-order asymptotically stable desired error dynamic behavior as

$$\ddot{\tilde{\mathbf{z}}} + 2\Lambda\dot{\tilde{\mathbf{z}}} + \Lambda^2\tilde{\mathbf{z}}, \tag{7.32}$$

where

$$\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{z}^d, \tag{7.33}$$

and

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}. \tag{7.34}$$

The new control input $\mathbf{v}$ must be determined such that the error behavior (7.32) is achieved. Combining Eqs. (7.31) and (7.32) results in the new control as

$$\mathbf{v} = \ddot{\mathbf{z}}^d - 2\Lambda\dot{\tilde{\mathbf{z}}} - \Lambda^2\tilde{\mathbf{z}}. \tag{7.35}$$

The original control input $\mathbf{u}$ can be obtained based on the control law (7.35) found for the new control input and the definition of the new control input (7.30).

$$\mathbf{u} = \mathbf{b}^{-1}(\ddot{\mathbf{z}}^d - 2\Lambda\dot{\tilde{\mathbf{z}}} - \Lambda^2\tilde{\mathbf{z}} - \mathbf{f}). \tag{7.36}$$

*Example 7.1.* Consider a surface vessel whose dynamics can be reduced to a 3-DOF model. The parameters of the 3-DOF dynamic model of the surface vessel are listed

**Table 7.1** Vessel's dynamic parameters

| | | |
|---|---|---|
| $m_{11} = 200$ kg | $m_{22} = 250$ kg | $I_{zz} = 700$ kg.m$^2$ |
| $d_{11} = 70$ kg/s | $d_{22} = 100$ kg/s | $d_{66} = 50$ kg.m$^2$/s |

in Table 7.1. Using the control law (7.36), simulate the autonomous motion of the surface vessel on a desired circular path for the control point with a radius of 10 m centered at the origin of the inertial coordinate system. Assume a constant linear velocity of 1 m/s. Plot the orientation stability parameter $\delta$ introduced in Section 7.4 and comment on the stability of the zero dynamics of the surface vessel. The vessel is moving with a longitudinal velocity of 1 m/s at the beginning of the motion, its longitudinal axis is parallel to the positive $x_2$ axis, and its control point is initially at $(11, -3)$ m. Assume $d = 1$ m.

*Solution.* The desired trajectory of the control point of the surface vessels can be parameterized in terms of time as follows.

$$\begin{aligned}
x_{p1}^d(t) &= x_c + R\cos(Vt/R),\\
x_{p2}^d(t) &= y_c + R\sin(Vt/R),\\
\dot{x}_{p1}^d(t) &= -V\sin(Vt/R),\\
\dot{x}_{p2}^d(t) &= V\cos(Vt/R),\\
\ddot{x}_{p1}^d(t) &= -V^2/R\cos(Vt/R),\\
\ddot{x}_{p2}^d(t) &= -V^2/R\sin(Vt/R),
\end{aligned} \tag{7.37}$$

where

$$R = 10, \quad V = 1, \quad x_c = 0, \quad y_c = 0. \tag{7.38}$$

The initial conditions for the surface vessel are

$$\begin{aligned}
x_1(0) &= 11 \quad \text{m}, & u(0) &= 1 \quad \text{m/s},\\
x_2(0) &= -4 \quad \text{m}, & v(0) &= 0 \quad \text{m/s},\\
\psi(0) &= \pi/2 \quad \text{rad}, & r(0) &= 0 \quad \text{rad/s}.
\end{aligned} \tag{7.39}$$

The first-oder form of the dynamic equations of motion used for the simulations are

$$\begin{aligned}
\dot{x}_1 &= u\cos\psi - v\sin\psi,\\
\dot{x}_2 &= u\sin\psi + v\cos\psi,\\
\dot{\psi} &= r,\\
\dot{u} &= (F + m_{22}vr - d_{11}u)/m_{11},\\
\dot{v} &= (-m_{11}ur - d_{22}v)/m_{22},\\
\dot{r} &= (T - (m_{22} - m_{11})uv - d_{66}r)/I_{zz}.
\end{aligned} \tag{7.40}$$

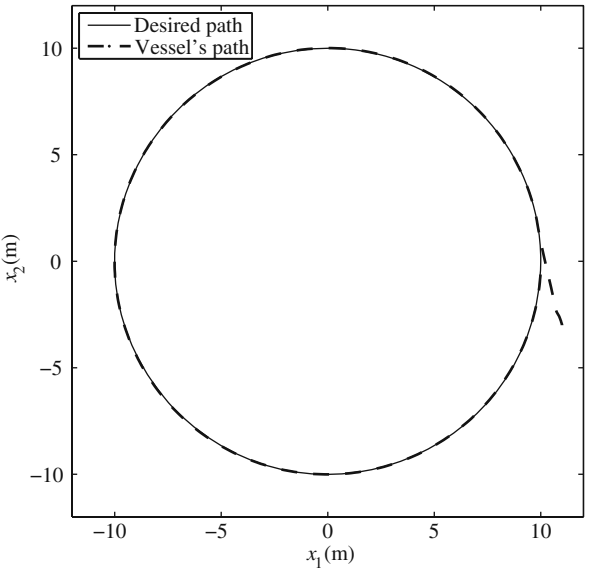


**Fig. 7.3** The circular path of the control point

The control law (7.36) determines the control inputs $\mathbf{u} = [F, T]^T$. The controller gains $\lambda_1$ and $\lambda_2$ are selected to be 3. These control inputs are directly applied to the first-order equations of motion of the surface vessel (7.40) for simulations.

Figure 7.3 shows the path of the vessel's control point drawn on top of the desired path of the control point. It can be seen that the controller successfully brings the surface vessel from its initial location onto the desired path. Since the vessel's control point desired speed is 1 m/s, it takes 62.8 s for the vessel to complete the desired circle with a 62.8 m circumference. The simulation time has been set to 62.8 s. Although Fig. 7.3 shows how the controller performs, the velocity information cannot be directly seen from this figure.

The linear and angular velocity components of the surface vessel are shown in Fig. 7.4. The longitudinal (surge) speed of the center of mass of the vessel is very close to 1.00 m/s. The lateral (sway) speed of the center of gravity of the vessel has a significant nonzero magnitude of 0.1992 m/s. This is an expected result, since there is no kinematic lateral motion constraint. This limit speed is solely determined by the lateral hydrodynamic damping forces. These magnitudes, at first, may seem to be contradicting the desired speed of 1 m/s for the control point. However, a closer analysis shows that the magnitude of the speed of the control point is in fact very close to the desired speed of 1 m/s. The speed of the control point in terms of the speed component of the center of gravity can be written as

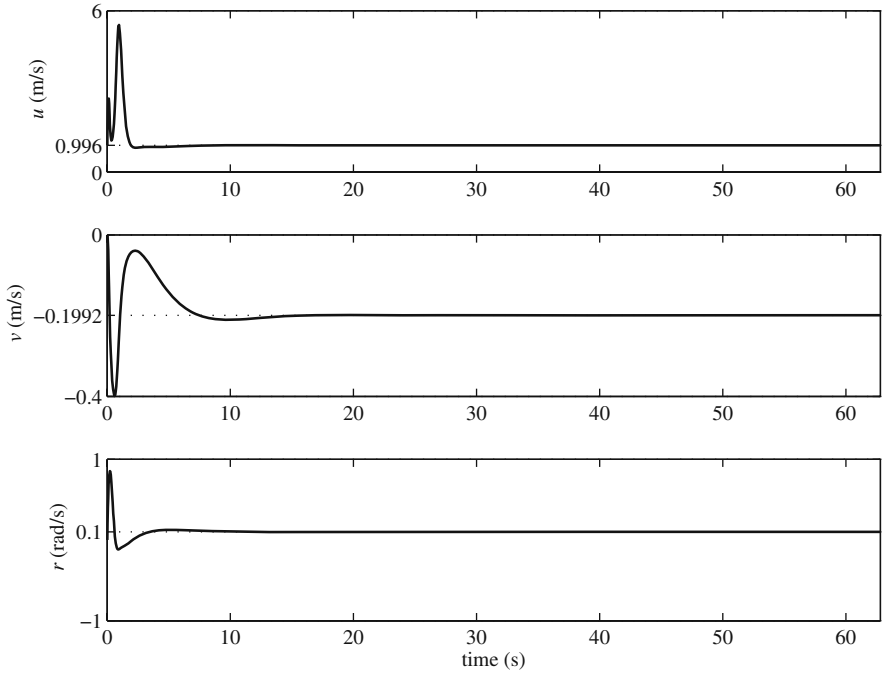


**Fig. 7.4** The linear and angular velocity components of the surface vessel in its local frame

$$\begin{aligned} V &= \sqrt{u_p^2 + v_p^2}, \\ &= \sqrt{u^2 + (v + rd)^2}, \\ &= \sqrt{0.9960^2 + (-0.1992 + (0.1)(1))^2}, \\ &= 1.00 \text{ m/s}, \end{aligned} \tag{7.41}$$

in which $r = 0.1$ rad/s has been used. This value for the yaw rate is also expected because the vessel's direction changes $2\pi$ radians in almost 62.8 s. The third plot of Fig. 7.4 confirms this value.

The control force and torque are shown in Fig. 7.5. The steady-state value of the driving force is 70 N to mostly compensate for the longitudinal hydrodynamic damping of $d_{11} = 70$ kg/s acting on a speed of 1 m/s. The magnitude of the steady-state torque is very close to zero due to small hydrodynamic effects on the yaw motion. This completes the solution to this example.

Figure 7.6 shows the orientation difference between the velocity vector of the control point and the heading of the surface vessel. This orientation difference $\delta$, defined in Section 7.4, is the representative of the unactuated DOF of the surface vessel. In Section 7.4, it was concluded that for a circular motion, $\delta$ converges to a constant value, i.e., the orientation of the vessel will have a constant offset with the orientation of the control point's velocity. Figure 7.6 confirms that result.

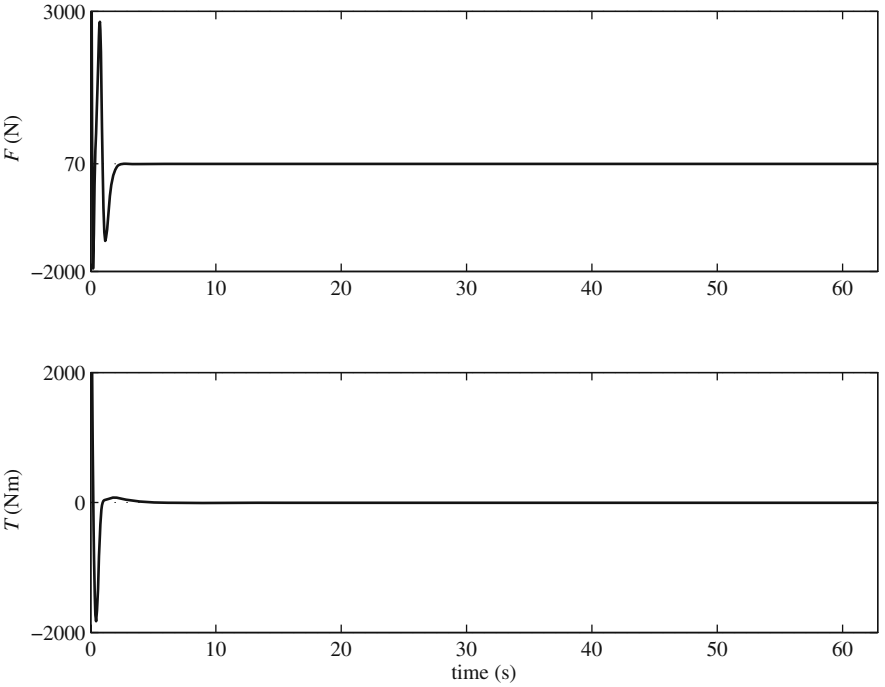


**Fig. 7.5** The driving force and torque

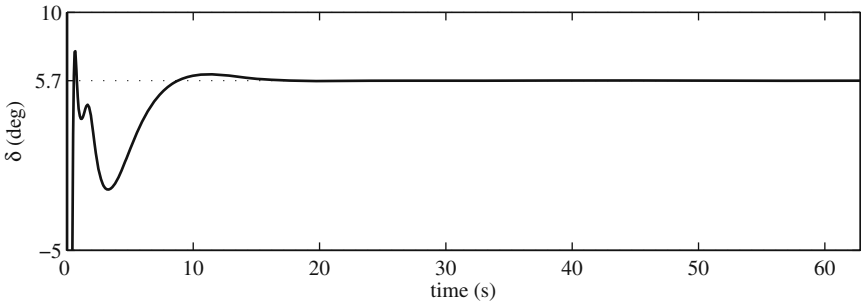


**Fig. 7.6** The orientation difference between the velocity vector of the control point and the heading of the surface vessel

The performance of the control law based on feedback linearization is acceptable if there are very small unknown external disturbaces and dynamic model uncertainties. However, for large disturbances and uncertainties, the performance deterioration may not be acceptable. Since, large disturbances and uncertainties can be easily present for a surface vessel, there is a need for more robust controllers. Investigating this fact remains as an exercise for the interested reader. In the next section, a robust controller is designed for the surface vessel based on the robust sliding mode control.

### 7.5.3 Robust Control Using the Sliding Mode Method

The controller derived in the previous section may not have an acceptable performance if the external disturbances and model uncertainties are too high. In these situtations, a robust controller is needed. In this section, a robust controller is designed using the sliding mode control method.

An asymptotically stable surface is defined for the error in each of the controller's output. The matrix form of this surface is written as

$$\dot{\tilde{\mathbf{z}}} + \Lambda \tilde{\mathbf{z}} = \mathbf{0}, \tag{7.42}$$

where $\Lambda$ is a positive-definite $2\times 2$ matrix. Since $\Lambda$ is a positive-definite matrix, if the errors' conditions satisfy the surface equation (7.42) at all times, they asymptotically approach the zero equilibrium point of $\tilde{\mathbf{z}} = 0$. However, this is in no way guaranteed. Therefore, the actual error behavior is more precisely defined by

$$\dot{\tilde{\mathbf{z}}} + \Lambda \tilde{\mathbf{z}} = \mathbf{s}, \tag{7.43}$$

where $\mathbf{s}$ is a parameter reflecting the offset of the error trajectory with the desired error trajectory of Eq. (7.42). A complete sliding mode control law must guarantee that, first, if the offset $\mathbf{s}$ is zero, the desired error trajectory becomes equal to Eq. (7.42), and second, if the offset $\mathbf{s}$ is not zero, it will approach zero and stay zero. The first part of the sliding mode controller is called the equivalent control. The second part is a nonlinearity term. These two parts are derived in the following.

#### 7.5.3.1 Equivalent Control

Consider the input–output relation (7.29). This relation is written with the nominal parameters of the dynamic model as

$$\ddot{\mathbf{z}} = \hat{\mathbf{f}} + \hat{\mathbf{b}}\mathbf{u}. \tag{7.44}$$

Equation (7.43) can be written in a more consize form as

$$\mathbf{s} = \dot{\mathbf{z}} - \mathbf{s}_r, \tag{7.45}$$

where

$$\mathbf{s}_r = \dot{\mathbf{z}}^d - \Lambda \tilde{\mathbf{z}}. \tag{7.46}$$

The equivalent control $\hat{\mathbf{u}}$ should guarantee that if $\mathbf{s}$ is zero, the error behavior is given by Eq. (7.42) or equivalently by Eq. (7.45). Therefore, one should combine the input–output relation (7.44) with Eq. (7.45) to find $\hat{\mathbf{u}}$. To do this, the second derivative of $\mathbf{z}$ must appear in Eq. (7.45). For $\mathbf{s}$ equal to zero, Eq. (7.45) is differentiated,

$$\ddot{\mathbf{z}} - \dot{\mathbf{s}}_r = \mathbf{0}. \tag{7.47}$$

Substituting for $\ddot{\mathbf{z}}$ from the input–output relation (7.44) and solving the result for $\mathbf{u}$ results in the equivalent control denoted by $\hat{\mathbf{u}}$,

$$\hat{\mathbf{u}} = \hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r). \tag{7.48}$$

#### 7.5.3.2 Robust Control Law

To complement the first part of the sliding mode controller, a second part is needed to guarantee that the surface offset parameter $\mathbf{s}$ approaches zero regardless of the error initial condition and uncertainty in the model parameters. This is done by adding a discontinuous term to the equivalent control,

$$\mathbf{u} = \hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})), \tag{7.49}$$

where $\mathbf{K}$ is a positive-definite diagonal matrix and sgn($\mathbf{s}$) returns a vector with the sign of the components of $\mathbf{s}$. Although the control law (7.49) seems complete, the discontinuity gain $\mathbf{K}$ must still be determined such that the surface offset parameter $\mathbf{s}$ converges to zero despite uncertainties in the dynamic model parameters.

A Lyapunov function is defined as

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{s}. \tag{7.50}$$

This Lyapunov function is admissible because it satisfies all the required properties of such a function. It is positive for all values of $\mathbf{s}$ and is only zero when $\mathbf{s}$ is identically zero. With these properties, if one can show that the time derivative of the Lyapunov function is always negative and is only zero when $\mathbf{s}$ is identically zero, then, one can conclude that $\mathbf{s}$ converges to zero from any initial condition and remains at zero.

The first derivative of the Lyapunov function (7.50) is

$$\dot{V} = \mathbf{s}^T\dot{\mathbf{s}}. \tag{7.51}$$

Substituting for $\dot{\mathbf{s}}$ from Eq. (7.45) results in

$$\dot{V} = \mathbf{s}^T(\ddot{\mathbf{z}} - \dot{\mathbf{s}}_r). \tag{7.52}$$

Substituting for $\ddot{\mathbf{z}}$ from the input–output relation yields

$$\dot{V} = \mathbf{s}^T(\mathbf{f} + \mathbf{bu} - \dot{\mathbf{s}}_r). \tag{7.53}$$

Substituting for $\mathbf{u}$ from Eq. (7.49) gives

$$\dot{V} = \mathbf{s}^T(\mathbf{f} + \mathbf{b}\hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})) - \dot{\mathbf{s}}_r). \tag{7.54}$$

For simplicity, it is assumed that there are no uncertainty in parameters that appear in $\mathbf{b}$. Therefore, $\mathbf{b} = \hat{\mathbf{b}}$. Equation (7.54) reduces to

$$\dot{V} = \mathbf{s}^T(\tilde{\mathbf{f}} - \mathbf{K}\text{sgn}(\mathbf{s})), \tag{7.55}$$

where

$$\tilde{\mathbf{f}} = \mathbf{f} - \hat{\mathbf{f}}. \tag{7.56}$$

Since the components of $\mathbf{K}$ are to be designed, Eq. (7.55) is written in component notation.

$$\begin{aligned}
\dot{V} &= \sum_{i=1}^{2} s_i(\tilde{f}_i - K_i\ \text{sgn}(s_i)), \\
&= \sum_{i=1}^{2} s_i\tilde{f}_i - K_i|s_i|, \\
&\leq \sum_{i=1}^{2} |s_i||\tilde{f}_i| - K_i|s_i|, \\
&\leq -\sum_{i=1}^{2} |s_i|(K_i - |\tilde{f}_i|).
\end{aligned} \tag{7.57}$$

It can be seen from Eq. (7.57) that if one selects

$$K_i \geq |\tilde{f}_i| + \eta_i, \tag{7.58}$$

where $\eta_i$'s ($i = 1, 2$) are positive numbers, then, the rate of the Lyapunov function becomes

$$\dot{V} \leq -\sum_{i=1}^{2} \eta_i|s_i|. \tag{7.59}$$

This means that by choosing $K_i$'s that satisfy Eq. (7.58), the rate of the Lyapunov function is always negative, implying that $s_i$'s decrease monotonically from any initial condition to zero, at which point the rate of Lyapunov function becomes zero, which means that $s_i$'s stay at zero.

*Example 7.2.* Consider the surface vessel and its dynamic model discussed in Example 7.1. Assume that the maximum uncertainty in parameters $m_{22}$, $d_{11}$, $d_{22}$, and $d_{66}$ is 25% and there is no uncertainty in $m_{11}$ and $I_{zz}$. Apply the control law (7.49) with

$K_i$'s determined from Eq. (7.58) to the dynamic model. Investigate the performance of the controller for the cases where there are −20%, 0%, and 20% uncertainty in the uncertain parameters. The vessel's control point is at the origin of the inertial frame at time zero. Assume a desired trajectory for the control point defined by

$$x_{p1}^d(t) = t \quad \text{for all } t, \tag{7.60}$$

$$x_{p2}^d(t) = \begin{cases} 0 & \text{if } 0 < t \leq 5 \\ (0.0008t^4 - 0.0099t^3 + 0.0588t^2 & \\ -0.1715t + 0.1969) & \text{if } 5 < t \leq 35 \\ 5 & \text{if } 35 < t \leq 40 \\ (0.0003t^5 - 0.0254t^4 + 1.3353t^3 - 41.4047t^2 & \\ +702.6246t - 5033.6465) & \text{if } 40 < t \leq 70 \\ 0 & \text{if } 70 < t \end{cases}. \tag{7.61}$$

These polynomials have been planned such that the position, velocity, acceleration, jerk, and curvature at the transition points between the segments are continuous. This is important for a practical trajectory because the controller resulting forces and torques will become continuous and are more likely to be applicable.

*Solution.* The first-order dynamic equations presented in Example 7.1 are also used here for simulating the motion of the surface vessel under control. The control law (7.49) with $K_i$'s determined from Eq. (7.58) is used to obtain the control inputs. To cover a maximum uncertainty of 25%, $\tilde{\mathbf{f}} = \mathbf{f} - \hat{\mathbf{f}}$, whose components appear in Eq. (7.58), is calculated as follows. $\hat{\mathbf{f}}$ is computed using the nominal dynamic parameters listed in Table 7.1. $\mathbf{f}$, on the other hand, is calculated using the uncertain parameters as

$$m_{22} = 1.25\hat{m}_{22},\ d_{11} = 1.25\hat{d}_{11},\ d_{22} = 1.25\hat{d}_{22},\ d_{66} = 1.25\hat{d}_{66}, \tag{7.62}$$

where the nominal parameters with hat are from Table 7.1.

In order to investigate the performace of the designed controller under parameter uncertainty, the uncertain parameters $m_{22}$, $d_{11}$, $d_{22}$, and $d_{66}$ used in this dynamic model are varied. For the three simulations, the uncertain parameters are multiplied by 0.8 (−20%), 1.0 (0%), and 1.2 (20%), respectively. Note that in this way, the controller is not aware of the actual value of the uncertain parameters, but only the bound of these parameters.

The path of the vessel with different degrees of uncertainty are shown in Fig. 7.7. As can be seen in this figure, the trajectory-tracking performance of the vessel does not deteriorate due to parameter uncertainty, as long as the uncertainty is within the assumed bounds.

The linear and angular velocity components of the center of gravity of the surface vessel are shown in Fig. 7.8. Note that these components are different than that of the control point. For the underactuated surface vessel, the controller is actually

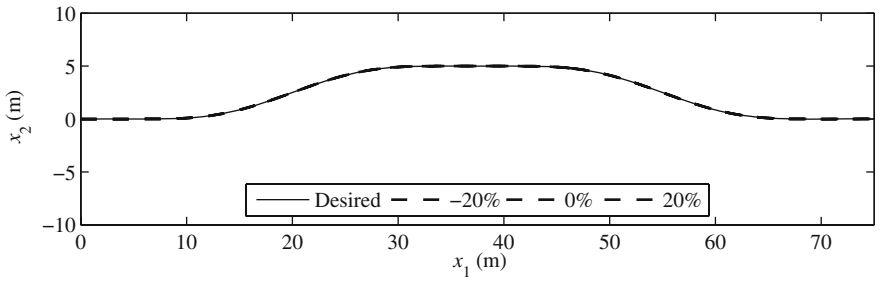


**Fig. 7.7** The polynomial path of the control point

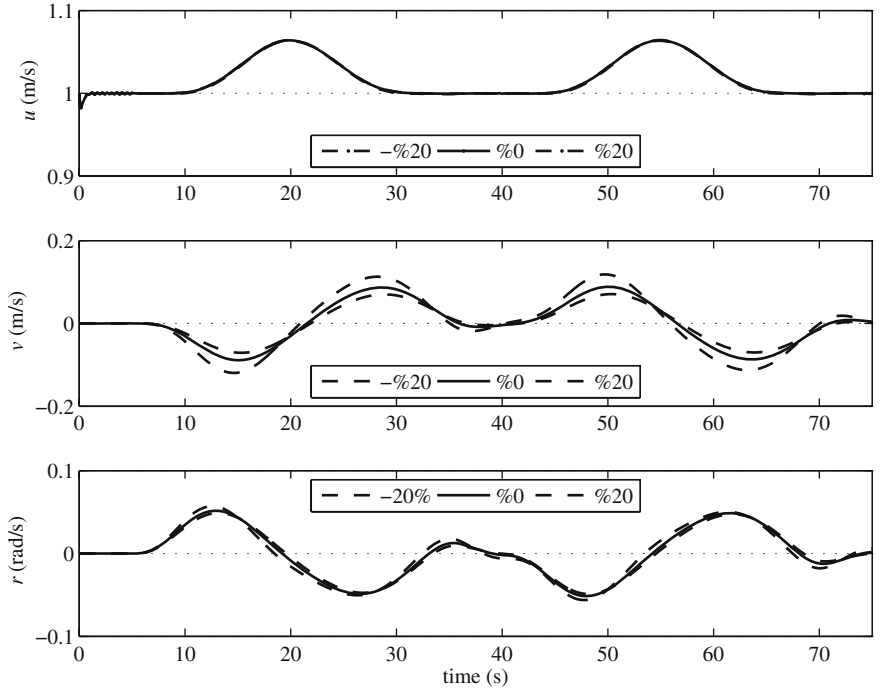


**Fig. 7.8** The linear and angular velocity components of the surface vessel in its local frame

controlling the velocity and position of the control point. The controller is not capable of controlling the unactuated mode of the dynamics of the vessel. The behavior of this DOF is determined by the zero dynamics of the vehicle. When the dynamic parameters change, the behavior of the unactuated DOF also changes.

This fact is somehow reflected in Fig. 7.8. The unactuated DOF for this surface vessel is the lateral direction of motion. Since the control point has a controlled motion, the unactuated lateral motion of the center of gravity shows itself as an oscillation about the control point. The longitudinal speed of the center of gravity is

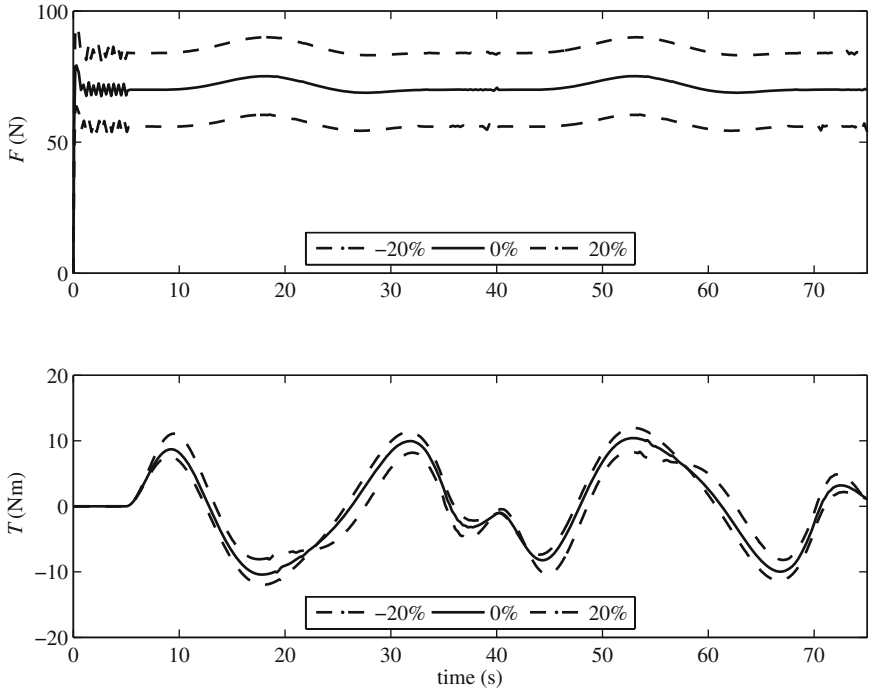


**Fig. 7.9** The driving force and torque

affected less by this oscillation than the longitudinal speed and the yaw rate, as seen in Fig. 7.8.

The control force and torque are shown in Fig. 7.9. The control force, when there is no uncertainty (0%), is about 70 N to drive the vessel with a constant speed of 1 m/s on the straight portions of the trajectory. The curved portions of the trajectory have a slightly higher desired velocity due to motion in the $x_2$ direction, hence, the driving force increases for those portions. For the uncertain cases, the driving force has been adjusted by the controller to address the lower and higher longitudinal dampings. As can be seen from Fig. 7.9, the steering torque is nonzero to steer the vessel on the trajectory. The magnitude of the torque slightly varies between the different cases of uncertainty to compensate for different dampings and hydrodynamic effects.

Figure 7.10 shows the orientation difference between the velocity vector of the control point and the heading of the surface vessel. This orientation difference $\delta$, defined in Section 7.4, is the representative of the unactuated DOF of the surface vessel. In Section 7.4, it was shown that this variable has a second-order damped response and is excited by the rate of the change of the desired trajectory's slope. It was also shown in Section 7.4 that, for a straight line trajectory, the equilibrium point of this variable is zero. Figure 7.10 confirms these findings. For the first portion of the trajectory, which is a straight line, $\delta$ remains zero. For the other portions where

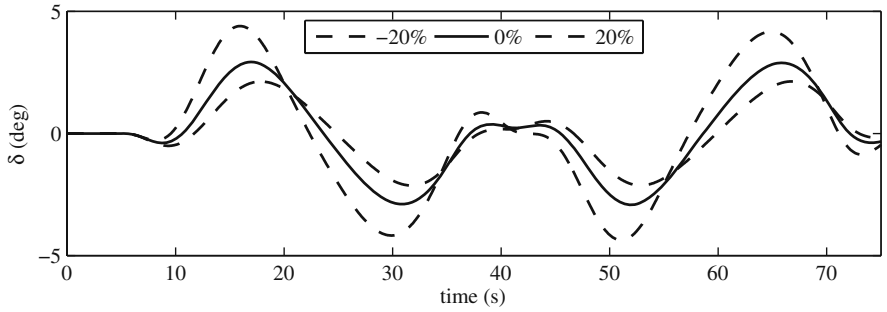


**Fig. 7.10** The orientation difference between the velocity vector of the control point and the heading of the surface vessel

the slope of the desired trajectory changes, $\delta$ has a damped response. It comes back to its equilibrium point of zero (after some time lag) wherever the trajectory is a straight line. The absolute of $\delta$ remains less than 5° for this maneover.

The phase planes for the controller output errors are shown in Fig. 7.11. As is seen in this figure, both the controller outputs are attracted to an equilibrium point. The equilibrium point of the error in $x_{p1}$ control point's position component has a small offset with zero in the order of millimeters, whereas the equilibrium point for the error in $x_{p2}$ component has no offset with zero. This may be due to the fact that

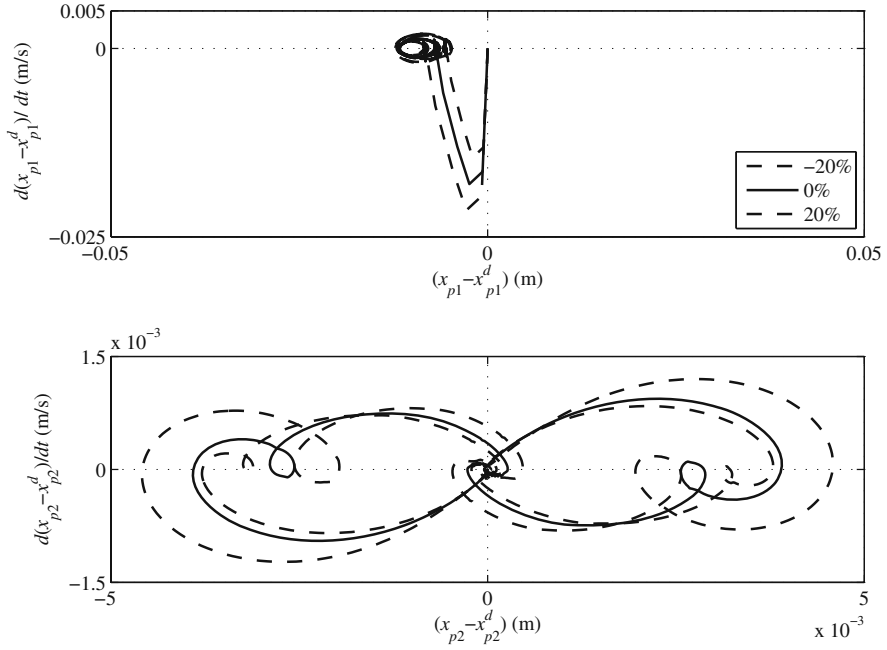


**Fig. 7.11** The phase plane for the controller output errors

the $x_{p1}$ component is tracking a dynamic function, whereas the $x_{p2}$ component is mostly constant throughout the motion.

## 7.6 Formation Control for Surface Vessels

In the previous section, the trajectory control for a single ASV was investigated. While a single surface vessel may be useful for simple tasks, the use of multiple cooperative vessels may be crucial or may accelerate the accomplishment of more complicated tasks. For a successful cooperative work, the autonomous vessels must intelligently maintain user-specified distances to each other, which is known as formation control.

During the last several years, the formation control problem for different types of vehicles is investigated by researchers, The studied vehicles include ground vehicles (indoor mobile robots and large outdoor vehicles), aerial and space vehicles, and underwater and surface vessels. Three major approaches that have been introduced are virtual structure [54], behavior-based [7, 26], or leader-follower approaches [74, 23]. The virtual structure approach a generic method in which the holonomic kinematics of the motion for the group is planned. For the actual navigation, the vehicles rely on existing conventional controllers. In contrary to the virtual structure method, in behavior-based and leader-follower approaches, the controller design is an integral part of the formation control design. Therefore, for these approaches, the accuracy of the kinematic model and consideration of the nonholonomic constraints of the vehicles are important.

The formation control problem for different types of marine vehicles has been investigated by many researches. The the leader-follower approach combined with the sliding mode control method [29] and the virtual structure method [42] have been used for formation control of marine craft. A method for formation control of marine surface craft inspired by Lagrangian mechanics has also been presented [41].

In this section, the problem of control and coordination for many unmanned surface vessels moving in general user-defined formations is investigated. The planar motion of surface vessels for control development with three DOFs of surge, sway, and yaw is considered. With this assumption, two modular leader-follower geometrical formation schemes [23] can be used. In these schemes, local parameters are utilized to define the internal geometry of the formation.

### *7.6.1 Geometrical Leader-Follower Formation Schemes*

Let us assume that a trajectory planning and obstacle avoidance algorithm characterizes the gross motion of the group of vessels. A real or hypothetical leader vessel adapts the planned gross trajectory. If the leader vessel is a real vessel, it can use the trajectory-tracking controller designed in the previous section for tracking the desired gross trajectory. Other vessels of the group follow either the hypothetical

group leader or their neighboring vessels. The follower vessels must have controllers that control the internal geometry of the formation. The internal formation structure is defined by using two geometrical formation schemes. These schemes provide the building blocks for defining a general formation structure for any type of vehicles with planar motion [23].

To define a 2D formation structure uniquely, a mesh with triangular cells must be defined in which the vertices of the triangular cells represent the control points of the surface vessels. The legs of the triangular cells represent the distances that the vessels must keep while moving to maintain the formation. These triangular cells can be formed with two types of formation building blocks, also known as the formation schemes.

The first geometrical scheme is called the $l - \alpha$ scheme. As the name implies, the desired relative distance and view angle of a vessel with respect to a neighboring vessel are defined as the geometrical formation parameters. With this formation control scheme, one can define the formation structure of vessels marching at an edge of the 2D formation structure or in a single file, if the formation structure is one dimensional.

Note that the $l - \alpha$ scheme alone can only define one side of a triangular cell in a 2D formation structure. A second geometrical scheme is needed to define the two other legs of the triangular cell. The second geometrical scheme is called the $l - l$ scheme, in which the distances of a vessel to two neighboring vessels or to one vessel and an obstacle are controlled.

The $l - \alpha$ scheme is used for the vessels at the edge of the formation structure geometry and the $l - l$ scheme is used for other vessels such that a formation mesh with triangular cells connecting all the vessels is generated.

Controllers must be designed to ensure that the vehicles autonomously reach and keep the desired formation parameters. These controllers determine the required physical control signals to obtain the desired formation parameters as controller outputs. In the following, the details of formation controller design for surface vessels based on the mentioned geometrical schemes are discussed.

### *7.6.2 Design of the l – α Controller*

Two neighboring vessels in the formation are shown in Fig. 7.12. The distance of the center of mass of vessel 1 and the control point, $p$, of vessel 2 is $l_{12}$. The control point is on the longitudinal axis of vessel 2 and has a distance $d$ from the center of mass of vessel 2. In order to maintain the formation, vessel 2 must keep a desired distance of $l_{12}^d$ and view angle $\alpha_{12}^d$ with vessel 1. A control law for the driving inputs of vessel 2, $F_2$ and $T_2$, must be determined such that vessel 2 maintains the desired distance and view angle.

To design such a control law, explicit relations between the control inputs ($F_2$ and $T_2$) and the control outputs ($l_{12}$ and $\alpha_{12}$) are required. These relations are known

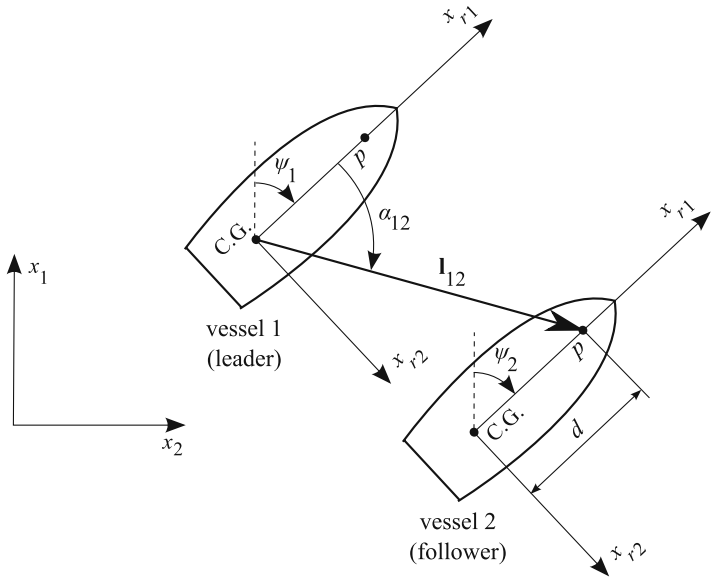


**Fig. 7.12** The $l - \alpha$ scheme geometry

as the input–output relations. In the following, first, these input–output relations are found; then, the control laws are derived.

The explicit relations between the input and the output are found with the following procedure. First, the dynamic equations (7.4) for vessel 2 are written. These equations relate the control inputs $F_2$ and $T_2$ to the acceleration components of vessel 2, $(\ddot{x}_2, \ddot{y}_2, \ddot{\psi}_2)$. Second, some kinematic equations are used to realted these acceleration components to the derivatives of $l_{12}$ and $\alpha_{12}$. Finally, the input–output relations are obtained by eliminating the acceleration components between the dynamic and kinematic equations.

Writing the dynamic equations (7.4) for vessel 2 is simply done by adding a subscript 2 to the variables. However, writing the kinematic equations require the acceleration analysis of the used $l - \alpha$ scheme.

### 7.6.2.1 Kinematic Analysis

A moving coordinate system is assumed with an origin at the center of mass of vessel 1. This coordinate system is rotating with the vector $\mathbf{l}_{12}$, which is the line of sight vessel 2 from vessel 1's center of gravity (Fig. 7.12). Also, two coincident points are assumed. The first one is $p_1$ and is attached to this moving coordinate system. The second one is $p_2$, which is attached to vessel 2. Both points are coincident with the instantaneous location of the control point $p$. One can see that, to the eye of an observer attached to the defined moving coordinate system standing at $p_1$, point $p_2$ moves along the vector $\mathbf{l}_{12}$. Mathematically, this can be expressed by the following kinematic equation for the inertial acceleration of $p_2$:

$$\mathbf{a}_{p2} = \mathbf{a}_{p1} + \mathbf{a}_{p2/1}, \tag{7.63}$$
$$= (\mathbf{a}_1 + \ddot{\alpha}_0 \hat{\mathbf{k}} \times \mathbf{l}_{12} - \dot{\alpha}_0^2 \mathbf{l}_{12}) + (2\dot{\alpha}_0 \hat{\mathbf{k}} \times \dot{\mathbf{l}}_{12} + \ddot{\mathbf{l}}_{12}), \tag{7.64}$$

where $\mathbf{a}_1$ is the acceleration of the center of mass of vessel 1, and

$$\alpha_0 = \psi_1 + \alpha_{12}. \tag{7.65}$$

Now, the the acceleration of point $p_2$ on vessel 2 can also be written in terms the acceleration of the center of mass of the vessel 2, $\mathbf{a}_2$:

$$\mathbf{a}_{p2} = \mathbf{a}_2 + \ddot{\psi}_2 \hat{\mathbf{k}} \times \mathbf{d} - \dot{\psi}_2^2 \mathbf{d}. \tag{7.66}$$

The vectorial equations (7.64) and (7.66) are combined, expanded, and solved for the highest derivatives of the outputs, $\ddot{l}_{12}$ and $\ddot{\alpha}_{12}$, to result in

$$\ddot{l}_{12} = (\ddot{y}_2 - \ddot{y}_1)\sin\alpha_0 + (\ddot{x}_2 - \ddot{x}_1)\cos\alpha_0 + d\ddot{\psi}_2 \sin\gamma_1 - d\dot{\psi}_2^2 \cos\gamma_1 + l_{12}\dot{\alpha}_0^2, \tag{7.67}$$

$$\ddot{\alpha}_{12} = \frac{1}{l_{12}}[(\ddot{y}_2 - \ddot{y}_1)\cos\alpha_0 - (\ddot{x}_2 - \ddot{x}_1)\sin\alpha_0 + d\ddot{\psi}_2 \cos\gamma_1 + d\dot{\psi}_2^2 \sin\gamma_1 - 2\dot{l}_{12}\dot{\alpha}_0 - l_{12}\ddot{\psi}_1], \tag{7.68}$$

where

$$\gamma_1 = \psi_1 + \alpha_{12} - \psi_2. \tag{7.69}$$

#### 7.6.2.2 Input–Output Equations

Now, $\ddot{x}_2$, $\ddot{y}_2$, and $\ddot{\psi}_2$ in the kinematic equations (7.67) and (7.68) are replaced by terms from the dynamic equations (7.4) written for the vessel 2 to obtain the following input–output equations:

$$\ddot{l}_{12} = \left[ f_l + \frac{1}{m_{11}} F_2 \cos\gamma_1 + \frac{1}{I_{zz}} T_2 d \sin\gamma_1 \right], \tag{7.70}$$

$$\ddot{\alpha}_{12} = \frac{1}{l_{12}} \left[ f_\alpha - \frac{1}{m_{11}} F_2 \sin\gamma_1 + \frac{1}{I_{zz}} T_2 d \cos\gamma_1 \right], \tag{7.71}$$

where

$$f_l = \frac{1}{m_{11}}(f_x \cos\alpha_0 + f_y \sin\alpha_0) + \frac{1}{I_{zz}} f_\psi d \sin\gamma_1 - \ddot{x}_1 \cos\alpha_0 - \ddot{y}_1 \sin\alpha_0 - d\dot{\psi}_2^2 \cos\gamma_1 + l_{12}\dot{\alpha}_0^2,$$

$$f_\alpha = 7\frac{1}{m_{11}}(-f_x \sin\alpha_0 + f_y \cos\alpha_0) + \frac{1}{I_{zz}} f_\psi d \cos\gamma_1 + \ddot{x}_1 \sin\alpha_0 - \ddot{y}_1 \cos\alpha_0 + d\dot{\psi}_2^2 \sin\gamma_1 - 2\dot{l}_{12}\dot{\alpha}_0 - l_{12}\ddot{\psi}_1.$$

The input–output relations (7.70) and (7.71) can be written in the matrix form.

$$\ddot{\mathbf{z}} = \mathbf{f} + \mathbf{b}\mathbf{u}, \tag{7.72}$$

where

$$\mathbf{z} = \begin{bmatrix} l_{12} \\ \alpha_{12} \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} f_l \\ \frac{f_\alpha}{l_{12}} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \frac{\cos\gamma_1}{m_{11}} & \frac{d\sin\gamma_1}{I_{zz}} \\ -\frac{\sin\gamma_1}{m_{11}l_{12}} & \frac{d\cos\gamma_1}{I_{zz}l_{12}} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} F_2 \\ T_2 \end{bmatrix}. \tag{7.73}$$

#### 7.6.2.3 Control Laws

With the input–output equations at hand, nonlinear control laws can be proposed. Here, a nonlinear controller is designed based on the input–output equation (7.72). A stable second order nonlinear error dynamics is introduced ($k_i(\tilde{z}_i) > 0$):

$$\ddot{\tilde{z}}_i + 2\sqrt{k_i(\tilde{z}_i)}\dot{\tilde{z}}_i + k_i(\tilde{z}_i)\tilde{z}_i = 0, \qquad i = 1, 2, \tag{7.74}$$

where $\tilde{z}_1$ and $\tilde{z}_2$ are the output errors defined as

$$\tilde{z}_1 = l_{12} - l_{12}^d, \qquad \tilde{z}_2 = \alpha_{12} - \alpha_{12}^d. \tag{7.75}$$

For a constant $k_i$, the error dynamics (7.74) results in a critically damped linear second-order closed-loop input–output system, whose behavior is similar to that of a mass-spring-damper system. Such a system can show a smooth convergence rate when $k_i$ is relatively low, however, it exhibits large offsets when external disturbace (equivalent of a force on the mass) are present. The offset can be reduced if a high $k_i$ is selected at the cost of an extra fast transient response.

To find a compromise between a response with a reasonable rate and a low offset in presence of external disturbances, $k_i$ must be defined as a function of the error. The continuous function $k_i(\tilde{z}_i)$ must have a relatively small value when the error is high for a reasonable rate of convergence, and must have a relatively large value when the error is zero for an acceptable offset in the presence of external disturbances. The following function meets the above criteria:

$$k_i(\tilde{z}_i) = n\lambda_i + (1-n)\frac{\lambda_i^2}{\lambda_i + a_i\tilde{z}_i^2}, \tag{7.76}$$

where $\lambda_i > 0$ is the controller gain when the error is zero and $n << 1$ determines the reduction ratio in the controller gain when the error is infinite. The parameter $a_i > 0$ affects how fast the controller gain is increased as the error approaches zero. By using the following Lyapunov function,

$$V = \frac{1}{2}(\dot{\tilde{z}}_i^2 + n\lambda_i\tilde{z}_i^2 + \frac{\lambda_i^2(1-n)}{a}\ln(\frac{\lambda_i + a_i\tilde{z}_i^2}{\lambda_i})) \geq 0, \tag{7.77}$$

one can show that

$$\dot{V} = -2\sqrt{k_i(\tilde{z}_i)}\dot{\tilde{z}}_i^2 \leq 0, \tag{7.78}$$

which implies that the nonlinear error dynamics (7.74) is in fact stable with an equilibrium position $(\tilde{z}_i, \dot{\tilde{z}}_i) = (0, 0)$.

The desired error behavior (7.74) is written in the matrix form for convenience.

$$\ddot{\tilde{\mathbf{z}}} + 2\sqrt{\mathbf{K}}\,\dot{\tilde{\mathbf{z}}} + \mathbf{K}\,\tilde{\mathbf{z}} = \mathbf{0}, \tag{7.79}$$

where

$$\sqrt{\mathbf{K}} = \begin{bmatrix} \sqrt{k}_1 & 0 \\ 0 & \sqrt{k}_2 \end{bmatrix}, \quad \mathbf{K} = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}. \tag{7.80}$$

Equation (7.79) can be solved for $\ddot{\mathbf{z}}$. This yields

$$\ddot{\mathbf{z}} = \ddot{\mathbf{z}}^d - 2\sqrt{\mathbf{K}}\,\dot{\tilde{\mathbf{z}}} - \mathbf{K}\,\tilde{\mathbf{z}}. \tag{7.81}$$

Substituting for $\ddot{\mathbf{z}}$ from Eq. (7.81) into Eq. (7.72) and solving for the control input $\mathbf{u}$ results in

$$\mathbf{u} = \mathbf{b}^{-1}(\ddot{\mathbf{z}}^d - 2\sqrt{\mathbf{K}}\,\dot{\tilde{\mathbf{z}}} - \mathbf{K}\,\tilde{\mathbf{z}} - \mathbf{f}). \tag{7.82}$$

Note that the matrix $\mathbf{b}$ must be inverted for calculating the control law. Therefore, the determinant of this matrix must be nonzero at all times. This determinant is obtained as

$$\det(\mathbf{b}) = \frac{d}{m_{11} I_{zz} l_{12}}. \tag{7.83}$$

As can be seen, this determinant is nonzero as long as the parameter $d$, defining the location of the control point, is not zero. This emphasizes the importance of the distance between the control point of the vessel 2 and its center of mass, $d$. If the control point was chosen to be coincident with the center of mass of vessel 2 ($d = 0$), the formation scheme would have become uncontrollable. This distance also has an important effect on the quality of the system's zero dynamics response, which was discussed in Section 7.4.

*Example 7.3.* Consider a 6-DOF surface vessel 2 whose dynamic parameters are listed in Table 7.2. The vessel 2, initially located at (3, 0) m, is commanded to keep

**Table 7.2** Vessel's dynamic parameters

| | | | |
|---|---|---|---|
| $m = 300\,\text{kg}$ | $I_{xx} = 300\,\text{kg.m}^2$ | $I_{yy} = 700\,\text{kg.m}^2$ | $I_{zz} = 700\,\text{kg.m}^2$ |
| $m_{11} = 200\,\text{kg}$ | $m_{22} = 250\,\text{kg}$ | $m_{33} = 300\,\text{kg}$ | $m_{66} = 80\ \text{kg.m}^2$ |
| $d_{11} = 70\ \text{kg/s}$ | $d_{22} = 100\,\text{kg/s}$ | $d_{33} = 100\,\text{kg/s}$ | $d_{44} = 500\,\text{kg.m}^2\text{/s}$ |
| $d_{55} = 500\,\text{kg.m}^2\text{/s}$ | $d_{66} = 50\,\text{kg.m}^2\text{/s}$ | $\bar{MT}_p = 0.5\,\text{m}$ | $\bar{MT}_q = 0.5\,\text{m}$ |
| $\bar{GF} = 0.5\,\text{m}$ | $\rho = 1000\,\text{kg/m}^3$ | $g = 9.81\,\text{m/s}^2$ | $A_{wp} = 1\,\text{m}^2$ |

a specified distance, $l_{12}^d = 3.0\,\text{m}$, and view angle, $\alpha_{12}^d = \pi/2$ rad, with respect to vessel 1. Vessel 1 is moving on a circle (dashed curve) with a radius of 15 m with a constant longitudinal speed of 1.57 m/s.

Apply the $l - \alpha$ control law (7.82) to this 6-DOF vessels to more closely investigate the performance of the designed controller on a real 6-DOF vessel. Simulate the response of vessel 2 for two different scenarios to investigate the robustness of the controller to external disturbances. For both scenarios, assume that the steady-state longitudinal speed of the vessel is $u_{ss} \approx 1.5\,\text{m/s}$. In the first scenario, assume that no sea current is present. For the second scenario, assume that a constant global sea current is flowing along the negative $x_2$-axis that exerts an equivalent force of 50 N on the vessel.

*Solution.* A list of the controller parameters can be found in Table 7.3. The path of the leader and the follower vessels are shown in Fig. 7.13. The path of vessel 2, the follower, is shown by a solid curve. The initial position of the follower vessel is such that there is an initial error in the formation parameters. The follower reaches the desired formation successfully after some time. The small line segments show the

**Table 7.3** Controller parameters for Example 7.3

| $l - \alpha$ |
|---|
| $n = 0.05$ |
| $d = 2.0$ |
| $\lambda_1 = 5, \lambda_2 = 8$ |
| $a_1 = 1000, a_2 = 1000$ |
| $l_{12}^d = 2.0\,\text{m}, \alpha_{12}^d = \pi/2$ rad |

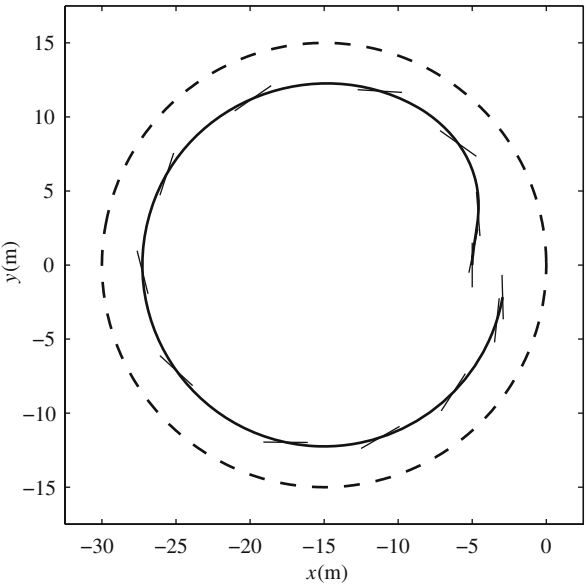


**Fig. 7.13** The $l - \alpha$ scheme – circular motion with constant speed

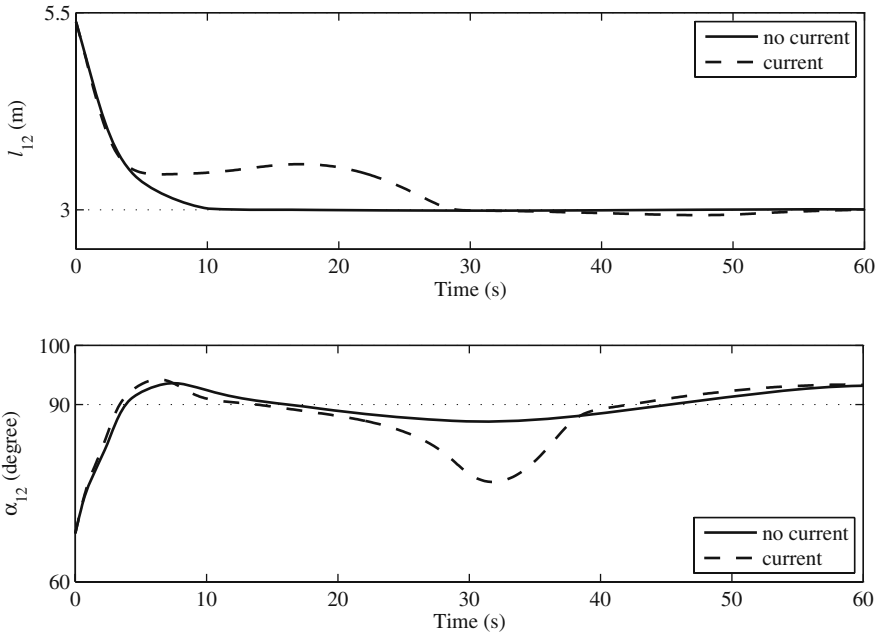


**Fig. 7.14** The $l - \alpha$ scheme – formation parameters

snapshots of the orientation of the follower. The orientation of the surface vessel is not tangent to the vessel's path as expected from zero-dynamics stability analysis. This orientation is stabilized.

Figure 7.14 shows the response of the controller outputs, $[l_{12}, \alpha_{12}]$. It can be seen that the controller outputs converge to their corresponding desired values for both scenarios. The control performance has been minimally affected by the disturbing current. The profile of the output responses confirms the stable error dynamics described in Eq. (7.74).

The difference in the orientation of vessel 2 and its control point's velocity direction, $\delta$, is shown in Fig. 7.15. As predicted by Eq. (7.20), this difference becomes a

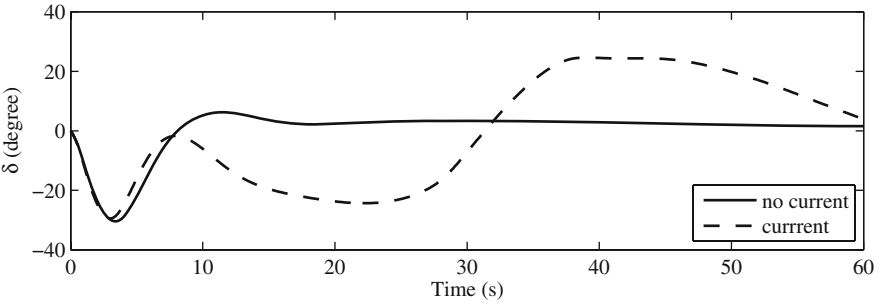


**Fig. 7.15** The $l - \alpha$ scheme – orientation difference

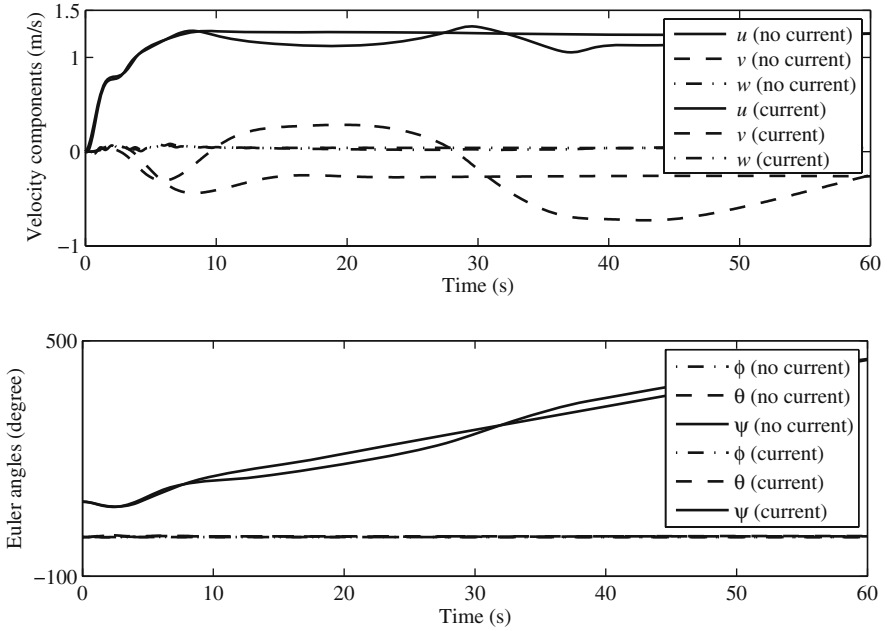


**Fig. 7.16** The $l - \alpha$ scheme – speed local components and the ZYX Euler angles $(\psi, \theta, \phi)$

constant value when there is no disturbance from the sea current. When the current is present, $\delta$ fluctuates. This is because the direction of the vessel relative to that of the current changes as the vehicle moves on a circular path. This affects the orientation of the vessel as it moves along the circle.

Some selected states of the 6-DOF model for the two scenarios are shown in Fig. 7.16. The longitudinal and lateral speeds are somewhat different for the two scenarios. However, the current has minimal effect on the the balance of the buoyancy force and the vessel's weight. That is the reason why, for both scenarios, the normal velocity vanishes. Also, the vessel's change of direction in a circular motion is indicated by the yaw Euler angle's $(\psi)$ monotonic increase. The roll $(\phi)$ and pitch $(\theta)$ angles are stabilized close to zero.

The result of this simulation shows that the $l - \alpha$ controller designed based on the 3-DOF vessel's model is suitable for controlling a real 6-DOF vessel, and it performs well when disturbance is present. Furthermore, the results of zero dynamics stability analysis, which predicated a constant $\delta$ for a circular motion is confirmed by the simulation results.

### *7.6.3 Design of the l – l Controller*

Three neighboring vessels in the formation structure are shown in Fig. 7.17. Assume that the formation structure is defined such that vessel 3 is designated to keep

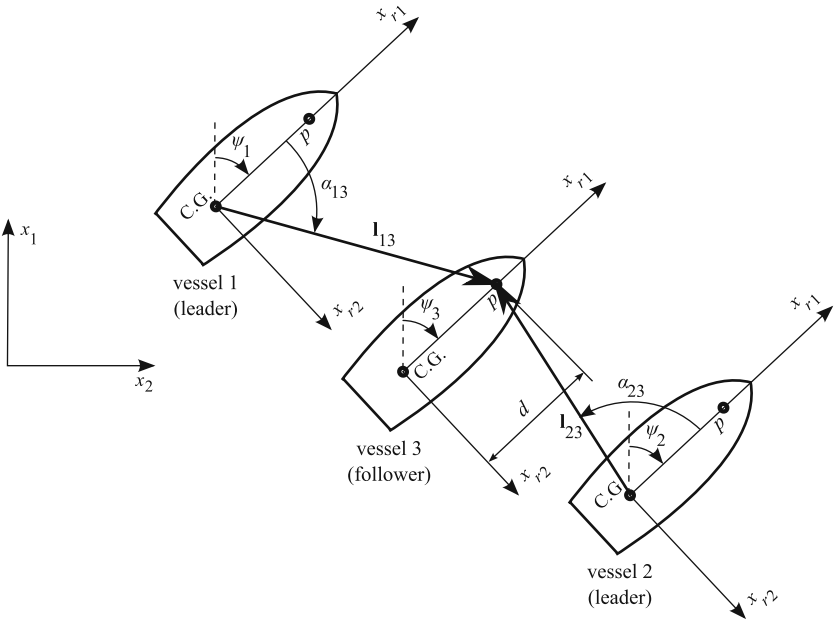


**Fig. 7.17** The $l - l$ scheme geometry

specified distances $l_{13}^d$ and $l_{23}^d$, with vessels 1 and 2, respectively. A controller must be designed to stabilize these distances, which are measured from the centres of mass of vessels 1 and 2 to a control point on vessel 3. The driving force and the steering torque of vessel 3, $F_3$ and $T_3$, are the control inputs, whereas the distances $l_{13}$ and $l_{23}$ are the control outputs. The equations relating the control inputs to the output, known as the input–output relations, must be obtained for designing the controller.

### 7.6.3.1 Kinematic Analysis

Similar to the $l - \alpha$ case, the relative acceleration equations for the control point $p$ on the follower vessel 3 with respect to the center of gravity of the leader vessel 1 is written. Similar relative acceleration relations are written for the follower vessel 3 and the leader vessel 3. These acceleration equations are, then, solved for $(\ddot{l}_{13}, \ddot{\alpha}_{13})$ and $(\ddot{l}_{23}, \ddot{\alpha}_{23})$, respectively. However, since the distances are the controller outputs, only the following resultant kinematic equations are relevant.

$$\begin{aligned}\ddot{l}_{13} &= (\ddot{y}_3 - \ddot{y}_1)\sin\alpha_1 + (\ddot{x}_3 - \ddot{x}_1)\cos\alpha_1 + d\ddot{\psi}_3\sin\gamma_2 \\ &\quad - d\dot{\psi}_3^2\cos\gamma_2 + l_{13}\dot{\alpha}_1^2, \qquad (7.84)\\ \ddot{l}_{23} &= (\ddot{y}_3 - \ddot{y}_2)\sin\alpha_2 + (\ddot{x}_3 - \ddot{x}_2)\cos\alpha_2 + d\ddot{\psi}_3\sin\gamma_3 \\ &\quad - d\dot{\psi}_3^2\cos\gamma_3 + l_{23}\dot{\alpha}_2^2, \qquad (7.85)\end{aligned}$$

where

$$\alpha_1 = \psi_1 + \alpha_{13}, \qquad \gamma_2 = \psi_1 + \alpha_{13} - \psi_3,$$
$$\alpha_2 = \psi_2 + \alpha_{23}, \qquad \gamma_3 = \psi_2 + \alpha_{23} - \psi_3.$$

#### 7.6.3.2 Input–Output Equations

Then, the input-output equations are obtained by writing the dynamic equations (7.4) for vessel 3 and substituting the results for $\ddot{x}_3$, $\ddot{y}_3$, and $\ddot{\psi}_3$ into the kinematics equations (7.84) and (7.85), which yields

$$\ddot{l}_{13} = f_1 + \tfrac{1}{m_{11}} F_3 \cos\gamma_2 + \tfrac{1}{I_{zz}} T_3 d \sin\gamma_2, \tag{7.86}$$
$$\ddot{l}_{23} = f_2 + \tfrac{1}{m_{11}} F_3 \cos\gamma_3 + \tfrac{1}{I_{zz}} T_3 d \sin\gamma_3, \tag{7.87}$$

where

$$f_1 = \frac{1}{m_{11}}(f_x \cos\alpha_1 + f_y \sin\alpha_1) + \frac{1}{I_{zz}} f_\psi d \sin\gamma_2 - \ddot{x}_1 \cos\alpha_1 - \ddot{y}_1 \sin\alpha_1$$
$$-d\dot{\psi}_3^2 \cos\gamma_2 + l_{13}\dot{\alpha}_1^2,$$
$$f_2 = \frac{1}{m_{11}}(f_x \cos\alpha_2 + f_y \sin\alpha_2) + \frac{1}{I_{zz}} f_\psi d \sin\gamma_3 - \ddot{x}_2 \cos\alpha_2 - \ddot{y}_2 \sin\alpha_2$$
$$-d\dot{\psi}_3^2 \cos\gamma_3 + l_{23}\dot{\alpha}_2^2.$$

The input–output relations (7.86) and (7.87) can be written in the matrix form.

$$\ddot{\mathbf{z}} = \mathbf{f} + \mathbf{b}\mathbf{u}, \tag{7.88}$$

where

$$\mathbf{z} = \begin{bmatrix} l_{13} \\ l_{23} \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \frac{\cos\gamma_2}{m_{11}} & \frac{d\sin\gamma_2}{I_{zz}} \\ \frac{\cos\gamma_3}{m_{11}} & \frac{d\sin\gamma_3}{I_{zz}} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} F_3 \\ T_3 \end{bmatrix}. \tag{7.89}$$

#### 7.6.3.3 Control Law

The controller is proposed based on the input–output equation (7.88). The output errors are defined as

$$\tilde{z}_1 = l_{13} - l_{13}^d \qquad \tilde{z}_2 = l_{23} - l_{23}^d, \tag{7.90}$$

and a stable second order nonlinear error dynamics is assumed ($k_i(e_i) > 0$):

$$\ddot{\tilde{z}}_i + 2\sqrt{k_i(\tilde{z}_i)}\dot{\tilde{z}}_i + k_i(\tilde{z}_i)\tilde{z}_i = 0, \qquad i = 1, 2. \tag{7.91}$$

The desired error behavior (7.91) is written in the matrix form for convenience.

$$\ddot{\tilde{\mathbf{z}}} + 2\sqrt{\mathbf{K}}\,\dot{\tilde{\mathbf{z}}} + \mathbf{K}\,\tilde{\mathbf{z}} = \mathbf{0}, \tag{7.92}$$

where

$$\sqrt{\mathbf{K}} = \begin{bmatrix} \sqrt{k_1} & 0 \\ 0 & \sqrt{k_2} \end{bmatrix}, \quad \mathbf{K} = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}. \tag{7.93}$$

Equation (7.92) can be solved for $\ddot{\mathbf{z}}$.

$$\ddot{\mathbf{z}} = \ddot{\mathbf{z}}^d - 2\sqrt{\mathbf{K}}\,\dot{\tilde{\mathbf{z}}} - \mathbf{K}\,\tilde{\mathbf{z}}. \tag{7.94}$$

Substituting for $\ddot{\mathbf{z}}$ from Eq. (7.94) into Eq. (7.88) and solving for the control input $\mathbf{u}$ results in

$$\mathbf{u} = \mathbf{b}^{-1}(\ddot{\mathbf{z}}^d - 2\sqrt{\mathbf{K}}\,\dot{\tilde{\mathbf{z}}} - \mathbf{K}\,\tilde{\mathbf{z}} - \mathbf{f}). \tag{7.95}$$

Note that the matrix $\mathbf{b}$ must be inverted for calculating the control law. Therefore, the determinant of this matrix must be nonzero at all times. This determinant is obtained as

$$\begin{aligned} \det(\mathbf{b}) &= \frac{d}{m_{11} I_{zz}} \sin(\gamma_3 - \gamma_2), \\ &= \frac{d}{m_{11} I_{zz}} \sin(\alpha_3 - \alpha_2). \end{aligned} \tag{7.96}$$

The determinant of $\mathbf{b}$ is zero, if $d = 0$. Once again, one can see the importance of using a control point other than the center of mass of the vessel, that is, $d \neq 0$. The determinant, also, reveals another condition under which the vessel becomes uncontrollable. That is when $\sin(\alpha_2 - \alpha_1)$ becomes zero. Substituting the values for $\alpha_1$ and $\alpha_2$, this condition reduces to

$$(\psi_2 + \alpha_{23}) - (\psi_1 + \alpha_{13}) = n\pi. \tag{7.97}$$

This represents the configuration at which the control point of the follower vehicle becomes collinear with the center of mass of the leaders (vessels 1 and 2) (Fig. 7.17). One must try to avoid this situation by defining proper formation structures. If this situation arises during a transient part of a formation change, the follower must temporarily switch to using the $l - \alpha$ scheme with one of the leaders until the singularity is cleared, after which the follower must switch back to using the $l - l$ scheme. This switch can be triggered if $|\alpha_2 - \alpha_1|$ becomes less than a predefined threshold.

It should be noted that, for a given set of desired formation parameters $[l_{13}, l_{23}]$, the $l - l$ scheme has two distinct equilibrium points. These two equilibrium points

have two distinct corresponding angle sets $[\alpha_{13}, \alpha_{23}]$. Only one of these angle sets define the true desired relative position of the follower with respect to the leaders. During a formation transition, the on-board computer can detect if the wrong equilibrium is being reached by observing the angle sets $[\alpha_{13}, \alpha_{23}]$. In that case, the controller switches to using the $l - \alpha$ scheme with one of the neighbors until the follower is at the correct equilibrium point. It then switches back to using the $l - l$ scheme with both its neighbors.

### 7.6.4 Implementation Notes

The formation control laws (7.82) and (7.95) need the inertial information of the leader vessel(s) for calculating the control action. The leader(s) can easily provide these information to the followers via communication using an on-board navigational sensor package, which is already present on the autonomous vessels. If the formation mesh is defined using triangular building blocks such that the vessels are not over-constrained, the communications can be minimized. Figure 7.18 shows two examples of formations with triangular building blocks. Each vessels, as a follower,

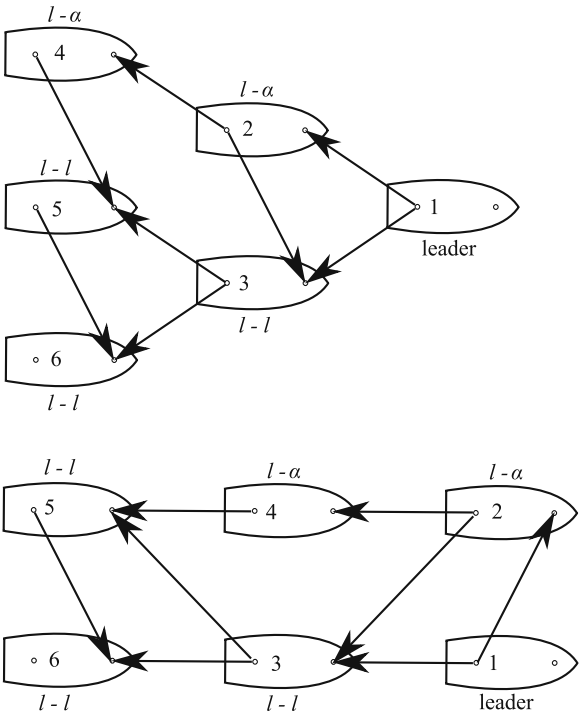


**Fig. 7.18** Six vessels in triangular and rectangular formations. In both formation meshes, 1 is the group leader; 2 follows 1; 3 follows 1 and 2; 4 follows 2; 5 follows 3 and 4; 6 follows 3 and 5. Table 7.4 shows the used control schemes for each of the vessels

**Table 7.4** Formation structure setup

| Vessel | Scheme | Follows |
|---|---|---|
| 1 | – | A predefined trajectory |
| 2 | $l-\alpha$ | Vessel 1 |
| 3 | $l-l$ | Vessels 1 & 2 |
| 4 | $l-\alpha$ | Vessel 2 |
| 5 | $l-l$ | Vessels 3 & 4 |
| 6 | $l-l$ | Vessels 3 & 5 |

has to receive linear and angular position, velocity, and acceleration from two leader vessels. If the formation mesh is defined correctly, each vessel has only to send its motion information to a very limited number of followers. For example, in the formations defined in Fig. 7.18, vessels 1, 2 ,and 3 have to sent their motion information to only two vessels and the rest of the vessels have to sent their motion information to only one vessel. From this discussion, one can conclude that the total number of vehicles in the group is not a factor in determining the required communication bandwidth. The communication bandwith can at most limit the number of followers that can be defined for a single vessel in the formation.

*Example 7.4.* Consider a 6-DOF surface vessel 3 whose dynamic parameters are listed in Table 7.2. The vessel 3, initially located at $(2, -10)$ m, is commanded to keep specified distances, $l_{13}^d = 5.0$ and $l_{23}^d = 5.0$ m, from vessels 1 and 2, respectively. The vessels 1 and 2 are moving on a straight line, shown by dashed lines, at a constant linear velocity of 1.5 m/s.

Apply the $l-l$ control law (7.95) to this 6-DOF vessels to more closely investigate the performance of the designed controller on a real 6-DOF vessel. Simulate the response of vessel 2 for two different scenarios to investigate the robustness of the controller to external disturbances. In the first scenario, assume that no sea current is present. For the second scenario, assume that a constant global sea current is flowing along the negative $x_2$-axis that exerts an equivalent force of 50 N on the vessel.

*Solution.* The controller parameters are listed in Table 7.5. The path of the three vessels are shown in Fig. 7.19. Vessel 3's path is shown by a solid curve. The path of vessels 1 and 2 are shown by dashed lines.

As can be seen in the figure, vessel 3 is not initially at the desired formation. However, after some time, it reaches its correct formation. The orientation of vessel 3 is shown by small line segments. Since the desired trajectory of the control point of the follower is a line, the orientation parameters of vessel 3 is stabilized at zero.

**Table 7.5** Controller parameters for Example 7.4

| $l-l$ |
|---|
| $n = 0.05$ |
| $d = 1.0$ |
| $\lambda_1 = 10, \lambda_2 = 10$ |
| $a_1 = 1000, a_2 = 1000$ |
| $l_{13}^d = 6.0\,\text{m}, l_{23}^d = 6.0\,\text{m}$ |

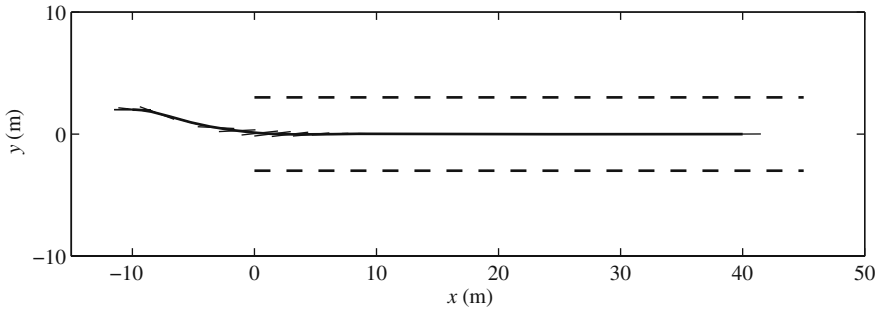


**Fig. 7.19** The $l - l$ scheme – linear motion with constant speed

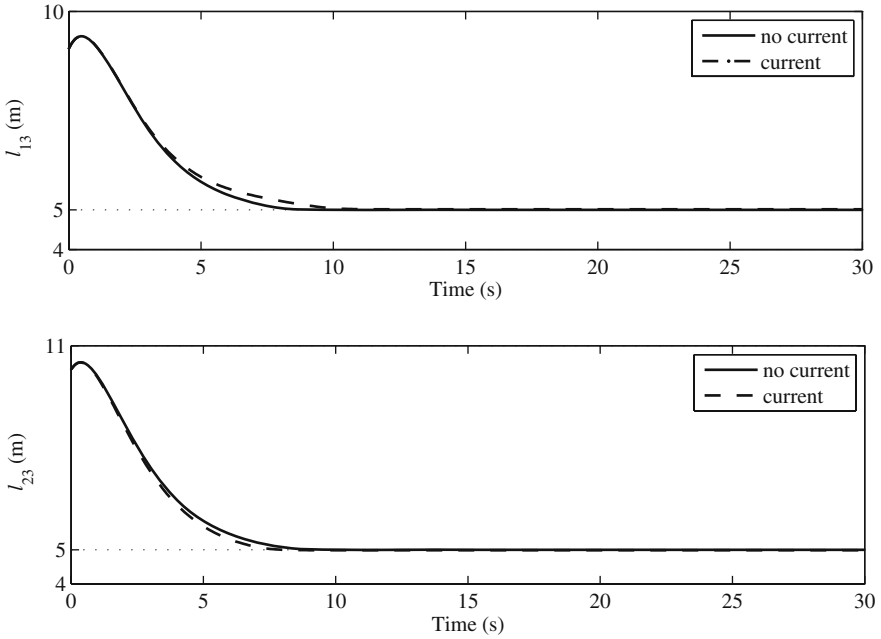


**Fig. 7.20** The $l - l$ scheme – formation parameters

This also indicated the stability of the zero dynamics of the vessel. Figure 7.20 shows the responses of the controller outputs. It can be seen that the error dynamics as indicated by Eq. (7.91) is stable for both the sea current scenarios. The representative of the zero dynamics of the vessel, $\delta$, which indicates the difference in the orientation of vessel 3 and its control point's velocity direction, is shown in Fig. 7.21. When the sea current is absent, this angle is zero. However, when the sea current exists, the vessel must have a constant angle with the direction of its motion such that the propeller thrust can compensate for the lateral force caused by the sea current. Therefore, $\delta$ becomes nonzero when the sea current exists. It can be seen in Fig. 7.22 that the longitudinal and lateral speeds of the vessel are not affected very

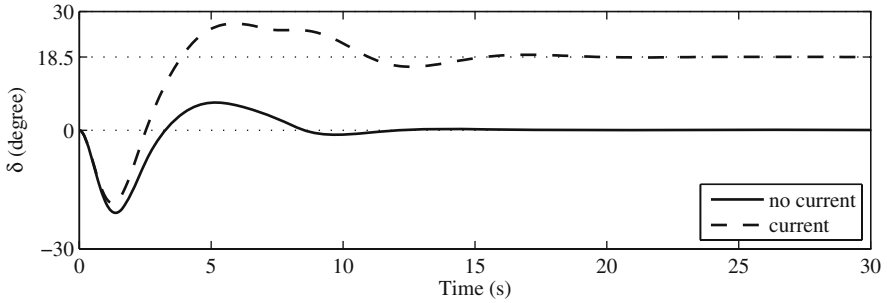


**Fig. 7.21** The $l-l$ scheme – orientation difference

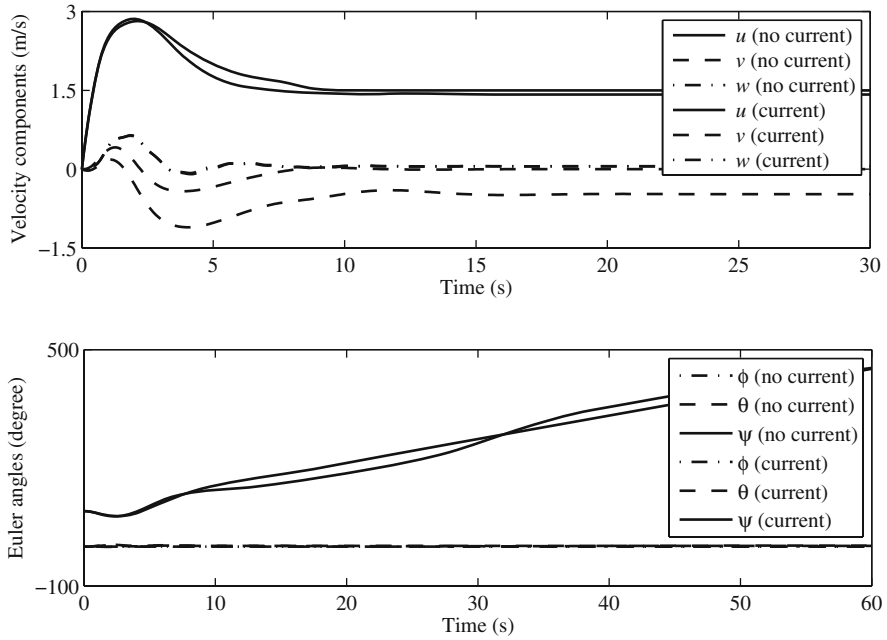


**Fig. 7.22** The $l-l$ scheme – speed local components and the ZYX Euler angles $(\psi, \theta, \phi)$

much by the sea current. Also, the normal velocity vanishes for both scenarios. The yaw Euler angle ($\psi$), representing the heading of the vessel's motion, stabilizes at $0^\circ$ when there is no disturbance, while the roll ($\phi$) and pitch ($\theta$) angles are stabilized close to zero.

The results of this simulation shows that the $l-l$ controller designed based on the 3-DOF vessel's model is suitable for controlling a real 6-DOF vessel. The controller performace is acceptable when disturbance is present. Furthermore, the results of zero-dynamics stability analysis, which predicated a $\delta$ equal to zero for a linear motion, is confirmed by the simulation results.

## 7.7 Summary

In this chapter, first, the 6-DOF dynamic model of a surface vessel was presented. Normally, only two control actuations are available for a surface vessel. The large unbalance in the number of DOF and number of control actuations lead to difficulties in controller design. However, in practice, three of the 6 DOFs are inherently stable because of the design of a surface vessel. These DOFs were assumed to be close to their corresponding equilibrium states. With this assumption, the dynamic equations of motion of a surface vessel were reduced to a 3-DOF dynamic model.

Since the vessels are underactuated, the stability of the internal dynamics of the system had to be investigated. It was shown that for a general motion with constant speed, the internal dynamics of a 3-DOF surface vessel has a second-order damped response. The equilibrium points of the zero dynamics were derived for circular and linear motions of the formation.

The 3-DOF dynamic model was used for trajectory-tracking controller design. Two methods were presented: the input–output linearization and the sliding mode control method. The performance of these two methods in presence of uncertainties and disturbances were compared. The sliding mode control method proved to be more robust.

Then, the problem of controlling multiple surface vessels in user-defined formations was investigated. Two control schemes were presented than can define the internal geometry of the a desired formation. These schemes define the desired relative positions of neighboring vessels to form a unique formation mesh. Since the geometrical schemes are local, each vessel has to have information about only one or two of its neighbors, depending on the location of the vessel in the formation. The 3-DOF dynamic model of the surface vessels were used for designing two nonlinear controllers for the surface vessels. The parameters that define the internal geometry of the formation are directly used as the controller output. It was shown that the nonlinear controllers can stabilize the formation parameters even in the presence of sea disturbances, hence, they can position the vessels in user-defined formations. The effectiveness of the control laws was shown by numerical simulations using a comprehensive 6-DOF dynamic model.

## Problems

**Problem 7.1.** Consider a surface vessel whose dynamics can be reduced to a 3-DOF model. The parameters of the 3-DOF dynamic model of the surface vessel are listed in Table 7.1. Using the control law (7.36), simulate the autonomous motion of the surface vessel on a desired sinusoidal path for the control point.

$$x_2 = 2\sin(\frac{x_1}{2}), \qquad 0 < x_1 < 4\pi, \tag{7.98}$$

where all the distances are in meters. Assume that the vessel has to maintain a constant linear velocity of 0.5 m/s in the $x_1$ direction. Plot the orientation stability parameter $\delta$ introduced in Section 7.4 and comment on the response of the zero dynamics of the surface vessel. The vessel is moving with a longitudinal velocity of 0.5 m/s at the beginning of the motion, its longitudinal axis makes an angle of $\psi = \arctan(2)$ rad with the positive $x_1$ axis, and its control point is initially at (0, 0) m. Assume $d = 1$ m.

**Problem 7.2.** Consider simplified 3-DOF dynamic model for a surface vessel as presented in Eq. (7.3). Assume the presence of unknown wave disturbances represented by $W_u$ and $W_v$ and uncertainty in parameters $m_{22}$, $d_{11}$, $d_{22}$, and $d_{66}$. Assume no uncertainty in $m_{11}$ and $I_{zz}$.

(a) Design a robust sliding mode controller to pilot the control point of the vessel on a desired trajectory in presence of an unknown wave disturbance and parameter uncertainty.
(b) Apply the control law derived in part (a) of this problem to the dynamic model. Assume that the nominal wave is zero. Use the nominal parameters given in Table 7.1. Consider a $\pm 5\%$ uncertainty bound for the uncertain parameters of the dynamic model and $\pm 10$ N bounds for the longitudinal and the lateral wave forces.
(c) Using the 3-DOF simplified model of the surface vessel (Eq. (7.3)), investigate the performance of the controller for a case in which the uncertainty in the dynamic parameters are 2% of their corresponding nominal value. Suppose a wave force of $W = 5\sin(2\pi t/10)$ N is acting on the vessel parallel to the global $x_2$ axis. Simulate the motion of the surface vessel for 20 s on a desired linear path.

$$x_1^d(t) = 0.5t,$$
$$x_2^d(t) = 0.$$

The vessel is initially at rest at the origin oriented parallel to the $x_2$ axis.
(d) Plot the path of the control point of the vessel on top of the desired path. Plot the control force and moment. Tune the controller parameters for an acceptable response. Can the controller be tuned to perform well even in the presence of the unknown wave disturbance?
(e) Plot the path of the center of mass of the vessel on top of the desired path. Discuss the difference of the path of the center of mass and that of the control point. Why does this difference exist?
(f) Plot the orientation stability parameter $\delta$ introduced in Section 7.4 and comment on the response of the zero dynamics of the surface vessel. Is the zero-dynamics response reasonable?
(g) Apply the controller designed based on the input–output linearization method to the vessel's model that includes the wave disturbance. Simulate the motion of the vessel for the desired linear motion in part (7.2). How do the results of the sliding mode control and the input–output linearization method compare?

**Problem 7.3.** Nine ASVs have to move in a square formation with three vessels at each side of the square. The sides of the square are 15 m long.

(a) Assign either a $l - l$ or a $l - \alpha$ scheme to each vessel in the formation to form a formation mesh with triangular building blocks for a better mesh interconnection. Assume that the group leader (vessel 1) is at one of the leading edges of the formation.
(b) Number the vessels such that each vessels follows a vessel or vessels with a number lower than themself. Although the numbering is optional for real-time implementation, this numbering scheme makes the simulations possible. The motion of a lower numbered vessel must be simulated first to provide information for simulating the proceeding vessels.
(c) If the distance of the control point to the center of mass of the vessels is chosen to be $d = 1$ m, calculate and tabulate the desired formation parameters for each of the vessels in the formation.
(d) Suppose that the center of the group is designated to move on a desired line as given in part (7.2) of Problem 7.2. Calculate the initial position of all the vessels and the desired trajectory of the group leader. Simulate the motion of the nine vessels using the following sequence. First, simulate the motion of the group leader by applying a trajectory-tracking method presented in this chapter. Then, use the response of the group leader to simulate the motion of its immediate followers. Use the response of the followers for simulating the motion of the their second-level followers in sequence until all the vessels have been processed.
(e) Is this sequential scheme needed for the real-time implementation? How would the real-time implementation work if the leaders are able to communicate with their corresponding followers in real-time?

# Chapter 8
# Autonomous Helicopters

## 8.1 Introduction

The use of UAVs has been increasing in the past few years. The serious application of these vehicles, similar to many advanced technologies, started in military in different countries. Different types of unmanned vehicles have been used. However, the fixed wing unmanned aerial vehicles have been more common in military applications due to their longer range, flight efficiency, and ease of control. Helicopters, on the other hand, are more maneuverable. Unmanned helicopters already have far more civilian applications than the fixed wing aircraft. They are used for aerial photography, timber survey, agriculture, etc.

Most of these UAVs are controlled remotely by a pilot on the ground. Piloting these vehicles remotely is not a trivial task. Hours of training and good intuitive skills are necessary for a pilot to acquire enough expertise to become a safe pilot. Even then, piloting these vehicles is prone to human errors. In addition, mostly the range of operation of the remote controlled vehicle is limited because the pilot has to see the vehicle's exact motion to be able to control it. As the application areas and the number of UAVs grow, there is a need for more intelligent aerial vehicles that are able to fly with minimum to no human interaction. There are so many problems in different areas related to intelligent (autonomous) aerial vehicles that have to be solved before a fully autonomous aerial vehicle can materialize. These areas, without any particular order of importance, include mission planning, machine vision, path planning and obstacle avoidance, platform development, navigational sensor development, dynamic modeling and model identification, and, finally, control.

The goal of this chapter is to introduce the dynamics of unmanned helicopters and present the classical and modern methods for controlling autonomous helicopters. First, the 6-DOF rigid body model of a helicopter is presented. Then, the position control for the helicopter using the classical PID method is discussed. Later, the trajectory-tracking control of autonomous helicopters will be presented.

F. Fahimi, *Autonomous Robots*, DOI 10.1007/978-0-387-09538-7_8

## 8.2 A 6-DOF Dynamic Model of a Helicopter

This section presents the dynamic model of a helicopter, shown in Fig. 8.1. Two frames are defined for this dynamic model: the inertial frame {0}, and the helicopter body frame $\{B\}$ with an origin at the center of mass. Six DOFs are assumed for each helicopter. Three translational DOFs, the surge, sway, and bounce of the center of mass, are expressed in the inertial frame {0} and are denoted by $(x_1, x_2, x_3)$. Three rotational DOFs are represented by the yaw-pitch-roll $(ZYX)$ Euler angles $(\psi, \theta, \phi)$. These Euler angles define the orientation of the body frame with respect to the inertial frame through the following transformation matrix [34]:

$$\mathbf{R}_{0B} = \begin{bmatrix} \mathrm{c}\psi\,\mathrm{c}\theta & (-\mathrm{s}\psi\,\mathrm{c}\phi + \mathrm{c}\psi\,\mathrm{s}\theta\,\mathrm{s}\phi) & (\mathrm{s}\psi\,\mathrm{s}\phi + \mathrm{c}\psi\,\mathrm{s}\theta\,\mathrm{c}\phi) \\ \mathrm{s}\psi\,\mathrm{c}\theta & (\mathrm{c}\psi\,\mathrm{c}\phi + \mathrm{s}\psi\,\mathrm{s}\theta\,\mathrm{s}\phi) & (-\mathrm{c}\psi\,\mathrm{s}\phi + \mathrm{s}\psi\,\mathrm{s}\theta\,\mathrm{c}\phi) \\ -\mathrm{s}\theta & \mathrm{c}\theta\,\mathrm{s}\phi & \mathrm{c}\theta\,\mathrm{c}\phi \end{bmatrix}, \tag{8.1}$$

where $\mathrm{c} = \cos$ and $\mathrm{s} = \sin$. The inertial position of the helicopter along with the $ZYX$ Euler angles determine the geometrical configuration of the helicopter at any given time. These variables represent the generalized coordinates (or configuration variables) of the helicopter as a dynamic system.

$$\mathbf{q} = \begin{bmatrix} x_1 & x_2 & x_3 & \phi & \theta & \psi \end{bmatrix}^T. \tag{8.2}$$

The configuration variables represent only the geometrical configuration of the helicopter as a dynamic system. To define the dynamic state of the helicopter at any given time, the rate of change of these configuration variables must also be known. Using the rate of change of the configuration variables for completing the state variables for the helicopter is the first choice that comes to mind. However, for some dynamic systems, and especially for spatial dynamic systems, the equations of motion become less complicated, if linear combinations of the rate of the configuration variables are used. These linear combination of the rate of configuration variables are called the generalized speeds. Here, the inertial velocity of the center of

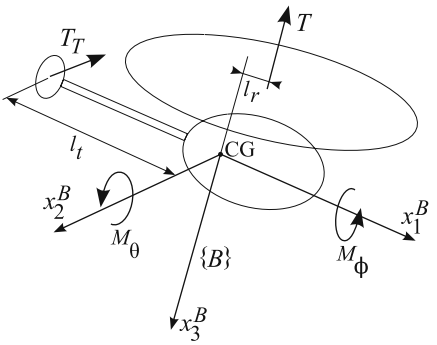


**Fig. 8.1** A 6-DOF dynamic model of a helicopter

mass of the helicopter and the components of the angular velocity of the helicopter in its body frame are selected as the generalized speeds.

$$\mathbf{v} = \left[\dot{x}_1\ \dot{x}_2\ \dot{x}_3\ \omega_1^B\ \omega_2^B\ \omega_3^B\right]. \tag{8.3}$$

These generalized speeds are related to the rate of the configuration variables through the following relation.

$$\mathbf{v} = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & -\sin\theta \\ 0 & 0 & 0 & 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & 0 & 0 & 0 & -\sin\phi & \cos\theta\cos\phi \end{array}\right] \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}, \tag{8.4}$$

or

$$\mathbf{v} = \mathbf{T}_R \dot{\mathbf{q}}, \tag{8.5}$$

where $\mathbf{T}_R$ is called the rate transformation matrix. Although this matrix is singular at $\theta = \pm\pi/2$, the helicopter is not expected to operate in that orientation (pointing straight up or down). The $12 \times 1$ state vector of the helicopter as a dynamic system is

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \mathbf{v} \end{bmatrix}. \tag{8.6}$$

Now that the generalized coordinates and the generalized speeds are defined, one can derive the equations of motion. Here, the Newton-Euler approach is used.

The external force and torque expressed in the body frame are $\mathbf{F}^B$ and $\mathbf{M}^B$. The external force includes the aerodynamic drag force vector in the body frame $\mathbf{D}^B$, the main and tail rotor thrust vector in the body frame $\mathbf{T}^B$, and the gravitational force in the inertial frame $\mathbf{W}$:

$$\mathbf{F}^B = \mathbf{T}^B + \mathbf{D}^B + \mathbf{R}_{0B}^T \mathbf{W}, \tag{8.7}$$

where

$$\mathbf{T}^B = \begin{bmatrix} 0 \\ -T_T \\ -T \end{bmatrix}, \tag{8.8}$$

and

$$\mathbf{W} = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}. \tag{8.9}$$

The external torque includes the three main directional torques $M_\phi$, $M_\theta$, and $T_T l_t$ as well as a torque $T l_r$ due to the offset of the rotor hinge with respect to the body $z$-axis, and the motor torque $\tau_m$:

$$\mathbf{M}^B = \begin{bmatrix} M_\phi \\ M_\theta + T l_r \\ T_T l_t + \tau_m \end{bmatrix}. \tag{8.10}$$

Usually, $\tau_m$ is assumed to be proportional to the main rotor thrust, $T$. That is,

$$\tau_m = K_m T. \tag{8.11}$$

The translational and rotational equations of motion of the helicopter can be written as

$$m \begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \ddot{x}_3 \end{bmatrix} = \mathbf{R}_{0B}\mathbf{T}^B + \mathbf{R}_{0B}\mathbf{D}^B + \mathbf{W}, \tag{8.12}$$

$$\mathbf{I} \begin{bmatrix} \dot{\omega}_1^B \\ \dot{\omega}_2^B \\ \dot{\omega}_3^B \end{bmatrix} = \mathbf{M}^B - \boldsymbol{\omega}^B \times \mathbf{I}\boldsymbol{\omega}^B, \tag{8.13}$$

where $m$ is the helicopter mass and $\mathbf{I}$ is

$$\mathbf{I} = \begin{bmatrix} I_{11} & 0 & 0 \\ 0 & I_{22} & 0 \\ 0 & 0 & I_{33} \end{bmatrix}. \tag{8.14}$$

By combining Eqs. (8.5), (8.6), (8.12), and (8.13), one can write the first-order form of the equations of motion for the helicopters as follows.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \hline \dot{\mathbf{v}} \end{bmatrix} = \left[\begin{array}{c} \mathbf{T}_R^{-1}\mathbf{v} \\ \hline m^{-1}(\mathbf{R}_{0B}\mathbf{T}^B + \mathbf{R}_{0B}\mathbf{D}^B + \mathbf{W}) \\ \mathbf{I}^{-1}(\mathbf{M}^B - \boldsymbol{\omega}^B \times \mathbf{I}\boldsymbol{\omega}^B) \end{array}\right]. \tag{8.15}$$

The independent inputs that control the helicopter's motion are organized in a column vector.

$$\mathbf{u} = \begin{bmatrix} T & M_\phi & M_\theta & T_T \end{bmatrix}^T. \tag{8.16}$$

If the control input $\mathbf{u}$ is known, one can find the trajectory of the helicopter's motion by integrating Eq. (8.15). Our goal is to find a control law that determines

**u** such that the helicopter is stabilized at a given point or follows a desired 3D trajectory.

## 8.3 Position Control for Autonomous Helicopters

In the previous section, we defined six generalized coordinates to describe the configuration of the helicopter uniquely at any given time. However, we also saw that only four control inputs are available for controlling these six generalized coordinates. This indicates that a helicopter is an underactuated system. The control development for a helicopter as an underactuated dynamic system is challenging. Understanding the physics of how these four control inputs affect the configuration variables of the helicopter is essential for taking advantage of the limited number of inputs for control design.

By observing the control inputs shown in Fig. 8.1, one can see that, when the helicopter is in hover, the four control inputs, $T$, $M_\phi$, $M_\theta$, and $T_T$ directly affect the configuration variables $x_3$, $\phi$, $\theta$, and $\psi$, respectively. At hover with no wind, $\phi$ and $\theta$ are approximately zero.[1] This allows the main rotor thrust to mainly balance the weight of the helicopter. The pilot can accelerate the helicopter forward or backward by producing a non zero pitch angle $\theta$, which is generated by the applying the control input $M_\theta$.[2] Similarly, the helicopter accelerates sideways if the pilot changes the roll angle $\phi$ by using the control input $M_\phi$. A diagonal acceleration can be generated by applying the two control inputs $M_\theta$ and $M_\phi$. Therefore, the pitch and the roll angle affect the acceleration in the $x_1$ and $x_2$, respectively, hence, they affect the $x_1$ and $x_2$ positions indirectly.

The above explanation helps us to formulate a position control strategy. The control strategy and the steps involved in deriving the control laws follow.

1. First, the helicopter should be able to hover at any height ($x_3^d$) with any heading ($\psi^d$) before it can be commanded to move from one position to another. The hovering condition corresponds to a unique set of roll and pitch angles ($\phi_h$, $\theta_h$) (close to zero), at which the main rotor's thrust is in equilibrium with the helicopter's weight and the tail rotor thrust. These angles are called "the hover trimming angles." The values of the trimming angles at hover (equilibrium) can be found using the dynamic equations of motion for the helicopter.
2. Control laws for the four control input ($T$,$M_\phi$,$M_\theta$,$T_T$) are derived that can stabilize the four configuration variables that correspond to hover at their desired values ($x_3^d$,$\phi_h$,$\theta_h$,$\psi^d$). These control laws receive no direct feedback from the

[1] A helicopter with a tail rotor has a slight non zero roll angle $\phi$ such that a lateral component of the main rotor thrust can compensate for the tail rotor's thrust.

[2] Note that when the helicopter leans forward, the vertical component of the main rotor thrust decreases. The pilot has to increase the thrust at the same time to ensure that the vertical component of the main rotor thrust can still balance the weight of the helicopter.

other two configuration variables ($x_1$, $x_2$). Hence, they cannot control these two position components.

3. In order to move the helicopter in the $x_1$ and $x_2$ directions, the roll and pitch angles must depart from their corresponding hover equilibrium positions. These departures from the hover equilibrium angles must be such that the thrust of the main rotor leans toward the desired position ($x_1^d$ $x_2^d$), which causes the helicopter to accelerate toward the desired position. Once the helicopter gets close to the desired position, these departure angles must vanish. Therefore, a control law that determines the roll and pitch departure angles ($\phi_p$, $\theta_p$) based on the errors in $x_1$ and $x_2$ positions must be derived.
4. Now, the actual desired roll and pitch angles are a combination of the ones corresponding to the hovering position and the departure angles corresponding to the positioning. If the combination angles ($\phi_c = \phi_h + \phi_p$ and $\theta_c = \theta_h + \theta_p$) are fed to the control law derived in the first step of this strategy (instead of $\phi_h$ and $\theta_h$), hovering and positing can be performed simultaneously.

Figure 8.2 shows the block diagram of the PID control strategy. In the following, the different blocks and parts of this control strategy are discussed and their corresponding control laws and relations are derived.

### 8.3.1 The Hover Trimming Angles

The hover trimming angles, $\phi_h$ and $\theta_h$, can be found by considering the dynamic equations of motion of the helicopter at equilibrium, where the accelerations and velocities are zero. In other words, the inertia force and damping forces vanish at equilibrium. Let us start with Eq. (8.12), which, for zero accelerations and zero damping, becomes

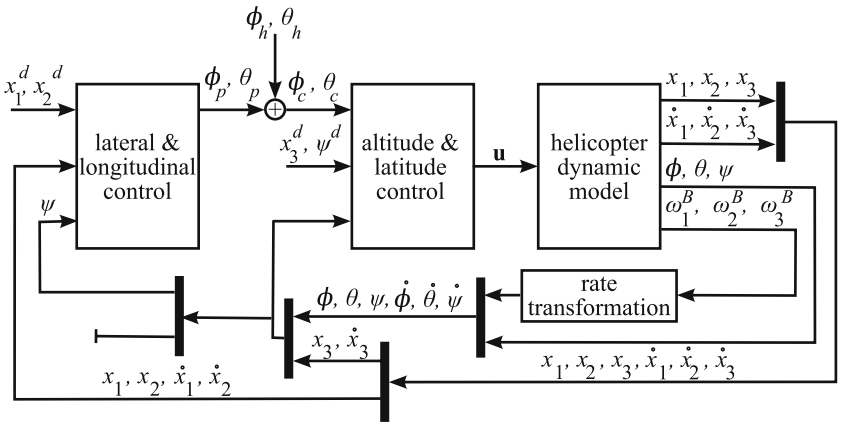


**Fig. 8.2** The block diagram for the PID position control

$$(\mathbf{R}_{0B})_h \mathbf{T}_h^B + \mathbf{W} = 0, \tag{8.17}$$

where the subscript $h$ indicates the hovering condition. The rotation matrix in Eq. (8.17) is the result of three consequent rotations defined by the Euler $ZYX$ angles and can be expanded as follows:

$$(\mathbf{R}_{0B})_h = \mathbf{R}_{x_3}(\psi^d).\mathbf{R}_{x_2}(\theta_h).\mathbf{R}_{x_1}(\phi_h), \tag{8.18}$$

Substituting the above relation in Eq. (8.17) and rearranging results in

$$\mathbf{R}_{x_2}(\theta_h).\mathbf{R}_{x_1}(\phi_h)\mathbf{T}_h^B = -\mathbf{R}_{x_3}^{-1}(\psi^d)\mathbf{W}. \tag{8.19}$$

Assuming small trimming angles, one can expand the above equation as follows:

$$\begin{bmatrix} 1 & 0 & \theta_h \\ 0 & 1 & 0 \\ -\theta_h & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -\phi_h \\ 0 & \phi_h & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -T_{Th} \\ -T_h \end{bmatrix} = \begin{bmatrix} \cos\psi^d & \sin\psi^d & 0 \\ -\sin\psi^d & \cos\psi^d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}. \tag{8.20}$$

Note that the tail and the main rotor thrusts at the hover condition (two trimming inputs $T_{Th}$ and $T_h$) are present in this equation. These are unknown at this stage. Therefore, the above equation cannot result in a definite solution for the trimming angles. The above equation must be completed by using Eq. (8.13) at the hover condition, where the angular accelerations and velocities are zero. Equation (8.13) at equilibrium becomes

$$(\mathbf{M}_B)_h = \begin{bmatrix} M_{\phi h} \\ M_{\theta h} + T_h l_r \\ T_{Th} l_t + K_m T_h \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{8.21}$$

Solving the six equations resulting from Eqs. (8.20) and (8.21) for the two unknown trimming angles and and the four unknown trimming inputs results in the following:

$$\begin{aligned} \phi_h &= \frac{K_m}{l_t}, \\ \theta_h &= 0, \\ T_h &= \frac{mg}{1-(K_m/l_t)^2}, \\ M_{\phi h} &= 0, \\ M_{\theta h} &= \frac{-mgl_r}{1-(K_m/l_t)^2}, \\ T_{Th} &= \frac{-mg(K_m/l_t)}{1-(K_m/l_t)^2}. \end{aligned} \tag{8.22}$$

Note that finding the trimming angles led to finding the trimming inputs. In fact, these trimming inputs will prove very useful when deriving the PID control laws.

### 8.3.2 The Longitudinal and Lateral Control Law

In this section, the roll and pitch angles that will cause the helicopter to move from its current $x_1$ and $x_2$ position toward the desired $x_1^d$ and $x_2^d$ position are derived. The helicopter must initially accelerate in the direction connecting its current position to the desired position. This can be done by tilting the helicopter such that the vector representing the main rotor's thrust force has a horizontal component, $\mathbf{T}_p$, pointing toward the desired position. This component must vanish gradually as the helicopter gets closer to the desired position. It is intuitive to define a control law for this component based on the lateral and longitudinal position errors. In the following, the mathematical formulation representing the discussed idea are derived.

Let the notation $\hat{\mathbf{T}}$ represent the unit vector of representing the main rotor's thrust direction. The projection of this unit vector on the horizontal plane $x_1 - x_2$ can be derived as:

$$\hat{\mathbf{T}}_p = \hat{\mathbf{T}} - (\hat{\mathbf{T}}.\hat{\mathbf{i}}_3)\hat{\mathbf{i}}_3. \tag{8.23}$$

A control law for the unit horizontal projection of the main thrust vector must be defined such that it vanishes when the lateral and the longitudinal position errors become zero. A PID control law can be selected as follows.

$$\hat{\mathbf{T}}_p = \begin{bmatrix} -(k_{x1P})e_1 - (k_{x1D})\dot{e}_1 - (k_{x1I})\int_0^t e_1 dt \\ -(k_{x2P})e_2 - (k_{x2D})\dot{e}_2 - (k_{x2I})\int_0^t e_2 dt \\ 0 \end{bmatrix}, \tag{8.24}$$

where all $k$'s are positive numbers and the helicopter's longitudinal and lateral position errors projected are expressed as

$$e_1 = x_1 - x_1^d, \tag{8.25}$$

$$e_2 = x_2 - x_2^d. \tag{8.26}$$

Note that the derivative of the desired values are assumed to be zero for position control. The unit horizontal thrust component is related to the departure angles $\phi_p$ and $\theta_p$. In the following, this relation is used to write the control law (8.24) in terms of the departure angles.

The unit vector of representing the main rotor's thrust direction when the departure angles are applied can be written as

$$\hat{\mathbf{T}} = \mathbf{R}_{x_3}(\psi).\mathbf{R}_{x_2}(\theta_p).\mathbf{R}_{x_1}(\phi_p) \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}. \tag{8.27}$$

Substituting Eq. (8.23) into the above relation yields

$$\mathbf{R}_{x_2}(\theta_p).\mathbf{R}_{x_1}(\phi_p)\begin{bmatrix}0\\0\\-1\end{bmatrix} = \mathbf{R}_{x_3}^{-1}(\psi)(\hat{\mathbf{T}}_p + (\hat{\mathbf{T}}.\hat{\mathbf{i}}_3)\hat{\mathbf{i}}_3). \tag{8.28}$$

For small departure angles $\phi_p$ and $\theta_p$, the above equation yields to a simple form:

$$\begin{bmatrix}-\theta_p\\ \phi_p\\ -1\end{bmatrix} = \mathbf{R}_{x_3}^{-1}(\psi)(\hat{\mathbf{T}}_p + (\hat{\mathbf{T}}.\hat{\mathbf{i}}_3)\hat{\mathbf{i}}_3). \tag{8.29}$$

The first two components of the above equation give the longitudinal and lateral control law for the departure angles, with yields

$$\begin{bmatrix}\phi_p\\ \theta_p\end{bmatrix} = \begin{bmatrix}-\sin\psi & \cos\psi\\ -\cos\psi & -\sin\psi\end{bmatrix}\begin{bmatrix}(k_{x1P})e_1 + (k_{x1D})\dot{e}_1 + (k_{x1I})\int_0^t e_1 dt\\ (k_{x2P})e_2 + (k_{x2D})\dot{e}_2 + (k_{x2I})\int_0^t e_2 dt\end{bmatrix}. \tag{8.30}$$

The control law (8.30) allows the helicopter to gain the correct roll and pitch angle that moves it toward the lateral and longitudinal desired positions. When the helicopter arrives at the desired position, these departure angles become zero. However, the helicopter has to maintain the hovering roll and pitch angles to be able to stay at the desired postion. Therefore, the internal desired roll and pitch angles for the controller are

$$\phi_c = \phi_h + \phi_p, \tag{8.31}$$

$$\theta_c = \theta_h + \theta_p. \tag{8.32}$$

Now, another control law is required to assure that the internal desired roll and pitch angles are maintained during the motion of the helicopter from the initial position to the desired postion. This control law also makes sure that the helicopter remains at the desired height and heading angle. Since this control deals with the roll, pitch, and heading angles and the height, it is called the latitude and altitude controller, whose associated control law is derived in the next section.

### *8.3.3 The Latitude and Altitude Control Law*

In this section, the latitude and altitude control law is derived such that the helicopter maintains the internal desired roll and pitch angles for the controller, $\phi_c$ and $\theta_c$, the desired heading angle $\psi^d$, and the desired height $x_3^d$. The physical control inputs available for the helicopter, $T$, $M_\phi$, $M_\theta$, and $T_T$, directly affect the four desired parameters list above, respectively. Therefore, selecting PID control laws appears straight forward. Here are the control laws for the physical inputs:

$$T = T_h + [(k_{x3P})e_3 + (k_{x3D})\dot{e}_3 + (k_{x3I})\int_0^t e_3 dt], \tag{8.33}$$

$$M_\phi = M_{\phi h} - [(k_{\phi P})e_4 + (k_{\phi D})\dot{e}_4 + (k_{\phi I})\int_0^t e_4 dt], \tag{8.34}$$

$$M_\theta = M_{\theta h} - [(k_{\theta P})e_5 + (k_{\theta D})\dot{e}_5 + (k_{\theta I})\int_0^t e_5 dt], \tag{8.35}$$

$$T_T = T_{Th} - [(k_{\psi P})e_6 + (k_{\psi D})\dot{e}_6 + (k_{\psi I})\int_0^t e_6 dt], \tag{8.36}$$

where all $k$'s are positive numbers and the errors are defined as

$$e_3 = x_3 - x_3^d, \tag{8.37}$$

$$e_4 = \phi - \phi_c, \tag{8.38}$$

$$e_5 = \theta - \theta_c, \tag{8.39}$$

$$e_6 = \psi - \psi^d. \tag{8.40}$$

Note that the derivative of the desired values are assumed to be zero for position control. The control laws (8.33), (8.34), (8.35), and (8.36) are selected such that when the errors vanish, the controller maintains the control inputs at the required level for hovering, which allows the helicopter to hover after it arrives at the desired position and heading.

Now, almost all the control blocks shown in Fig. 8.2 have been derived. In that figure, the "lateral and longitudinal control" block represents Eq. (8.30), the "altitude and latitude control" block corresponds to Eqs. (8.33) (8.34), (8.35), and (alt-lat-cont-law-4), the "helicopter dynamic model" block reflects Eq. (8.15), and the "rate transformation" block can be derived by investigating the submatrices of Eq. (8.4). In other words, this block corresponds to the following relation.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} \omega_1^B \\ \omega_2^B \\ \omega_3^B \end{bmatrix}. \tag{8.41}$$

This concludes the PID position controller design for an autonomous helicopter.

*Example 8.1.* Consider a small autonomous helicopter with the mass and geometrical properties listed in Table 8.1. In this table, $C$'s are the aerodynamic damping coefficients to be used for calculating the damping force matrix in the helicopter's dynamic equations. The helicopter is hovering at an initial position of (3.0, 15.0, 0.0) m. The autonomous controller is given a desired position and heading of $(x_1^d, x_2^d, x_3^d, \psi^d) = (4.0\,\text{m}, 14.0\,\text{m}, -1.0\,\text{m}, -\pi/4\,\text{rad})$. Using the dynamic model (8.15) and the PID control laws derived in the previous section, simulate the motion of the helicopter under autonomous position control.

**Table 8.1** Properties of a small autonomous helicopter

| | | |
|---|---|---|
| $m = 1.36$ kg | $l_t = 0.635$ m | $C_{x1} = 1.0$ Ns/m |
| $I_{11} = 0.137$ kg m$^2$ | $l_r = 0$ m | $C_{x2} = 1.0$ Ns/m |
| $I_{22} = 0.221$ kg m$^2$ | $K_m = 0.0178$ N m/N | $C_{x3} = 1.0$ Ns/m |
| $I_{22} = 0.0323$ kg m$^2$ | | |

$C$'s are the aerodynamic damping coefficients to be used for calculating the damping force matrix in the helicopter's dynamic equations.

**Table 8.2** PID controller gains for position control

| | | | | | |
|---|---|---|---|---|---|
| $k_{x1P} = 1.0$ | $k_{x2P} = 1.0$ | $k_{x3P} = 1.5$ | $k_{\phi P} = 2.0$ | $k_{\theta P} = 2.0$ | $k_{\psi P} = 2.0$ |
| $k_{x1D} = 1.0$ | $k_{x2D} = 1.0$ | $k_{x3D} = 1.0$ | $k_{\phi D} = 5.0$ | $k_{\theta D} = 5.0$ | $k_{\psi D} = 1.0$ |
| $k_{x1I} = 0.1$ | $k_{x2I} = 0.1$ | $k_{x3I} = 0.1$ | $k_{\phi I} = 0.1$ | $k_{\theta I} = 0.1$ | $k_{\psi I} = 1.0$ |

*Solution.* The selected controller gains are shown in Table 8.2. These gains have been tuned by trial and error, since the PID tuning rules are not easy to implement to the nonlinear dynamic model of the helicopter. The simulations results are discussed below. The response of the position components of the helicopter are shown in Fig 8.3. This figure shows that all of the position components are stabilized at their corresponding desired positions by the 30th second of the simulation. Because of the introduction of the integral term in the control laws, there is no steady-state error.

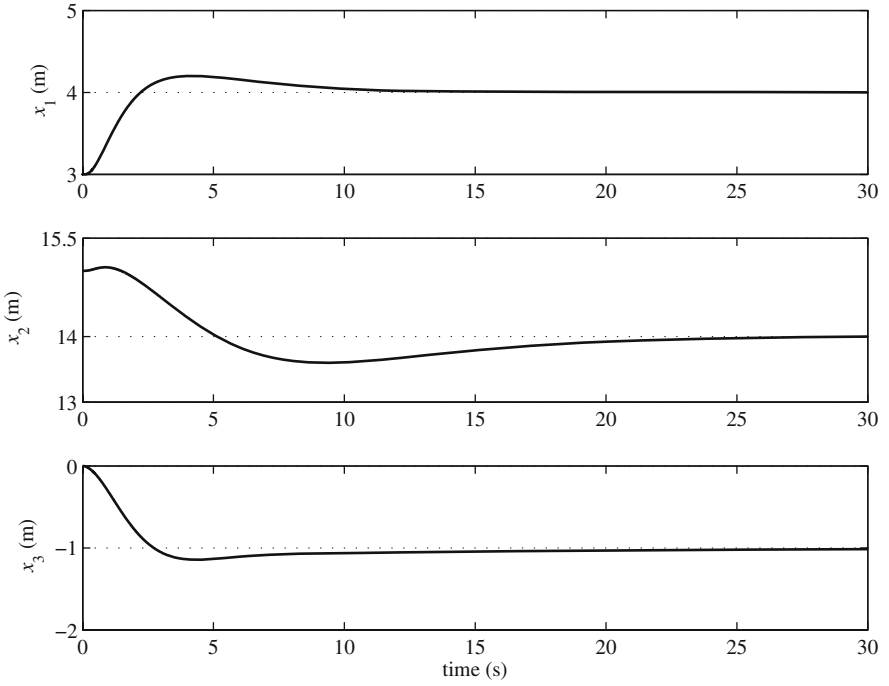


**Fig. 8.3** The time response of the position components

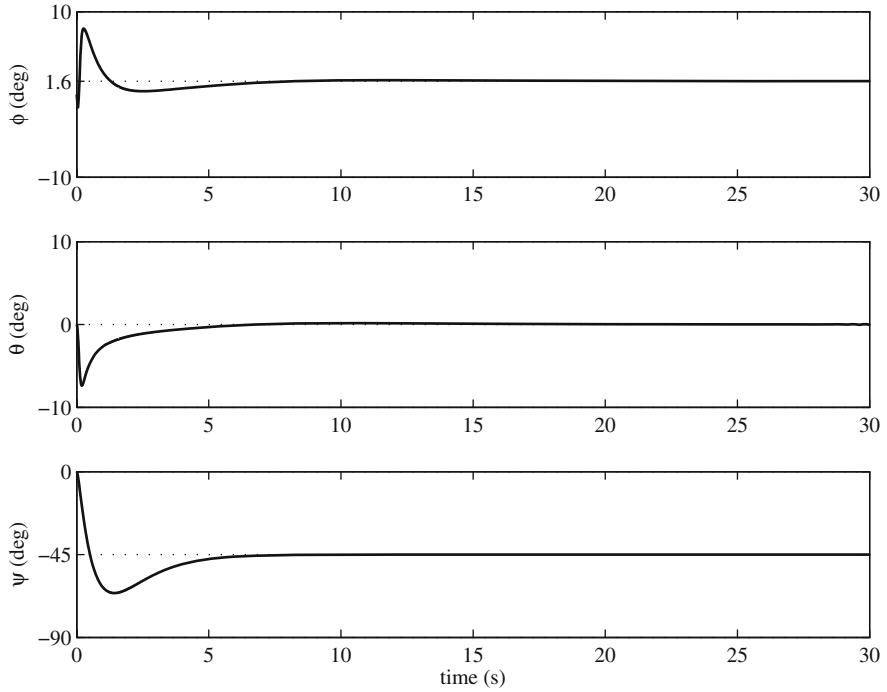


**Fig. 8.4** The time response of the Euler angles

The time history of the Euler angles as representatives of the altitude of the helicopter are shown in Fig. 8.4. In order for the helicopter to move forward and to the right toward the desired position, the roll and pitch angles gain an initial positive and negative values, respectively. These departure angles vanish when the helicopter reaches its desired position. The hover trimming roll and pitch angles can be calculated using Eq. (8.22), which results in $\phi_h \approx 1.6^\circ$ and $\theta_h = 0^\circ$. These values are confirmed by Fig. 8.4. The heading angle has also converged to its desired value without any steady-state error.

Figure 8.5 shows the control effort for this position control scenario. The main rotor thrust starts from a large magnitude to elevate the helicopter for 1 m. This thrust stabilizes at a constant value that corresponds to hovering. This values is 13.34 N, which can also be verified by Eq. (8.22). The roll and pitch moments are initially non zero to generate the required roll and pitch departure angles. These moments vanish at hovering as is suggested by Eq. (8.22). The tail rotor thrust is non zero initially, which causes the change in the helicopter's heading. The steady-state of this thrust is also non zero because it has to counter balance the main rotor's reaction torque. The magnitude of the tail rotor thrust at hovering is consistant with what Eq. (8.22) suggests.

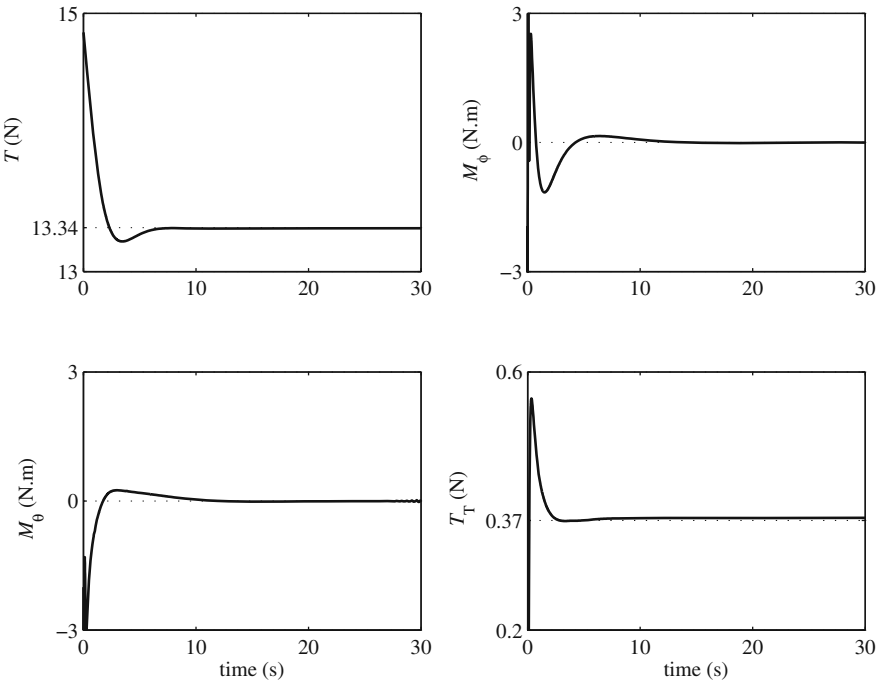


**Fig. 8.5** The time response of the control effort

The spatial path of the helicopter is shown in Fig. 8.6. The overshoot of the helicopter response is easier to see in this figure. This overshoot could have been improved or prevented by fine tuning the controller gains.

## 8.4 The Control Point Concept for Underactuated Vehicles

As was shown previously, the dynamic model of the helicopter has six DOFs, whereas there are only four control inputs available. At the first glance, it may seem impossible to control a 6-DOF system with only four control inputs. In fact, an underactuated system must have some inherent stability to be controllable. For a helicopter, the four control inputs affect the bounce, roll, pitch, and yaw motions directly. The longitudinal and the lateral motion of the helicopter are not directly actuated. However, one can reflect on their real world experience with helicopters and remember that humans can successfully control these vehicles using only four inputs. This can only mean that the unactuated DOFs for a helicopter is inherently stable. In fact, this stability is inherent in the helicopter design, in which the lateral and longitudinal accelerations are a direct function of the roll and pitch angles. Also, the flybar mechanism of the main rotor enhances that stability of the zero dynamics of the system. Using such real world experience with the system, a control engineer

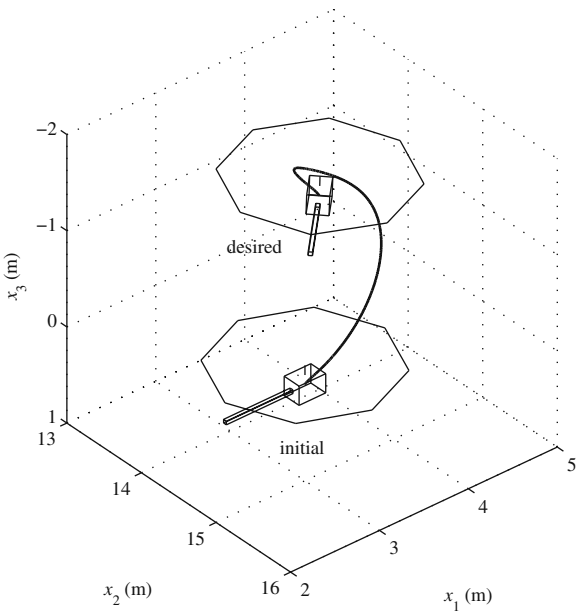


**Fig. 8.6** The spatial path of the helicopter between the initial and the desired positions

can confidently choose a representative of four of the six DOFs of the helicopter as the output of the controller and design a controller to make them track a desired trajectory.

### *8.4.1 The Role of the Control Point*

For an underactuated system, one has to select the most important DOFs to control directly. The number of these DOFs must be equal to the number of control inputs for a simpler controller design. For a helicopter, if the goal is to make the vehicle follow a path 3D (or a trajectory, to be more precise) with a user-specified heading, three position components and the yaw angle of the helicopter are the most important out of the six DOFs. Based on this justification, the simplest choice of the controller position outputs seems to be the three position components of the helicopter's center of mass. However, this simple choice is not the best choice. If the components of the helicopter's center of gravity are used as the controller output, the controller will not sense any disturbance in the remaining DOFs, the roll and pitch. It is never a good idea to make a controller unaware of any states of the system. Controlling the position components of a point on the helicopter's body, except the center of mass, can solve this problem. This point is called the "control point." The postion of this point is a function of all the DOFs of the helicopter. When the control point's position components are chosen as the controller output, a disturbance generated by an external source only in the roll and pitch orientation of

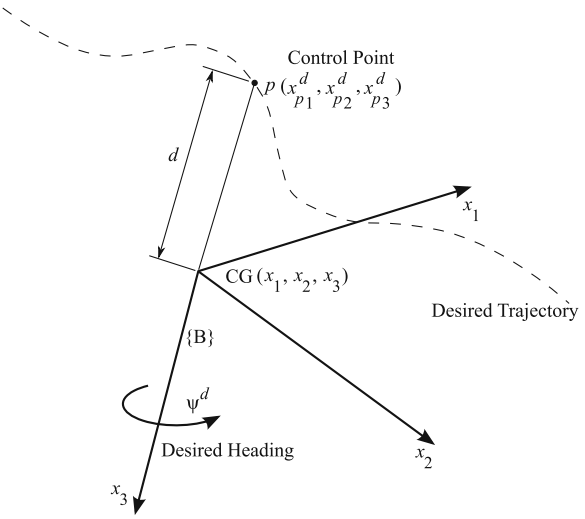


**Fig. 8.7** The control point $p$ and the desired trajectory defined for the control point

the helicopter will also disturb the control point's position, to which the controller can react. This reaction would not happen if the postion of the center of gravity was chosen as the controller output. Let us pick the control point $p$ on the negative local yaw axis of the helicopter with a distance $d$ from the helicopter's center of gravity as shown in Fig. 8.7. This choice simplifies the relations between the inputs and outputs of the controller. The controller outputs are defined as

$$\mathbf{z} = [x_{p1}, x_{p2}, x_{p3}, \psi]^T. \tag{8.42}$$

The desired values of these outputs (the trajectory of the helicopter) is defined by functions of time.

## 8.5 Robust Trajectory-Tracking Control for Autonomous Helicopters

In this section, a robust trajectory-tracking controller is designed based on the sliding mode control method. This controller must determine the four inputs $\mathbf{u} = [T, M_\phi, M_\theta, T_T]^T$, such that the outputs defined in Eq. (8.42) follow a given desired trajectory $\mathbf{z}^d$. First, an input–output equation relating the second-order dynamics of $\mathbf{z}$ to the control inputs $\mathbf{u}$ is derived. Then, this input–output equation is used to design a robust sliding mode controller.

### 8.5.1 The Input–Output Equations

The components of the inertial position of the control point $p$ are among the controller outputs. The dynamics of these components can be found by formulating the acceleration vector of the control point in terms of the acceleration of the helicopter's center of gravity.

$$\ddot{\mathbf{x}}_p = \ddot{\mathbf{x}} + \mathbf{R}_{0B}(\boldsymbol{\omega}^B \times (\boldsymbol{\omega}^B \times \mathbf{d}^B) + \dot{\boldsymbol{\omega}}^B \times \mathbf{d}^B), \tag{8.43}$$

where

$$\mathbf{d}^B = [0, 0, -d]^T \tag{8.44}$$

is the position vector of the control point with respect to the helicopter's center of gravity and $d > 0$. $\ddot{\mathbf{x}}$ is the inertial acceleration of the helicopter's center of mass.

The outputs must be realted to the inputs. Therefore, $\ddot{\mathbf{x}}$ and $\dot{\boldsymbol{\omega}}^B$ are substituted from Eqs. (8.12) and (8.13). Equation (8.43) becomes

$$\begin{aligned}\ddot{\mathbf{x}}_p = {} & \mathbf{R}_{0B}(m^{-1}\mathbf{T}^B + \mathbf{I}^{-1}(\mathbf{M}^B \times \mathbf{d}^B)) \\ & + m^{-1}(\mathbf{R}_{0B}\mathbf{D}_B + \mathbf{W}) + \mathbf{R}_{0B}(\boldsymbol{\omega}^B \times (\boldsymbol{\omega}^B \times \mathbf{d}^B) - \mathbf{I}^{-1}(\boldsymbol{\omega}^B \times \mathbf{I}\boldsymbol{\omega}^B)).\end{aligned} \tag{8.45}$$

In order to write Eq. (8.45) in terms of the input vector $\mathbf{u} = [T, M_\phi, M_\theta, T_T]^T$, the terms $\mathbf{T}^B$ and $\mathbf{M}^B$ must be expanded in terms of $\mathbf{u}$.

$$\mathbf{T}^B = \begin{bmatrix} 0 & 0\,0 & 0 \\ 0 & 0\,0 & -1 \\ -1 & 0\,0 & 0 \end{bmatrix} \begin{bmatrix} T \\ M_\phi \\ M_\theta \\ T_T \end{bmatrix} = \mathbf{T}_u \mathbf{u}, \tag{8.46}$$

and

$$\begin{aligned}\mathbf{M}^B \times \mathbf{d}^B &= (M_1^B \hat{\mathbf{i}} + M_2^B \hat{\mathbf{j}} + M_3^B \hat{\mathbf{k}}) \times (-d\hat{\mathbf{k}}), \\ &= M_1^B d\hat{\mathbf{j}} - M_2^B d\hat{\mathbf{i}}, \\ &= \begin{bmatrix} -(M_\theta + T l_r)d \\ M_\phi d \\ 0 \end{bmatrix}, \\ &= \begin{bmatrix} -l_r d & 0 & -d & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} T \\ M_\phi \\ M_\theta \\ T_T \end{bmatrix}, \\ &= \mathbf{M}_u \mathbf{u}.\end{aligned} \tag{8.47}$$

Substituting Eqs. (8.46) and (8.47) into Eq. (8.45) and rearranging yields

$$\ddot{\mathbf{x}}_p = \mathbf{R}_{0B}(m^{-1}\mathbf{T}_u + \mathbf{I}^{-1}\mathbf{M}_u)\mathbf{u} + m^{-1}(\mathbf{R}_{0B}\mathbf{D}_B + \mathbf{W}) + \mathbf{R}_{0B}(\boldsymbol{\omega}^B \times (\boldsymbol{\omega}^B \times \mathbf{d}^B) - \mathbf{I}^{-1}(\boldsymbol{\omega}^B \times \mathbf{I}\boldsymbol{\omega}^B)). \quad (8.48)$$

Equation (8.48) can be written in a standard short form as

$$\ddot{\mathbf{x}}_p = \mathbf{f}_1 + \mathbf{b}_1\mathbf{u}, \quad (8.49)$$

where

$$\mathbf{f}_1 = m^{-1}(\mathbf{R}_{0B}\mathbf{D}_B + \mathbf{W}) + \mathbf{R}_{0B}(\boldsymbol{\omega}^B \times (\boldsymbol{\omega}^B \times \mathbf{d}^B) - \mathbf{I}^{-1}(\boldsymbol{\omega}^B \times \mathbf{I}\boldsymbol{\omega}^B)),$$
$$\mathbf{b}_1 = \mathbf{R}_{0B}(m^{-1}\mathbf{T}_u + \mathbf{I}^{-1}\mathbf{M}_u). \quad (8.50)$$

Note that Eq. (8.50) describes the dynamic behavior of only three components of the controller output $\mathbf{z} = [x_{p1}, x_{p2}, x_{p3}, \psi]^T$. The dynamic behavior of the last component of $\psi$ can be augmented to Eq. (8.50) to give the complete input–output description of the control system. The dynamics of $\psi$ can be derived from Eq. (8.4).

The last three equations in Eqs. (8.4) are

$$\begin{bmatrix} \omega_1^B \\ \omega_2^B \\ \omega_3^B \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}. \quad (8.51)$$

Solving this equation for $\dot{\psi}$ results in

$$\dot{\psi} = \sin\phi\sec\theta\omega_2^B + \cos\phi\sec\theta\omega_3^B. \quad (8.52)$$

Differentiating Eq. (8.52) and substituting for $\dot{\omega}_2^B$ and $\dot{\omega}_3^B$ from Eq. (8.13) and for $\tau_m$ from Eq. (8.11) in the results yield

$$\ddot{\psi} = f_2 + \mathbf{b}_2\mathbf{u}, \quad (8.53)$$

where

$$f_2 = (\dot{\phi}\cos\phi\sec\theta + \dot{\theta}\sin\phi\sec\theta\tan\theta)\omega_2^B + (-\dot{\phi}\sin\phi\sec\theta + \dot{\theta}\cos\phi\sec\theta\tan\theta)\omega_3^B + (\sin\phi\sec\theta)((I_{33} - I_{11})\omega_1^B\omega_3^B/I_{22}) + (\cos\phi\sec\theta)((I_{11} - I_{22})\omega_1^B\omega_2^B/I_{33}), \quad (8.54)$$

and

$$\mathbf{b}_2 = \left[\frac{\sin\phi\sec\theta l_r}{I_{22}} \; -\frac{\cos\phi\sec\theta K_m}{I_{33}} \; 0 \; \frac{\sin\phi\sec\theta}{I_{22}} \; \frac{\cos\phi\sec\theta l_r}{I_{33}}\right]. \quad (8.55)$$

Finally, Eqs. (8.50) and (8.53) are combined to give the final form of the input–output equations.

$$\begin{bmatrix} \ddot{\mathbf{x}}_p \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ f_2 \end{bmatrix}_{(4\times 1)} + \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}_{(4\times 4)} \mathbf{u}, \tag{8.56}$$

or in a more concise form

$$\ddot{\mathbf{z}} = \mathbf{f} + \mathbf{b}\mathbf{u}. \tag{8.57}$$

The standard form of Eq. (8.57) represents the relation between four outputs $\mathbf{z}$ and four inputs $\mathbf{u}$ of the helicopter as a control system. By taking advantage of the control point concept and the wise definition of the control point, this input–output system takes the simple form of Eq. (8.57), which represents a square system in which the number of inputs and outputs are equal. This simple form makes the application of many control theories possible for controlling an underactuated helicopter with a complex dynamics. Note that this simple form can still be preserved even if the aerodynamic model of the helicopter's main and tail rotors are also considered.

In the next section, the standard form is used for designing a sliding mode controller for robust trajectory tracking.

## *8.5.2 Robust Control Using the Sliding Mode Method*

In sliding mode control method, the desired dynamic behavior of the system error is defined as

$$\dot{\tilde{\mathbf{z}}} + \Lambda\tilde{\mathbf{z}} = \mathbf{s}, \tag{8.58}$$

where

$$\tilde{\mathbf{z}} = \mathbf{z} - \mathbf{z}^d. \tag{8.59}$$

and $\Lambda$ is a diagonal matrix with positive numbers on the diagonal and $\mathbf{s}$ is called the surface parameter. When $\mathbf{s}$ is zero, Eq. (8.58) is asymptotically stable. A sliding mode control law must achieve two goals. First, the control law must guarantee that the error behavior follows the surface to the equilibrium point. Second, the control law must guarantee that the surface variable $\mathbf{s}$ approaches zero and stays zero. The first goal is met by deriving $\hat{\mathbf{u}}$ called the equivalent control, while the second goal is reached by adding a discontinuous term to the equivalent control.

### 8.5.2.1 Equivalent Control

The input–output equations for the nominal parameters of the helicopter are assumed.

$$\ddot{\mathbf{z}} = \hat{\mathbf{f}} + \hat{\mathbf{b}}\mathbf{u}. \tag{8.60}$$

Interestingly enough, the input–output form (8.57) is similar to that of the surface vessel trajectory-tracking problem introduced in Section 7.5.1 (Eq. (7.29)). Since the form of the input–output relations (7.29) and (8.57) are similar, the same procedure as presented in Section 7.5.3.1 for surface vessels can be used to find the equivalent control for the helicopter trajectory-tracking control law, which results in a similar form to Eq. (7.48).

$$\hat{\mathbf{u}} = \hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r), \tag{8.61}$$

where

$$\dot{\mathbf{s}}_r = \ddot{\mathbf{z}}^d - \Lambda\dot{\tilde{\mathbf{z}}}. \tag{8.62}$$

#### 8.5.2.2 Robust Control Law

To complement the first part of the sliding mode controller, a second part is needed to guarantee that the surface offset parameter $\mathbf{s}$ approaches zero regardless of the error initial condition and uncertainty in the model parameters. This is done by adding a discontinuous term to the equivalent control.

$$\mathbf{u} = \hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})), \tag{8.63}$$

where $\mathbf{K}$ is a positive-definite diagonal matrix and sgn($\mathbf{s}$) returns a vector with the sign of the components of $\mathbf{s}$. Although the control law (7.49) seems complete, the discontinuity gain $\mathbf{K}$ must still be determined such that the surface offset parameter $\mathbf{s}$ converges to zero despite uncertainties in the dynamic model parameters.

A Lyapunov function is defined as

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{s}. \tag{8.64}$$

This Lyapunov function is admissible because it satisfies all the required properties of such a function. It is positive for all values of $\mathbf{s}$ and is only zero when $\mathbf{s}$ is identically zero. With these properties, if one can show that the time derivative of the Lyapunov function is always negative and is only zero when $\mathbf{s}$ is identically zero, then, one can conclude that $\mathbf{s}$ converges to zero from any initial condition and remains at zero.

With the same procedure that is used in Section 7.5.3.2 for surface vessels, it can be shown that the rate of the Lyapunov function is

$$\dot{V} = \mathbf{s}^T(\mathbf{f} + \mathbf{b}\hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})) - \dot{\mathbf{s}}_r). \tag{8.65}$$

Note that in the ideal conditions (e.g., when there is no uncertainty and $\mathbf{b} = \hat{\mathbf{b}}$), $\mathbf{b}\hat{\mathbf{b}}^{-1}$ is the identity matrix $\mathbf{I}_m$. Therefore, it is logical to define the uncertainty in $\mathbf{b}$ in terms of the difference of $\mathbf{b}\hat{\mathbf{b}}^{-1}$ and the identity matrix $\mathbf{I}_m$.

$$\boldsymbol{\delta} = \mathbf{b}\hat{\mathbf{b}}^{-1} - \mathbf{I}_m., \tag{8.66}$$

Equation (8.65) is rewritten in terms of the uncertainty $\boldsymbol{\delta}$.

$$\begin{aligned}\dot{V} &= \mathbf{s}^T(\mathbf{f} + (\mathbf{I}_m + \boldsymbol{\delta})(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})) - \dot{\mathbf{s}}_r), \\ &= \mathbf{s}^T(\mathbf{f} - \hat{\mathbf{f}} + \boldsymbol{\delta}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r) - \mathbf{K}\text{sgn}(\mathbf{s}) - \boldsymbol{\delta}\mathbf{K}\text{sgn}(\mathbf{s})).\end{aligned} \tag{8.67}$$

As this stage, some bounds must be assumed for the parameter uncertainties. These bounds are defined for the components of $\mathbf{f} - \hat{\mathbf{f}}$ and $\boldsymbol{\delta}$.

$$|f_i - \hat{f}_i| \leq F_i, \quad |\delta_{ij}| \leq \Delta_{ij}, \qquad i = 1, \ldots, 4. \tag{8.68}$$

If $\mathbf{f} - \hat{\mathbf{f}}$ and $\boldsymbol{\delta}$ in Eq. (8.67) are replaced by the uncertainty bounds defined in Eq. (8.68), the right hand side of the resulting equation increases in value. Hence, the equal sign should be replaced by an inequality sign.

$$\dot{V} \leq \mathbf{s}^T(\mathbf{F} + \Delta| - \hat{\mathbf{f}} + \dot{\mathbf{s}}_r| - \mathbf{K}\text{sgn}(\mathbf{s}) + \Delta\mathbf{K}\text{sgn}(\mathbf{s})). \tag{8.69}$$

Equation (8.69) in component notation becomes

$$\begin{aligned}\dot{V} &\leq \sum_{i=1}^{4} s_i(F_i + \sum_{j=1}^{4}(\Delta_{ij}| - \hat{f}_j + \dot{s}_{rj}|) - K_i\text{sgn}(s_i) + \sum_{j=1}^{4}\Delta_{ij}K_j\text{sgn}(s_j)), \\ &\leq -\sum_{i=1}^{4}|s_i|(-F_i - \sum_{j=1}^{4}(\Delta_{ij}| - \hat{f}_i + \dot{s}_{ri}|) + K_i - \sum_{j=1}^{4}\Delta_{ij}K_j).\end{aligned} \tag{8.70}$$

Equation (8.70) implies that if $K_i$'s are found such that

$$-F_i - \sum_{j=1}^{4}(\Delta_{ij}| - \hat{f}_i + \dot{s}_{ri}|) + K_i - \sum_{j=1}^{4}\Delta_{ij}K_j = \eta_i, \quad i = 1, \ldots, 4, \tag{8.71}$$

where $\eta_i$ are positive numbers, then, the rate of the Lyapunov function becomes

$$\dot{V} \leq -\sum_{i=1}^{4}|s_i|\eta_i. \tag{8.72}$$

In other words, if $K_i$'s are determined from Eq. (8.71), the rate of the Lyapunov function is always negative, except when all $s_i$'s are zero, at which point the rate of the Lyapunov function is zero. This implies that the Lyapunov function decreases to zero despite of any initial value and remains at zero. Consequently, $s_i$'s approach zero and remain zero, because $V$ is zero if and only if all $s_i$'s are zero.

For simplicity of implementation, Eq. (8.71) is written in the matrix form.

$$\mathbf{K}_v = (\mathbf{I}_m - \Delta)^{-1}(\mathbf{F} + \Delta|\,-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r| + \boldsymbol{\eta}), \tag{8.73}$$

where $\mathbf{K}_v = [K_1, K_2, K_3, K_4]^T$ and $\boldsymbol{\eta} = [\eta_1, \eta_2, \eta_3, \eta_4]^T$.

*Example 8.2.* Consider the autonomous helicopter introduced in Example 8.1, whose inertial and geometrical properties are shown in Table 8.1. Assume that the distance of the control point with the helicopter's center of gravity $d = 1$ m. Use the sliding mode control law derived in this section to control the control point of the helicopter on the following desired trajectory:

$$\begin{aligned} x_{p1}^d(t) &= R\cos\frac{2\pi t}{\tau}, \\ x_{p2}^d(t) &= R\sin\frac{2\pi t}{\tau}, \\ x_{p3}^d(t) &= -H\sin\frac{4\pi t}{\tau}, \\ \psi^d(t) &= \frac{4\pi t}{\tau} + \frac{\pi}{2}, \end{aligned} \tag{8.74}$$

where $R = 10$ m, $H = 2$ m, and $\tau = 60$ s. Note that the projection of this 3D desired trajectory on the $x_1 - x_2$ plane is a circle with a radius $R$. It takes $\tau$ seconds for the helicopter to complete this circle. The altitude of the helicopter starts at zero, increases $H$ meters in $\tau/4$ seconds, returns back to zero altitude in the next $\tau/4$ seconds, decreases $H$ meters in the third $\tau/4$ seconds, and once again returns to zero altitude in the last $\tau/4$ seconds.

The helicopter is at rest at time zero and its initial position and orientation are as follows.

$$x_1(0) = 10 \text{ m}, \quad x_2(0) = 0 \text{ m}, \quad x_3(0) = 1 \text{ m}, \tag{8.75}$$

$$\phi(0) = 0 \text{ rad}, \quad \theta(0) = 0 \text{ rad}, \quad \psi(0) = \pi/2 \text{ rad}. \tag{8.76}$$

Assume that the inertial properties of the helicopter can be uncertain up to 1.3 times of the nominal properties. This uncertainty can cover the pay load of the helicopter. Simulate the motion of the helicopter under control for three cases of uncertainty in the inertial properties of the helicopter. Assume that the actual inertial properties are

(a) 0.8 times the nominal values (−20% uncertainty),
(b) equal the nominal values (0% uncertainty),
(c) and 1.2 times the nominal values (20% uncertainty).

*Solution.* The first- and second-order derivatives of the desired trajectory is required for calculating the sliding mode surfaces in the sliding mode control method. These derivatives are calculated by differentiating Eq. (8.74) versus time.

To calculate the discontinuity gain $\mathbf{K}_v$ from Eq. (8.73), the uncertainty bounds $\mathbf{F}$ and $\Delta$ must be determined. These bounds are determined from Eq. (8.68), in which $\hat{\mathbf{f}}$ and $\hat{\mathbf{b}}$ are evaluated using the helicopter's nominal parameters listed in Table 8.1, while $\mathbf{f}$ and $\mathbf{b}$ are evaluated based on the values of 1.3 times the nominal values.

The controller parameters are selected as follows.

$$\Lambda = \begin{bmatrix} 0.6 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0.6 & 0 \\ 0 & 0 & 0 & 0.6 \end{bmatrix}, \tag{8.77}$$

$$\boldsymbol{\eta} = \begin{bmatrix} 5 & 5 & 5 & 5 \end{bmatrix}^T, \tag{8.78}$$

$$\Phi = \begin{bmatrix} 0.05 & 0.05 & 0.05 & 0.05 \end{bmatrix}^T. \tag{8.79}$$

The above parameters and the discontinuity gains mentioned above are used to calculate the surface parameter $\mathbf{s}$ from Eq. (8.58) and the auxiliary parameter $\dot{\mathbf{s}}_r$ from Eq. (8.62). Finally, the control law is evaluated using Eq. (8.63). The control input evaluated from the control law is applied to the dynamic model (8.15). The simulations are run three times, in which the actual inertial properties defined in (a) to (b) are used.

Figure 8.8 shows the output of the controller for the three cases of uncertainty and the desired profile for the outputs. The difference between the behavior of the helicopters for the three cases of uncertainty is so small that the curves representing them seem to completely overlap. The response curves also overlap the desired curve very closely. This means that the controller has a good performance with minimal errors and is very robust to parameter uncertainty.

The control effort is shown in Fig. 8.9. The magnitude of all the control forces and moments are reasonable and achievable by the modeled helicopter. It is seen that the magnitude of the main rotor thrust is automatically adjusted by the controller to compensate for less or more actual mass. The tail rotor thrust, which balances the reaction torque of the main rotor on the helicopter body, is also smartly decreased or increased by the controller according to the magnitude of the main rotor. The roll and pitch moments for the three different cases are very close to zero and no variations can be seen in their corresponding figures. This is because the roll and pitch acceleration for the desired motion are negligible.

Since the helicopter is an underactuated dynamic system, observing the controlled outputs are not enough for performance verifications. While the response of the outputs are as expected due to the effect of a well-designed controller, the states of the helicopter could have oscillatory responses. For example, for the presented control strategy, which is based on the control point, the control point could perform the desired trajectory while the roll and pitch angle responses could be oscillatory.

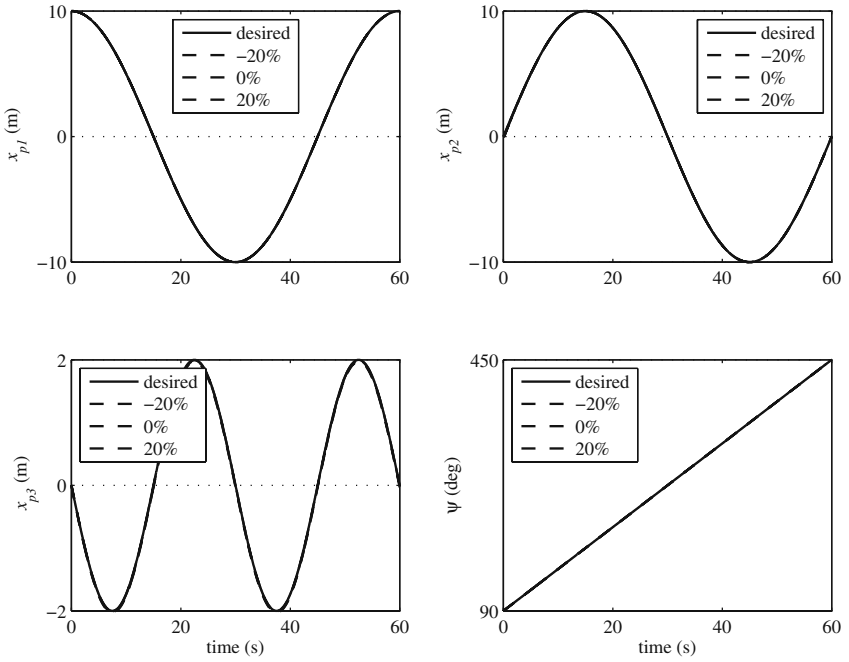


**Fig. 8.8** The controller outputs, the control point position components and the yaw of the helicopter

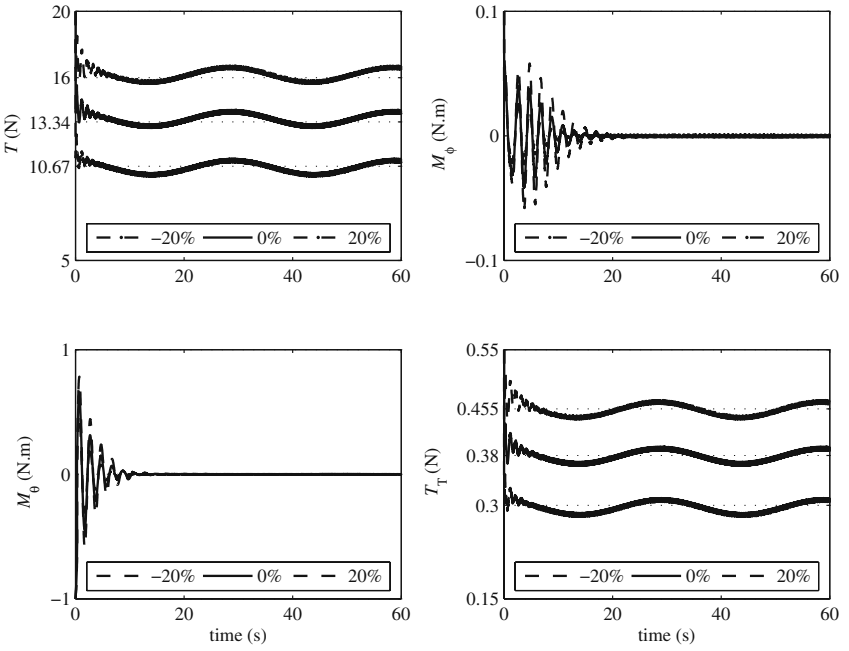


**Fig. 8.9** The controller inputs, the main rotor thrust and moments and the tail rotor thrust

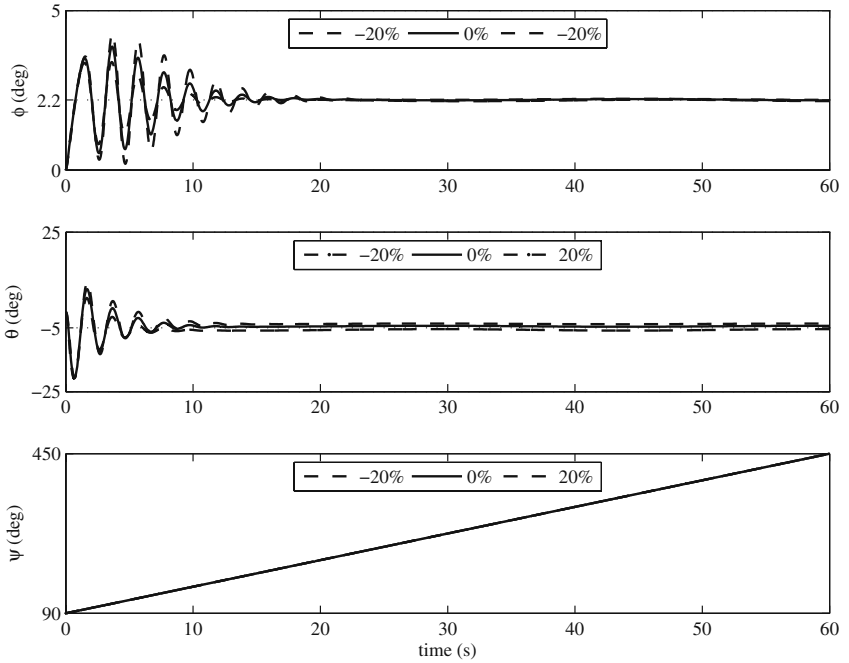


**Fig. 8.10** The helicopter's roll, pitch, and yaw Euler angles

This would not be desirable. To check if such oscillations have happened for the simulated scenario, one has to check the response of the states of the helicopter.

Figure 8.10 shows the the roll, pitch, and yaw Euler angles of the helicopter. The roll and pitch components are stabilized at constant values after a transient response. The transient response is due to the fact that the helicopter is initially at rest and is trying to catch up with the desired velocity dictated by the desired trajectory. The responses of the helicopter's roll and pitch angles are different for the three difference uncertainty cases in the transient portion because of the effect of rotational inertia in dynamic situations. The difference in responses are negligible when the roll and pitch angles approach their constant equilibrium.

Figure 8.11 shows the position of the center of mass of the helicopter. Since the center of gravity of the helicopter has an offset with the control point, its motion is directly related to the control point motion and the roll and pitch responses. In this example, the roll and pitch angles are approximately constant after a transient response, therefore, the response of the postion components of the helicopter's center of mass are the offset of that of the control point. It can be concluded from Fig. 8.11 that the center of mass of the helicopter has a smooth motion. The projection of this motion on the $x_1 - x_2$ plane is a circle with an approximate radius of 10 m. The change in altitude of the helicopter's center of mass for the whole 60-s motion is approximately 4 m, which is in agreement with that of the desired trajectory of the

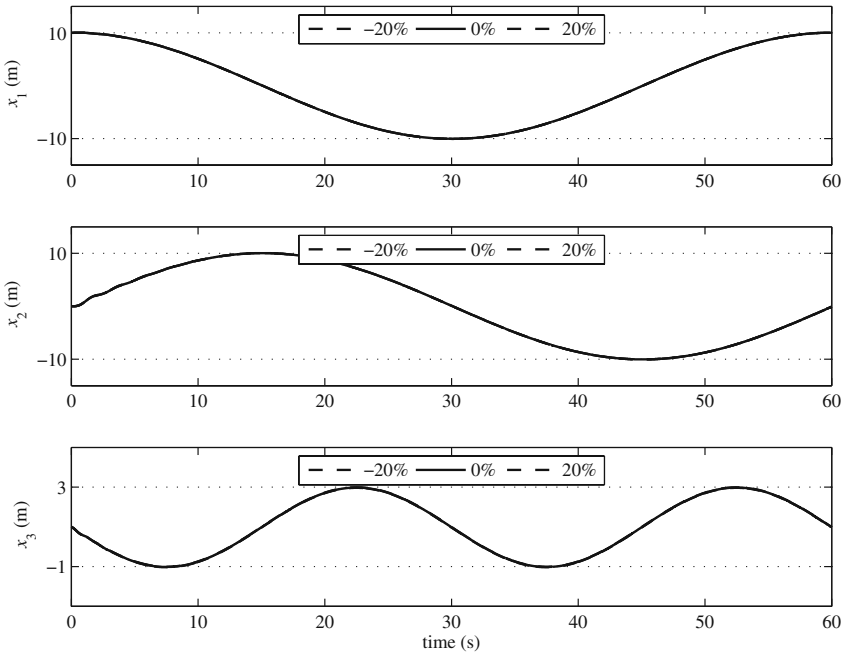


**Fig. 8.11** The helicopter's center of mass position components

control point. The 1-m offset that is observed in $x_3$ compared to $x_{p3}$ in Fig. 8.8 is due to the distance of the control point with the center of mass ($d = 1$ m).

Figure 8.12 shows the path of the control point of the helicopter for the three cases of uncertainty. As expected from the small differences in the output responses for the three uncertainty cases in Fig. 8.8, the path of the control point for the three cases are so close that they seem to overlap one another. This, once a gain, shows the high degree of robustness of the controller to parameter uncertainty. The helicopter icons on this path is shown at times 0, 15, 30, 45, and 60 s, respectively. The helicopter icons for time 0–60 s are coincident, indicating that the helicopter has successfully finished the desired trajectory on time.

## 8.6 Leader-Follower Formation Control for Autonomous Helicopters

Recently, the research in the area of autonomous aerial vehicles has moved beyond considering a single vehicle.[3] Researchers have considered aerial pursuit/evasion games in three dimensions on a fixed wing aircraft by implementing and testing a

[3] Some of the material of this section has been adapted from the article titled: "Full formation control for autonomous helicopter groups," by Farbod Fahimi, Robotica (2008) volume 26, pp. 143–156. Copyright 2008 Cambridge University Press, UK.

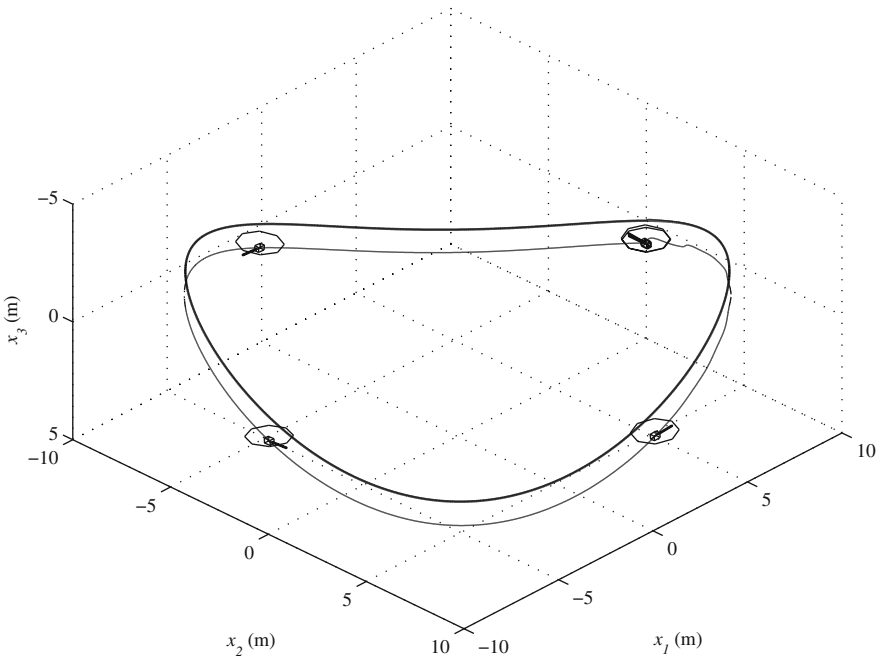


**Fig. 8.12** The path of the helicopter's center of mass and control point

Nonlinear Model Predictive tracking controller [28]. The pursuit/evasion games has been extended to heterogeneous teams of autonomous agents, in which the problem of having a team of agents pursue a second team of evader while building a map of the environment has been considered [77]. Another aspect of formation control is formation planning. In formation planning, the initial and final configurations are given for a group of autonomous vehicles and the nominal input trajectory for each vehicle is determined such that the group can start from the initial configuration and reach their final configuration at a specified time [83]. Another approach to formation control is to introduce carefully designed inter-agent coupling terms in each performance index of a Nonlinear Model Predictive controller for the vehicles [18].

The objective of the current paper is to introduce a new approach for formation control of autonomous helicopters. Formation control of helicopters shares the same challenges with that of other types of vehicles. Need for decentralized controllers, minimum communication, and scalability are among these challenges. The nonlinear dynamics, parameter and model uncertainty, and disturbances add to the common formation control problem difficulties. The current paper contributes to the low-level formation control design for autonomous helicopters, while addressing these difficulties.

The problem of control and coordination for small helicopters moving in a formation is investigated by introducing a leader-follower approach. The overall motion plan for a single virtual lead helicopter is assumed. This motion plan defines the gross motion of the formation. A 6-DOF dynamic model of the helicopters is

considered for designing the controllers. It is assumed that four independent actuators control the four control inputs: the main and the tail rotor thrust, and the roll and pitch moments. Two nonlinear decentralized control schemes are required to define a unique 3D formation. In the first scheme, one helicopter controls its relative distance and orientation with respect to a neighboring helicopter. In the second scheme, a helicopter maintains its position in the formation by maintaining specified distances from two neighboring helicopters.

The proposed control schemes only use the state information of the neighboring helicopters. The sliding mode method is used. It is shown that the relative distances and orientations of the helicopters are stabilized even in the presence of wind disturbance. Numerical simulations are presented to demonstrate the efficiency of these techniques.

### *8.6.1 Formation Control Schemes*

The bulk motion of the group of helicopters can be characterized by trajectory planning and obstacle avoidance algorithms, for example, the method of artificial potential fields. It is assumed that a virtual helicopter as a group leader adapts the bulk motion of the group as its planned trajectory. Other helicopters of the group follow either the virtual group leader or their neighboring helicopters. Therefore, our attention is focused on controlling the internal geometry of the formation. Two types of feedback controllers are introduced for controlling the internal geometry.

The first feedback controller is called the $l - \alpha$ controller. It controls the relative distance and view angle of a helicopter with respect to a neighboring helicopter. This controller is used for helicopters marching in a single file (for example, in a line formation) or at an edge of the formation geometry. Note that when the $l - \alpha$ controller is used, a follower can only be related to one leader, which may not be very safe for the formations in which each helicopter is surrounded by more than one helicopter (for example, a rectangular formation). In these situations, forming a triangular formation mesh (Fig. 8.13) is desirable. The mesh generates a more

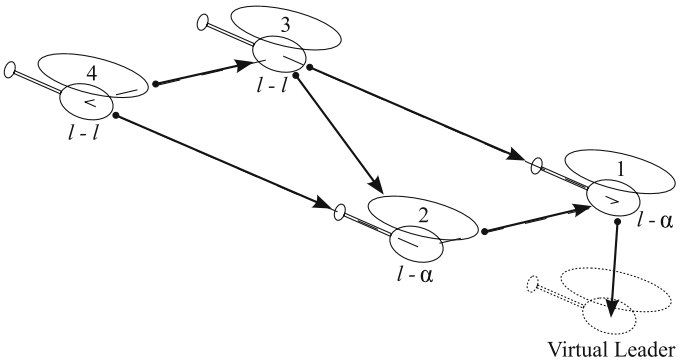


**Fig. 8.13** General formation control configuration

dense interconnection between the helicopters, which is safer and more robust. To complete a triangular formation mesh, a second controller is needed to control the 3D distances of the helicopter from two neighboring helicopters. This controller is called the $l - l$ controller.

These two local control schemes can be used to define a solid general formation (Fig. 8.13). Usually, the helicopters at an edge of the formation geometry control their distance with their immediate front helicopter using the $l - \alpha$ controller. The other helicopters control their distances to their immediate front and side helicopters using the $l - l$ controller. This is necessary so that a helicopter can also avoid its side helicopter.

The sliding mode control method is used for deriving low-level control laws for each of the mentioned schemes. Designing a sliding mode control law requires the input–output description of the control system, whereas the equations of motion of the helicopter has been written in state-space form in the previous section. In the next two subsections, the control outputs of the two formation control schemes are defined and the input–output descriptions are derived for the two control system.

### 8.6.1.1 Input–Output Description for the $l - \alpha$ Control Scheme

In Fig. 8.14, a system of two neighboring helicopters in the formation is shown. The helicopters are separated by a vectorial distance $\mathbf{l}_{12} + \mathbf{z}_{12}$ between an arbitrary control point, $p_1$, on helicopter 1 (the leader) and the control point, $p_2$, on helicopter 2 (the follower). The control point has a fixed distance $d$ with the helicopter center of mass along the negative $z$ direction of the helicopter's body frame. Note that the helicopters are not physically coupled in any way. A feedback control law for control inputs $\mathbf{u} = [T, M_\phi, M_\theta, T_T]^T$ must be determined to control helicopter 2 such that the desired distance $l_{12}^d$, view angle $\alpha_{12}^d$, height offset $z_{12}^d$, all defined in body frame {1}, to helicopter 1 are maintained, while the yaw angle of helicopter

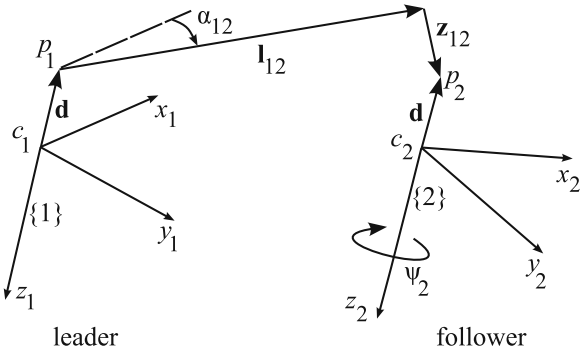


**Fig. 8.14** $l - \alpha$ control configuration. Frames 1 and 2 correspond to the leader and the follower respectively. The helicopters' centres of mass are denoted by $c$ and their control points are denoted by $p$

2, $\psi_2^d$, follows the yaw angle of helicopter 1. Therefore, the outputs of the control system are $\mathbf{y} = [l_{12}, \alpha_{12}, z_{12}, \psi_2]^T$.

Here, the input–output description of the control system are derived, which relate the output $\mathbf{y}$ to the input $\mathbf{u}$ directly. First, through a kinematic analysis, the state variables of helicopter 2, which are

$$\begin{aligned}\mathbf{q}_2 = [&x_2^{(0)} \;\; y_2^{(0)} \;\; z_2^{(0)} \;\; \dot{x}_2^{(0)} \;\; \dot{y}_2^{(0)} \;\; \dot{z}_2^{(0)} \\ &\phi_2 \;\; \theta_2 \;\; \psi_2 \;\; \omega_{2x}^{(2)} \;\; \omega_{2y}^{(2)} \;\; \omega_{2z}^{(2)}]^T,\end{aligned} \tag{8.80}$$

are related to the output $\mathbf{y}$. Then, the equations of motion, containing $\mathbf{u}$ are substituted in the resulting equations to give the input–output relations. The details follow.

**Kinematic Analysis**

Let us consider the moving body frames of helicopters 1 and 2 (Fig. 8.14). We assume two coincident points; point $p_2'$, attached to frame {1}, and point $p_2$, attached to frame {2}, both coincident with the instantaneous location of the follower's control point. If $\mathbf{v}_{p2/1}^{(1)}$ and $\mathbf{a}_{p2/1}^{(1)}$ are the apparent velocity and acceleration of the point $p_2$ as seen by an observer at point $p_2'$ attached to frame {1} expressed in {1}, one can write

$$\mathbf{v}_{p2/1}^{(1)} = (\dot{\mathbf{l}}_{12} + \mathbf{z}\dot{\mathbf{i}}_{12}), \tag{8.81}$$

$$\mathbf{a}_{p2/1}^{(1)} = (\ddot{\mathbf{l}}_{12} + \ddot{\mathbf{z}}_{12}). \tag{8.82}$$

If $\mathbf{a}_{p2/1}^{c(1)}$ is the Coriolis acceleration of the point $p_2$ as seen by an observer at point $p_2'$ expressed in {1}, one can write

$$\mathbf{a}_{p2}^{(1)} = \mathbf{a}_{p'2}^{(1)} + \mathbf{a}_{p2/1}^{c(1)} + \mathbf{a}_{p2/1}^{(1)}, \tag{8.83}$$

where

$$\mathbf{a}_{p'2}^{(1)} = \mathbf{a}_{p1}^{(1)} + \dot{\boldsymbol{\omega}}_1^{(1)} \times (\mathbf{l}_{12} + \mathbf{z}_{12}) + \boldsymbol{\omega}_1^{(1)} \times (\boldsymbol{\omega}_1^{(1)} \times (\mathbf{l}_{12} + \mathbf{z}_{12})), \tag{8.84}$$

$$\mathbf{a}_{p1}^{(1)} = \mathbf{R}_{01}^T(\ddot{x}_1^{(0)}\hat{\mathbf{i}}_0 + \ddot{y}_1^{(0)}\hat{\mathbf{j}}_0 + \ddot{z}_1^{(0)}\hat{\mathbf{k}}_0) + \boldsymbol{\omega}_1^{(1)} \times \mathbf{d}^{(1)}, \tag{8.85}$$

$$\mathbf{a}_{p2/1}^{c(1)} = 2\boldsymbol{\omega}_1^{(1)} \times \mathbf{v}_{p2/1}^{(1)}. \tag{8.86}$$

After combining these relations, the absolute acceleration of the control point $p_2$ becomes

$$\begin{aligned}\mathbf{a}_{p2}^{(1)} = \; &\mathbf{R}_{01}^T\mathbf{a}_{p1}^{(0)} + \dot{\boldsymbol{\omega}}_1^{(1)} \times (\mathbf{l}_{12} + \mathbf{z}_{12}) \\ &+ \boldsymbol{\omega}_1^{(1)} \times (\boldsymbol{\omega}_1^{(1)} \times (\mathbf{l}_{12} + \mathbf{z}_{12})) \\ &+ 2\boldsymbol{\omega}_1^{(1)} \times \mathbf{v}_{p2/1}^{(1)} + (\ddot{\mathbf{l}}_{12} + \ddot{\mathbf{z}}_{12}).\end{aligned} \tag{8.87}$$

On the other hand, the acceleration of the same point $p_2$ can be calculated in terms of the absolute acceleration of the center of mass of helicopter 2 as

$$\mathbf{a}_{p2}^{(1)} = \mathbf{R}_{01}^T \mathbf{R}_{02}(\mathbf{a}_{c2}^{(2)} + \dot{\boldsymbol{\omega}}_2^{(2)} \times \mathbf{d}^{(2)} + \boldsymbol{\omega}_2^{(2)} \times (\boldsymbol{\omega}_2^{(2)} \times \mathbf{d}^{(2)})), \tag{8.88}$$

where

$$\mathbf{a}_{c2}^{(2)} = \mathbf{R}_{02}^T(\ddot{x}_2^{(0)}\hat{\mathbf{i}}_0 + \ddot{y}_2^{(0)}\hat{\mathbf{j}}_0 + \ddot{z}_2^{(0)}\hat{\mathbf{k}}_0), \tag{8.89}$$

$$\mathbf{d}^{(2)} = -d\hat{\mathbf{k}}_2. \tag{8.90}$$

The absolute acceleration of point $p_2$ must be the same, independent of how it is calculated. Therefore, one can equate Eqs. (8.87) and (8.88). By solving the resulting vectorial equation for $\ddot{\mathbf{l}}_{12} + \ddot{\mathbf{z}}_{12}$, one can obtain the following vectorial kinematic equation:

$$(\ddot{\mathbf{l}}_{12} + \mathbf{z}\ddot{\mathbf{1}}_2) = \mathbf{R}_{01}^T[\mathbf{a}_{c2}^{(0)} + \mathbf{R}_{02}(\dot{\boldsymbol{\omega}}_2^{(2)} \times \mathbf{d}^{(2)})] + \mathbf{B}_0, \tag{8.91}$$

where

$$\begin{aligned} \mathbf{B}_0 = \; & \mathbf{R}_{01}^T[-\mathbf{a}_{p1}^{(0)} + \mathbf{R}_{02}(\boldsymbol{\omega}_2^{(2)} \times (\boldsymbol{\omega}_2^{(2)} \times \mathbf{d}^{(2)}))] \\ & - \dot{\boldsymbol{\omega}}_1^{(1)} \times (\mathbf{l}_{12} + \mathbf{z}_{12}) \\ & - \boldsymbol{\omega}_1^{(1)} \times (\boldsymbol{\omega}_1^{(1)} \times (\mathbf{l}_{12} + \mathbf{z}_{12})) \\ & - 2\boldsymbol{\omega}_1^{(1)} \times (\dot{\mathbf{l}}_{12} + \dot{\mathbf{z}}_{12}). \end{aligned} \tag{8.92}$$

Since $\mathbf{l}_{12} + \mathbf{z}_{12} = [l_{12}\,\mathrm{c}\alpha_{12}, l_{12}\,\mathrm{s}\alpha_{12}, z_{12}]^T$, the left hand side of Eq. (8.91) can be expanded as

$$(\ddot{\mathbf{l}}_{12} + \ddot{\mathbf{z}}_{12}) = \mathbf{A}_1\ddot{\mathbf{y}}_1 + \mathbf{B}_1, \tag{8.93}$$

in which

$$\begin{aligned} \mathbf{A}_1 &= \begin{bmatrix} \mathrm{c}\alpha_{12} & -l_{12}\,\mathrm{s}\alpha_{12} & 0 \\ \mathrm{s}\alpha_{12} & l_{12}\,\mathrm{c}\alpha_{12} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{y}_1 = \begin{bmatrix} l_{12} \\ \alpha_{12} \\ z_{12} \end{bmatrix}, \\ \mathbf{B}_1 &= \begin{bmatrix} -2\dot{l}_{12}\dot{\alpha}_{12}\,\mathrm{s}\alpha_{12} - l_{12}\dot{\alpha}_{12}^2\,\mathrm{c}\alpha_{12} \\ 2\dot{l}_{12}\dot{\alpha}_{12}\,\mathrm{c}\alpha_{12} - l_{12}\dot{\alpha}_{12}^2\,\mathrm{s}\alpha_{12} \\ 0 \end{bmatrix}. \end{aligned} \tag{8.94}$$

By combining Eqs. (8.91) and (8.93), one can arrive at an equation that relates a subset of the output vector, $\mathbf{y}_1$, to the state variables of the helicopters:

$$\ddot{\mathbf{y}}_1 = \mathbf{A}_1^{-1}[\mathbf{R}_{01}^T[\mathbf{a}_{c2}^{(0)} + \mathbf{R}_{02}(\dot{\boldsymbol{\omega}}_2^{(2)} \times \mathbf{d}^{(2)})] + \mathbf{B}_0 - \mathbf{B}_1]. \tag{8.95}$$

Note that $\mathbf{A}_1$ is invertible as long as $l_{12} \neq 0$, which can be easily avoided by defining an appropriate desired formation.

### Input–Output Equations

The linear and angular acceleration of helicopter 2, the follower, appear in Eq. (8.95). In this subsection, first, these accelerations are substituted by the dynamics equations of helicopter 2, which include the inputs, to give a subset of the input–output equations. Then, the input–output equations are completed by including the dynamics of the follower's yaw degree of freedom, $\psi_2$.

The translational dynamic equation (8.12) is customized for helicopter 2 by adding an index 2 to the variables and noting that the body frame notation $\{B\}$ is replaced by notion $\{2\}$. The resulting dynamic equation can be rearranged and written in the following matrix form:

$$\mathbf{a}_{c2}^{(0)} = \mathbf{C}_1\mathbf{u} + \mathbf{D}_1 + \mathbf{W}_1, \tag{8.96}$$

where

$$\mathbf{C}_1 = \tfrac{1}{m}\mathbf{R}_{02}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{D}_1 = \tfrac{1}{m}\mathbf{R}_{02}\mathbf{D}^{(2)}, \quad \mathbf{W}_1 = \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}. \tag{8.97}$$

Also, the term $(\dot{\boldsymbol{\omega}}_2^{(2)} \times \mathbf{d}^{(2)})$ is derived by customizing the rotational equation of motion (8.13) for helicopter 2 and calculating the cross product. This results in

$$\dot{\boldsymbol{\omega}}_2^{(2)} \times \mathbf{d}^{(2)} = \mathbf{C}_2\mathbf{u} + \mathbf{D}_2, \tag{8.98}$$

where

$$\mathbf{C}_2 = \begin{bmatrix} -\frac{dl_r}{I_{yy}} & 0 & -\frac{d}{I_{yy}} & 0 \\ 0 & \frac{d}{I_{xx}} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$
$$\mathbf{D}_2 = -[\mathbf{I}^{-1}(\boldsymbol{\omega}_2^{(2)} \times \mathbf{I}\boldsymbol{\omega}_2^{(2)})] \times \mathbf{d}^{(2)}. \tag{8.99}$$

Now, a subset of the required input–output equations can be obtained by substituting Eqs. (8.96) and (8.98) into Eq. (8.95):

$$\ddot{\mathbf{y}}_1 = \mathbf{f}_1 + \mathbf{b}_1\mathbf{u}, \tag{8.100}$$

where

$$\mathbf{f}_1 = \mathbf{A}_1^{-1}[\mathbf{R}_{01}^T(\mathbf{W}_1 + \mathbf{R}_{02}\mathbf{D}_2) + \mathbf{B}_0 - \mathbf{B}_1 + \mathbf{R}_{01}^T\mathbf{D}_1],$$
$$\mathbf{b}_1 = \mathbf{A}_1^{-1}[\mathbf{R}_{01}^T(\mathbf{C}_1 + \mathbf{R}_{02}\mathbf{C}_2)]. \tag{8.101}$$

Note that the yaw angle of helicopter 2 ($\psi_2$) as the last output component is missing from this subset of input–output equations because $\mathbf{y}_1$ only contains the three outputs $l_{12}$, $\alpha_{12}$, and $z_{12}$. This component must also be included in the input–output equations. The dynamics of the yaw angle is derived by differentiating the third component of Eq. (8.41). The result of this differentiation takes the following standard matrix form:

$$\ddot{\psi}_2 = f_2 + \mathbf{b}_2\mathbf{u}, \tag{8.102}$$

where

$$\begin{aligned} f_2 = \; & (\dot{\phi}_2 \cos\phi_2 \sec\theta_2 + \dot{\theta}_2 \sin\phi_2 \sec\theta_2 \tan\theta_2)\omega_{2y}^{(2)} \\ & + (-\dot{\phi}_2 \sin\phi_2 \sec\theta_2 + \dot{\theta}_2 \cos\phi_2 \sec\theta_2 \tan\theta_2)\omega_{2z}^{(2)} \\ & + (\sin\phi_2 \sec\theta_2)((I_{zz} - I_{xx})\omega_{2x}^{(2)}\omega_{2z}^{(2)}/I_{yy}) \\ & + (\cos\phi_2 \sec\theta_2)((I_{xx} - I_{yy})\omega_{2x}^{(2)}\omega_{2y}^{(2)}/I_{zz}), \end{aligned} \tag{8.103}$$

$$\mathbf{b}_2 = \left[ \frac{\sin\phi_2 \sec\theta_2 l_r}{I_{yy}} - \frac{\cos\phi_2 \sec\theta_2 K_m}{I_{zz}},\; 0,\; \frac{\sin\phi_2 \sec\theta_2}{I_{yy}},\; \frac{\cos\phi_2 \sec\theta_2 l_t}{I_{zz}} \right]. \tag{8.104}$$

The full set of input–output equations are obtained by combining Eqs. (8.100) and (8.102) into a single matrix form:

$$\begin{bmatrix} \ddot{l}_{12} \\ \ddot{\alpha}_{12} \\ \ddot{z}_{12} \\ \hline \ddot{\psi}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{1(3\times1)} \\ \hline f_2 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_{1(3\times4)} \\ \hline \mathbf{b}_{2(1\times4)} \end{bmatrix} \begin{bmatrix} T \\ M_\phi \\ M_\theta \\ T_T \end{bmatrix}, \tag{8.105}$$

or in a more concise form:

$$\ddot{\mathbf{y}} = \mathbf{f} + \mathbf{b}\mathbf{u}. \tag{8.106}$$

### 8.6.1.2 Input–Output Description for the *l* – *l* Control Scheme

In Fig. 8.15, a system of three neighboring helicopters in the formation is shown. The control point of the follower helicopter, $p_3$, is separated from the control points of leader 1 and leader 2, $p_1$ and $p_2$, by two 3D vectorial distances $\mathbf{l}_{13}$ and $\mathbf{l}_{23}$, respectively. The formation plane $p_1p_2p_3$ makes an angle of $\beta_{123}$ with a reference direction (Fig. 8.16). Note that the helicopters are not physically coupled in any way. A feedback control law for control inputs $\mathbf{u} = [T, M_\phi, M_\theta, T_T]^T$ must be determined to control helicopter 3 such that the desired distances $l_{13}^d$, $l_{23}^d$, and angle $\beta_{123}^d$

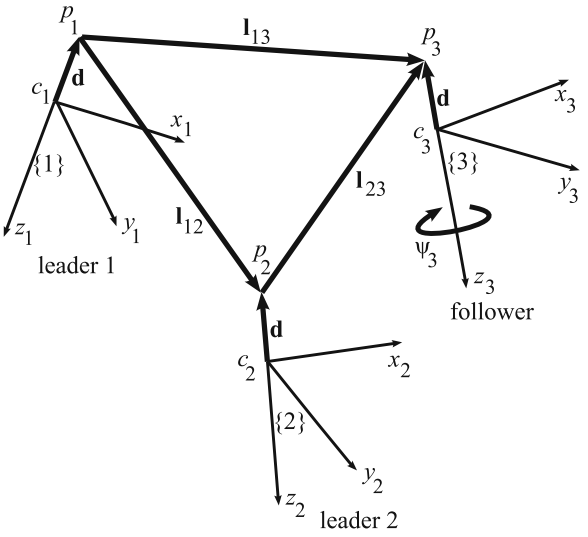


**Fig. 8.15** $l - l$ control configuration. Frames 1, 2, and 3 correspond to the first leader, the second leader, and the follower, respectively. The helicopters' centres of mass are denoted by $c$ and their control points are denoted by $p$

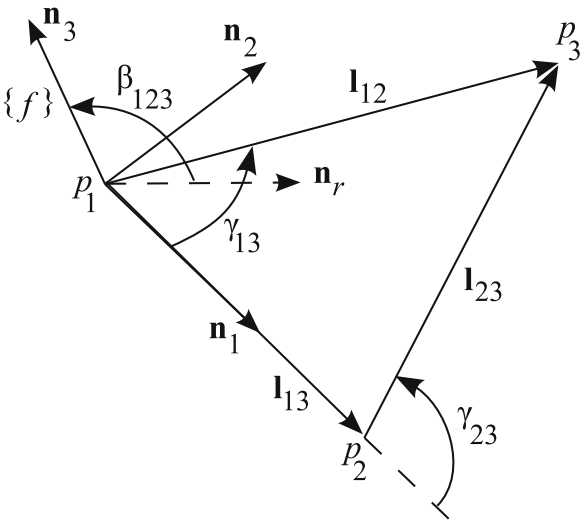


**Fig. 8.16** Definition of the formation frame for the $l - l$ control scheme. The three helicopters' control points are denoted by $p_1$, $p_2$, and $p_3$. Unit vector $\mathbf{n}_1$ points from $p_1$ to $p_2$. Unit vector $\mathbf{n}_3$ is perpendicular to the formation plane $p_1p_2p_3$. Unit vector $\mathbf{n}_2$ lies in the formation plane and makes a right-hand frame $\{f\}$ with $\mathbf{n}_1$ and $\mathbf{n}_3$. The frame $\{f\}$ is called the formation frame. Unit vector $\mathbf{n}_r$ lies in the global horizontal plane and is perpendicular to $\mathbf{n}_1$. When the formation plane is horizontal, $\beta_{123} = \frac{\pi}{2}$

are maintained, while the yaw angle of helicopter 3, $\psi_3$, follows a desired trajectory. With these definitions, the outputs of the control system are $\mathbf{y} = [l_{13}, l_{23}, \beta_{123}, \psi_3]^T$.

Here, the input–output description of the control system are derived, which relate the output $\mathbf{y}$ to the input $\mathbf{u}$ directly. First, through a kinematic analysis, the state variables of helicopter 3, which are

$$\mathbf{q}_3 = \left[x_3^{(0)}, y_3^{(0)}, z_3^{(0)}, \dot{x}_3^{(0)}, \dot{y}_3^{(0)}, \dot{z}_3^{(0)}, \phi_3, \theta_3, \psi_3, \omega_{3x}^{(3)}, \omega_{3y}^{(3)}, \omega_{3z}^{(3)}\right]^T, \tag{8.107}$$

are related to the output $\mathbf{y}$. Then, the equations of motion, containing $\mathbf{u}$ are substituted in the resulting equations to give the input-output relations. The details follow.

**Kinematic Analysis**

The control points of the three helicopters in an $l - l$ scheme form a three-dimensional plane (the formation plane). The plane of formation $p_1 p_2 p_3$ and its local coordinate frame $\{f\}$ are shown in Fig. 8.16. Since the unit vectors of the coordinate frame $\{f\}$ are defined based on the location of the three helicopters, this frame moves and rotates when the three helicopters move. The angular velocity and acceleration of this frame is required for calculating the rate of change of the formation parameters (control outputs) defined in the previous section. In the following, first, the formation frame is defined. Then, the angular velocity and acceleration of this frame are determined. And finally, the rate of change of the formation parameters (control outputs) are calculated.

The mutually perpendicular unit vectors of the formation frame $\{f\}$, whose origin is at $p_1$, are defines as

$$\mathbf{n}_1 = \frac{\mathbf{l}_{12}}{|\mathbf{l}_{12}|}, \quad \mathbf{n}_3 = \frac{\mathbf{l}_{12} \times \mathbf{l}_{13}}{|\mathbf{l}_{12} \times \mathbf{l}_{13}|}, \quad \mathbf{n}_2 = \mathbf{n}_3 \times \mathbf{n}_1. \tag{8.108}$$

A reference $\mathbf{n}_r$ for rotation of the formation plane about $\mathbf{l}_{12}$ is needed to define the formation parameter $\beta_{123}$. This reference unit vector vector is assumed to lie in the global horizontal plane and to be perpendicular to $\mathbf{l}_{12}$. It is calculated as

$$\mathbf{n}_r = \frac{\mathbf{l}_{12} \times \mathbf{k}_0}{|\mathbf{l}_{12} \times \mathbf{k}_0|}. \tag{8.109}$$

Now, the rotation of the formation plane about $\mathbf{l}_{12}$ in reference to $\mathbf{n}_r$, which is one of the formation parameters, is defines as

$$\beta_{123} = \arccos(\mathbf{n}_r . \mathbf{n}_3). \tag{8.110}$$

The angular velocity of the formation frame described in the formation frame is defined as

$$\boldsymbol{\omega}_f^{(f)} = \dot{\beta}_{123}\mathbf{n}_1 + \omega_{f2}\mathbf{n}_2 + \omega_{f3}\mathbf{n}_3. \tag{8.111}$$

The second and third components of $\boldsymbol{\omega}_f^{(f)}$ can be found by observing the relative velocity of points $p_1$ and $p_2$.

$$\mathbf{v}_{p2}^{(f)} = \mathbf{v}_{p1}^{(f)} + \boldsymbol{\omega}_f^{(f)} \times \mathbf{l}_{12}^{(f)} + \dot{\mathbf{l}}_{12}^{(f)}, \tag{8.112}$$

where $\mathbf{l}_{12}^{(f)} = l_{12}\mathbf{n}_1$ and $\dot{\mathbf{l}}_{12}^{(f)} = \dot{l}_{12}\mathbf{n}_1$. Two of the three components of $\boldsymbol{\omega}_f^{(f)}$ can be obtained by rearranging Eq. (8.112) as

$$\begin{bmatrix} \dot{l}_{12} \\ l_{12}\omega_{f3} \\ -l_{12}\omega_{f2} \end{bmatrix} = \mathbf{R}_{0f}^T(\mathbf{v}_{p2}^{(0)} - \mathbf{v}_{p1}^{(0)}). \tag{8.113}$$

The first component of $\boldsymbol{\omega}_f^{(f)}$ is found by writing a relative velocity equation between points $p_1$ and $p_3$.

$$\mathbf{v}_{p3}^{(f)} = \mathbf{v}_{p1}^{(f)} + \boldsymbol{\omega}_f^{(f)} \times \mathbf{l}_{13}^{(f)} + \dot{\mathbf{l}}_{13}^{(f)}. \tag{8.114}$$

where $\mathbf{l}_{13}^{(f)} = l_{13}(\cos\gamma_{13}\mathbf{n}_1 + \sin\gamma_{13}\mathbf{n}_2)$ and $\dot{\mathbf{l}}_{13}^{(f)} = \dot{l}_{13}(\cos\gamma_{13}\mathbf{n}_1 + \sin\gamma_{13}\mathbf{n}_2)$, as concluded from Fig. 8.16. $\dot{\beta}_{123}$ can be found by simplifying the third component of Eq. (8.114).

$$\dot{\beta}_{123} = \frac{v_{p3z}^{(f)} - v_{p1z}^{(f)} + \omega_{f2} l_{13}\cos\gamma_{13}}{l_{13}\sin\gamma_{13}}. \tag{8.115}$$

$\dot{\beta}_{123}$ can be calculated as long as $l_{13}$ and $\gamma_{13}$ are nonzero. These situations can be avoided when defining the desired formation parameters. $l_{13}^d = 0$ means that the helicopters are coincident, which is physically impossible and must not be used. $\gamma_{13}^d = 0$ corresponds to the situation when the three helicopters' control points are on the same line. In this situation, two $l-\alpha$ schemes must be used to define the desired formation. Note that if the control points $p_2$ and $p_3$ are used for finding the rate $\dot{\beta}_{123}$, the same result will be obtained. This remains for the reader to prove as an exercise.

The angular acceleration of the formation frame, $\dot{\boldsymbol{\omega}}_f^{(f)}$, is also required for deriving the input–output equations of the $l-l$ control scheme. Two components of this acceleration can be obtained by observing the relative acceleration of points $p_1$ and $p_2$.

$$\mathbf{a}_{p2}^{(f)} = \mathbf{a}_{p1}^{(f)} + \dot{\boldsymbol{\omega}}_f^{(f)} \times \mathbf{l}_{12}^{(f)} + \boldsymbol{\omega}_f^{(f)} \times (\boldsymbol{\omega}_f^{(f)} \times \mathbf{l}_{12}^{(f)}) + 2\boldsymbol{\omega}_f^{(f)} \times \dot{\mathbf{l}}_{12}^{(f)} + \ddot{\mathbf{l}}_{12}^{(f)}, \tag{8.116}$$

where $\ddot{\mathbf{l}}_{12}^{(f)} = \ddot{l}_{12}\mathbf{n}_1$ and $\dot{\boldsymbol{\omega}}_f^{(f)} = \ddot{\beta}_{123}\mathbf{n}_1 + \alpha_{f2}\mathbf{n}_2 + \alpha_{f3}\mathbf{n}_3$. The second and third components of the formation frame angular acceleration are obtained by rearranging Eq. (8.116).

$$\begin{bmatrix} \ddot{l}_{12} \\ l_{12}\alpha_{f3} \\ -l_{12}\alpha_{f2} \end{bmatrix} = \mathbf{R}_{0f}^{T}(\mathbf{a}_{p2}^{(0)} - \mathbf{a}_{p1}^{(0)}) - \boldsymbol{\omega}_{f}^{(f)} \times (\boldsymbol{\omega}_{f}^{(f)} \times \mathbf{l}_{12}^{(f)}) - 2\boldsymbol{\omega}_{f}^{(f)} \times \dot{\mathbf{l}}_{12}^{(f)}. \quad (8.117)$$

After the angular motion of the formation frame is known, the rate of change of the formation parameters $l_{13}$, $l_{23}$, and $\beta_{123}$ can be determined by investigating the relative motion of point $p_3$ with respect to points $p_1$ and $p_2$ separately. This is first shown by considering the relative motion of point $p_3$ with respect to $p_1$ to obtain $\ddot{l}_{13}$ and $\ddot{\beta}_{123}$. Then, the results are simply extended for $\ddot{l}_{23}$. The acceleration of $p_3$ can be formulated as

$$\mathbf{a}_{p3}^{(f)} = \mathbf{a}_{p1}^{(f)} + \dot{\boldsymbol{\omega}}_{f}^{(f)} \times \mathbf{l}_{13}^{(f)} + \boldsymbol{\omega}_{f}^{(f)} \times (\boldsymbol{\omega}_{f}^{(f)} \times \mathbf{l}_{13}^{(f)}) + 2\boldsymbol{\omega}_{f}^{(f)} \times \dot{\mathbf{l}}_{13}^{(f)} + \ddot{\mathbf{l}}_{13}^{(f)}. \quad (8.118)$$

On the other hand, the same acceleration can be derived with respect to the follower's center of mass $c_3$.

$$\mathbf{a}_{p3}^{(f)} = \mathbf{a}_{c3}^{(f)} + \dot{\boldsymbol{\omega}}_{3}^{(f)} \times \mathbf{d}^{(f)} + \boldsymbol{\omega}_{3}^{(f)} \times (\boldsymbol{\omega}_{3}^{(f)} \times \mathbf{d}^{(f)}). \quad (8.119)$$

Combining Eqs. (8.118) and (8.119) and defining the terms

$$\mathbf{N}_1 = \boldsymbol{\omega}_{3}^{(f)} \times (\boldsymbol{\omega}_{3}^{(f)} \times \mathbf{d}^{(f)}) - \mathbf{a}_{p1}^{(f)} - \boldsymbol{\omega}_{f}^{(f)} \times (\boldsymbol{\omega}_{f}^{(f)} \times \mathbf{l}_{12}^{(f)}) - 2\boldsymbol{\omega}_{f}^{(f)} \times \dot{\mathbf{l}}_{12}^{(f)}, \quad (8.120)$$

$$\mathbf{M}_1 = \begin{bmatrix} 0 & 0 & -\alpha_{f2} l_{13} \cos\gamma_{13} \end{bmatrix}^{T}, \quad (8.121)$$

results in

$$\begin{bmatrix} \ddot{l}_{13}\cos\gamma_{13} - \alpha_{f3} l_{13}\sin\gamma_{13} \\ \ddot{l}_{13}\sin\gamma_{13} + \alpha_{f3} l_{13}\cos\gamma_{13} \\ \ddot{\beta}_{123} l_{13}\sin\gamma_{13} \end{bmatrix} = \mathbf{a}_{c3}^{(f)} + \dot{\boldsymbol{\omega}}_{3}^{(f)} \times \mathbf{d}^{(f)} + \mathbf{N}_1 - \mathbf{M}_1. \quad (8.122)$$

Equation (8.122) can be further simplified to obtain $\ddot{l}_{13}$ and $\ddot{\beta}_{123}$.

$$\begin{bmatrix} \ddot{l}_{13} \\ l_{13}\alpha_{f3} \\ \ddot{\beta}_{123} \end{bmatrix} = \mathbf{A}_3^{-1}(\mathbf{a}_{c3}^{(f)} + \dot{\boldsymbol{\omega}}_{3}^{(f)} \times \mathbf{d}^{(f)} + \mathbf{N}_1 - \mathbf{M}_1), \quad (8.123)$$

where

$$\mathbf{A}_3 = \begin{bmatrix} \cos\gamma_{13} & -\sin\gamma_{13} & 0 \\ \sin\gamma_{13} & \cos\gamma_{13} & 0 \\ 0 & 0 & l_{13}\sin\gamma_{13} \end{bmatrix}. \quad (8.124)$$

This matrix is invertible as long as $l_{13}$ and $\gamma_{13}$ are nonzero.

The rate of change for $l_{23}$ can be obtained with the same procedure as mentioned above. The results is

$$\begin{bmatrix} \ddot{l}_{23} \\ l_{23}\alpha_{f3} \\ \ddot{\beta}_{123} \end{bmatrix} = \mathbf{A}_4^{-1}(\mathbf{a}_{c3}^{(f)} + \dot{\boldsymbol{\omega}}_3^{(f)} \times \mathbf{d}^{(f)} + \mathbf{N}_2 - \mathbf{M}_2), \tag{8.125}$$

where

$$\mathbf{A}_4 = \begin{bmatrix} \cos\gamma_{23} & -\sin\gamma_{23} & 0 \\ \sin\gamma_{23} & \cos\gamma_{23} & 0 \\ 0 & 0 & l_{23}\sin\gamma_{23} \end{bmatrix}. \tag{8.126}$$

$$\mathbf{N}_2 = \boldsymbol{\omega}_3^{(f)} \times (\boldsymbol{\omega}_3^{(f)} \times \mathbf{d}^{(f)}) - \mathbf{a}_{p2}^{(f)} - \boldsymbol{\omega}_f^{(f)} \times (\boldsymbol{\omega}_f^{(f)} \times \mathbf{l}_{23}^{(f)}) - 2\boldsymbol{\omega}_f^{(f)} \times \dot{\mathbf{l}}_{23}^{(f)}, \tag{8.127}$$

$$\mathbf{M}_2 = \begin{bmatrix} 0 & 0 & -\alpha_{f2} l_{23}\cos\gamma_{23} \end{bmatrix}^T. \tag{8.128}$$

$\mathbf{A}_4$ can be inverted as long as $l_{23}$ and $\gamma_{23}$ are nonzero.

The singularities arising from Eqs. (8.124) and (8.126) can be avoided when defining the desired formation parameters. $l_{13}^d = 0$ or $l_{23}^d = 0$ means that the leader and the follower helicopters are coincident, which is physically impossible and must not be used. $\gamma_{13}^d = 0$ or $\gamma_{23}^d = 0$ corresponds to the situation when the three helicopters' control points are on the same line. In this situation, two $l - \alpha$ schemes must be used to define the desired formation.

It can be shown that the result for $\ddot{\beta}_{123}$ from Eq. (8.125) is equal to the result obtained in Eq. (8.123). This remains for the reader to prove as an exercise. Now, a subset of the output vector $\mathbf{y}_2 = [l_{13}, l_{23}, \beta_{123}]^T$ can be formed by combining Eqs. (8.123) and (8.125) as follows.

$$\ddot{\mathbf{y}}_2 = \begin{bmatrix} \ddot{l}_{13} \\ \ddot{l}_{23} \\ \ddot{\beta}_{123} \end{bmatrix} = \mathbf{C}_3 \begin{bmatrix} \ddot{l}_{13} \\ l_{13}\alpha_{f3} \\ \ddot{\beta}_{123} \end{bmatrix} + \mathbf{C}_4 \begin{bmatrix} \ddot{l}_{23} \\ l_{23}\alpha_{f3} \\ \ddot{\beta}_{123} \end{bmatrix}, \tag{8.129}$$

where

$$\mathbf{C}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{C}_4 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{8.130}$$

**Input–Output Equations**

Part of the input–output description of the $l - l$ control scheme is derived by substituting Eqs. (8.123) and (8.125) into Eq. (8.129).

$$\ddot{\mathbf{y}}_2 = (\mathbf{C}_3\mathbf{A}_3^{-1} + \mathbf{C}_4\mathbf{A}_4^{-1})(\mathbf{a}_{c3}^{(f)} + \dot{\boldsymbol{\omega}}_3^{(f)} \times \mathbf{d}^{(f)}) + \mathbf{C}_3\mathbf{A}_3^{-1}(\mathbf{N}_1 - \mathbf{M}_1) + \mathbf{C}_4\mathbf{A}_4^{-1}(\mathbf{N}_2 - \mathbf{M}_2). \tag{8.131}$$

This part of the input–output description is completed by using frame conversions

$$\mathbf{a}_{c3}^{(f)} = \mathbf{R}_{0f}^T \mathbf{a}_{c3}^{(0)}, \qquad \dot{\boldsymbol{\omega}}_3^{(f)} \times \mathbf{d}^{(f)} = \mathbf{R}_{0f}^T \mathbf{R}_{03}(\dot{\boldsymbol{\omega}}_3^{(3)} \times \mathbf{d}^{(3)}), \tag{8.132}$$

and substituting for $\mathbf{a}_{c3}^{(0)}$ and $\dot{\boldsymbol{\omega}}_3^{(3)} \times \mathbf{d}^{(3)}$ using equations similar to Eqs. (8.96) and (8.98) for the dynamics of the follower. The result can be rearranged as

$$\ddot{\mathbf{y}}_2 = \mathbf{f}_3 + \mathbf{b}_3\mathbf{u}, \tag{8.133}$$

where

$$\mathbf{b}_3 = (\mathbf{C}_3\mathbf{A}_3^{-1} + \mathbf{C}_4\mathbf{A}_4^{-1})(\mathbf{R}_{0f}^T\mathbf{C}_1 + \mathbf{R}_{0f}^T\mathbf{R}_{03}\mathbf{C}_2), \tag{8.134}$$

$$\begin{aligned}\mathbf{f}_3 = {} & (\mathbf{C}_3\mathbf{A}_3^{-1} + \mathbf{C}_4\mathbf{A}_4^{-1})(\mathbf{R}_{0f}^T(\mathbf{D}_1 + \mathbf{W}_1) + \mathbf{R}_{0f}^T\mathbf{R}_{03}\mathbf{D}_2) \\ & + \mathbf{C}_3\mathbf{A}_3^{-1}(\mathbf{N}_1 - \mathbf{M}_1) + \mathbf{C}_4\mathbf{A}_4^{-1}(\mathbf{N}_2 - \mathbf{M}_2).\end{aligned} \tag{8.135}$$

The full set of input–output equations are obtained by augmenting the yaw dynamics $\ddot{\psi}_3$ of the follower as the fourth formation parameter with Eqs. (8.133). Relations similar to Eqs. (8.102), (8.103), and (8.104) can be used, in which the subscript 2 for the follower states is replaced by 3, indicating the use of the follower 3 states.

$$\ddot{\psi}_3 = f_4 + \mathbf{b}_4\mathbf{u}. \tag{8.136}$$

This results in a single matrix form for the $l$ –$l$ input–output description:

$$\begin{bmatrix} \ddot{l}_{13} \\ \ddot{l}_{23} \\ \ddot{\beta}_{123} \\ \hline \\ \ddot{\psi}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{3(3\times1)} \\ \hline \\ f_4 \end{bmatrix} + \begin{bmatrix} \mathbf{b}_{3(3\times4)} \\ \hline \\ \mathbf{b}_{4(1\times4)} \end{bmatrix} \begin{bmatrix} T \\ M_\phi \\ M_\theta \\ T_T \end{bmatrix}, \tag{8.137}$$

or in a more concise form:

$$\ddot{\mathbf{y}} = \mathbf{f} + \mathbf{b}\mathbf{u}. \tag{8.138}$$

### *8.6.2 Designing the Sliding Mode Control Law*

In previous sections, the input–output description for both the $l - \alpha$ and $l - l$ control schemes were derived. They were written in a similar general matrix form

[Eqs. (8.106) and (8.138)] to simplify the control law development. The sliding mode control method is used to design a controller based on the matrix form of the input–output equations. In this method, four first-order asymptotically stable surface functions are assumed:

$$\mathbf{s} = (\dot{\mathbf{y}} - \dot{\mathbf{y}}^d) + \Lambda(\mathbf{y} - \mathbf{y}^d), \tag{8.139}$$

where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$, and all $\lambda_i$'s are positive. Eq. (8.139) is written in following form for convenience:

$$\mathbf{s} = \dot{\mathbf{y}} - \mathbf{s}_r, \tag{8.140}$$

where

$$\mathbf{s}_r = \dot{\mathbf{y}}^d - \Lambda(\mathbf{y} - \mathbf{y}^d). \tag{8.141}$$

If the trajectory of the system can be controlled such that $\mathbf{s}$ approaches zero and remains zero at all times, since the surface (8.139) is asymptotically stable, it is guaranteed that the output $\mathbf{y}$ converges to its desired value. Therefore, the sliding mode controller design reduces to finding a control law that brings and keeps the output of the system on the sliding surface.

Since the input–output Eqs. (8.106) and (8.138) have the same form as the input–output Eq. (8.57) for the helicopter trajectory-tracking control, a sliding mode control law in the form of Eq. (8.63) derived in Section 8.5.2.2 for trajectory tracking can be used here as well.

$$\mathbf{u} = \hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})). \tag{8.142}$$

The results of Section 8.5.2.2 also indicate that if the bounds of model uncertainties are defined by

$$|f_i - \hat{f}_i| \leq F_i, \quad |\delta_{ij}| \leq \Delta_{ij}, \qquad i = 1, \ldots, 4, \tag{8.143}$$

the diagonal elements of the discontinuity gain matrix $\mathbf{K}$ are found from

$$\mathbf{K}_v = (\mathbf{I}_m - \Delta)^{-1}(\mathbf{F} + \Delta|\ - \hat{\mathbf{f}} + \dot{\mathbf{s}}_r| + \boldsymbol{\eta}), \tag{8.144}$$

where $\mathbf{K}_v = [K_1, K_2, K_3, K_4]^T$ and $\boldsymbol{\eta} = [\eta_1, \eta_2, \eta_3, \eta_4]^T$. For a detailed derivation of Eq. (8.144), refer to Section 8.5.2.2.

Note that the control law (8.142) requires the formation parameters, their first order derivatives, and the states of the leader(s) and the follower helicopter at any given time. The states of the leader helicopter(s), especially their Euler angles and angular velocities are difficult to measure using vision sensors on the follower helicopter. Therefore, it is assumed that the leader(s) communicate their states with the follower. Once the follower is aware of the states of the leader(s), it can calculate

the formation parameters based on the leader(s)' and its own positions, and the rate of formation parameters based on the leader(s)' and its own linear and angular velocities. The required communication bandwith is bounded and is not a function of the number of helicopters in the formation because the controllers are decentralized. Each helicopter has to receive information from at most two helicopters and send information to at most two helicopters.

*Example 8.3.* Consider the autonomous helicopter introduced in Example 8.1, whose inertial and geometrical properties are shown in Table 8.1. Assume that the center of gravity of the helicopter is not coincident with the main rotor's axis (Fig. 8.1), such that $l_r = 0.1$ m. Assume that the distance of the control point with the helicopter's center of gravity $d = 1$ m. Use the $l - \psi$ control law to make the helicopter keep a distance of 10 m ($l_{12}^d = 10$ m) to the right side of a leader ($\alpha_{12}^d = 90^\circ$), while flying at the same altitude of the leader ($z_{12}^d = 0$ m) and heading parallel to the positive global $x$ direction ($\psi_2^d = 0^\circ$). The motion of the leader is defined by

$$\begin{aligned} x_1(t) &= V_x t, \\ y_1(t) &= R_y \sin(\frac{2\pi t}{\tau}), \\ z_1(t) &= R_z \sin(\frac{2\pi t}{\tau}), \end{aligned} \tag{8.145}$$

where $V_x = 1$ m/s, $\tau = 30$ s, $R_y = R_z = 2$ m. The follower helicopter is initially at $(0, 10, 0)$ m and facing toward the positive global $x$ direction and has an initial velocity of $\frac{2\pi t}{\tau}(R_y\hat{\mathbf{j}} + R_z\hat{\mathbf{k}})$ m/s. Simulate the motion of the helicopter for the following two conditions.

(a) There is no wind.
(b) There is a constant wind with a 5 m/s speed blowing along the negative global $y$ direction.

*Solution.* To include the wind disturbance, the shape of the helicopter fuselage is approximated by a box to estimate the wind forces, where $L$ is the length and $H$ is the height of the helicopter fuselage. Also, $\rho$ is the air density and $v_w$ is the wind velocity. The following numerical values are used:

$$\begin{aligned} L &= 0.3 \text{ m}, \qquad H = 0.2 \text{ m}, \\ \rho &= 1.2 \text{ kg/m}^3, \qquad v_w = 5 \text{ m/s}. \end{aligned} \tag{8.146}$$

This wind results is an approximate force of

$$F_w = \frac{1}{2}\rho v_w^2 HL = 1.8 \text{ N}, \tag{8.147}$$

which is approximately 13.5% of the helicopter's weight (or main rotor's thrust at hovering condition). These data are used with the dynamic model when the effect of wind on the helicopter's performance is to be shown. Note that the formation controllers are not aware of the presence of the wind forces and do not directly compensate for them.

The control law (8.142) calculates the required inputs $\mathbf{u}$ without being aware of the presence or absence of the wind. The controller parameters are as follows.

$$\Lambda = \begin{bmatrix} 0.3 & 0 & 0 & 0 \\ 0 & 0.3 & 0 & 0 \\ 0 & 0 & 0.3 & 0 \\ 0 & 0 & 0 & 0.3 \end{bmatrix}, \tag{8.148}$$

$$\boldsymbol{\eta} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^T, \tag{8.149}$$

$$\Phi = \begin{bmatrix} 0.1 & 0.1 & 0.1 & \pi/40 \end{bmatrix}^T. \tag{8.150}$$

The control commands are directly applied to the equations of motion of the follower helicopter. In the no-wind simulation case, the equations of motion (8.15) are used for integration. In the case where wind is present, a force term representing the wind force is added to the translational equations of motion of the follower. The results of the simulations and the discussions are presented in the following.

Figure 8.17 shows the paths of the motion for the two cases. Two coincident curves show the resulting motion of the follower helicopter in no-wind and lateral wind situations. As is seen in the figure, there is not much difference in the path of the follower in the two cases.

Figure 8.18 shows the global position components of the follower helicopter. It can be seen that the $x$ and $z$ components of motion are not affected by the presence of the lateral wind. Only the $y$ component is slightly disturbed because of the wind. However, the controller is successful in rejecting the disturbances and bringing the $y$ component to the desired dynamic equilibrium.

Figure 8.19 shows the orientation of the follower helicopter. The pitch and yaw motions are minimally affected by the wind force. After a transient response, the roll angle fluctuates around $1.6^\circ$ when there is no wind. This small offset angle is necessary so that the lateral components of the main rotor and the tail rotor forces reach an equilibrium. When the lateral wind is blowing, after a transient response, the roll angle fluctuates around $7^\circ$. In such a situation, the helicopter has to lean more against the wind direction to equalize the wind force with the lateral component of the main rotor thrust, which explains the higher roll angle compared to when the wind is not present.

Figure 8.20 shows the four $l - \alpha$ formation parameters or the control outputs. Once again, it is seen that the effect of the wind on these outputs are minimal.

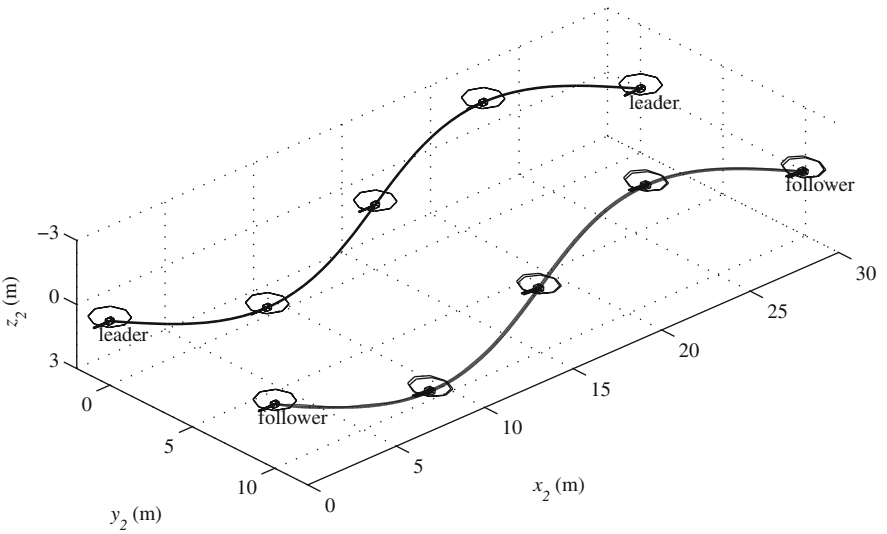


**Fig. 8.17** Path of the motion for the $l - \alpha$ scheme. The follower robustly follows the leader with a given relative position despite of a 10 m/s lateral wind. The effect of the wind on the path of the follower is negligible

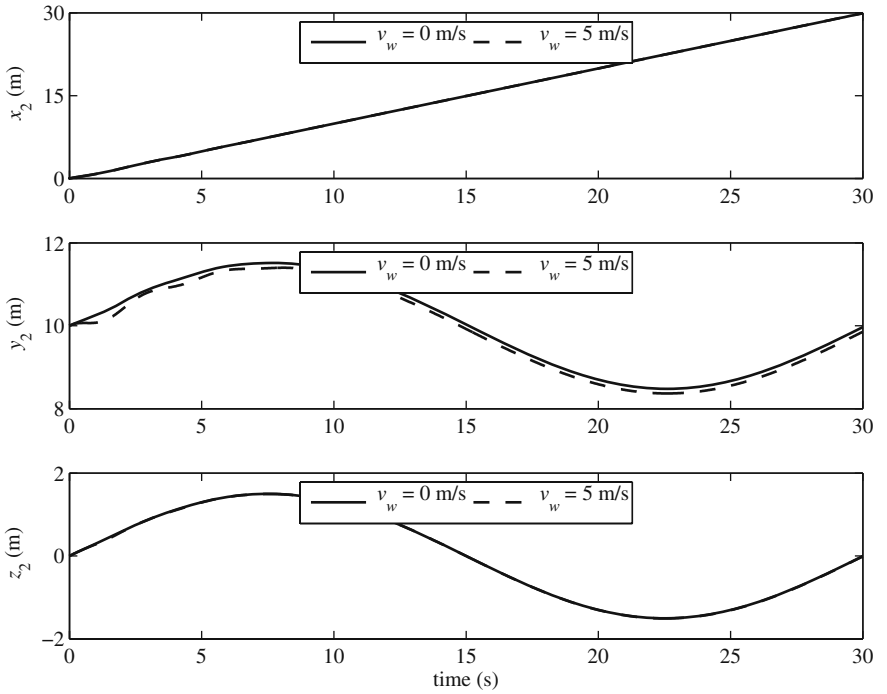


**Fig. 8.18** Position trajectories for the $l - \alpha$ scheme. The follower's position trajectories in the presence and the absence of the wind disturbance are fairly close. The difference in the $y$ component is because the follower tries to reach a steady-state roll angle to resist the lateral wind force when the wind is present

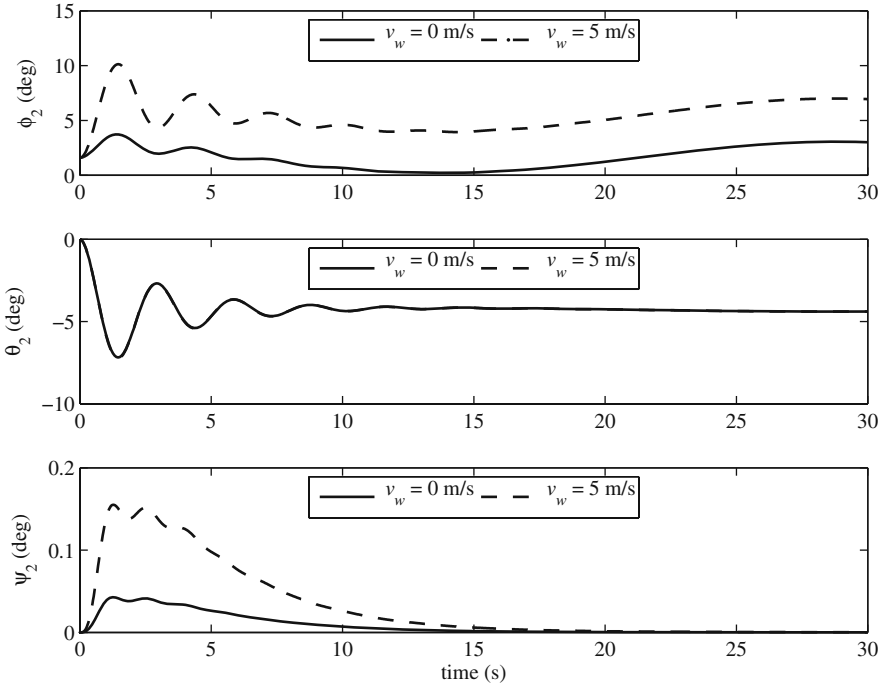


**Fig. 8.19** Orientation trajectories for the $l - \alpha$ scheme. The follower's orientation trajectories in the presence and the absence of the wind disturbance are fairly close, except for the roll angle. The steady-state roll angle is larger when wind is present because the follower tries to counteract the lateral wind force

The steady-state values of the formation parameters do not experience any offsets, except for the $l_{12}$, which is directly affected by the wind in the global $y$ direction. The control points of the leader and the follower helicopters keep a distance of approximately 10 m ($l_{12} \approx 10$ m). The helicopters move side by side ($\alpha_{12} \approx 90^\circ$) and at the same height ($z_{12} \approx 0$ m). And the follower helicopter faces straight ahead ($\psi_2 \approx 0^\circ$).

Figure 8.21 shows the four helicopter control inputs. The thrust force $T$ is higher when the wind blows, because it has to provide a lateral component to counterbalance the lateral wind force. The larger roll angle in the wind situation allows for this component. Since the main rotor axis does not pass through the helicopter's center of mass, the main rotor's thrust generates a moment about this point. This moment is counterbalanced by the pitch moment, which is not fluctuating around zero. An increased thrust requires a higher pitch moment. This fact is reflected in the $M_\theta$ plot. The tail rotor thrust also has to equalize the reaction torque on the fuselage caused by $T$. Hence, a higher tail rotor thrust $T_T$ is required when the wind is present.

*Example 8.4.* Consider the autonomous helicopter introduced in Example 8.1, whose inertial and geometrical properties are shown in Table 8.1. Assume that the center of gravity of the helicopter is not coincident with the main rotor's axis (Fig. 8.1), such

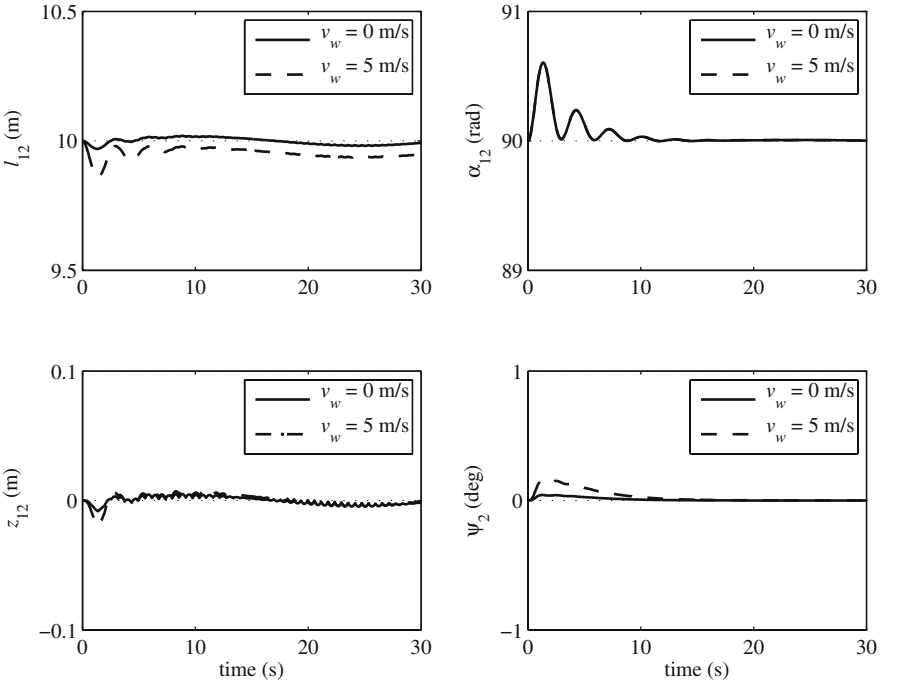


**Fig. 8.20** Output trajectories for the $l - \alpha$ scheme. The formation parameters' steady-state values are not significantly affected by the lateral wind. The initial disturbance in $l_{12}$ is because the follower tries to reach an equilibrium roll angle to counteract the lateral wind force

that $l_r = 0.1$ m. Assume that the distance of the control point with the helicopter's center of gravity is $d = 1$ m. Use the $l - l$ control law to make the helicopter keep a distance of 10 m with two leader helicopters ($l_{13}^d = l_{23}^d = 10$ m), while flying at the same altitude of the leaders ($\beta_{123}^d = 90^\circ$) and heading parallel to the positive global $x$ direction ($\psi_3^d = 0^\circ$). The motion of the leaders are defined by

$$\begin{aligned} x_1(t) &= V_x t, \\ y_1(t) &= R_y \sin(\frac{2\pi t}{\tau}) + y_{10}, \\ z_1(t) &= R_z \sin(\frac{2\pi t}{\tau}), \end{aligned} \tag{8.151}$$

$$\begin{aligned} x_2(t) &= V_x t, \\ y_2(t) &= R_y \sin(\frac{2\pi t}{\tau}) + y_{20}, \\ z_2(t) &= R_z \sin(\frac{2\pi t}{\tau}), \end{aligned} \tag{8.152}$$

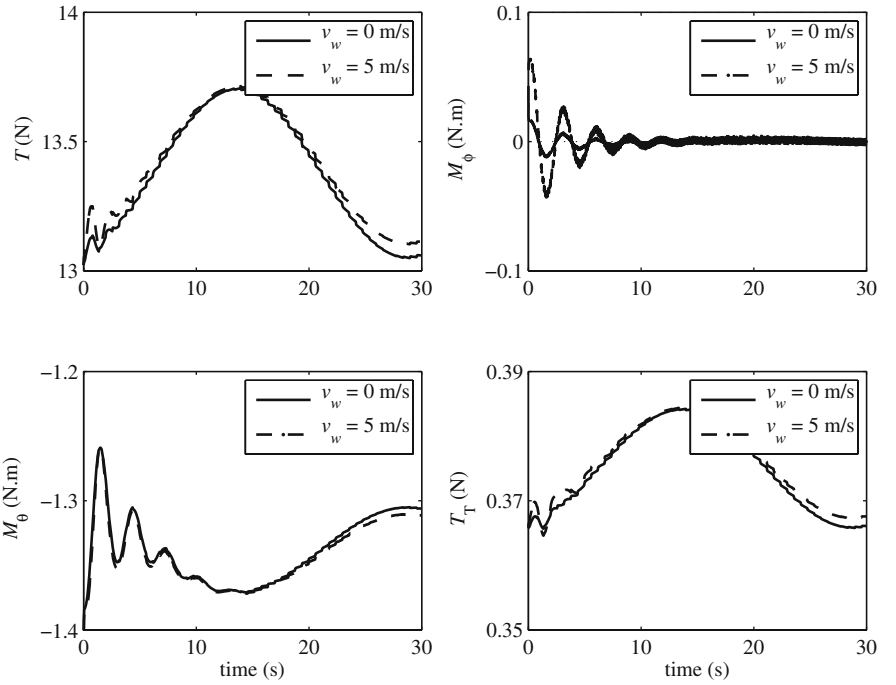


**Fig. 8.21** Input history for the $l - \alpha$ scheme. The required thrust $T$ when wind exists is higher than that of the no wind situation. Other control inputs are also higher in presence of wind to counterbalance the higher thrust force, except $M_\phi$. This is because $T$ does not produce a moment about the helicopter's roll axis

where $V_x = 1$ m/s, $\tau = 30$ s, $y_{10} = -5$ m, $y_{20} = 5$ m, $R_y = R_z = 1.5$ m. The follower helicopter is initially at $(0, -8.66, 0)$ m and facing toward the positive global $x$ direction and has an initial velocity of $\frac{2\pi t}{\tau}(R_y\hat{\mathbf{j}} + R_z\hat{\mathbf{k}})$ m/s. Simulate the motion of the helicopter for the following two conditions.

(a) There is no wind.
(b) There is a constant wind with a 5 m/s speed blowing along the negative global $y$ direction.

*Solution.* The wind force is calculated similar to the calculations in Example 8.3. The control law (8.142) calculates the required inputs $\mathbf{u}$ without being aware of the presence or absence of the wind. The controller parameters are as follows.

$$\Lambda = \begin{bmatrix} 0.3 & 0 & 0 & 0 \\ 0 & 0.3 & 0 & 0 \\ 0 & 0 & 0.3 & 0 \\ 0 & 0 & 0 & 0.3 \end{bmatrix}, \tag{8.153}$$

$$\boldsymbol{\eta} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}^T, \tag{8.154}$$

$$\Phi = \begin{bmatrix} 0.1 & 0.1 & \pi/40 & \pi/40 \end{bmatrix}^T. \tag{8.155}$$

The equations of motion of the follower helicopter are used with the control commands to simulate the follower's motion. In the case where wind is present, a force term representing the wind force is added to the translational equations of motion of the follower. The results of the simulations and the discussions are presented in the following.

The paths of the motion of the leaders and the follower are shown in Fig. 8.22 for the two cases. The simulation is run for 30 s. The figure shows a minimal difference in the steady-state path of the follower in the two cases.

The global position components of the follower helicopter are plotted in Fig. 8.23. The lateral wind does not affect $x$ and $z$ components of motion, whereas the $y$ component is initially disturbed. However, the controller successfully keeps the $y$ component very close to the desired dynamic equilibrium state.

The orientation of the follower helicopter can be seen in Fig. 8.24. Once again, the mean values of the roll angles, after a transient response, are 1.6 and 7.0° for the with and without wind cases, respectively. These values are compatible with the result of the $l - \alpha$ controller because the equilibrium roll angle is inherent in the dynamics of the helicopter and the disturbances.

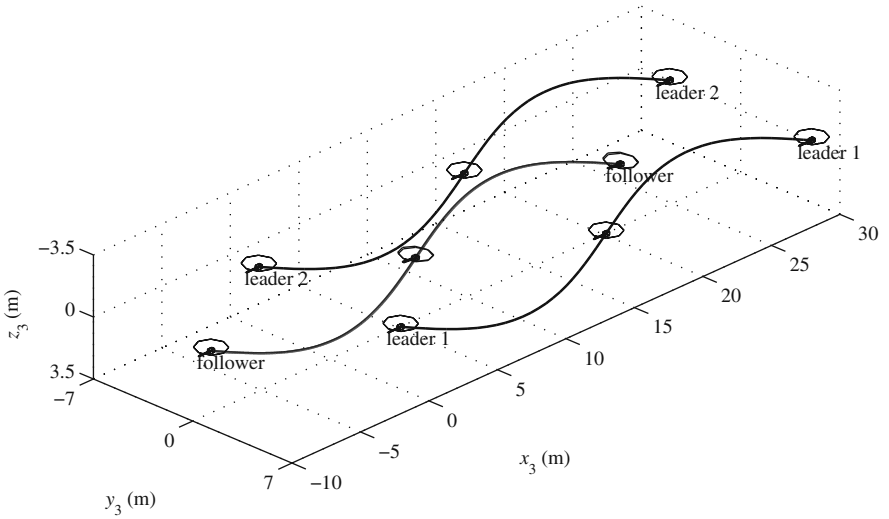


**Fig. 8.22** Path of the motion for the $l - l$ scheme. The effect of the wind on the path of the follower is negligible. Despite a 10 m/s lateral wind, the follower robustly follows the leaders with a given relative position

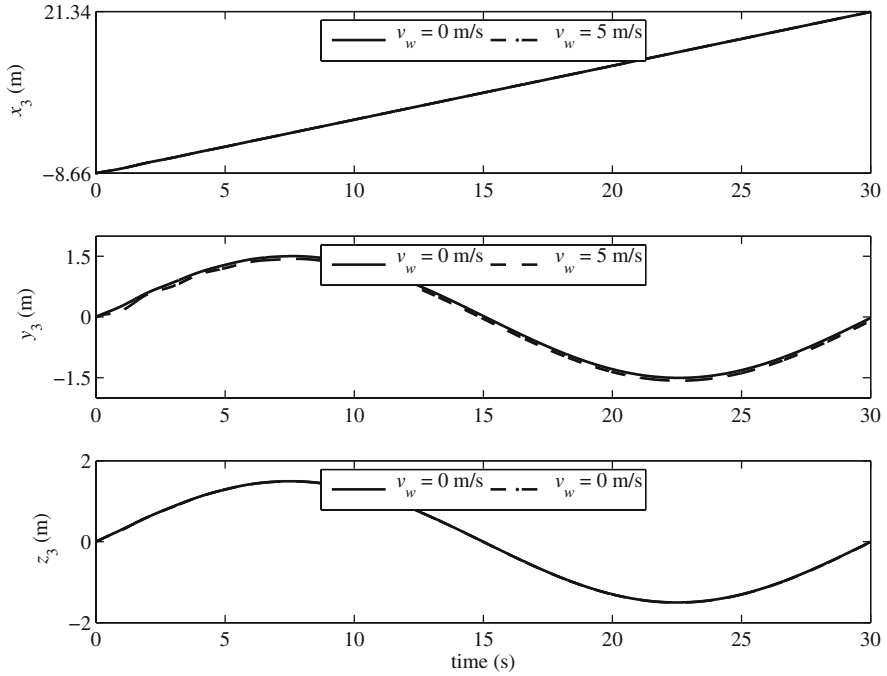


**Fig. 8.23** Position trajectories for the $l - l$ scheme. The follower's position trajectories for the zero and non zero wind force situations are very similar. The difference in the $y$ component is because the follower tries to reach a larger steady-state roll angle at which it can resist the lateral wind force when the wind is present

The four $l - l$ formation parameters or the control outputs are presented in Fig. 8.25. The steady-state values of the formation parameters do not experience any offsets in any case. The control points of the follower stays at a distance of 5 m with the control point of both leaders during the motion ($l_{13} = l_{23} = 5$ m). The follower helicopter moves at the same height of the two leaders ($\beta_{123} = 90^\circ$). And the follower helicopter faces straight ahead as instructed ($\psi_3 = 0^\circ$).

The four helicopter control inputs are shown in Fig. 8.26. The control forces are in general similar to that of the $l - \alpha$ controller. However, the inputs show more chatter. The chatter can be reduced by fine tuning the controller gain nonlinearity factor $\boldsymbol{\eta}$ and the boundary layer $\Phi$ for the $l - l$ controller.

*Example 8.5.* Consider eight helicopters with the inertail and geometrical properties given in Table 8.1. Assume that they initially make a rectangular formation with a grid size of 5 m while moving forward with a constant speed of 1 m/s. At time zero, they are commanded to form a desired spatial cubic formation with 5-m dimensions. Use the developed formation control schemes as building blocks to interconnect the group of eight autonomous helicopters in the initial rectangular formation. Derive the desired formation parameters that can map the initial formation parameters into the desired cubic formation. Show the change of formation resulting from the new

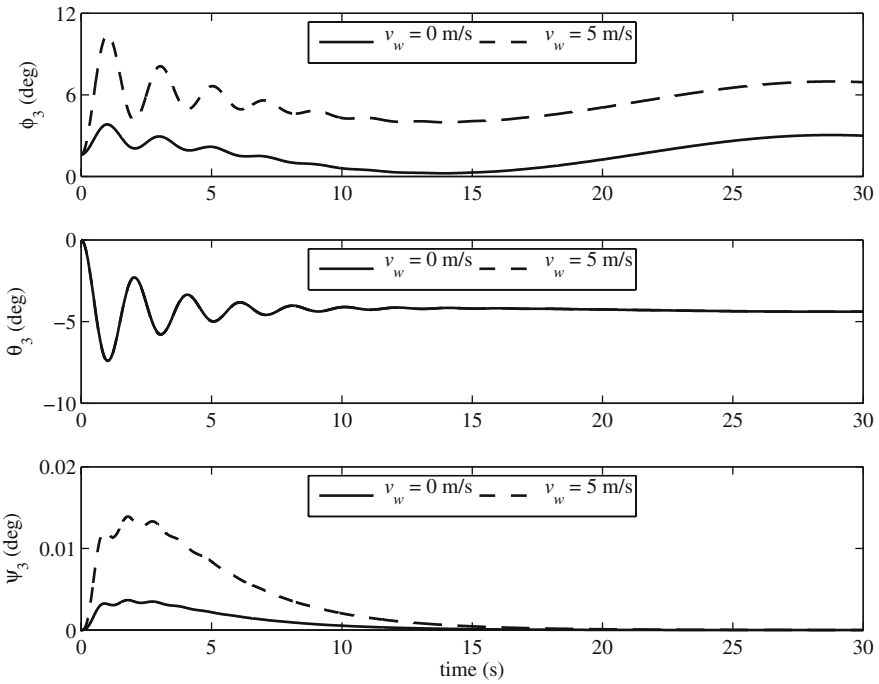


**Fig. 8.24** Orientation trajectories for the $l - l$ scheme. Except for the roll angle, the follower's orientation trajectories in the presence and the absence of the wind disturbance are fairly close. Since the follower tries to counteract the lateral wind force, the steady-state roll angle is larger when wind is present

desired formation parameters by simulation. Investigate the performance of the controllers for the following two cases.

(a) Assume no uncertainty in the inertial properties listed in Table 8.1.
(b) Assume +20% uncertainty in the inertial properties listed in Table 8.1.

*Solution.* The eight helicopters in their initial formation can be interconnected as shown in Fig. 8.27. In this figure, the leader-follower relationship is shown by arrows. The tip of the arrow correspond to a follower, while the tail of an arrow points to a leader. The formation scheme assignments, consistant with the arrows are as follows. Helicopter 1 is assigned to be the group leader. Helicopter 2 follows helicopters 1 using the $l$ –$\alpha$ scheme. Helicopter 3 follows helicopters 2 and 1 using the $l$ –$l$ scheme. Helicopter 4 follows helicopter 3 and 1 using the $l$ –$l$ scheme. Helicopters 5, 6, 7, and 8 follow helicopters 1, 2, 4, and 3, respectively, using the $l$ –$\alpha$ scheme.

The desired formation can be achieved by the same structure as that shown in Fig. 8.27. If the squares in the front and back of this structure are folded upward as shown in Fig. 8.28, a cubic formation is obtained. The desired formation parame-

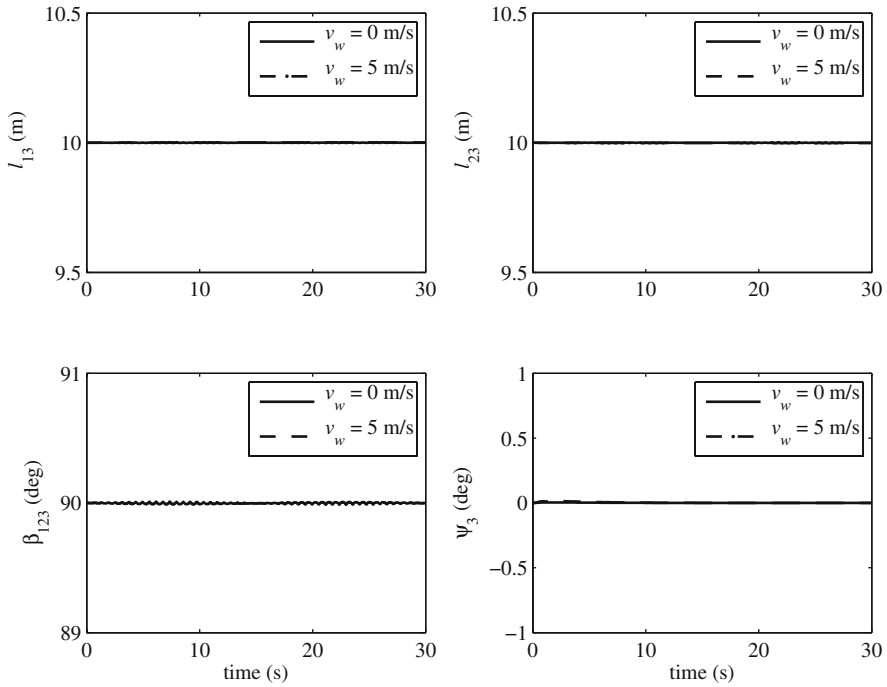


**Fig. 8.25** Output trajectories for the $l - l$ scheme. The formation parameters' steady-state values are not significantly affected by the lateral wind. The difference between the output trajectories for the two wind conditions are negligible

ters are derived using the sketch for the desired formation. The desired formation parameters are listed in Table 8.3.

The described change of formation is simulated for two different cases to show the robustness of the controllers in the presence of parameter uncertainty. In both cases, the control law is calculated based on the helicopters' nominal parameters listed in Table (8.1). However, the inertia parameters used with the dynamic model to simulate the response of the helicopters are different for the two cases. In the first case, the nominal mass and moment of inertia are used. In the second case, the helicopters' mass and moment of inertia is assumed to be 20% higher than the nominal values.

Figure 8.29 shows the 35-s motion of the helicopters for the two cases. The curves that show the path of the motion of each helicopter for the two different cases are either very close or completely coincident with each other. This shows the robustness of the formation controllers to parameter uncertainty. The helicopters successfully achieve the new desired formation after 35 s despite of the wind and inertia uncertainties.

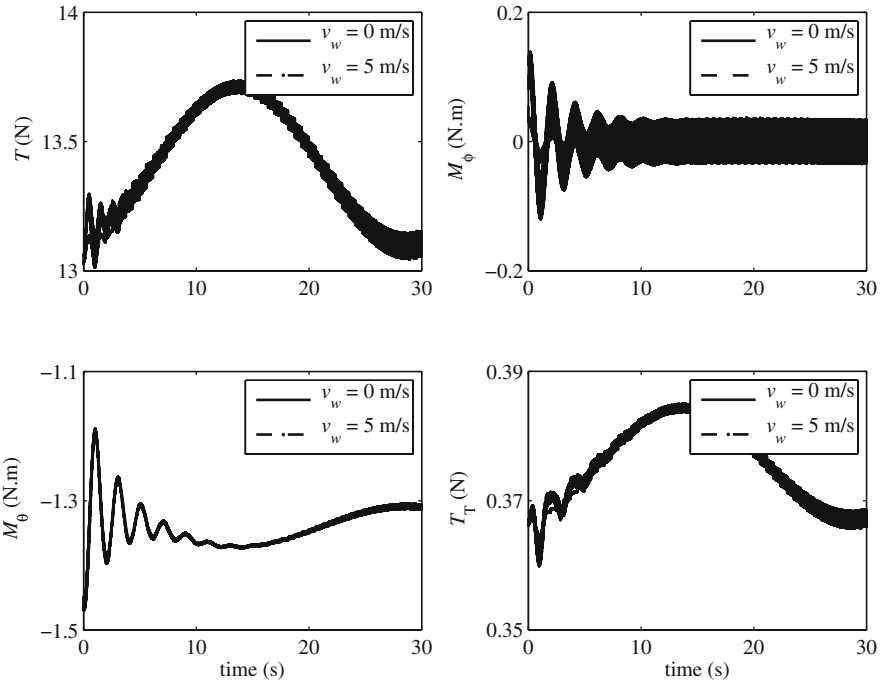


**Fig. 8.26** Input history for the $l - l$ scheme. The required thrust $T$ when wind exists is higher than that of the no wind situation. Other control inputs are also higher in presence of wind to counterbalance the higher thrust force, except $M_\phi$. This is because $T$ does not produce a moment about the helicopter's roll axis

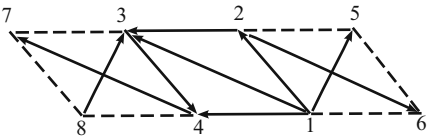


**Fig. 8.27** Eight helicopters in a planar rectangular formation. Helicopter 1 is the group leader. 2 follows 1 using the $l$ –$\alpha$ scheme. 3 follows 2 and 1 using the $l$ –$l$ scheme. 4 follows 3 and 1 using the $l$ –$l$ scheme. 5, 6, 7, and 8 follow 1, 2, 4, and 3, respectively, using the $l$ –$\alpha$ scheme

## Problems

**Problem 8.1.** Consider a small autonomous helicopter with the mass and geometrical properties listed in Table 8.1. Use Eq. (8.22) to calculate the trimming values.

**Problem 8.2.** Consider a small autonomous helicopter with the mass and geometrical properties listed in Table 8.1. The helicopter is hovering at an initial position of (0.0, 0.0, −2.0) m, while facing the positive $x_1$ axis. Simulate the motion of the helicopter under PID position control for the following given desired positions and headings.

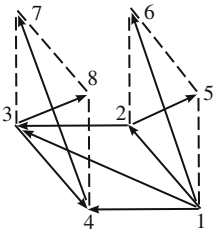


**Fig. 8.28** Eight helicopters in a spatial cubic formation. The formation structure of these eight helicopters are the same as that of the group shown in Fig. 8.27. However, the desired formation parameters for the helicopters 5, 6, 7, and 8 are different than that of the planar configuration of Fig. 8.27

**Table 8.3** Desired formation parameters for the cubic formation shown in Fig. 8.28

| Helicopter | Control scheme | Desired | | | |
|---|---|---|---|---|---|
| 1 | Leader | | | | |
| 2 | $l-\alpha$ | $l_{12}^d = 5\,\text{m}$ | $\alpha_{12}^d = 90^\circ$ | $z_{12}^d = 0\,\text{m}$ | $\psi_2^d = 0^\circ$ |
| 3 | $l-l$ | $l_{13}^d = 5\sqrt{2}\,\text{m}$ | $l_{23}^d = 5\,\text{m}$ | $\beta_{123}^d = 90^\circ$ | $\psi_3^d = 0^\circ$ |
| 4 | $l-l$ | $l_{13}^d = 5\,\text{m}$ | $l_{23}^d = 5\,\text{m}$ | $\beta_{123}^d = 90^\circ$ | $\psi_3^d = 0^\circ$ |
| 5 | $l-\alpha$ | $l_{12}^d = 5\,\text{m}$ | $\alpha_{12}^d = -90^\circ$ | $z_{12}^d = -5\,\text{m}$ | $\psi_2^d = 0^\circ$ |
| 6 | $l-\alpha$ | $l_{12}^d = 5\,\text{m}$ | $\alpha_{12}^d = 90^\circ$ | $z_{12}^d = -5\,\text{m}$ | $\psi_2^d = 0^\circ$ |
| 7 | $l-\alpha$ | $l_{12}^d = 5\,\text{m}$ | $\alpha_{12}^d = -90^\circ$ | $z_{12}^d = -5\,\text{m}$ | $\psi_2^d = 0^\circ$ |
| 8 | $l-\alpha$ | $l_{12}^d = 5\,\text{m}$ | $\alpha_{12}^d = 90^\circ$ | $z_{12}^d = -5\,\text{m}$ | $\psi_2^d = 0^\circ$ |

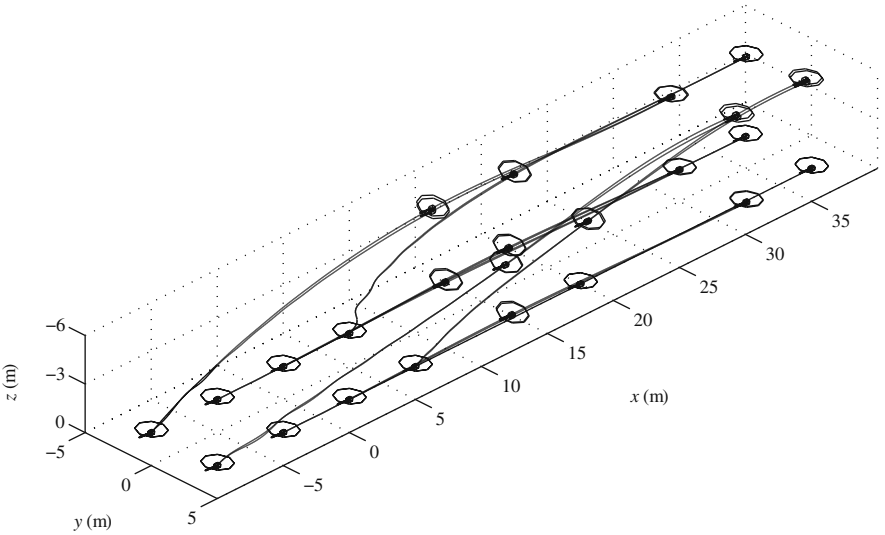


**Fig. 8.29** Formation change of a group of eight helicopters with and without model inertia uncertainty. The helicopters are initially in a planar rectangular formation as defined in Fig. 8.27. They change formation while continuing their maneuver after a new set of desired formation parameters are defined. The curves showing the path of the helicopters for the cases of 0% and +20% inertia uncertainty are either very close or coincident

(a) $(x_1^d, x_2^d, x_3^d, \psi^d) = (0.5\,\text{m}, 0.5\,\text{m}, -3.0\,\text{m}, -\pi/4\,\text{rad})$.
(b) $(x_1^d, x_2^d, x_3^d, \psi^d) = (1.5\,\text{m}, 1.6\,\text{m}, -4.0\,\text{m}, -\pi/4\,\text{rad})$.
(c) $(x_1^d, x_2^d, x_3^d, \psi^d) = (3.0\,\text{m}, 3.0\,\text{m}, -6.0\,\text{m}, -\pi/4\,\text{rad})$.

Is the PID controller successful in stabilizing the helicopter for all of the above scenarios? Why not? Discuss the simulation results.

**Problem 8.3.** Consider a small autonomous helicopter introduced in Problem 8.2. The helicopter is hovering under PID position control at a desired position of $(0.0, 0.0, -2.0)$ m, while facing the positive $x_1$ axis. Simulate the response of the helicopter if a constant wind starts blowing in the positive $x_2$ direction starting at time $t = 5$ s. Assume the wind force is

(a) 5% of the helicopter's weight,
(b) 10% of the helicopter's weight,
(c) 15% of the helicopter's weight,
(d) 20% of the helicopter's weight.

Is the PID controller successful in stabilizing the helicopter in presence of wind disturbance? Discuss the simulation results.

**Problem 8.4.** Consider the autonomous helicopter introduced in Problem 8.2, whose inertial and geometrical properties are shown in Table 8.1. Assume that the distance of the control point with the helicopter's center of gravity $d = 1$ m. The control point of the helicopter is initially at $(0.0, 0.0, -2.0)$ m, while the helicopter is facing the positive $x_1$ axis. Use a sliding mode control law to position the control point of the helicopter at the following desired points.

(a) $(x_{p1}^d, x_{p2}^d, x_{p3}^d, \psi^d) = (0.5\,\text{m}, 0.5\,\text{m}, -3.0\,\text{m}, -\pi/4\,\text{rad})$.
(b) $(x_{p1}^d, x_{p2}^d, x_{p3}^d, \psi^d) = (1.5\,\text{m}, 1.6\,\text{m}, -4.0\,\text{m}, -\pi/4\,\text{rad})$.
(c) $(x_{p1}^d, x_{p2}^d, x_{p3}^d, \psi^d) = (3.0\,\text{m}, 3.0\,\text{m}, -6.0\,\text{m}, -\pi/4\,\text{rad})$.

Is the sliding mode controller successful in stabilizing the helicopter for all of the above scenarios? Why? Discuss the simulation results.

**Problem 8.5.** Consider the autonomous helicopter introduced in Problem 8.2. Assume that the distance of the control point with the helicopter's center of gravity $d = 1$ m. The control point of the helicopter is stabilized using an sliding mode controller at the desired point $(0.0, 0.0, -2.0)$ m, while the helicopter is facing the positive $x_1$ axis. Simulate the response of the helicopter if a constant wind starts blowing in the positive $x_2$ direction starting at time $t = 5$ s. Assume the wind force is

(a) 5% of the helicopter's weight,
(b) 10% of the helicopter's weight,

(c) 15% of the helicopter's weight,
(d) 20% of the helicopter's weight.

Is the sliding mode controller successful in stabilizing the helicopter in presence of wind disturbance? What controller parameter can be tuned for a better performance? What is the effect of that controller parameter on input chatter?

**Problem 8.6.** Consider the autonomous helicopter introduced in Problem 8.2. Assume that the distance of the control point with the helicopter's center of gravity $d = 1$ m. Use a sliding mode control law to control the control point of the helicopter on the following desired trajectory.

$$\begin{aligned} x_{p1}^d(t) &= R\cos\frac{2\pi t}{\tau}, \\ x_{p2}^d(t) &= R\sin\frac{2\pi t}{\tau}, \\ x_{p3}^d(t) &= -H, \\ \psi^d(t) &= 0, \end{aligned}$$

where $R = 10$ m, $H = 2$ m, and $\tau = 60$ s. Note that this desired trajectory is a circle with radius $R$ parallel to the $x_1$-$x_2$ plane. It takes $\tau$ seconds for the helicopter to complete this circle. The helicopter is at rest at time zero and its initial position and orientation are as follows.

$$x_1(0) = 0 \text{ m}, \quad x_2(0) = 0 \text{ m}, \quad x_3(0) = -2 \text{ m},$$

$$\phi(0) = 0 \text{ rad}, \quad \theta(0) = 0 \text{ rad}, \quad \psi(0) = \pi/2 \text{ rad}.$$

Assume that the inertial properties of the helicopter can be uncertain up to 1.3 times of the nominal properties. This uncertainty can cover the pay load of the helicopter. Simulate the motion of the helicopter under control for three cases of uncertainty in the inertial properties of the helicopter. Assume that the actual inertial properties are

(a) 0.8 times the nominal values (−20% uncertainty),
(b) equal the nominal values (0% uncertainty),
(c) and 1.2 times the nominal values (20% uncertainty).

**Problem 8.7.** Find the singularities of the $l - \alpha$ control scheme by investigating the relative postions of the leader and follower helicopters for which the $l - \alpha$ formation parameters are undefined.

**Problem 8.8.** Find the singularities of the $l - l$ control scheme by investigating the relative postions of the leader and follower helicopters for which the $l - l$ formation parameters are undefined.

**Problem 8.9.** Three helicopters must maintain an equilateral triangular formation with 5-m legs in a horizontal plane, while two of them must fly side by side behind the other helicopter. Determine the desired geometrical formation parameters.

***Hint*:** Use both the $l-\alpha$ and $l-l$ formation control schemes to define a rigid formation structure consisting of one triangular cell. Note that using two $l-\alpha$ schemes does not connect two of the helicopters. If the motion of one of the unconnected helicopters is disturbed, the other helicopter will not sense the disturbance and cannot adjust its motion. Therefore, the triangular formation will be lost.

**Problem 8.10.** Consider the autonomous helicopter introduced in Problem 8.2. Assume that the center of gravity of the helicopter is coincident with the main rotor's axis (Fig. 8.1), such that $l_r = 0$ m. Assume that the distance of the control point with the helicopter's center of gravity $d = 1$ m. Use the $l-\psi$ control law to make the helicopter keep a distance of 8 m at a relative view angle of $\alpha_{12}^d = 90^\circ$ with a leader helicopter, while flying at the same altitude of the leader ($z_{12}^d = 0$ m) and heading parallel to the positive global $x$ direction ($\psi_2^d = 0^\circ$). The motion of the leader is defined by

$$\begin{aligned} x_1(t) &= V_x t, \\ y_1(t) &= V_y t, \\ z_1(t) &= V_z t, \end{aligned}$$

where $V_x = 1$ m/s and $V_y = V_z = 0.5$ m/s. The follower helicopter is initially at (0, 10, 0) m and is facing toward the positive global $x$ direction and has a zero initial velocity. Simulate the motion of the helicopter for the following two conditions.

(a) There is no wind.
(b) There is a constant wind with a 5 m/s speed blowing along the negative global $y$ direction.

**Problem 8.11.** Consider the autonomous helicopter introduced in Problem 8.2. Assume that the center of gravity of the helicopter is coincident with the main rotor's axis (Fig. 8.1), such that $l_r = 0$ m. Assume that the distance of the control point with the helicopter's center of gravity is $d = 1$ m. Use the $l-l$ control law to make the helicopter keep a distance of 8 m with two leader helicopters ($l_{13}^d = l_{23}^d = 8$ m), while flying slightly above the leaders ($\beta_{123}^d = 120^\circ$) and heading parallel to the positive global $x$ direction ($\psi_3^d = 0^\circ$). The motion of the leaders are defined by

$$\begin{aligned} x_1(t) &= V_x t, \\ y_1(t) &= V_y t + y_{10}, \\ z_1(t) &= V_z t, \end{aligned}$$

$$x_2(t) = V_x t,$$
$$y_2(t) = V_y t + y_{20},$$
$$z_2(t) = V_z t,$$

where $V_x = 1\,\text{m/s}$, $V_y = V_z = 0.5\,\text{m/s}$, $y_{10} = -5\,\text{m}$, $y_{20} = 5\,\text{m}$, and $R_y = R_z = 1.5\,\text{m}$. The follower helicopter is initially at $(0, -10)$ m and is facing toward the positive global $x$ direction, and has an initial velocity of 0 m/s. Simulate the motion of the helicopter for the following two conditions.

(a) There is no wind.
(b) There is a constant wind with a 5 m/s speed blowing along the negative global $y$ direction.

# Appendix A
# Mathematics

In Appendix A, some more information about the mathematical methods used on this book are presented.

## A.1 Null Space

**Definition**. The null space of an $m \times n$ matrix $\mathbf{A}$ is the set

$$\aleph(\mathbf{A}) = \left\{\mathbf{x} \in \Re^n : \mathbf{A}\mathbf{x} = \mathbf{0}\right\}, \tag{A.1}$$

where $\mathbf{0}$ denotes the zero vector with m components. The matrix equation $\mathbf{A}\mathbf{x} = \mathbf{0}$ is equivalent to a homogeneous system of linear equations:

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & 0 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & 0 \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & 0. \end{array} \tag{A.2}$$

From this viewpoint, the null space of A is the same as the solution set to the homogeneous system.

*Example A.1.* Consider the matrix

$$\mathbf{A} = \begin{bmatrix} 2 & 3 & 5 \\ -4 & 2 & 3 \end{bmatrix}. \tag{A.3}$$

The matrix can be seen as a transformation that maps a 3D space onto a 2D space. The null space of this matrix consists of all vectors $(x, y, z) \in \Re^3$ for which

$$\begin{bmatrix} 2 & 3 & 5 \\ -4 & 2 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{A.4}$$

This can be written as a homogeneous system of linear equations involving $x$, $y$, and $z$:

$$\begin{aligned} 2x + 3y + 5z &= 0, \\ -4x + 2y + 3z &= 0. \end{aligned} \tag{A.5}$$

The null space of $\mathbf{A}$ is precisely the set of solutions to these equations, which represent two planes in the 3D space. The solution to the system (A.5) is the 3D line of the intersection of the two planes. That line is the null space of the matrix $\mathbf{A}$. This means that the transformation $\mathbf{A}$ transforms any point on this line in the 3D space to the origin of the destination 2D space, i.e., the 2D $\mathbf{0}$ vector.

## A.2 Rank

**Definition**. The maximal number of linearly independent columns of $\mathbf{A}$ is called the *column rank* of a matrix $\mathbf{A}$. Similarly, the maximal number of linearly independent rows of $\mathbf{A}$ is called the *row rank*.

The column rank and the row rank are always equal. Therefore, they are simply called the rank of $\mathbf{A}$. The common notation used for showing the rank of a matrix $\mathbf{A}$ is either rk($\mathbf{A}$) or rank$\mathbf{A}$.

The maximum possible rank of an $m \times n$ matrix is the smaller of the number of columns or the number of rows, or in mathematical notation, rank ($\mathbf{A}$) $\leq \min(m, n)$. A matrix is said to have a full rank when its rank is as large as possible. Otherwise, the matrix is rank deficient.

*Example A.2.* Consider the following $3 \times 4$ matrix.

$$\mathbf{A} = \begin{bmatrix} 2 & 4 & 1 & 3 \\ -1 & -2 & 1 & 0 \\ 0 & 0 & 2 & 2 \end{bmatrix}. \tag{A.6}$$

One can notice by observation that the second column is twice the first column and that the fourth column equals the sum of the first and the third. The first and the third columns are linearly independent. Since there are only two independent columns, the rank of $\mathbf{A}$ is two.

## A.3 Singular Value Decomposition (SVD)

**Definition**. The singular value decomposition (SVD) is an important method in linear algebra for factorizing a rectangular real or complex matrix. The SVD has several applications in signal processing and statistics such as computing the pseudo-inverse of a rectangular matrix, approximating matrices, and determining the rank, range, and null space of a matrix.

Suppose $\mathbf{A}$ is an $m \times n$ matrix whose entries come from the field of real numbers $\Re$.[1] Then, there exists a factorization of the form

$$\mathbf{A} = \mathbf{u}\boldsymbol{\sigma}\mathbf{v}^T, \tag{A.7}$$

where $\mathbf{u}$ is an $m \times m$ matrix over $\Re$, the matrix $\boldsymbol{\sigma}$ is $m \times n$ with nonnegative numbers on the diagonal (as defined for a rectangular matrix) and zeros off the diagonal, and $\mathbf{v}^T$ is the transpose[2] of $\mathbf{v}$, which denotes an $n \times n$ matrix over $\Re$. Such a factorization is called a SVD of $\mathbf{A}$. The matrix $\mathbf{A}$ can be looked at as a transformation between two input and output spaces, with sample vectors $\mathbf{x}$ and $\mathbf{y}$, as

$$\mathbf{y} = \mathbf{A}\mathbf{x} = \mathbf{u}.[\boldsymbol{\sigma}.(\mathbf{v}^T.\mathbf{x})]. \tag{A.8}$$

The matrices $\mathbf{v}$, $\mathbf{u}$, and the diagonal entries of $\boldsymbol{\sigma}$ can be interpreted as follows.

- The matrix $\mathbf{v}^T$, which contains a set of orthonormal vectors, "rotates" the input vector as a the first part of the transformation $\mathbf{A}$.
- The matrix $\boldsymbol{\sigma}$ contains the singular values on the diagonal. The diagonal values can be thought of as scalar factors by which each corresponding rotated input direction is "scaled" to give a corresponding output direction.
- The matrix $\mathbf{u}$, which contains a set of orthonormal vectors, "maps" the result of the "rotation" and "scaling" of the input vector onto the output space as the final part of the transformation $\mathbf{A}$.

A common convention is to order the diagonal values $\sigma_{ii}$ in non increasing fashion. In this case, the diagonal matrix is uniquely determined by $\mathbf{A}$ (though the matrices $\mathbf{u}$ and $\mathbf{v}$ are not).

### *A.3.1 Computing SVD*

The matrices $\mathbf{v}^T$, $\boldsymbol{\sigma}$, and $\mathbf{v}$ are claculated via the eigenvalue decomposion of the matrix $\mathbf{A}^T\mathbf{A}$. Assume an $m \times n$ size for the matrix $\mathbf{A}$. First, the eigenvalues and the eigenvectors of the matrix $\mathbf{A}^T\mathbf{A}$ are claculated. The eigenvalues are sorted from the greatest to the smallest values, which are named $\lambda_1$ to $\lambda_n$. The $m \times n$ matrix $\boldsymbol{\sigma}$ is formed with all zero enteries except the diagonal, which are filled with $\sigma_{ii} = \sqrt{\lambda_i}$ $(i = 1, \ldots, \max(m, n))$. The eignvectors corresponding to $\lambda_1$ to $\lambda_n$ form the $n$ columns of the matrix $\mathbf{v}$. Finally, the columns $\mathbf{u}_i$ of the $m \times m$ matrix $\mathbf{u}$ is formed using the following relation.

---

[1] The enteries of $\mathbf{A}$, in general, can be complex numbers. However, that case is not relevant to the redundant and hyper-redundant robot applications.

[2] If the enteries of $\mathbf{A}$ are complex numbers, $\mathbf{v}^T$ is the conjugate transpose of $\mathbf{v}$.

$$\mathbf{u}_i = \begin{cases} \frac{1}{\sigma_{ii}}\mathbf{A}\mathbf{v}_i & i = 1, \ldots, \max(m, n) \\ \mathbf{0} & i = \max(m, n) + 1, \ldots, m \end{cases}, \tag{A.9}$$

where $\mathbf{v}_i$ the column $i$ of the matrix $\mathbf{v}$. If for any $i$ the value of $\sigma_{ii}$ is zero, $\mathbf{u}_i$ is assigned zero. Also, if $m < n$, the second realtion in Eq. (A.9) is ignored.

*Example A.3.* Consider the following rectangular matrix.

$$\mathbf{A} = \begin{bmatrix} 1\,0\,0\,0\,2 \\ 0\,0\,3\,0\,0 \\ 0\,0\,0\,0\,0 \\ 0\,6\,0\,0\,0 \end{bmatrix}. \tag{A.10}$$

The eigenvalues of the matrix $\mathbf{A}^T\mathbf{A}$ are

$$\lambda_1 = 36, \lambda_2 = 9, \lambda_3 = 5, \lambda_4 = 0, \lambda_5 = 0.$$

The eigenvectors of the matrix $\mathbf{A}^T\mathbf{A}$ are

$$\begin{aligned} \mathbf{v}_1^T &= [0, 1, 0, 0, 0], \\ \mathbf{v}_2^T &= [0, 0, 1, 0, 0], \\ \mathbf{v}_3^T &= [\sqrt{0.2}, 0, 0, 0, \sqrt{0.8}], \\ \mathbf{v}_4^T &= [0, 0, 0, -1, 0], \\ \mathbf{v}_5^T &= [\sqrt{0.8}, 0, 0, 0, -\sqrt{0.2}]. \end{aligned}$$

Based on these eigenvalues and eigenvectors, the matrices $\boldsymbol{\sigma}$ and $\mathbf{v}$ are determined as

$$\boldsymbol{\sigma} = \begin{bmatrix} 6\,0 & 0 & 0\,0 \\ 0\,3 & 0 & 0\,0 \\ 0\,0 & \sqrt{5} & 0\,0 \\ 0\,0 & 0 & 0\,0 \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} 0\,0 & \sqrt{0.2} & 0 & \sqrt{0.8} \\ 1\,0 & 0 & 0 & 0 \\ 0\,1 & 0 & 0 & 0 \\ 0\,0 & 0 & -1 & 0 \\ 0\,0 & \sqrt{0.8} & 0 & -\sqrt{0.2} \end{bmatrix}. \tag{A.11}$$

Now, since $\max(m, n) = \max(4, 5) = 4$, the columns $\mathbf{u}_i$ $(i = 1, \ldots, 4)$ are determined from Eq. (A.9). Finally the matrix $\mathbf{u}$ is

$$\mathbf{u} = \begin{bmatrix} 0\,0\,1\,0 \\ 0\,1\,0\,0 \\ 0\,0\,0\,0 \\ 1\,0\,0\,0 \end{bmatrix}. \tag{A.12}$$

One can verify that $\mathbf{u}\boldsymbol{\sigma}\mathbf{v}^T$ is, indeed, equal to $\mathbf{A}$.

## A.4 Pseudo-Inverse for a Rectangular Matrix

The SVD can be used for computing the pseudo-inverse of a matrix. Indeed, the pseudo-inverse of the matrix $\mathbf{A}$ with SVD of $\mathbf{u}\boldsymbol{\sigma}\mathbf{v}^T$ is

$$\mathbf{A}^\dagger = \mathbf{v}\boldsymbol{\sigma}^*\mathbf{u}^T, \tag{A.13}$$

where $\boldsymbol{\sigma}^*$ is the transpose of $\boldsymbol{\sigma}$ with every nonzero entry replaced by its reciprocal.

*Example A.4.* Consider the matrix $\mathbf{A}$ given in Example A.3. The psedue-inverse of this matrix is

$$\mathbf{A}^\dagger = \mathbf{v}\boldsymbol{\sigma}^*\mathbf{u}^T = \begin{bmatrix} 1/5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/6 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 2/5 & 0 & 0 & 0 \end{bmatrix}, \tag{A.14}$$

where

$$\boldsymbol{\sigma}^* = \begin{bmatrix} 1/6 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 \\ 0 & 0 & 1/\sqrt{5} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{A.15}$$

## A.5 Bisection Method

The bisection method is an numerical method that finds the root of a single equation in a continuous given interval. It approaches the root by repeatedly dividing the interval in half and selecting the resulting subinterval in which a root exists. The bisection method is less efficient than Newton's method but it is numnerically much more robust.

Assume that the equation can be expressed as $f(x) = 0$, where $x$ is an independent variable. First, two points $a$ and $b$, for which $f(a)$ and $f(b)$ have opposite signs, must be selected heuristically. Also, the function $f$ must be continuous in the interval $[a, b]$. With these assumptions, the intermediate value theorem predicts that $f$ must have at least one root in the interval $[a, b]$. The bisection method divides the interval in two by computing $c = (a + b)/2$. There are now two possibilities: either $f(a)$ and $f(c)$ have opposite signs or $f(c)$ and $f(b)$ have opposite signs. Since the root belongs to the interval for which the sign change accurs, it is picked for the next bisection application. The bisection algorithm is then applied recursively to the sub-interval where the sign change occurs. The bisection method converges to a root of $f$, if $f$ is a continuous function on the interval $[a, b]$ and $f(a)f(b) < 0$.

**Table A.1** The results of the bisection algorithm

| Iteration | $a$ | $b$ | $c$ | $e$ |
|---|---|---|---|---|
| 1 | 1.500000 | 3.000000 | 2.250000 | 0.750000 |
| 2 | 1.500000 | 2.250000 | 1.875000 | 0.375000 |
| 3 | 1.875000 | 2.250000 | 2.062500 | 0.187500 |
| 4 | 1.875000 | 2.062500 | 1.968750 | 0.093750 |
| 5 | 1.968750 | 2.062500 | 2.015625 | 0.046875 |
| 6 | 1.968750 | 2.015625 | 1.992188 | 0.023438 |
| 7 | 1.992188 | 2.015625 | 2.003906 | 0.011719 |
| 8 | 1.992188 | 2.003906 | 1.998047 | 0.005859 |
| 9 | 1.998047 | 2.003906 | 2.000977 | 0.002930 |
| 10 | 1.998047 | 2.000977 | 1.999512 | 0.001465 |
| 11 | 1.999512 | 2.000977 | 2.000244 | 0.000732 |

Normally, when the half width of the resulting sub-interval is smaller than a treshold $\epsilon$, the iterations are stopped and the midpoint of the sub-interval in which the sign change happens is accepted as the approximate solution. In this case, the absolute error is halved at each iteration. After $n$ iterations, the maximum absolute error is

$$\frac{|b-a|}{2^{n+1}}. \tag{A.16}$$

The following algorithm summerizes the bisection method.

1. Initialize $a$, $b$, and the convergence treshold $\epsilon$.
2. Check if $f(a).f(b)$ is less than zero. If not, restart from Step 1.
3. Calculate $c = (a + b)/2$.
4. If $f(a).f(c)$ is less than zero, assign the value of $c$ to $b$, i.e., $b := c$. Otherwise, assign the value of $c$ to $a$, i.e., $a := c$.
5. The approximate solution is $x = (a + b)/2$.
6. If $|b - a|/2$ is smaller than the treshold $\epsilon$, accept the approximate solution $x$ and stop the iterations. Otherwise, repeat the steps from Step 3.

*Example A.5.* Consider the following equation:

$$f(x) = x^2 - 4. \tag{A.17}$$

Here, the solution must be found within the treshold $\epsilon = 0.001$. The interval $[a, b] = [0, 3]$ is picked, for which $f(0).f(3) = -20 < 0$. Applying the bisection method algorithm recursively results in the sub-intervals listed in Table A.1, along with the approximate solution and the maximum error of the approximation. Since the root of the assumed equation is known, it is easy to see the correctness of the maximum error estimates.

# Appendix B
# Control Methods Review

## B.1 Feedback Linearization

In the feedback linearization method, a different control input representation is defined for the system such that the dynamic equation seems similar to that of a linear system. Then, a controller is designed for the linear system using any classical control method. The nonlinear control law, then, is derived by transforming the equations for the new input back to the initial input representation. The steps that can be used for model based control of many mechanical systems is summerized below.

(a) Most of the mathematical models describing mechanical systems can be written in the *companion form* or the *controllability canonical form*.

$$\mathbf{x}^{(n)} = \mathbf{f}(\mathbf{x}) + \mathbf{b}(\mathbf{x})\mathbf{u}, \tag{B.1}$$

where the superscript $(n)$ indicates the $n$th time derivative, $\mathbf{u}$ is the original control input, $x$ is the vectorial output of the control system, and $\mathbf{f}(\mathbf{x})$ and $\mathbf{b}(\mathbf{x})$ are nonlinear matrix functions of the output vector $\mathbf{x}$. Note that the matrix $\mathbf{b}(\mathbf{x})$ must have full rank (non zero determinant) for all $\mathbf{x}$; otherwise, the output system will not be controllable.

(b) A new control input $\mathbf{v}(\mathbf{x}, \mathbf{u})$ is defined such that it can be determined uniquely for a given $\mathbf{x}$ and $\mathbf{u}$. The new control input must captures all the nonlinearities of the system. For a system with the campanion form, all the nonlinearities are in the terms $\mathbf{f}(\mathbf{x})$ and $\mathbf{b}(\mathbf{x})$. Therefore, the new input can be defined as

$$\mathbf{v} = \mathbf{f}(\mathbf{x}) + \mathbf{b}(\mathbf{x})\mathbf{u}. \tag{B.2}$$

Note that when the system's model is rewritten in terms of this new input, its form looks similar to a simple linear system (also called the multiple integrator form):

$$\mathbf{x}^{(n)} = \mathbf{v}. \tag{B.3}$$

(c) A desired asymptotically stable error behavior is defined for the output system. The error is defined as

$$\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_d(t). \tag{B.4}$$

where the subscript $d$ denotes the desired output trajectory. For each component of the error vector $\mathbf{e}$, denoted by $e_i$, the desired behavior is

$$e_i^{(n)} + k_{1,i} e_i^{(n-1)} + \ldots + k_{n,i} e = 0. \tag{B.5}$$

To ensure that the error behavior (B.5) is asymptotically stable, the gains $k_{i,j}$ are chosen such that all the roots of the characteristic polynomial of Eq. (B.5) have negative real parts. The charateristic polynomial corresponding to Eq. (B.5) is defined as

$$p^n + k_{1,i} p^{n-1} + \ldots + k_{n,i} p = 0. \tag{B.6}$$

(d) The new control input $\mathbf{v}$ is calculated such that, when applied to the system, it can generate the desired error behavior (B.5). This calculation is done by solving Eq. (B.5) for $\mathbf{x}^{(n)}$ and substituting the result into Eq. (B.3).

$$\mathbf{v} = \mathbf{x}_d^{(n)} - \mathbf{k}_1 \mathbf{e}^{(n-1)} - \ldots - \mathbf{k}_n \mathbf{e}. \tag{B.7}$$

where $\mathbf{k}_i$ is a diagonal matrix with the diagonal enteries $k_{1,i}$.

(e) Finally, the original control input is calculated using the first definition of the new control input in Eq. (B.2):

$$\mathbf{u} = \mathbf{b}^{-1}(\mathbf{x}) \left[ \mathbf{x}_d^{(n)} - \mathbf{f}(\mathbf{x}) - \mathbf{k}_1 \mathbf{e}^{(n-1)} - \ldots - \mathbf{k}_n \mathbf{e} \right]. \tag{B.8}$$

Note that since the output system is assumed to be controllable, the matrix $\mathbf{b}(\mathbf{x})$ has full rank for all $\mathbf{x}$ and is invertable.

## B.2 Sliding Mode Control

Modeling inaccuracies in the form of parameter uncertainties or unmodeled dyanmics can have strong negative effects on the performance of a nonlinear control system. There always have been a need for control methods that can perform well despite of modeling inaccuracies. The sliding mode method is one of them.

A sliding mode controller is composed of two parts, a nominal part and a discontinuous part. The nominal part, also known as the *equivalent control*, is similar to a feedback linearizing control law and captures the nonlinearity of the nominal plant's model. The discontinuous part, which is the unique feature of the sliding mode control method, aims at dealing with the modeling inaccuracies.

In the following, a systematic approach for designing a sliding mode controller for nonlinear systems, whose mathematical model can be written in the controllability canonical form, is presented.

(a) The nominal input–output representation of the system is written in the controllability canonocal form.

$$\mathbf{x}^{(n)} = \hat{\mathbf{f}}(\mathbf{x}) + \hat{\mathbf{b}}(\mathbf{x})\mathbf{u}, \tag{B.9}$$

where the superscript $(n)$ indicates the $n$th time derivative, $\mathbf{u}$ is the original control input, $x$ is the vectorial output of the control system, and $\hat{\mathbf{f}}(\mathbf{x})$ and $\hat{\mathbf{b}}(\mathbf{x})$ are nonlinear matrix functions of the output vector $\mathbf{x}$ with the "nominal" parameters of the system. Note that the matrix $\hat{\mathbf{b}}(\mathbf{x})$ must have full rank (non zero determinant) for all $\mathbf{x}$; otherwise, the output system will not be controllable.

(b) The tracking error of the outputs are defined.

$$\mathbf{e}(t) = \mathbf{x}(t) - \mathbf{x}_d(t), \tag{B.10}$$

where $\mathbf{x}_d(t)$ is the desired trajectroy for the outputs. For the tracking task to be achievable using a finite control input $\mathbf{u}$, the intial desired outputs must be equal to the outputs at time zero, i.e., $\mathbf{x}_d(0) = \mathbf{x}(0)$ or $\mathbf{e}(0) = \mathbf{0}$.

(c) The desired behavior of the error components are defined as

$$s_i(x_i, t) = \left(\frac{d}{dt} + \lambda_i\right)^{n-1} e_i. \tag{B.11}$$

where $\lambda_i$'s are stricyly positive. Note that, in general, the initial conditions of the output error derivatives are not zero. Therefore, there is no guarantee that $s_i(x_i(0), 0)$ is zero. The idea is that a control law must be designed to, first, ensure that variables $s_i$ approach zero and stay at zero despite modeling accuracies, and second, the error components follow the desired trajectory (B.11) when $s_i$'s have been stabilized at zero. A sliding mode control law that can meet the above mentioned requirements has two parts. The first part, called the *equivalent control*, ensures the realization of the second requirement. The second part, called the *discontinuous term*, ensures that the first requirement is satisfied.

(d) The equivalent control part of a sliding mode controller is designed solely based on the nominal model of the input–output system. Assuming that $s_i$'s are already stabilized at zero, one can determine the equivalent control. First, to simplify the notation, the desired error behavior is written as

$$s_i(x_i, t) = x_i^{(n-1)} - s_{ri}, \tag{B.12}$$

where

$$s_{ri} = x_{di}^{(n-1)} - \left(\frac{d}{dt} + \lambda_i\right)^{n-1} e_i + e_i^{(n-1)}. \tag{B.13}$$

The matrix form of Eq. (B.12) is

$$\mathbf{s}(\mathbf{x}, t) = \mathbf{x}^{(n-1)} - \mathbf{s}_r. \tag{B.14}$$

Since it is assumed that $s_i$'s are already stabilized, $\mathbf{s}(\mathbf{x}, t) = \mathbf{0}$. Now, the equivalent control can be derived by differentiating Eq. (B.14) and substituting for $\mathbf{x}^{(n)}$ from Eq. (B.9) and solving for $\mathbf{u}$. The notation $\hat{\mathbf{u}}$ is used for the equivalent control.

$$\hat{\mathbf{u}} = \hat{\mathbf{b}}^{-1}\left(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r\right). \tag{B.15}$$

(e) A discontinuous term is added to the equivalent control to guarantee that the parameter $\mathbf{s}$ approaches zero regardless of the error initial condition and uncertainty in the model parameters.

$$\mathbf{u} = \hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})), \tag{B.16}$$

where $\mathbf{K}$ is a positive-definite diagonal matrix and sgn($\mathbf{s}$) returns a vector with the sign of the components of $\mathbf{s}$. Although the control law (B.16) seems complete, the discontinuity gain $\mathbf{K}$ must still be determined such that the parameter $\mathbf{s}$ converges to zero despite inaccuracies in the nominal dynamic model.

(f) To determine the discontinuity gain $\mathbf{K}$, a Lyapunov function is defined as

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{s}. \tag{B.17}$$

This Lyapunov function is admissible because it is positive for all values of $\mathbf{s}$ and is only zero when $\mathbf{s}$ is identically zero. With these properties, if one can show that the time derivative of the Lyapunov function is always negative and is only zero when $\mathbf{s}$ is identically zero, then, one can conclude that $\mathbf{s}$ converges to zero from any initial condition and remains at zero. The first derivative of the Lyapunov function (B.17) is

$$\dot{V} = \mathbf{s}^T\dot{\mathbf{s}}. \tag{B.18}$$

Substituting for $\dot{\mathbf{s}}$ from Eq. (B.14) results in

$$\dot{V} = \mathbf{s}^T(\mathbf{x}^{(n)} - \dot{\mathbf{s}}_r). \tag{B.19}$$

Substituting for $\mathbf{x}^{(n)}$ from the input–output relation (B.9) yields

$$\dot{V} = \mathbf{s}^T(\mathbf{f} + \mathbf{bu} - \dot{\mathbf{s}}_r). \tag{B.20}$$

Note that in the above equation, "$\hat{.}$" is dropped from the notation to indicate the use of the uncertain input–output model. Substituting for $\mathbf{u}$ from Eq. (B.16) gives

$$\dot{V} = \mathbf{s}^T(\mathbf{f} + \mathbf{b}\hat{\mathbf{b}}^{-1}(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})) - \dot{\mathbf{s}}_r). \tag{B.21}$$

Note that in the ideal conditions (e.g., when there is no uncertainty and $\mathbf{b} = \hat{\mathbf{b}}$), $\mathbf{b}\hat{\mathbf{b}}^{-1}$ is the identity matrix $\mathbf{I}_m$. Therefore, it is logical to define the uncertainty in $\mathbf{b}$ in terms of the difference of $\mathbf{b}\hat{\mathbf{b}}^{-1}$ and the identity matrix $\mathbf{I}_m$.

$$\delta = \mathbf{b}\hat{\mathbf{b}}^{-1} - \mathbf{I}_m. \tag{B.22}$$

Equation (B.21) is rewritten in terms of the uncertainty $\delta$.

$$\begin{aligned}\dot{V} &= \mathbf{s}^T(\mathbf{f} + (\mathbf{I}_m + \delta)(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r - \mathbf{K}\text{sgn}(\mathbf{s})) - \dot{\mathbf{s}}_r),\\ &= \mathbf{s}^T(\mathbf{f} - \hat{\mathbf{f}} + \delta(-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r) - \mathbf{K}\text{sgn}(\mathbf{s}) - \delta\mathbf{K}\text{sgn}(\mathbf{s})).\end{aligned} \tag{B.23}$$

As this stage, some bounds must be assumed for the parameter uncertainties. These bounds are defined for the components of $\mathbf{f} - \hat{\mathbf{f}}$ and $\delta$.

$$|f_i - \hat{f}_i| \le F_i, \quad |\delta_{ij}| \le \Delta_{ij}, \qquad i = 1, \ldots, 4. \tag{B.24}$$

If $\mathbf{f} - \hat{\mathbf{f}}$ and $\delta$ in Eq. (B.23) are replaced by the uncertainty bounds defined in Eq. (B.24), the right hand side of the resulting equation increases in value. Hence, the equal sign should be replaced by an inequality sign.

$$\dot{V} \le \mathbf{s}^T(\mathbf{F} + \Delta| - \hat{\mathbf{f}} + \dot{\mathbf{s}}_r| - \mathbf{K}\text{sgn}(\mathbf{s}) + \Delta\mathbf{K}\text{sgn}(\mathbf{s})). \tag{B.25}$$

Equation (B.25) in component notation becomes

$$\begin{aligned}\dot{V} &\le \sum_{i=1}^{4} s_i(F_i + \sum_{j=1}^{4}(\Delta_{ij}| - \hat{f}_j + \dot{s}_{rj}|) - K_i\text{sgn}(s_i) + \sum_{j=1}^{4}\Delta_{ij}K_j\text{sgn}(s_j)),\\ &\le -\sum_{i=1}^{4}|s_i|(-F_i - \sum_{j=1}^{4}(\Delta_{ij}| - \hat{f}_i + \dot{s}_{ri}|) + K_i - \sum_{j=1}^{4}\Delta_{ij}K_j).\end{aligned} \tag{B.26}$$

Equation (B.26) implies that if $K_i$'s are found such that

$$-F_i - \sum_{j=1}^{4}(\Delta_{ij}| - \hat{f}_i + \dot{s}_{ri}|) + K_i - \sum_{j=1}^{4}\Delta_{ij}K_j = \eta_i, \quad i = 1, \ldots, 4, \tag{B.27}$$

where $\eta_i$ are positive numbers, then, the rate of the Lyapunov function becomes

$$\dot{V} \leq -\sum_{i=1}^{4} |s_i|\eta_i. \tag{B.28}$$

In other words, if $K_i$'s are determined from Eq. (B.27), the rate of the Lyapunov function is always negative, except when all $s_i$'s are zero, at which point the rate of the Lyapunov function is zero. This implies that the Lyapunov function decreases to zero despite of any initial value and remains at zero. Consequently, $s_i$'s approach zero and remain zero because $V$ is zero if and only if all $s_i$'s are zero.

For simplicity of implementation, Eq. (B.27) is written in the matrix form:

$$\mathbf{K}_v = (\mathbf{I}_m - \Delta)^{-1}(\mathbf{F} + \Delta|-\hat{\mathbf{f}} + \dot{\mathbf{s}}_r| + \eta). \tag{B.29}$$

where $\mathbf{K}_v = [K_1, K_2, K_3, K_4]^T$ and $\eta = [\eta_1, \eta_2, \eta_3, \eta_4]^T$.

# References


1. Akishita, S., Kawamura, S., Hisanobu, T.: Velocity potential approach to path planning for avoiding moving obstacles. Advanced Robotics **7**(5), 463–478 (1996)
2. Angeles, J.: Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms, 2nd edn. Springer-Verlag (2002)
3. Arai, T., Ota, J.: Motion planning of multiple mobile robots using virtual impedance. Journal of Robotics and Mechatronics **8**(1), 67–74 (1996)
4. Baillieul, J.: Kinematic programming alternatives for redundant manipulators. In: Proceeding IEEE International Conference on Robotics and Automation, pp. 722–728. St. Louis, MO (1985)
5. Baillieul, J.: Avoiding obstacles and resolving kinematic redundancy. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1698–1704 (1986)
6. Balch, T., Arkin, R.C.: Behavior-based formation control for multi-robot teams. IEEE Transactions on Robotics and Automation **6**, 926–939 (1998)
7. Balch, T., Arkin, R.C.: Behavior-based formation control for multirobot teams. IEEE Transactions on Robotics and Automation **14**, 926–939 (1998)
8. van den Berg, J.P., Overmars, M.H.: Roadmap-based motion planning in dynamic environments. IEEE Transactions on Robotics **21**(5), 885–897 (2005)
9. Buckingham, R., Graham, A.: Snaking around in a nuclear jungle. International Journal of Industrial Robot **32**(2), 120–127 (2005)
10. Canny, J.: The Complexity of Robot Motion Planning. The MIT Press, Cambridge, Massachusetts (1987)
11. Cao, Z., Xie, L., Zhang, B., Wang, S., Tan, M.: Formation constrained multi-robot system in unknown environments. In: Proceedings – IEEE International Conference on Robotics and Automation, vol. 1, pp. 735–740. Taipei, Taiwan (2003)
12. Chester, C.R.: Techniques in Partial Differential Equations. McGraw-Hill, New York, NY, USA (1971)
13. Chirikjian, G.S., Burdick, J.W.: A geometric approach to hyper-redundant manipulator obstacle avoidance. The ASME Journal of Mechanical Design **114**(4), 580–585 (1992)
14. Chirikjian, G.S., Burdick, J.W.: A modal approach to hyper-redundant manipulator kinematics. IEEE Transactions on Robotics and Automation **10**(3), 343–354 (1994)
15. Chirikjian, G.S., Burdick, J.W.: Kinematically optimal hyper-redundant manipulator configurations. IEEE Transactions on Robotics and Automation **11**(6), 794–806 (1995)
16. Chirikjian, G.S., Burdick, J.W.: Kinematics of hyper-redundant robot locomotion. IEEE Transactions on Robotics and Automation **11**(6), 781–793 (1995)
17. Chiu, S.: Task compatibility of manipulator postures. International Journal of Robotics Research **7**(5), 13–21 (1988)
18. Chung, H., Sastry, S.S.: Autonomous helicopter formation using model predictive control. Collection of Technical Papers – AIAA Guidance, Navigation, and Control Conference 2006 **1**, 459–473 (2006)

19. Colbaugh, R., Seraji, H., Glass, K.: Obstacle avoidance of redundant robots using configuration control. Internatioanl Journal of Robotics Research **6**, 721–744 (1989)
20. Connolly, C.I., Burnd, J.B., Weiss, R.: Path planning using laplace's equation. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 2102–2106 (1990)
21. Cosio, F.A., Castaneda, M.P.: Autonomous robot navigation using adaptive potential fields. Mathematical and Computer Modelling **40**(9–10), 1141–1156 (2004)
22. Desai, J., Ostrowski, J., Kumar, V.: Controlling formations of multiple mobile robots. In: Proceedings of 1998 IEEE International Conference on Robotics and Automation, pp. 2864–2869. Leuven, Belgium (1998)
23. Desai, J.P.: A graph theoretic approach for modeling mobile robot team formations. International Journal of Robotics and Automation **19**(11), 511–525 (2002)
24. Desai, J.P., Ostrowski, P.J., Kumar, V.: Modeling and control of formations of nonholonomic mobile robots. IEEE Transactions on Robotics and Automation **6**, 905–908 (2001)
25. Dubey, R., Euler, J., Babock, S.: An efficient gradient projection optimization scheme for a seven-degree-of-freedom redundant robot with spherical wrist. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 28–36. Philadelphia, PA (1988)
26. Duman, H., Hu, H.: United we stand, divided we fall: Team formation in multiple robot applications. Journal of Robotic Systems **16**(4), 153–161 (2001)
27. Egeland, O.: Task-space tracking with redundant manipulators. IEEE Journal of Robotics and Automation **3**, 471–475 (1987)
28. Eklund, J.M., Sprinkle, J., Sastry, S.: Implementing and testing a nonlinear model predictive tracking controller for aerial pursuit/evasion games on a fixed wing aircraft. In: Proceedings of the American Control Conference, vol. 3, pp. 1509–1514. Portland, OR, United States (2005)
29. Fahimi, F.: Sliding mode formation control for under-actuated surface vessels. IEEE Transactions on Robotics **23**(3), 617–622 (2007)
30. Fahimi, F., Ashrafiuon, H., Nataraj, C.: An improved inverse kinematic and velocity solution for spatial hyper-redundant robots. IEEE Transactions on Robotics and Automation **18**(1), 103–107 (2002)
31. Fahimi, F., Ashrafiuon, H., Nataraj, C.: Obstacle avoidance for spatial hyper-redundant manipulators using harmonic potential functions and the mode shape technique. Journal of Robotic Systems **20**(1), 23–33 (2003)
32. Fossen, T.I.: Guidance and control of ocean vehicles. John Wiley & Sons, UK (1994)
33. Ge, S.S., Cui, Y.J.: Dynamic motion planning for mobile robot using potential field method. Autonomous Robots **1**, 207–222 (2002)
34. Ginsberg, J.H.: Advanced Engineering Dynamics, 2 edn. Cambridge University Press, Cambridge (1998)
35. Golub, G., Loan, C.V.: Matrix Computations, 2nd edn. John Hopkins University Press, Baltimore, MD (1989)
36. Gravagne, I., Walker, I.D.: Properties of minimum infinity-norm optimization applied to kinematically redundant robots. In: Proceeding of the IEEE International Conference on Intelligent Robots and Systems, vol. 1, pp. 152–160 (1998)
37. Guldner, J., Utkin, V.I., Bauer, R.: A three-layered hierarchical path control system for mobile robots: Algorithms and experiments. Robotics and Automation Systems **14**, 133–147 (1995)
38. Gulec, N., Unel, M.: A novel algorithm for the coordination of multiple mobile robots. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **3733 NCS**, 422–431 (2005)
39. Hirota, K., Kuwabara, T., Kenichi, I., Miyanohara, A., Ohdachi, H., Ohsawa, T., Takeuchi, W., Yubazaki, N., Ohtani, M.: Robots moving in formation by using neural network and radial basis functions. In: Proceedings of the IEEE International Conference on Fuzzy Systems, vol. 1, pp. 91–94. Yokohama, Japan (1995)
40. Hopcrof, J.E., Schwartz, J.T., Sharir, M.: On the complexity of motion planning for multiple independent objects; pspace-hardness of the warehouseman's problem. International Journal of Robotics Research **3**(4), 76–88 (1984)

41. Ihle, I.F., Jouffroy, J., Fossen, T.I.: Formation control of marine surface craft: A Lagrangian approach. IEEE Journal of Oceanic Engineering **31**(4), 922–934 (2006)
42. Ihle, I.F., Skjetne, R., Fossen, T.I.: Nonlinear formation control of marine craft with experimental results. In: Proceedings of the IEEE Conference on Decision and Control, vol. 1, pp. 680–685. Nassau, Bahamas (2004)
43. Jing, X.J., Wang, Y.C.: Control of behavior dynamics for motion planning of mobile robots in uncertain environments. In: Proceedings of the 2005 IEEE International Conference on Mechatronics, ICM '05, vol. 2005, pp. 364–369. Taipei, Taiwan (2005)
44. Kazerounian, K., Wang, Z.: Global versus local optimization in redundancy resolution of robotics manipulators. International Journal of Robotics Research **7**(5), 3–12 (1988)
45. Kim, D.H., Wang, H.O., Ye, G., Shin, S.: Decentralized control of autonomous swarm systems using artificial potential functions: Analytical design guidelines. In: Proceedings of the IEEE Conference on Decision and Control, vol. 1, pp. 159–164. Nassau, Bahamas (2004)
46. Kim, J., Khosla, P.K.: Real-time obstacle avoidance using harmonic potential functions. IEEE Transactions on Robotics and Automation **3**, 338–349 (1992)
47. Klein, A., Huang, C.H.: Review of pseudo-inverse control for use with kinematically redundant manipulators. IEEE Transactions on Systems, Man, and Cybernetics **SMC-13**(2), 245–250 (1983)
48. Klein, C.: Use of redundancy in design of robotic systems. In: Proceedings of the 2nd International Symposium on Robotic Research, pp. 207–214. Kyoto, Japan (1985)
49. Kuethe, A.M., Chow, C.Y.: Foundation of Aerodynamics: Bases of Aerodynamic Design, 4th edn. Wiley, New York, NY, USA (1986)
50. Latombe, J.: Robot Motion Planning. Kluwer Academic Publishers, Boston, Massachusetts (1991)
51. LaValle, S.M., Hutchinson, S.A.: Optimal motion planning for multiple robots having independent goals. IEEE Transactions on Robotics and Automation **14**(6), 912–925 (1998)
52. Lawton, R.J., Young, B.J., Beard, R.W.: A decentralized approach to elementary formation maneuvers. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 1, pp. 2728–2743. San Francisco, CA, USA (2000)
53. Lee, S., Adams, T.M.: Spatial model for path planning of multiple mobile construction robots. Computer-Aided Civil and Infrastructure Engineering **19**(4), 231–245 (2004)
54. Lewis, M.A., Tan, K.H.: High precision formation control of mobile robots using virtual structures. Autonomous Robots **4**, 387–403 (1997)
55. Liang, Y., Lee, H.H.: Avoidance of multiple obstacles for a mobile robot with nonholonomic constraints. In: American Society of Mechanical Engineers, Dynamic Systems and Control Division (Publication) DSC, vol. 74 DSC, no. 2 PART B, pp. 1657–1663. Orlando, FL, United States (2005)
56. Lin, Z., Patel, R., Balafoutis, C.: Augmented impedance control: An approach to impact reduction for kinematically redundant manipulators. Journal of Robotic Systems **12**, 301–313 (1995)
57. Luca, A.D., Oriolo, G., Samson, C.: Robot Motion Planning and Control: Feedback Control of a Nonholonomic Car-Like Robot. Springer-Verlag, Berlin, DE (1998)
58. Ma, S., Kobayashi, I.: Kinematics of hyper-redundant robot locomotion. Obstacle Avoidance Control Scheme for the Moray Arm on the Basis of Posture Space Analysis **32**(2), 163–172 (2000)
59. Maciejewski, A., Klein, C.: The singular value decomposition: Computation and application to robotics. International Journal of Robotics Research **8**, 63–79 (1989)
60. Maciejewski, A., Klein, C.A.: Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. International Journal of Robotics Research **4**(3), 109–117 (1985)
61. Marchese, F.M., Negro, M.D.: Path-planning for multiple generic-shaped mobile robots with MCA. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) **3993 NCS–III**, 264–271 (2006)

62. Mochiyama, H., Shimemura, E., Kobayashi, H.: Shape control of manipulators with hyper degrees of freedom. International Journal of Robotics Research **18**(6), 584–600 (1999)
63. Nakamura, Y., Hanafusa, H.: Inverse kinematic solutions with singularity robustness for manipulator control. ASME Journal of Dynamic Systems, Measurment, and Control **108**, 163–171 (1986)
64. Nakamura, Y., Hanafusa, H.: Optimal redundancy control of robot manipulators. International Journal of Robotics Research **6**(1), 32–42 (1987)
65. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving non-symmetric linear systems. SIAM Journal on Scientific Computing **7**(3), 856–869 (1986)
66. Schwartz, J.T., Sharir, M.: On the piano movers' problem: III. Coordinating the motion of several independent bodies amidst polygon barriers. International Journal of Robotics Research **2**(3), 46–75 (1983)
67. Sciavicco, L., Siciliano, B.: A solution algorithm to the inverse kinematic problem of redundant manipulators. IEEE Journal of Robotics and Automation **4**, 403–410 (1988)
68. Seraji, C., Colbaugh, R.: Singularity-robustness and task prioritization in configuration control of redundant robots. In: Proceedings of the 29th IEEE Conference on Decision and Control, pp. 3089–3095 (1990)
69. Seraji, H.: Configuration control of redundant manipulators: Theory and implementation. IEEE Transactions on Robotics and Automation **5**, 472–490 (1989)
70. Seraji, H.: Task options for redundancy resolution using configuration control. In: Proceedings of the 30th IEEE Conference on Decision and Control, pp. 2793–2798 (1991)
71. Seraji, H., Colbaugh, R.: Improved configuration control for redundant robots. Journal of Robotic Systems **7**(6), 897–928 (1989)
72. Shao, J., Xie, G., Yu, J., Wang, L.: Leader-following formation control of multiple mobile robots. In: Proceedings of the 20th IEEE International Symposium on Intelligent Control, ISIC '05 and the 13th Mediterranean Conference on Control and Automation, MED '05, vol. 2005, pp. 808–813. Limassol, Cyprus (2005)
73. Sheu, C., Young, K.: Heuristic approach to robot path planning based on task requirements using a genetic algorithm. Journal of Intelligent and Robotic Systems: Theory and Applications **1**, 65–88 (1996)
74. Sugar, T., Kumar, V.: Decentralized control of cooperating mobile manipulators. In: IEEE Internatrional Conferences on Robotics and Automation, pp. 2916–2921. Leuven, Belgium (1998)
75. Sugiyama, S., Akishita, S.: Path planning for mobile robot at crossroads by using hydrodynamic potential. In: Proceedings of 1998 Japan-USA Symposium on Flexible Automation, pp. 595–602 (1998)
76. Suh, K., Hollerbac, J.: Local versus global torque optimization of redundant manipulators. In: Proceeding IEEE International Conference on Robotics and Automation, pp. 619–624 (1987)
77. Vidal, R., Rashid, S., Sharp, C., Shakernia, O., Kim, J., Sastry, S.: Pursuit-evasion games with unmanned ground and aerial vehicles. In: Proceedings – IEEE International Conference on Robotics and Automation, vol. 3, pp. 2948–2955. Seoul (2001)
78. Warren, C.: Multiple robot path coordination using artificial potential fields. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 1, pp. 500–505 (1990)
79. Wilton, D.R., Rao, S.M., Glisson, A.W., Schaubert, D.H., Al-Bundak, O.M., Butler, C.M.: Potential integrals for uniform and linear source distributions on polygonal and polyhedral domains. IEEE Transactions on Antennas and Propagation **AP-32**(3), 276–281 (1984)
80. Yamaguchi, H.: Cooperative hunting behavior by mobile robot troops. International Journal of Robotics Research **9**, 931–940 (1999)
81. Yoshikawa, T.: Analysis and control of robot manipulators with redundancy. In: Proceedings of the First International Symposium on Robotic Research, pp. 735–747. Cambridge, MA (1984)
82. Yun, X., Tan, K.: Wall-following method for escaping local minima in potential field based motion planning. In: Proceeding of the 8th International Conference on Advanced Robotics, ICAR'97, pp. 421–426. Monterey, CA, USA (1997)

83. Zelinski, S., Koo, T.J., Sastry, S.: Optimization-based formation reconfiguration planning for autonomous vehicles. In: Proceedings–IEEE International Conference on Robotics and Automation, vol. 3, pp. 3758–3763. Taipei, Taiwan (2003)
84. Zexiang, L., Canny, J.F.: Nonholonomic Motion Planning. Kluwer Academic Publishers, Norwell, Massachusetts (1993)
85. Zhang, Y., Valavanis, K.P.: Sensor-based 2-D potential panel method for robot motion planning. Robotica **1**, 81–89 (1996)
86. Zhang, Y., Valavanis, K.P.: A 3-D panel method for robot motion planning. Robotica **1**, 421–434 (1997)

# Index