INTRODUCTION TO ROBOTICS
MECHANICS, PLANNING, AND CONTROL

F. C. PARK AND K. M. LYNCH
# Contents

1 Preview ........................................... 1

2 Configuration Space ..................................... 9
   2.1 Degrees of Freedom of a Rigid Body ................ 10
   2.2 Degrees of Freedom of a Robot ..................... 13
       2.2.1 Robot Joints .................................... 13
       2.2.2 Grübler’s Formula ............................... 14
   2.3 Configuration Space: Topology and Representation .... 19
       2.3.1 Configuration Space Topology ................. 19
       2.3.2 Configuration Space Representation ........... 21
   2.4 Configuration and Velocity Constraints ............... 25
   2.5 Task Space and Workspace ........................... 28
   2.6 Summary ........................................... 31
   2.7 Exercises ......................................... 33

3 Rigid-Body Motions ..................................... 51
   3.1 Rigid-Body Motions in the Plane ................. 53
   3.2 Rotations and Angular Velocities ................ 59
       3.2.1 Rotation Matrices .................................. 59
       3.2.2 Angular Velocity .................................. 65
       3.2.3 Exponential Coordinate Representation of Rotation . 68
   3.3 Rigid-Body Motions and Twists ..................... 77
       3.3.1 Homogeneous Transformation Matrices ........... 77
       3.3.2 Twists ........................................... 83
       3.3.3 Exponential Coordinate Representation of Rigid-Body Motions ......................... 90
   3.4 Wrenches ........................................... 93
   3.5 Summary ........................................... 95
   3.6 Software .......................................... 97
   3.7 Notes and References ................................ 98
   3.8 Exercises ......................................... 99
## 4 Forward Kinematics

4.1 Product of Exponentials Formula ........................................... 120
  4.1.1 First Formulation: Screw Axes Expressed in Base Frame .......... 120
  4.1.2 Examples ............................................................ 122
  4.1.3 Second Formulation: Screw Axes Expressed in End-Effector Frame ............................................................ 127
4.2 The Universal Robot Description Format .................................. 130
4.3 Summary ................................................................. 135
4.4 Software ................................................................. 137
4.5 Exercises ................................................................. 138

## 5 Velocity Kinematics and Statics

5.1 Manipulator Jacobian ..................................................... 155
  5.1.1 Space Jacobian ...................................................... 155
  5.1.2 Body Jacobian ...................................................... 159
  5.1.3 Relationship between the Space and Body Jacobian ............... 161
  5.1.4 Alternative Notions of the Jacobian ................................ 162
  5.1.5 Inverse Velocity Kinematics ...................................... 163
5.2 Statics of Open Chains .................................................. 163
5.3 Singularity Analysis ..................................................... 164
5.4 Manipulability ............................................................ 169
5.5 Summary ................................................................. 172
5.6 Software ................................................................. 173
5.7 Notes and References .................................................... 174
5.8 Exercises ................................................................. 175

## 6 Inverse Kinematics

6.1 Analytic Inverse Kinematics ............................................. 193
  6.1.1 6R PUMA-Type Arm ............................................... 193
  6.1.2 Stanford-Type Arms ................................................ 196
6.2 Numerical Inverse Kinematics ........................................... 198
  6.2.1 Newton-Raphson Method .......................................... 198
  6.2.2 Optimization Basics ................................................ 199
  6.2.3 Numerical Inverse Kinematics Algorithm .......................... 201
6.3 Inverse Velocity Kinematics ............................................. 205
6.4 A Note on Closed Loops ................................................ 207
6.5 Summary ................................................................. 207
6.6 Software ................................................................. 208
6.7 Notes and References .................................................... 208
6.8 Exercises ................................................................. 209

## 7 Kinematics of Closed Chains

7.1 Inverse and Forward Kinematics ....................................... 218
  7.1.1 3×RPR Planar Parallel Mechanism .................................. 219
  7.1.2 Stewart-Gough Platform .......................................... 221
  7.1.3 General Parallel Mechanisms .................................... 222
## CONTENTS

7.2 Differential Kinematics .............................................. 223  
7.2.1 Stewart-Gough Platform ............................................. 224  
7.2.2 General Parallel Mechanisms ..................................... 225  
7.3 Singularities .......................................................... 227  
7.4 Summary ............................................................... 231  
7.5 Notes and References .................................................. 232  
7.6 Exercises ............................................................... 234  

8 Dynamics of Open Chains ................................................. 241  
8.1 Lagrangian Formulation .................................................. 242  
8.1.1 Basic Concepts and Motivating Examples ......................... 242  
8.1.2 General Formulation .................................................. 246  
8.1.3 Understanding the Mass Matrix .................................... 248  
8.1.4 Lagrangian Dynamics vs. Newton-Euler Dynamics ................. 250  
8.2 Dynamics of a Single Rigid Body ..................................... 251  
8.2.1 Classical Formulation ................................................. 251  
8.2.2 Twist-Wrench Formulation ........................................... 255  
8.3 Newton-Euler Inverse Dynamics ....................................... 257  
8.3.1 Derivation ............................................................. 257  
8.3.2 Newton-Euler Inverse Dynamics Algorithm ....................... 260  
8.4 Dynamic Equations in Closed Form .................................. 260  
8.5 Forward Dynamics of Open Chains ................................... 263  
8.6 Dynamics in Task Space Coordinates ................................ 264  
8.7 Constrained Dynamics ................................................... 266  
8.8 Robot Dynamics in the URDF .......................................... 267  
8.9 Actuation, Gearing, and Friction ..................................... 268  
8.9.1 DC Motors and Gearing .............................................. 270  
8.9.2 Apparent Inertia ..................................................... 273  
8.9.3 Friction ............................................................... 276  
8.9.4 Joint and Link Flexibility ......................................... 277  
8.10 Summary ............................................................... 277  
8.11 Software ............................................................... 280  
8.12 Notes and References .................................................. 282  
8.13 Exercises ............................................................... 283  

9 Trajectory Generation ...................................................... 289  
9.1 Definitions ............................................................. 289  
9.2 Point-to-Point Trajectories .......................................... 290  
9.2.1 Straight-Line Paths .................................................. 290  
9.2.2 Time Scaling a Straight-Line Path ................................ 292  
9.3 Polynomial Via Point Trajectories ................................... 297  
9.4 Time-Optimal Time Scaling ........................................... 299  
9.4.1 The $(s, s)$ Phase Plane ........................................... 301  
9.4.2 The Time-Scaling Algorithm ....................................... 304  
9.4.3 Assumptions and Caveats ......................................... 306  
9.5 Summary ............................................................... 307
CONTENTS

9.6 Software ........................................ 308
9.7 Notes and References ............................ 308
9.8 Exercises ........................................... 310

10 Motion Planning ..................................... 315
10.1 Overview of Motion Planning .................... 315
10.1.1 Types of Motion Planning Problems ........... 316
10.1.2 Properties of Motion Planners ................ 317
10.1.3 Motion Planning Methods ..................... 318
10.2 Foundations ....................................... 319
10.2.1 Configuration Space Obstacles ................. 319
10.2.2 Distance to Obstacles and Collision Detection 323
10.2.3 Graphs and Trees ................................ 324
10.2.4 Graph Search ................................... 325
10.3 Complete Path Planners ......................... 328
10.4 Grid Methods ....................................... 329
10.4.1 Multi-Resolution Grid Representation ......... 331
10.4.2 Grid Methods with Motion Constraints ....... 332
10.5 Sampling Methods .................................. 337
10.5.1 The RRT ....................................... 338
10.5.2 The PRM ....................................... 342
10.6 Virtual Potential Fields ......................... 344
10.6.1 A Point in C-space ............................ 344
10.6.2 Navigation Functions ......................... 347
10.6.3 Workspace Potential ............................ 348
10.6.4 Wheeled Mobile Robots ....................... 349
10.6.5 Use of Potential Fields in Planners .......... 349
10.7 Nonlinear Optimization ............................ 349
10.8 Smoothing .......................................... 351
10.9 Summary .......................................... 352
10.10 Notes and References ............................ 354
10.11 Exercises ......................................... 356

11 Robot Control ....................................... 359
11.1 Control System Overview ....................... 360
11.2 Motion Control .................................... 360
11.2.1 Motion Control of a Multi-Joint Robot with Velocity Input 361
11.2.2 Motion Control of a Single Joint with Torque or Force Input 363
11.2.3 Motion Control of a Multi-joint Robot with Torque or Force Input 374
11.2.4 Task Space Motion Control .................... 376
11.3 Force Control ..................................... 378
11.4 Hybrid Motion-Force Control .................... 380
11.4.1 Natural and Artificial Constraints ............. 380
11.4.2 A Hybrid Controller .......................... 382
11.5 Impedance Control ................................ 384
Chapter 1

Preview

As an academic discipline, robotics is a relatively young field with highly ambitious goals, the ultimate one being the creation of machines that can behave and think like humans. This attempt to create intelligent machines naturally leads us to first examine ourselves—to ask, for example, why our bodies are designed the way they are, how our limbs are coordinated, and how we learn and perform complex tasks. The sense that the fundamental questions in robotics are ultimately questions about ourselves is part of what makes robotics such a fascinating and engaging endeavor.

Our focus in this book will be on the mechanics, planning and control of robot mechanisms. Robot arms are one familiar example. So are wheeled vehicles, as are robot arms mounted on wheeled vehicles. Basically, a mechanism is constructed by connecting rigid bodies, called links, together with joints, so that relative motion between adjacent links becomes possible. Actuation of the joints, typically by electric motors, then causes the robot to move and exert forces in desired ways.

The links of a robot mechanism can be arranged in serial fashion, like the familiar open chain arm shown in Figure 1.1(a). Robot mechanisms can also have closed loops, such as the Stewart-Gough platform shown in Figure 1.1(b). In the case of an open chain, all of its joints are actuated, while in the case of mechanisms with closed loops only a subset of its joints may be actuated.

Let us examine more closely the current technology behind robot mechanisms. The links are moved by actuators, which are typically electrically driven (e.g., DC or AC servo motors, stepper motors, even shape memory alloys), or sometimes by pneumatic or hydraulic cylinders, or internal combustion engines. In the case of rotating electric motors, they should ideally be lightweight, operate at relatively low rotational speeds (e.g., in the range of hundreds of RPM) and be able to generate large forces and torques. Since most currently available motors operate in the range of thousands of RPM, speed reduction devices with low slippage and backlash are often required. Belts, sprockets, and spur gears are usually not well-suited for this purpose; instead, specially designed low backlash gears, harmonic drives, and ball screws are used to simultaneously...
reduce speed and amplify the delivered torque. Brakes may also be attached to quickly stop the robot or to maintain a stationary posture.

Robots are also equipped with sensors to measure the position and velocity at the joints. For both revolute and prismatic joints, optical encoders measure the displacement, while tachometers measure their velocity. Forces at the links or at the tip can be measured using various types of force-torque sensors. Additional sensors may be used depending on the nature of the task, e.g., cameras, sonar and laser range finders to locate and measure the position and orientation of objects.

This textbook is about the mechanics, motion planning, and control of such robots. We now provide a preview of the later chapters.

**Chapter 2: Configuration Space**

![Open chain industrial manipulator](image1.png) ![Stewart-Gough platform](image2.png)

Figure 1.1: Open chain and closed chain robot mechanisms.

At its most basic level, a robot consists of rigid bodies connected by joints, with the joints driven by actuators. In practice the links may not be completely rigid, and the joints may be affected by factors such as elasticity, backlash, friction, and hysteresis. In this book we shall ignore these effects for the most part and assume all links are rigid. The most commonly found joints are revolute joints (allowing for rotation about the joint axis) and prismatic joints (allowing for linear translation along the joint axis). Revolute and prismatic joints have one degree of freedom (either rotation or translation); other joints, such as the spherical joint (also called the ball-in-socket joint), have higher degrees of freedom.

In the case of an open chain robot such as the industrial manipulator of Figure 1.1(a), all of its joints are independently actuated. This is the essential idea behind the **degrees of freedom** of a robot: it is the sum of all the independently actuated degrees of freedom of the joints. For open chains the degrees of freedom is obtained simply by adding up all the degrees of freedom
associated with the joints.

For closed chains like the Stewart-Gough platform shown in Figure 1.1(b), the situation is somewhat more complicated. First, joints with multiple degrees of freedom like the spherical joint are quite common. Second, it is usually not possible to independently actuate all of the joints—fixing a certain set of joints to prescribed values automatically determines the values of the remaining joints. For even more complicated closed chains with multiple loops and different joint types, determining the degrees of freedom may not be straightforward or intuitive.

A more abstract but equivalent definition of the degrees of freedom of a robot begins with the notion of its **configuration space**: a robot’s configuration is a complete specification of the positions and orientations of each link of a robot, and its configuration space is the set of all possible configurations of the robot. The degrees of freedom, then, is the minimum number of independent parameters required to specify the position and orientation of each of the links. Based on this definition we obtain a formula—Grübler’s formula—that relates the number of links and joints (including the degrees of freedom of each joint) comprising a robot with its degrees of freedom.

Robot motion planning and control both begin by choosing coordinates that parametrize the robot’s configuration space. Often the coordinates of choice are the joint variables, and the configuration space can be parametrized either explicitly or implicitly in terms of these joint variables. Also, to grasp and manipulate objects, a robot is typically equipped with an **end-effector**, e.g., a mechanical hand or gripper. The **task space**, also called the workspace, is the configuration space of the end-effector. In this chapter we study the various ways in which the configuration and task spaces of a robot can be parametrized.

**Chapter 3: Rigid-Body Motions**

This chapter addresses the problem of how to mathematically describe the motion of a rigid body moving in three-dimensional physical space. One convenient way is to attach a reference frame to the rigid body, and to develop a way to quantitatively describe the frame’s position and orientation as it moves. As a first step, we introduce the $3 \times 3$ matrix representation for describing a frame’s orientation; such a matrix is referred to as a **rotation matrix**.

A rotation matrix is parametrized by three independent coordinates. The most natural and intuitive way to visualize a rotation matrix is in terms of its **exponential coordinate** representation. That is, given a rotation matrix $R$, there exists some unit vector $\hat{\omega} \in \mathbb{R}^3$ and angle $\theta \in [0, \pi]$ such that the rotation matrix can be obtained by rotating the identity frame (that is, the frame corresponding to the identity matrix) about $\hat{\omega}$ by $\theta$. The exponential coordinates are defined as $\omega = \hat{\omega} \theta \in \mathbb{R}^3$, which is a three-parameter representation. There are several other well-known coordinate representations, e.g., Euler angles, Cayley-Rodrigues parameters, unit quaternions, that are further discussed in the appendix.

Another reason for focusing on the exponential description of rotations is
that they lead directly to the exponential description of rigid-body motions. The latter can be viewed as a modern geometric interpretation of the classical screw theory, or a modern screw theory. Keeping the classical terminology (e.g., twists, wrenches) as much as possible, we cover in detail the linear algebraic constructs of screw theory, including the unified description of linear and angular velocities as six-dimensional spatial velocities (twists), and an analogous description of three-dimensional forces and moments as six-dimensional spatial forces (wrenches).

Chapter 4: Forward Kinematics

For an open chain, the position and orientation of the end-effector are uniquely determined from the joint positions. This is precisely the forward kinematics problem for a robot: given a set of input joint values, find the output position and orientation of the reference frame attached to the end-effector. In this chapter we present the product of exponentials (PoE) formula for describing the forward kinematics of open chains. As the name implies, the PoE formula is directly derived from the exponential coordinate representation for rigid body motions. Aside from providing an intuitive and easily visualizable interpretation of the exponential coordinates as the twists of the joint axes, the PoE formula offers other advantages, like eliminating the need for link frames (only the base frame and end-effector frame are required, and these can be chosen arbitrarily).

In the appendix we also present the Denavit-Hartenberg (D-H) representation for forward kinematics. The D-H representation uses a fewer number of parameters, but requires that reference frames be attached to each link following special rules of assignment, which can be cumbersome. Details of the transformation from the D-H to the PoE representation are also provided in the appendix.

Chapter 5: Velocity Kinematics and Statics

Velocity kinematics refers to the relationship between joint rates and the linear and angular velocities of the end-effector frame. Central to velocity kinematics is the Jacobian of the forward kinematics. By multiplying the vector of joint rates by this matrix, the linear and angular velocities of the end-effector frame can be obtained for any given robot configuration. Kinematic singularities, which are configurations in which the end-effector frame loses the ability to move or rotate in one or more directions—imagine, for example, a two-link planar chain with its two links folded over each other—correspond to those configurations at which the Jacobian matrix fails to have maximal rank. The closely related and more general notion of the manipulability ellipsoid, whose shape indicates the ease with which the robot can move in various directions, is also derived from the Jacobian.

Finally, the Jacobian is also central to static force analysis. In static equilibrium settings, the Jacobian is used to determine what forces and torques need to be exerted at the input joints in order for the end-effector to apply a certain
force or moment in a particular direction. In this chapter we show how to obtain the Jacobian for general open chains, and its many practical uses in the above and other settings.

**Chapter 6: Inverse Kinematics**

In the inverse kinematics problem, given a desired position and orientation of the end-effector frame, the objective is to determine the set of joint positions that achieve this desired end-effector configuration. For open chain robots, the inverse kinematics is in general more involved than the forward kinematics: for a given set of joint values there usually exist a unique end-effector position and orientation, but for a particular end-effector position and orientation, there may exist multiple solutions, or even none at all.

In this chapter we first examine a popular class of six-dof open chain structures whose inverse kinematics admits a closed-form analytic solution. Iterative numerical algorithms are then derived for solving the inverse kinematics of general six-dof open chains. We also examine the inverse kinematics of redundant spatial open chains (that is, those with seven or more degrees of freedom) in the context of tracking a desired end-effector trajectory. For this problem, we present a solution for obtaining the corresponding input joint rates that relies on the generalized inverse of the forward kinematics Jacobian.

**Chapter 7: Kinematics of Closed Chains**

While open chains have unique forward kinematics solutions, closed chains will often have multiple forward kinematics solutions, and sometimes even multiple solutions for the inverse kinematics as well. Also, because closed chains possess both actuated and passive joints, the kinematic singularity analysis of closed chains presents subtleties not encountered in open chains. In this chapter we study the basic concepts and tools for the kinematic analysis of closed chains. We begin with a detailed case study of mechanisms like the planar five-bar linkage and the Stewart-Gough Platform. These results are then generalized into a systematic methodology for the kinematic analysis of more general closed chains.

**Chapter 8: Dynamics of Open Chains**

Dynamics is the study of motion taking into account the forces and torques that cause it. In this chapter we study the dynamics of open chain robots. Analogous to the notions of a robot’s forward and inverse kinematics, the forward dynamics problem seeks to determine the resulting joint trajectory for a given input joint torque profile. The inverse dynamics problem is concerned with determining the input joint torque profile for a desired joint trajectory. The dynamic equations relating the forces and torques to the motion of the robot’s links are given by a set of second-order ordinary differential equations.
We begin with a formulation of the dynamic equations for a single rigid body in terms of spatial velocities (twists), spatial accelerations, and spatial forces (wrenches). The dynamics for an open chain robot can be derived using one of two approaches. In the Newton-Euler approach, the equations of motion derived for each rigid body are merged in an appropriate fashion. Because of the open chain structure, the dynamics can be formulated recursively along the links. In the Lagrangian approach, a set of coordinates—referred to as the generalized coordinates in the classical dynamics literature—are first chosen to parametrize the configuration space. The sum of the potential and kinetic energies of the robot’s links are then expressed in terms of the generalized coordinates. These are then substituted into the Euler-Lagrange equations, which then lead to a set of second-order differential equations for the dynamics, expressed in the chosen coordinates for the configuration space.

In this chapter we examine both approaches to deriving a robot’s dynamic equations. Recursive algorithms for both the forward and inverse dynamics, as well as analytical formulations of the dynamic equations, are also presented.

Chapter 9: Trajectory Generation

What sets a robot apart from an automated machine is that it should be easily reprogrammable for different tasks. Different tasks require different motions, and it would be unreasonable to expect the user to specify the entire time-history of each joint for every task; clearly it would be desirable for the computer to “fill in the details” from a small set of task input data.

This chapter is concerned with the automatic generation of joint trajectories from this set of task input data. Formally, a trajectory consists of a path, which is a purely geometric description of the sequence of configurations achieved by a robot, typically described by a curve in the joint configuration space. A trajectory also consists of a time scaling, which specifies the times at which those configurations are reached.

Often the input task data is given in the form of an ordered set of joint values, called control points, together with a corresponding set of control times. Based on this data the trajectory generation algorithm produces a trajectory for each joint that satisfies various user-supplied conditions. In this chapter we will focus on three cases: (i) point-to-point straight line trajectories in both joint space and task space; (ii) smooth trajectories passing through a sequence of timed via points; (iii) minimum-time trajectories along specified paths. Finding paths that avoid collisions is the subject of the next chapter on motion planning.

Chapter 10: Motion Planning

This chapter addresses the problem of finding a collision-free motion for a robot through a cluttered workspace, while avoiding joint and torque limits, and other physical constraints imposed on the robot. The path planning problem is a subproblem of the general motion planning problem, and seeks to determine a collision-free path between a start and goal configuration, usually without
regard to the dynamics, the duration of the motion, or other constraints on the
motion or control inputs. There is no single planner applicable to all motion
planning problems. In this chapter we shall consider three basic approaches:
grid methods, sampling methods, and methods based on virtual potential fields.

Chapter 11: Robot Control

A robot arm can exhibit a number of different behaviors depending on the task
and its environment. It can act as a source of programmed motions for tasks
such as moving an object from one place to another, or tracing a trajectory for
manufacturing applications. It can act as a source of forces, for example when
grinding or polishing a workpiece. In tasks such as writing on a chalkboard, it
must control forces in some directions (the force pressing the chalk against the
board) and motions in others (motion in the plane of the board). In certain
applications, e.g., haptic displays, we may want the robot to act like a spring,
damper, or mass, controlling its position, velocity, or acceleration in response
to forces applied to it.

In each of these cases, it is the job of the robot controller to convert the
task specification to forces and torques at the actuators. Control strategies
to achieve the behaviors described above are known as motion (or position)
control, force control, hybrid motion-force control, and impedance con-
trol. Which of these behaviors is appropriate depends on both the task and
the environment. For example, a force control goal makes sense when the end-
effector is in contact with something, but not when it is moving in free space. We
also have a fundamental constraint imposed by mechanics, irrespective of the
environment: the robot cannot independently control both motions and forces
in the same direction. If the robot imposes a motion, then the environment
determines the force, and vice versa.

Most robots are driven by actuators that apply a force or torque to each
joint. Hence, to precisely control a robot would require an understanding of
the relationship between joint forces and torques and the motion of the robot;
this is the domain of dynamics. Even for simple robots, however, the dynamic
equations are usually very complex. Also, to accurately derive the dynamics
requires, among other things, precise knowledge of the mass and inertia of each
link, which may not be readily available. Even if they were, the dynamic equa-
tions would still not reflect physical phenomena like friction, elasticity, backlash,
and hysteresis.

Most practical control schemes compensate for these errors by using feedback. One effective method of industrial robot control is to neglect the robot’s
dynamics, and instead model each actuator as a scalar second-order linear sys-
tem. As such we first introduce basic concepts from linear control, and show
how they can be used to effectively control complex multi-dof robots.

This chapter also introduces some basic robot control techniques that as-
sume a dynamic model of the robot is available; such feedforward control
techniques use the dynamic model of the robot and its environment to deter-
mine actuator control inputs that achieve the desired task. Because of modeling
and other errors, feedforward control is rarely used by itself, but is often used in conjunction with feedback control. After considering feedback and forward strategies for model-based motion control, we then examine force control, hybrid motion-force control, and impedance control.

**Chapter 12: Grasping and Manipulation**

The focus of the previous chapters has been mostly on the internal characterization of the robot—its kinematics and dynamics, as well as methods for motion planning and control. In this chapter we now explicitly consider physical interactions between the robot and its environment. A fundamental problem addressed in this chapter is how to restrain a rigid body using a fixed number of point contacts. The first order of business is to characterize the nature of contacts between the robot and objects, or more generally, contact constraints between rigid bodies. As such it becomes necessary to consider friction. We examine the equations of motion for rigid body mechanics with friction, formulate the general problem of manipulation planning and grasping, and examine simplifications and assumptions that lead to certain basic solutions.

**Chapter 13: Wheeled Robots**

This chapter addresses the kinematics, motion planning, and control of wheeled robots that are subject to no-slip rolling constraints. Such constraints are fundamentally different from the loop closure constraints found in closed chains—the former are **holonomic**, the latter **nonholonomic**—and as such we begin with a discussion of nonholonomic constraints. We then examine the kinematics of some popular wheeled robots: car-like, differential drive, Dubins, and omni-directional robots. The **controllability** problem of determining whether a wheeled robot is able to move from a given initial configuration to an arbitrary final configuration is then examined. The chapter concludes with a discussion of motion planning and control algorithms for wheeled robots, including the problem of characterizing and finding optimal paths, and feedback control of wheeled robots.
Chapter 2

Configuration Space

A robot is mechanically constructed by connecting a set of bodies, called links, to each other using various types of joints. Actuators, such as electric motors, deliver forces or torques to the joints that cause the robot’s links to move. Usually an end-effector, such as a gripper or hand for grasping and manipulating objects, is attached to a specific link. All of the robots considered in this book have links that can be modeled as rigid bodies.

Perhaps the most fundamental question one can ask about a robot is, where is it? (In the sense of where the links of the robot are situated.) The answer is the robot’s configuration: a specification of the positions of all points of the robot. Since the robot’s links are rigid and of known shape,1 only a few numbers are needed to represent the robot’s configuration. For example, the configuration of a door can be represented by a single number, the angle \( \theta \) that the door rotates about its hinge. The configuration of a point lying on a plane can be described by two coordinates, \((x, y)\). The configuration of a coin lying heads up on a flat table can be described by three coordinates: two coordinates \((x, y)\) that specify the location of a particular point on the coin, and one coordinate \(\theta\) that specifies the coin’s orientation. (See Figure 2.1).

The above coordinates all share the common feature of taking values over a continuous range of real numbers. The smallest number of real-valued coordinates needed to represent a robot’s configuration is its degrees of freedom (dof). In the example above, the door (regarded as a robot) has one degree of freedom. The coin lying heads up on a table has three degrees of freedom. Even if the coin could lie either heads up or tails up, its configuration space still would have only three degrees of freedom; introducing a fourth variable to represent which side of the coin is facing up, this variable would then take values in the discrete set \{heads, tails\}, and not over a continuous range of real values like the other three coordinates \((x, y, \theta)\).

Definition 2.1. The configuration of a robot is a complete specification of the positions of every point of the robot. The minimum number \(n\) of real-

\[\text{Compare with trying to represent the configuration of a soft object like a pillow.}\]
10

Configuration Space

Figure 2.1: (a) The configuration of a door is described by its angle $\theta$. (b) The configuration of a point in a plane is described by coordinates $(x, y)$. (c) The configuration of a coin on a table is described by $(x, y, \theta)$.

valued coordinates needed to represent the configuration is the **degrees of freedom** (dof) of the robot. The $n$-dimensional space containing all possible configurations of the robot is called the **configuration space** (or C-space).

In this chapter we study the C-space and degrees of freedom of general robots. Since our robots are constructed of rigid links, we first examine the degrees of freedom of a single rigid body, followed by the degrees of freedom of general multi-link robots. We then study the shape (or topology) and geometry of C-spaces and their mathematical representation. The chapter concludes with a discussion of the C-space of a robot’s end-effector, or its **task space**. In the next chapter we study in more detail the various mathematical representations for the C-space of a single rigid body.

### 2.1 Degrees of Freedom of a Rigid Body

Continuing with the example of the coin lying on the table, choose three points $A$, $B$, and $C$ on the coin (Figure 2.2(a)). Once a coordinate frame $\hat{x}-\hat{y}$ is attached to the plane, the positions of these points in the plane are written $(x_A, y_A)$, $(x_B, y_B)$, and $(x_C, y_C)$. If these points could be placed independently anywhere in the plane, the coin would have six degrees of freedom—two for each of the three points. However, according to the definition of a rigid body, the distance between point $A$ and point $B$, denoted $d(A, B)$, is always constant regardless of where the coin is. Similarly, the distances $d(B, C)$ and $d(A, C)$ must be constant.

The following equality constraints on the coordinates $(x_A, y_A)$, $(x_B, y_B)$, and $(x_C, y_C)$ must hold:

\[x_A = x_B + d(B, A), \quad y_A = y_B + d(B, A),\]
\[x_C = x_A + d(A, C), \quad y_C = y_A + d(A, C).\]

The unit axes of coordinate frames are written with a hat, indicating they are unit vectors, in a non-italic font, e.g., $\hat{x}$, $\hat{y}$, and $\hat{z}$.
2.1. Degrees of Freedom of a Rigid Body

(A)       (B)                                       (C)

Figure 2.2: (a) Choosing three points fixed to the coin. (b) Once the location of A is chosen, B must lie on a circle of radius $d_{AB}$ centered at A. Once the location of B is chosen, C must lie at the intersection of circles centered at A and B. Only one of these two intersections corresponds to the “heads up” configuration. (c) The configuration of a coin in three-dimensional space is given by the three coordinates of A, two angles to the point B on the sphere of radius $d_{AB}$ centered at A, and one angle to the point C on the circle defined by the intersection of the a sphere centered at A and a sphere centered at B.

\[(x_C, y_C)\] must therefore always be satisfied:

\[
d(A, B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2} = d_{AB}
\]

\[
d(B, C) = \sqrt{(x_B - x_C)^2 + (y_B - y_C)^2} = d_{BC}
\]

\[
d(A, C) = \sqrt{(x_A - x_C)^2 + (y_A - y_C)^2} = d_{AC}.
\]

To determine the number of degrees of freedom of the coin, first choose the position of point A in the plane (Figure 2.2(b)). We may choose it to be anything we want, so we have two degrees of freedom to specify, \((x_A, y_A)\). Once \((x_A, y_A)\) is specified, the constraint \(d(A, B) = d_{AB}\) restricts the choice of \((x_B, y_B)\) to those points on the circle of radius \(d_{AB}\) centered at A. A point on this circle can be specified by a single parameter, e.g., the angle specifying the location of B on the circle centered at A; let’s call this angle \(\phi_{AB}\), and define it to be the angle that the vector \(\vec{AB}\) makes with the \(\hat{x}\)-axis.

Once we have chosen the location of point B, there are only two possible locations for C: at the intersections of the circle of radius \(d_{AC}\) centered at A, and the circle of radius \(d_{BC}\) centered at B (Figure 2.2(b)). These two solutions correspond to heads or tails. In other words, once we have placed A and B and chosen heads or tails, the two constraints \(d(A, C) = d_{AC}\) and \(d(B, C) = d_{BC}\) eliminate the two apparent freedoms provided by \((x_C, y_C)\), and the location of C is fixed. The coin has exactly three degrees of freedom in the plane, which can be specified by \((x_A, y_A, \phi_{AB})\).

Suppose we were to choose an additional point D on the coin. This new point introduces three additional constraints: \(d(A, D) = d_{AD}\), \(d(B, D) = d_{BD}\), and \(d(C, D) = d_{CD}\). One of these constraints is redundant, i.e., it provides no
new information; only two of the three constraints are independent. The two freedoms apparently introduced by the coordinates \((x_D, y_D)\) are then immediately eliminated by these two independent constraints. The same would hold for any other newly chosen point on the coin, so that there is no need to consider additional points.

We have been applying the following general rule for determining the number of degrees of freedom of a system:

\[
\text{Degrees of freedom} = (\text{Sum of freedoms of the points}) - (\text{Number of independent constraints}).
\]  

This rule can also be expressed in terms of the number of variables and independent equations that describe the system:

\[
\text{Degrees of freedom} = (\text{Number of variables}) - (\text{Number of independent equations}).
\]

This general rule can also be used to determine the number of freedoms of a rigid body in three dimensions. For example, assume our coin is no longer confined to the table (Figure 2.2(c)). The coordinates for the three points \(A\), \(B\), and \(C\) are now given by \((x_A, y_A, z_A)\), \((x_B, y_B, z_B)\), and \((x_C, y_C, z_C)\), respectively. Point \(A\) can be placed freely (three degrees of freedom). The location of point \(B\) is subject to the constraint \(d(A, B) = d_{AB}\), meaning it must lie on the sphere of radius \(d_{AB}\) centered at \(A\). Thus we have \(3 - 1 = 2\) freedoms to specify, which can be expressed as the latitude and longitude for the point on the sphere. Finally, the location of point \(C\) must lie at the intersection of spheres centered at \(A\) and \(B\) of radius \(d_{AC}\) and \(d_{BC}\), respectively. In the general case the intersection of two spheres is a circle, and the location of point \(C\) can be described by an angle that parametrizes this circle. Point \(C\) therefore adds \(3 - 2 = 1\) freedom. Once the position of point \(C\) is chosen, the coin is fixed in space.

In summary, a rigid body in three-dimensional space has six freedoms, which can be described by the three coordinates parametrizing point \(A\), the two angles parametrizing point \(B\), and one angle parametrizing point \(C\). Other representations for the configuration of a rigid body are discussed in Chapter 3.

We have just established that a rigid body moving in three-dimensional space, which we call a **spatial rigid body**, has six degrees of freedom. Similarly, a rigid body moving in a two-dimensional plane, which we henceforth call a **planar rigid body**, has three degrees of freedom. This latter result can also be obtained by considering the planar rigid body to be a spatial rigid body with six degrees of freedom, but with the three independent constraints \(z_A = z_B = z_C = 0\).

Since our robots are constructed of rigid bodies, Equation (2.1) can be expressed as follows:

\[
\text{Degrees of freedom} = (\text{Sum of freedoms of the bodies}) - (\text{Number of independent constraints}).
\]  

\[(2.3)\]
Equation (2.3) forms the basis for determining the degrees of freedom of general robots, which is the topic of the next section.

2.2 Degrees of Freedom of a Robot

Consider once again the door example of Figure 2.1(a), consisting of a single rigid body connected to the wall by a hinge joint. From the previous section we know that the door has only one degree of freedom, conveniently represented by the hinge joint angle $\theta$. Without the hinge joint, the door is free to move in three-dimensional space and has six degrees of freedom. By connecting the door to the wall via the hinge joint, five independent constraints are imposed on the motion of the door, leaving only one independent coordinate ($\theta$). Alternatively, the door can be viewed from above and regarded as a planar body, which has three degrees of freedom. The hinge joint then imposes two independent constraints, again leaving only one independent coordinate ($\theta$). Its C-space is represented by some range in the interval $[0, 2\pi]$ over which $\theta$ is allowed to vary.

In both cases the joints have the effect of constraining the motion of the rigid body, and thus reducing the overall degrees of freedom. This observation suggests a formula for determining the number of degrees of freedom of a robot, simply by counting the number of rigid bodies and joints. In this section we derive precisely such a formula, called Grüber's formula, for determining the number of degrees of freedom of planar and spatial robots.

2.2.1 Robot Joints

Figure 2.3 illustrates the basic joints found in typical robots. Every joint connects exactly two links; joint that simultaneously connect three or more links are not allowed. The revolute joint (R), also called a hinge joint, allows for rotational motion about the joint axis. The prismatic joint (P), also called a sliding or linear joint, allows for translational (or rectilinear) motion along the direction of the joint axis. The screw joint (H), also called a helical joint, allows simultaneous rotation and translation about a screw axis. Revolute, prismatic, and screw joints all have one degree of freedom.

Joints can also have multiple degrees of freedom. The cylindrical joint (C) is a two-dof joint that allows for independent translations and rotations about a single fixed joint axis. The universal joint (U) is another two-dof joint that consists of a pair of revolute joints arranged so that their joint axes are orthogonal. The spherical joint (S), also called a ball-and-socket joint, has three degrees of freedom and functions much like our shoulder joint.

A joint can be viewed as providing freedoms to allow one rigid body to move relative to another. It can also be viewed as providing constraints on the possible motions of the two rigid bodies it connects. For example, a revolute joint can be viewed as allowing one freedom of motion between two rigid bodies in space, or it can be viewed as providing five constraints on the motion of one rigid body relative to the other. Generalizing, the number of degrees of freedom
Figure 2.3: Typical robot joints.

<table>
<thead>
<tr>
<th>Joint type</th>
<th>dof f</th>
<th>Constraints c between two planar rigid bodies</th>
<th>Constraints c between two spatial rigid bodies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revolute (R)</td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Prismatic (P)</td>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>Screw (H)</td>
<td>1</td>
<td>N/A</td>
<td>5</td>
</tr>
<tr>
<td>Cylindrical (C)</td>
<td>2</td>
<td>N/A</td>
<td>4</td>
</tr>
<tr>
<td>Universal (U)</td>
<td>2</td>
<td>N/A</td>
<td>4</td>
</tr>
<tr>
<td>Spherical (S)</td>
<td>3</td>
<td>N/A</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2.1: The number of degrees of freedom and constraints provided by common joints.

of a rigid body (three for planar bodies and six for spatial bodies) minus the number of constraints provided by a joint must equal the number of freedoms provided by the joint.

The freedoms and constraints provided by the various joint types are summarized in Table 2.1.

2.2.2 Grüber’s Formula

The number of degrees of freedom of a mechanism with links and joints can be calculated using Grüber’s formula, which is an expression of Equation (2.3).

Proposition 2.1. Consider a mechanism consisting of N links, where ground is also regarded as a link. Let J be the number of joints, m be the number of degrees of freedom of a rigid body (m = 3 for planar mechanisms and m = 6 for
2.2. Degrees of Freedom of a Robot

Figure 2.4: (a) Four-bar linkage. (b) Slider-crank mechanism.

spatial mechanisms), \( f_i \) be the number of freedoms provided by joint \( i \), and \( c_i \) be the number of constraints provided by joint \( i \) (it follows that \( f_i + c_i = m \) for all \( i \)). Then Grübler’s formula for the degrees of freedom (dof) of the robot is

\[
dof = m(N - 1) - \sum_{i=1}^{J} c_i = m(N - 1) - J \sum_{i=1}^{J} (m - f_i) = m(N - 1 - J) + \sum_{i=1}^{J} f_i. \quad (2.4)
\]

This formula only holds if all joint constraints are independent. If they are not independent, then the formula provides a lower bound on the number of degrees of freedom.

Below we apply Grübler’s formula to several planar and spatial mechanisms. We distinguish between two types of mechanisms: open-chain mechanisms (also known as serial mechanisms) and closed-chain mechanisms. A closed-chain mechanism is any mechanism that has a closed loop. A person standing with both feet on the ground is an example of a closed-chain mechanism, since a closed loop is traced from the ground, through the right leg, through the waist, through the left leg, and back to the ground (recall that ground itself is a link). An open-chain mechanism is any mechanism without a closed loop; an example is your arm when your hand is allowed to move freely in space.

Example 2.1. Four-bar linkage and slider-crank mechanism

The planar four-bar linkage shown in Figure 2.4(a) consists of four links (one of them ground) arranged in a single closed loop and connected by four revolute joints. Since all the links are confined to move in the same plane, \( m = 3 \). Substituting \( N = 4, J = 4, \) and \( f_i = 1, i = 1, \ldots, 4 \), into Grübler’s formula, we see that the four-bar linkage has one degree of freedom.

The slider-crank closed-chain mechanism of Figure 2.4(b) can be analyzed in two ways: (i) the mechanism consists of three revolute joints and one prismatic
Example 2.2. Some classical planar mechanisms

Let us now apply Grüber’s formula to several classical planar mechanisms. The $k$-link planar serial chain of revolute joints in Figure 2.5(a) (called a $kR$ robot for its $k$ revolute joints) has $N = k + 1$ ($k$ links plus ground) and $J = k$, and since all the joints are revolute, each $f_i = 1$. Therefore,

$$\text{dof} = 3((k + 1) - 1 - k) + k = k$$

as expected. For the planar five-bar linkage of Figure 2.5(b), $N = 5$ (four links plus ground), $J = 5$, and since all joints are revolute, each $f_i = 1$. Therefore,

$$\text{dof} = 3(5 - 1 - 5) + 5 = 2.$$ 

For the Stephenson six-bar linkage of Figure 2.5(c), we have $N = 6$, $J = 7$, and $f_i = 1$ for all $i$, so that

$$\text{dof} = 3(6 - 1 - 7) + 7 = 1.$$
Finally, for the Watt six-bar linkage of Figure 2.5(d), we have $N = 6$, $J = 7$, and $f_i = 1$ for all $i$, so that like the Stephenson six-bar linkage,

$$dof = 3(6 - 1 - 7) + 7 = 1.$$  

**Example 2.3. A planar mechanism with overlapping joints**

The planar mechanism illustrated in Figure 2.6 has three links that meet at a single point on the right of the large link. Recalling that a joint by definition connects exactly two links, the joint at this point of intersection should not be regarded as a single revolute joint. Rather, it is correctly interpreted as two revolute joints overlapping each other. Again, there is more than one way to derive the number of degrees of freedom of this mechanism using Grüber's formula: (i) The mechanism consists of eight links ($N = 8$), eight revolute joints, and one prismatic joint. Substituting into Grüber’s formula,

$$dof = 3(8 - 1 - 9) + 9(1) = 3.$$  

(ii) Alternatively, the lower-right revolute-prismatic joint pair can be regarded as a single two-dof joint. In this case the number of links is $N = 7$, with seven revolute joints, and a single two-dof revolute-prismatic pair. Substituting into Grüber’s formula,

$$dof = 3(7 - 1 - 8) + 7(1) + 1(2) = 3.$$  

**Example 2.4. Grüber’s formula and singularities**

For the parallelogram linkage of Figure 2.7(a), $N = 5$, $J = 6$, and $f_i = 1$ for each joint. From Grüber’s formula, the degrees of freedom is given by

$$3(5 - 1 - 6) + 6 = 0.$$  

A mechanism with zero degrees of freedom is by definition a rigid structure. However, if the three parallel links are of the same length, and the two horizontal rows of joints are also collinear as implied by the figure, the mechanism can in fact move with one degree of freedom.

A similar situation occurs for the two-dof planar five-bar linkage of Figure 2.7(b). If the two joints connected to ground are locked at some fixed angle,
the five-bar linkage should then become a rigid structure. However, if the two middle links are of equal length and overlap each other as illustrated in the figure, then these overlapping links can rotate freely about the two overlapping joints. Of course, the link lengths of the five-bar linkage must meet certain specifications in order for such a configuration to even be possible. Also note that if a different pair of joints is locked in place, the mechanism does become a rigid structure as expected.

Grübler’s formula provides a lower bound on the degrees of freedom for singular cases like those just described. Configuration space singularities arising in closed chains are discussed in Chapter 7.

**Example 2.5. Delta robot**
The Delta robot of Figure 2.8 consists of two platforms—the lower one mobile,
the upper one stationary—connected by three legs: each leg consists of an RR serial chain connected to a closed-loop parallelogram linkage. Naively applying the spatial version of Grüber’s formula yields $N = 17$, $J = 21$ (all revolute joints), and $\text{dof} = 17(17 - 1 - 21) + 21 = -9$, which would imply that the mechanism is overconstrained and therefore a rigid structure. However, each parallelogram linkage has one degree of freedom of motion, so that each leg of the Delta robot is kinematically equivalent to an RUU chain. In this case $N = 8$, $J = 9$ (six U joints and three R joints), and

$$\text{dof} = 6(8 - 1 - 9) + 3(1) + 6(2) = 3.$$ 

The Delta robot has three degrees of freedom. The moving platform in fact always remains parallel to the fixed platform and in the same orientation, so that the Delta robot in effect acts as a Cartesian positioning device.

Example 2.6. Stewart-Gough Platform

The Stewart-Gough platform of Figure 1.1(b) consists of two platforms—the lower one stationary and regarded as ground, the upper one mobile—connected by six universal-prismatic-spherical (UPS) serial chains. The total number of links is fourteen ($N = 14$). There are six universal joints (each with two degrees of freedom, $f_i = 2$), six prismatic joints (each with a single degree of freedom, $f_i = 1$), and six spherical joints (each with three degrees of freedom, $f_i = 3$). The total number of joints is 18. Substituting these values into Grüber’s formula with $m = 6$,

$$\text{dof} = 6(14 - 1 - 18) + 6(1) + 6(2) + 6(3) = 6.$$ 

In some versions of the Stewart-Gough platform the six universal joints are replaced by spherical joints. By Grüber’s formula this mechanism would have twelve degrees of freedom; replacing each universal joint by a spherical joint introduces an extra degree of freedom in each leg, allowing torsional rotations about the leg axis. Note however that this torsional rotation has no effect on the motion of the mobile platform.

The Stewart-Gough platform is a popular choice for car and airplane cockpit simulators, as the platform moves with the full six degrees of freedom of motion of a rigid body. The parallel structure means that each leg only needs to support a fraction of the weight of the payload. On the other hand, the parallel design also limits the range of translational and rotational motion of the platform.

2.3 Configuration Space: Topology and Representation

2.3.1 Configuration Space Topology

Until now we have been focusing on one important aspect of a robot’s C-space—its dimension, or the number of degrees of freedom. However, the shape of the space is also important.
Figure 2.9: An open interval of the real line, denoted \((a, b)\), can be deformed to an open semicircle. This open semicircle can then be deformed to the real line by the mapping illustrated: beginning from a point at the center of the semicircle, draw a ray that intersects the semicircle and then a line above the semicircle. These rays show that every point of the semicircle can be stretched to exactly one point on the line, and vice-versa. Thus an open interval can be continuously deformed to a line, so an open interval and a line are topologically equivalent.

Consider a point moving on the surface of a sphere. The point’s C-space is two-dimensional, as the configuration can be described by two coordinates, e.g., latitude and longitude. As another example, a point moving on a plane also has a two-dimensional C-space, with coordinates \((x, y)\). While both a plane and the surface of a sphere are two-dimensional, clearly they do not have the same shape—the plane extends infinitely while the sphere wraps around.

On the other hand, a larger sphere has the same shape as the original sphere, in that it wraps around in the same way. Only its size is different. For that matter, an oval-shaped American football also wraps around similarly to a sphere. The only difference between a football and a sphere is that the football has been stretched in one direction.

The idea that the two-dimensional surfaces of a small sphere, a large sphere, and a football all have the same kind of shape, which is different from the shape of a plane, is expressed by the topology of the surfaces. We do not attempt a rigorous treatment in this book\(^3\), but we say that two spaces are topologically equivalent if one can be continuously deformed into the other without cutting or gluing. A sphere can be deformed into a football simply by stretching, without cutting or gluing, so those two spaces are topologically equivalent. You cannot turn a sphere into a plane without cutting it, however, so a sphere and a plane are not topologically equivalent.

Topologically distinct one-dimensional spaces include the circle, the line, and a closed interval of the line. The circle is written mathematically as \(S\) or \(S^1\), i.e., a one-dimensional “sphere.” The line can be written \(E\) or \(E^1\), indicating a one-dimensional Euclidean (or “flat”) space. Since a point in \(E^1\) is so commonly represented by a real number (after choosing an origin and a length scale), it is often written \(\mathbb{R}\) or \(\mathbb{R}^1\) instead. A closed interval of the line, which contains its endpoints, can be written \([a, b] \subset \mathbb{R}^1\). (An open interval \((a, b)\) does not include the endpoints \(a\) and \(b\) and is topologically equivalent to a line, since the open interval can be stretched to a line, as shown in Figure 2.9. A closed interval is not topologically equivalent to a line, since a line does not contain endpoints.)

\(^3\)For those familiar with concepts in topology, all spaces we consider can be viewed as embedded in a higher-dimensional Euclidean space, inheriting the Euclidean topology of that
2.3. Configuration Space: Topology and Representation

In higher dimensions, $\mathbb{R}^n$ is the $n$-dimensional Euclidean space and $S^n$ is the $n$-dimensional surface of a sphere in $(n+1)$-dimensional space. For example, $S^2$ is the two-dimensional surface of a sphere in three-dimensional space.

Note that the topology of a space is a fundamental property of the space itself, and it is independent of how we choose to use coordinates to represent points in the space. For example, to represent a point on a circle, we could refer to the point by the angle $\theta$ from the center of the circle to the point, relative to a chosen zero angle. Or we could choose a reference frame with its origin at the center of the circle and represent the point by the two coordinates $(x, y)$, subject to the constraint $x^2 + y^2 = 1$. No matter our choice of coordinates, the space itself does not change.

Some C-spaces can be expressed as the Cartesian product of two or more spaces of lower dimension; that is, points in such a C-space can be represented as the union of the representation of points in the lower-dimensional spaces. For example:

- The C-space of a rigid body in the plane can be written as $\mathbb{R}^2 \times S^1$, since the configuration can be represented as the concatenation of the coordinates $(x, y)$ representing $\mathbb{R}^2$ and an angle $\theta$ representing $S^1$.
- The C-space of a PR robot arm can be written $\mathbb{R}^1 \times S^1$. (Typically we will not worry about joint limits when expressing the topology of the C-space.)
- The C-space of a 2R robot arm can be written as $S^1 \times S^1 = T^2$, where $T^n$ is the $n$-dimensional surface of a torus in an $(n+1)$-dimensional space. (See Table 2.2.) Note that $S^1 \times S^1 \times \ldots \times S^1$ ($n$ copies of $S^1$) is equal to $T^n$, not $S^n$; for example, a sphere $S^2$ is not topologically equivalent to a torus $T^2$.
- The C-space of a planar rigid body (e.g., the chassis of a mobile robot) with a 2R robot arm can be written as $\mathbb{R}^2 \times S^1 \times T^2 = \mathbb{R}^2 \times T^3$.
- As we saw in Section 2.1 when we counted the degrees of freedom of a rigid body in three dimensions, the configuration of a rigid body can be described by a point in $\mathbb{R}^3$, plus a point on a two-dimensional sphere $S^2$, plus a point on a one-dimensional circle $S^1$, for a total C-space of $\mathbb{R}^3 \times S^2 \times S^1$.

### 2.3.2 Configuration Space Representation

To perform computations, we must have a numerical representation of the space, consisting of a set of real numbers. We are familiar with this idea from linear algebra—a vector is a natural way to represent a point in a Euclidean space. It is important to keep in mind that the representation of a space involves a choice, and therefore it is not as fundamental as the topology of the space itself.
Table 2.2: Four topologically different two-dimensional C-spaces and example coordinate representations. In the latitude-longitude representation of the sphere, the latitudes $-90^\circ$ and $90^\circ$ each correspond to a single point (the South Pole and the North Pole, respectively), and the longitude parameter wraps around at $180^\circ$ and $-180^\circ$: the edges with the arrows are glued together. Similarly, the coordinate representations of the torus and cylinder wrap around at the edges marked with identical arrows. To turn the torus into its coordinate representation (a subset of $\mathbb{R}^2$), the torus can be cut along the small circle shown (representing the range of angles $\theta_2$ of the second joint while $\theta_1 = 0$) and straightened out to make a cylinder, then cut along the length of the cylinder (representing the range of angles of the first joint while $\theta_2 = 0$) and flattened.

which is independent of the representation. For example, the same point in a 3D space can have different coordinate representations depending on the choice
of the reference frame (the origin and the direction of the coordinate axes) and the choice of length scale, but the topology of the underlying space is the same regardless of our choice.

While it is natural to choose a reference frame and length scale and use a vector to represent points in a Euclidean space, representing a point on a curved space, like a sphere, is less obvious. One solution for a sphere is to use latitude and longitude coordinates. A choice of \( n \) coordinates, or parameters, to represent an \( n \)-dimensional space is called an explicit parametrization of the space. The explicit parametrization is valid for a particular range of the parameters (e.g., \([-90^\circ, 90^\circ]\) for latitude and \([-180^\circ, 180^\circ]\) for longitude for a sphere, where, on Earth, negative values correspond to “South” and “West,” respectively).

The latitude-longitude representation of a sphere is dissatisfying if you are walking near the North Pole (latitude equals \( 90^\circ \)) or South Pole (latitude equals \( -90^\circ \)), where taking a very small step can result in a large change in the coordinates. The North and South Poles are singularities of the representation, and the existence of singularities is a result of the fact that a sphere does not have the same topology as a plane, i.e., the space of the two real numbers that we have chosen to represent the sphere (latitude and longitude). The location of these singularities has nothing to do with the sphere itself, which looks the same everywhere, and everything to do with the chosen representation of it. Singularities of the parametrization are particularly problematic when representing velocities as the time rate of change of coordinates, since these representations may tend to infinity near singularities even if the point on the sphere is moving at a constant speed \( \sqrt{\dot{x}^2 + \dot{y}^2 + \dot{z}^2} \) (if you had represented the point as \((x, y, z)\) instead).

If you assume that the configuration never approaches a singularity of the representation, you can ignore this issue. If you cannot make this assumption, there are two ways to overcome the problem:

- Define more than one coordinate chart on the space, where each coordinate chart is an explicit parametrization covering only a portion of the space. Within each chart, there is no singularity. For example, we could define two coordinate charts on the sphere: the usual latitude \( \phi \in [-90^\circ, 90^\circ] \) and longitude \( \psi \in [-180^\circ, 180^\circ] \), and alternative coordinates \((\phi', \psi')\) in a rotated coordinate frame, where the alternative latitude \( \phi' \) is \( 90^\circ \) at the “East Pole” and \(-90^\circ \) at the “West Pole.” Then the first coordinate chart can be used when \(-90^\circ + \epsilon < \phi < 90^\circ - \epsilon \), for some small \( \epsilon > 0 \), and the second coordinate chart can be used when \(-90^\circ + \epsilon < \phi' < 90^\circ - \epsilon \).

If we define a set of singularity-free coordinate charts that overlap each other and cover the entire space, like the two charts above, the charts are said to form an atlas of the space, much like an atlas of the Earth consists of several maps that together cover the Earth. An advantage of using an atlas of coordinate charts is that the representation always uses the minimum number of numbers. A disadvantage is the extra bookkeeping...
required to switch the representation between coordinate charts to avoid singularities. (Note that Euclidean spaces can be covered by a single coordinate chart without singularities.)

- Instead of using an explicit parametrization, use an implicit representation of the space. An implicit representation views the $n$-dimensional space as embedded in a Euclidean space of more than $n$ dimensions, just like a two-dimensional unit sphere can be viewed as a surface embedded in a three-dimensional Euclidean space. An implicit representation uses the coordinates of the higher-dimensional space (e.g., $(x, y, z)$ in the three-dimensional space), but subjects these coordinates to constraints that reduce the number of degrees of freedom (e.g., $x^2 + y^2 + z^2 = 1$ for the unit sphere).

A disadvantage of this approach is that the representation has more numbers than the number of degrees of freedom. An advantage is that there are no singularities in the representation—a point moving smoothly around the sphere is represented by a smoothly changing $(x, y, z)$, even at the North and South Poles. A single representation is used for the whole sphere; multiple coordinate charts are not needed.

Another advantage is that while it may be very difficult to construct an explicit parametrization, or atlas, for a closed-chain mechanism, it is easy to find an implicit representation: the set of all joint coordinates subject to the loop-closure equations that define the closed loops (Section 2.4).

We use implicit representations throughout the book, beginning in the next chapter. In particular, we use nine numbers, subject to six constraints, to represent the three orientation freedoms of a rigid body in space. This is called a rotation matrix. In addition to being singularity-free (unlike three-parameter representations such as roll-pitch-yaw angles), the rotation matrix representation has the benefit of allowing us to use linear algebra to perform computations such as (1) rotating a rigid body or (2) changing the reference frame in which the orientation of a rigid body is expressed.

In summary, the non-Euclidean shape of many C-spaces motivates the use of implicit representations of C-space throughout this book. We return to this topic in the next chapter.
2.4 Configuration and Velocity Constraints

For robots containing one or more closed loops, usually an implicit representation is more easily obtained than an explicit parametrization. For example, consider the planar four-bar linkage of Figure 2.10, which has one degree of freedom. The fact that the four links always form a closed loop can be expressed in the form of the following three equations:

\[
\begin{align*}
L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + \ldots + L_4 \cos(\theta_1 + \ldots + \theta_4) &= 0 \\
L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + \ldots + L_4 \sin(\theta_1 + \ldots + \theta_4) &= 0 \\
\theta_1 + \theta_2 + \theta_3 + \theta_4 - 2\pi &= 0.
\end{align*}
\]

These equations are obtained by viewing the four-bar linkage as a serial chain with four revolute joints, in which (i) the tip of link $L_4$ always coincides with the origin and (ii) the orientation of link $L_4$ is always horizontal.

These equations are sometimes referred to as loop-closure equations. For the four-bar linkage they are given by a set of three equations in four unknowns. The set of all solutions forms a one-dimensional curve in the four-dimensional joint space and constitutes the C-space.

For general robots containing one or more closed loops, the configuration space can be implicitly represented by $\theta = (\theta_1, \ldots, \theta_n) \in \mathbb{R}^n$ and loop-closure equations of the form

\[
g(\theta) = \begin{bmatrix}
g_1(\theta_1, \ldots, \theta_n) \\
\vdots \\
g_k(\theta_1, \ldots, \theta_n)
\end{bmatrix} = 0, \quad (2.5)
\]

\(^4\)Roll-pitch-yaw angles and Euler angles use three parameters for the space of rotations $S^2 \times S^1$ (two for $S^2$ and one for $S^1$), and therefore are subject to singularities as discussed above.

\(^5\)Another singularity-free implicit representation of orientations, the unit quaternion, uses only four numbers subject to the constraint that the four-vector be unit length. In fact, this representation is a double covering of the set of orientations: for every orientation, there are two unit quaternions.
where \( g : \mathbb{R}^n \rightarrow \mathbb{R}^k \) is a set of \( k \) independent equations, with \( k \leq n \). Such constraints are known as \textbf{holonomic constraints}, constraints that reduce the dimension of the C-space.\(^6\) The C-space can be viewed as a surface of dimension \( n - k \) (assuming all constraints are independent) embedded in \( \mathbb{R}^n \).

Suppose a closed-chain robot with loop-closure equations \( g(\theta) = 0, g : \mathbb{R}^n \rightarrow \mathbb{R}^k \), is in motion, following the time trajectory \( \theta(t) \). Differentiating both sides of \( g(\theta(t)) = 0 \) with respect to \( t \), we obtain

\[
\frac{d}{dt} g(\theta(t)) = 0
\]

\[
\begin{bmatrix}
\frac{\partial g_1}{\partial \theta_1}(\theta) \dot{\theta}_1 + \ldots + \frac{\partial g_n}{\partial \theta_n}(\theta) \dot{\theta}_n \\
\vdots \\
\frac{\partial g_1}{\partial \theta_1}(\theta) \dot{\theta}_1 + \ldots + \frac{\partial g_k}{\partial \theta_n}(\theta) \dot{\theta}_n \\
\frac{\partial g_k}{\partial \theta_1}(\theta) \dot{\theta}_1 + \ldots + \frac{\partial g_k}{\partial \theta_n}(\theta) \dot{\theta}_n \\
\end{bmatrix}
\begin{bmatrix}
\dot{\theta}_1 \\
\vdots \\
\dot{\theta}_n \\
\end{bmatrix}
= 0
\]

\[
\frac{\partial g}{\partial \dot{\theta}}(\theta) \dot{\theta} = 0. \tag{2.6}
\]

Here \( \dot{\theta}_i \) denotes the time derivative of \( \theta_i \) with respect to time \( t \), \( \frac{\partial g_i}{\partial \dot{\theta}}(\theta) \in \mathbb{R}^{k \times n} \), and \( \theta, \dot{\theta} \in \mathbb{R}^n \). From the above we see that the joint velocity vector \( \dot{\theta} \in \mathbb{R}^n \) cannot be arbitrary, but must always satisfy

\[
\frac{\partial g}{\partial \dot{\theta}}(\theta) \dot{\theta} = 0. \tag{2.7}
\]

These constraints can be written in the form

\[
A(\theta) \dot{\theta} = 0, \tag{2.8}
\]

where \( A(\theta) \in \mathbb{R}^{k \times n} \). Velocity constraints of this form are called \textbf{Pfaffian constraints}. For the case of \( A(\theta) = \frac{\partial g}{\partial \dot{\theta}}(\theta) \), one could regard \( g(\theta) \) as being the “integral” of \( A(\theta) \); for this reason, holonomic constraints of the form \( g(\theta) = 0 \) are also called \textbf{integrable constraints}—the velocity constraints that they imply can be integrated to give equivalent configuration constraints.

We now consider another class of Pfaffian constraints that is fundamentally different from the holonomic type. To illustrate with a concrete example, consider an upright coin of radius \( r \) rolling on the plane as shown in Figure 2.11. The configuration of the coin is given by the contact point \((x, y)\) on the plane, the steering angle \( \phi \), and the angle of rotation (see Figure 2.11). The C-space of the coin is therefore \( \mathbb{R}^2 \times T^2 \), where \( T^2 \) is the two-dimensional torus parametrized by the angles \( \phi \) and \( \theta \). This C-space is four-dimensional.

\(^6\)Viewing a rigid body as a collection of points, the distance constraints between the points, as we saw earlier, can be viewed as holonomic constraints.
2.4. Configuration and Velocity Constraints

We now express, in mathematical form, the fact that the coin rolls without slipping. The coin must always roll in the direction indicated by \((\cos \phi, \sin \phi)\), with forward speed \(r\dot{\theta}\):

\[
\begin{bmatrix}
\dot{x} \\
\dot{y}
\end{bmatrix} = r\dot{\theta} \begin{bmatrix}
\cos \phi \\
\sin \phi
\end{bmatrix}.
\] (2.9)

Collecting the four C-space coordinates into a single vector \(q = (q_1, q_2, q_3, q_4) = (x, y, \phi, \theta) \in \mathbb{R}^2 \times T^2\), the above no-slip rolling constraint can then be expressed in the form

\[
\begin{bmatrix}
1 & 0 & 0 & -r \cos q_3 \\
0 & 1 & 0 & -r \sin q_3
\end{bmatrix} \dot{q} = 0.
\] (2.10)

These are Pfaffian constraints of the form \(A(q)\dot{q} = 0\), \(A(q) \in \mathbb{R}^{2 \times 4}\).

These constraints are not integrable; that is, for the \(A(q)\) given in (2.10), there does not exist any differentiable \(g : \mathbb{R}^4 \to \mathbb{R}^2\) such that \(\frac{\partial g}{\partial q} = A(q)\). To see why, there would have to exist a differentiable \(g_1(q)\) that satisfied the following four equalities:

\[
\begin{align*}
\frac{\partial g_1}{\partial q_1} &= 1 & \rightarrow & g_1(q) = q_1 + h_1(q_2, q_3, q_4) \\
\frac{\partial g_1}{\partial q_2} &= 0 & \rightarrow & g_1(q) = h_2(q_1, q_3, q_4) \\
\frac{\partial g_1}{\partial q_3} &= 0 & \rightarrow & g_1(q) = h_3(q_1, q_2, q_4) \\
\frac{\partial g_1}{\partial q_4} &= -r \cos q_3 & \rightarrow & g_1(q) = -rq_4 \cos q_3 + h_4(q_1, q_2, q_3),
\end{align*}
\]

for some \(h_i, i = 1, \ldots, 4\), differentiable in each of its variables. By inspection it should be clear that no such \(g_1(q)\) exists. Similarly, it can be shown that \(g_2(q)\) does not exist, so that the constraint (2.10) is nonintegrable. A Pfaffian constraint that is nonintegrable is called a nonholonomic constraint. Such constraints reduce the dimension of the feasible velocities of the system, but do not reduce the dimension of the reachable C-space. The rolling coin can reach any point in its four-dimensional C-space despite the two constraints on its velocity.\(^7\) See Exercise 28.

\(^7\)Some texts define the number of degrees of freedom of a system to be the dimension of
Nonholonomic constraints arise in a number of robotics contexts that involve conservation of momentum and rolling without slipping, e.g., wheeled vehicle kinematics and grasp contact kinematics. We examine nonholonomic constraints in greater detail in the later chapter on wheeled robots.

2.5 Task Space and Workspace

We introduce two more concepts relating to the configuration of a robot: the task space and the workspace. Both relate to the configuration of the end-effector of a robot, not the configuration of the entire robot.

The task space is a space in which the robot’s task can be naturally expressed. For example, if the robot’s task is to plot with a pen on a piece of paper, the task space would be \( \mathbb{R}^2 \). If the task is to manipulate a rigid body, a natural representation of the task space is the C-space of a rigid body, representing the position and orientation of a frame attached to the robot’s end-effector. This is the feasible velocities, e.g., two for the rolling coin. We uniformly refer to the dimension of the C-space as the number of degrees of freedom.
the default representation of task space. The decision of how to define the task space is driven by the task, independent of the robot.

The **workspace** is a specification of the configurations the end-effector of the robot can reach. The definition of the workspace is primarily driven by the robot’s structure, independent of the task.

Both the task space and the workspace involve a choice by the user; in particular, the user may decide that some freedoms of the end-effector (e.g., its orientation) do not need to be represented.

The task space and the workspace are distinct from the robot’s C-space. A point in the task space or the workspace may be achievable by more than one robot configuration, meaning that the point is not a full specification of the robot’s configuration. For example, for an open-chain robot with seven joints, the six-dof position and orientation of its end-effector does not fully specify the robot’s configuration.

Some points in task space may not be reachable at all by the robot, e.g., a point on a chalkboard that the robot cannot reach. By definition, all points in the workspace are reachable by at least one configuration of the robot.

Two mechanisms with different C-spaces may have the same workspace. For example, considering the end-effector to be the Cartesian tip of the robot (e.g., the location of a plotting pen) and ignoring orientations, the planar 2R open chain with links of equal length three (Figure 2.12(a)) and the planar 3R open chain with links of equal length two (Figure 2.12(b)) have the same workspace despite having different C-spaces.

Two mechanisms with the same C-space may also have different workspaces. For example, taking the end-effector to be the Cartesian tip of the robot and ignoring orientations, the 2R open chain of Figure 2.12(a) has a planar disk as its workspace, while the 2R open chain of Figure 2.12(c) has the surface of a sphere as its workspace.

Attaching a coordinate frame to the tip of the tool of the 3R open chain “wrist” mechanism of Figure 2.12(d), we see that the frame can achieve any orientation by rotating the joints, but the Cartesian position of the tip is always fixed. This can be seen by noting that the three joint axes always intersect at the tip. For this mechanism, we would likely define the workspace to be the three-dof space of orientations of the frame, $S^2 \times S^1$, which is different from the C-space $T^3$. The task space depends on the task; if the job is to point a laser pointer, then rotations about the axis of the laser beam are immaterial, and the task space would be $S^2$, the set of directions the laser can point.

**Example 2.7.** The SCARA robot of Figure 2.13 is an RRRP open chain that is widely used for tabletop pick-and-place tasks. The end-effector configuration is completely described by the four parameters $(x, y, z, \phi)$, where $(x, y, z)$ denotes the Cartesian position of the end-effector center point, and $\phi$ denotes the orientation of the end-effector in the $x$-$y$ plane. Its task space would typically be defined as $\mathbb{R}^3 \times S^1$, and its workspace would typically be defined as the reachable points in $(x, y, z)$ Cartesian space, since all orientations $\phi \in S^1$ can be achieved at all reachable points.
Example 2.8. A standard 6R industrial manipulator can be adapted to spray-painting applications as shown in Figure 2.14. The paint spray nozzle attached to the tip can be regarded as the end-effector. What is important to the task is the Cartesian position of the spray nozzle, together with the direction in which the spray nozzle is pointing; rotations about the nozzle axis (which points in the direction in which paint is being sprayed) do not matter. The nozzle configuration can therefore be described by five coordinates: \((x, y, z)\) for the
2.6 Summary

- A robot is mechanically constructed from links that are connected by various types of joints. The links are usually modeled as rigid bodies. An end-effector such as a gripper may be attached to some link of the robot. Actuators deliver forces and torques to the joints, thereby causing motion of the robot.

- The most widely used one-dof joints are the revolute joint, which allows for rotation about the joint axis, and the prismatic joint, which allows for translation in the direction of the joint axis. Some common two-dof joints include the cylindrical joint, which is constructed by serially connecting a revolute and prismatic joint, and the universal joint, which is constructed by orthogonally connecting two revolute joints. The spherical joint, also known as ball-in-socket joint, is a three-dof joint whose function is similar to the human shoulder joint.

- The configuration of a rigid body is a specification of the location of all of its points. For a rigid body moving in the plane, three independent parameters are needed to specify the configuration. For a rigid body moving in three-dimensional space, six independent parameters are needed to specify the configuration.

- The configuration of a robot is a specification of the configuration of all of its links. The robot’s configuration space is the set of all possible robot configurations. The dimension of the C-space is the number of degrees of freedom of a robot.

- The number of degrees of freedom of a robot can be calculated using Grübler’s formula,

\[
dof = m(N - 1 - J) + \sum_{i=1}^{J} f_i,
\]

where \( m = 3 \) for planar mechanisms and \( m = 6 \) for spatial mechanisms, \( N \) is the number of links (including the ground link), \( J \) is the number of joints, and \( f_i \) is the number of degrees of freedom of joint \( i \).

- A robot’s C-space can be parametrized explicitly or represented implicitly. For a robot with \( n \) degrees of freedom, an explicit parametrization uses \( n \) coordinates, the minimum necessary. An implicit representation...
Configuration Space

involves \( m \) coordinates with \( m \geq n \), with the \( m \) coordinates subject to \( m - n \) constraint equations. With the implicit parametrization, a robot’s C-space can be viewed as a surface of dimension \( n \) embedded in a space of higher dimension \( m \).

- The C-space of an \( n \)-dof robot whose structure contains one or more closed loops can be implicitly represented using \( k \) loop-closure equations of the form \( g(\theta) = 0 \), where \( \theta \in \mathbb{R}^m \) and \( g : \mathbb{R}^m \to \mathbb{R}^k \). Such constraint equations are called holonomic constraints. Assuming \( \theta(t) \) varies with time \( t \), the holonomic constraints \( g(\theta(t)) = 0 \) can be differentiated with respect to \( t \) to yield
  \[
  \frac{\partial g}{\partial \theta}(\theta)\dot{\theta} = 0,
  \]
  where \( \frac{\partial g}{\partial \theta}(\theta) \) is a \( k \times m \) matrix.

- A robot’s motion can also be subject to velocity constraints of the form
  \[
  A(\theta)\dot{\theta} = 0,
  \]
  where \( A(\theta) \) is a \( k \times m \) matrix that cannot be expressed as the differential of some function \( g(\theta) \), i.e., there does not exist any \( g(\theta), g : \mathbb{R}^m \to \mathbb{R}^k \), such that
  \[
  A(\theta) = \frac{\partial g}{\partial \theta}(\theta).
  \]
  Such constraints are said to be nonholonomic constraints, or nonintegrable constraints. These constraints reduce the dimension of feasible velocities of the system but do not reduce the dimension of the reachable C-space. Nonholonomic constraints arise in robot systems subject to conservation of momentum or rolling without slipping.

- A robot’s task space is a space in which the robot’s task can be naturally expressed. A robot’s workspace is a specification of the configurations the end-effector of the robot can reach.

Notes and References

In the kinematics literature, structures that consist of links connected by joints are also called mechanisms or linkages. The degrees of freedom of mechanisms is treated in most texts on mechanism analysis and design, e.g., [29]. A robot’s configuration space has the mathematical structure of a differentiable manifold. Some accessible introductions to differential manifolds and differential geometry are [88], [18]. Configuration spaces are further examined in a motion planning context in [59], [19].
2.7 Exercises

1. Using the methods of Section 2.1, derive a formula, in terms of \( n \), for the degrees of freedom of a rigid body in \( n \)-dimensional space. Indicate how many of those dof are translational and how many are rotational. Describe the topology of the C-space (e.g., for \( n = 2 \), the topology is \( \mathbb{R}^2 \times S^1 \)).

2. Find the degrees of freedom of your arm, from your torso to your palm (just past the wrist, not including finger degrees of freedom). Do this in two ways: (a) add up the degrees of freedom at the shoulder, elbow, and wrist joints, and (b) fix your palm flat on a table with your elbow bent, and without moving your torso, investigate how many degrees of freedom with which you can still move your arm. Do your answers agree? How many constraints were placed on your arm when you placed your palm at a fixed configuration on the table?

3. Assume each of your arms has \( n \) degrees of freedom. You are driving a car, your torso is stationary relative to the car (a tight seatbelt!), and both hands are firmly grasping the wheel, so that your hands do not move relative to the wheel. How many degrees of freedom does your arms-plus-steering wheel system have? Explain your answer.

4. Figure 2.15 shows a robot used for human arm rehabilitation. Determine the degrees of freedom of the chain formed by the human arm and robot.

5. The mobile manipulator of Figure 2.16 consists of a 6R arm and multifingered hand mounted on a mobile base with a single wheel. The wheel rotates
(a) Ignoring the multifingered hand, describe the configuration space of the mobile manipulator.

(b) Now suppose the robot hand rigidly grasps the refrigerator door handle, and with its wheel completely stationary, opens the door using only its arm. With the door open, how many degrees of freedom does the mechanism formed by the arm and open door have?

(c) A second identical mobile manipulator comes along, and after parking its mobile base, also rigidly grasps the refrigerator door handle. How many degrees of freedom does the mechanism formed by the two arms and the open refrigerator door have?

6. Three identical SRS open chain arms are grasping a common object as shown in Figure 2.17.

(a) Find the degrees of freedom of this system.

(b) Suppose there are now a total of $n$ such arms grasping the object. What is the degrees of freedom of this system?
(c) Suppose the spherical wrist joint in each of the \( n \) arms is now replaced by a universal joint. What is the degrees of freedom of the overall system?

7. Consider a spatial parallel mechanism consisting of a moving plate connected to the fixed plate by \( n \) identical open chain legs. In order for the moving plate to have six degrees of freedom, how many degrees of freedom should each leg have, as a function of \( n \)? For example, if \( n = 3 \), then the moving plate and fixed plate are connected by three open chains; how many degrees of freedom should each open chain have in order for the moving plate to move with six degrees of freedom? Solve for arbitrary \( n \).

8. Use the planar version of Gr"ubler’s formula to determine the degrees of freedom of the mechanisms shown in Figure 2.18. Comment on whether your results agree with your intuition about the possible motions of these mechanisms.

9. Use the spatial version of Gr"ubler’s formula to determine the degrees of freedom of the mechanisms shown in Figure 2.19. Comment on whether your results agree with your intuition about the possible motions of these mechanisms.

10. In the parallel mechanism shown in Figure 2.20, six legs of identical length are connected to a fixed and moving platform via spherical joints. Determine the degrees of freedom of this mechanism using Gr"ubler’s formula. Illustrate all possible motions of the upper platform.

11. The \( 3 \times \text{UPU} \) platform of Figure 2.21 consists of two platforms—the lower one stationary, the upper one mobile—connected by three UPU serial chains.
   (a) Using the spatial version of Gr"ubler’s formula, Verify that it has three degrees of freedom.
   (b) Construct a physical model of the \( 3 \times \text{UPU} \) platform to see if it indeed has three degrees of freedom. In particular, lock the three P joints in place; does the robot become a structure as predicted by Gr"ubler’s formula, or does it move? Try reversing the order of the U joints, i.e., with the rotational axes connecting the leg to the platforms arranged parallel to the platforms, and also arranged orthogonal to the platforms. Does the order in which the U joints are connected matter?

12. (a) The mechanism of Figure 2.22(a) consists of six identical squares arranged in a single closed loop, connected serially by revolute joints. The bottom square is fixed to ground. Determine its degrees of freedom using an appropriate version of Gr"ubler’s formula.
   (b) The mechanism of Figure 2.22(b) also consists of six identical squares connected by revolute joints, but arranged differently as shown. Determine its degrees of freedom using an appropriate version of Gr"ubler’s formula. Does your result agree with your intuition about the possible motions of this mechanism?
Configuration Space

(a) Fork Joint
(b) Slider
(c) Fork Joint
(d) Slider
(e) Beam
(f) Arm
(g) Beam
(h) Arm
13. Figure 2.23 shows a spherical four-bar linkage, in which four links (one of the links is the ground link) are connected by four revolute joints to form a single-loop closed chain. The four revolute joints are arranged so that they lie on a sphere, and such that their joint axes intersect at a common point.
(a) Use an appropriate version of Grübler’s formula to find the degrees of freedom. Justify your choice of formula.
(b) Describe the configuration space.
(c) Assuming a reference frame is attached to the center link, describe its workspace.

14. Figure 2.24 shows a parallel robot for surgical applications. As shown in Figure 2.24(a), leg A is an RRRP chain, while legs B and C are RRRUR chains. A surgical tool is attached to the end-effector as shown.
(a) Use an appropriate version of Grübler’s formula to find the degrees of freedom of the mechanism in Figure 2.24(a).
2.7. Exercises

Figure 2.19: A collection of spatial parallel mechanisms.

Figure 2.20: A $6 \times SS$ platform.

(b) Now suppose the surgical tool must always pass through point $A$ in Figure 2.24(a). How many degrees of freedom does the manipulator have?

(c) Legs A, B, and C are now replaced by three identical RRRR legs as shown in Figure 2.24(b). Furthermore the axes of all R joints pass through point $A$. Use an appropriate version of Grüber’s formula to derive the degrees of freedom of this mechanism.

15. Figure 2.25 shows a $3 \times PUP$ platform, in which three identical PUP legs connect a fixed base to a moving platform. As shown in the figure, the P joints on both the fixed base and moving platform are arranged symmetrically. Recalling that the U joint consists of two revolute joints connected orthogonally,
Figure 2.21: The $3 \times UPU$ platform.

Figure 2.22: Two mechanisms.

each R joint connected to the moving platform has its joint axis aligned in the
same direction as the platform’s P joint. The R joint connected to the fixed base has its joint axis orthogonal to the base P joint. Use an appropriate version of Grüber’s formula to find the degrees of freedom. Does your answer agree with your intuition about this mechanism? If not, try to explain any discrepancies without resorting to a detailed kinematic analysis.

The dual-arm robot of Figure 2.26 is rigidly grasping a box as shown. The box can only slide on the table; the bottom face of the box must always be in contact with the table. How many degrees of freedom does this system have?
17. The dragonfly robot of Figure 2.27 has a body, four legs, and four wings as shown. Each leg is connected to its adjacent leg by a USP open chain. Use appropriate versions of Grübler’s formula to answer the following questions:
(a) Suppose the body is fixed, and only the legs and wings can move. How many degrees of freedom does the robot have?
(b) Now suppose the robot is flying in the air. How many degrees of freedom does the robot have?
(c) Now suppose the robot is standing with all four feet in contact with the ground. Assume the ground is uneven, and that each foot-ground contact can be modeled as a point contact with no slip. How many degrees of freedom does the robot have?
18. (a) A caterpillar robot is hanging by its tail as shown in Figure 2.28(a). The caterpillar robot consists of eight rigid links (one head, one tail, and six body links) connected serially by revolute-prismatic pairs as shown. Find the degrees of freedom of this robot.

(b) The caterpillar robot is now crawling on a leaf as shown in Figure 2.28(b). Suppose all six body links must be in contact with the leaf at all times (each link-leaf contact can be modelled as a frictionless point contact). Find the de-
Figure 2.29: (a) A four-fingered hand with palm. (b) The hand grasping an ellipsoidal object. (c) A rounded fingertip that can roll on the object without sliding.

19. The four-fingered hand of Figure 2.29(a) consists of a palm and four URR fingers (the U joints connect the fingers to the palm).
   (a) Assume the fingertips are points, and that one of the fingertips is in contact with the table surface (sliding of the fingertip point contact is allowed). How many degrees of freedom does the hand have? What if two fingertips are in sliding point contact with the table? Three? All four?
   (b) Repeat part (a) but with each URR finger replaced by an SRR finger (each universal joint is replaced by a spherical joint).
   (c) The hand (with URR fingers) now grasps an ellipsoidal object as shown in Figure 2.29(b). Assume the palm is fixed in space, and that no slip occurs between the fingertips and object. How many degrees of freedom does the system have?
   (d) Now assume the fingertips are spheres as shown in Figure 2.29(c). Each of the fingertips can roll on the object, but cannot slip or break contact with the object. How many degrees of freedom does the system have? For a single fingertip in rolling contact with the object, comment on the dimension of the space of feasible fingertip velocities relative to the object versus the number of parameters needed to represent the fingertip configuration relative to the object (its degrees of freedom). (Hint: You may want to experiment by rolling a ball around on a tabletop to get some intuition.)

20. Consider the slider-crank mechanism of Figure 2.4(b). A rotational motion at the revolute joint fixed to ground (the “crank”) causes a translational motion at the prismatic joint (the “slider”). Suppose the two links connected to the crank and slider are of equal length. Determine the configuration space of this mechanism, and draw its projected version on the space defined by the crank.
2.7. Exercises

21. (a) Use an appropriate version of Grüber’s formula to determine the degrees of freedom of a planar four-bar linkage floating in space.
(b) Derive an implicit parametrization of the four-bar’s configuration space as follows. First, label the four links 1, 2, 3, 4, and choose three points $A, B, C$ on link 1, $D, E, F$ on link 2, $G, H, I$ on link 3, and $J, K, L$ on link 4. The four-bar linkage is constructed such that the four following pairs of points are each connected by a revolute joint: $C$ with $D$, $F$ with $G$, $I$ with $J$, and $L$ with $A$. Write down explicit constraints on the coordinates for the eight points $A, \ldots, H$ (assume a fixed reference frame has been chosen, and denote the coordinates for point $A$ by $p_A = (x_A, y_A, z_A)$, and similarly for the other points). Based on counting the number of variables and constraints, what is the degrees of freedom of the configuration space? If it differs from the result you obtained in (a), try to explain why.

22. In this exercise we examine in more detail the representation of the C-space for the planar four-bar linkage of Figure 2.30. Attach a fixed reference frame and label the joints and link lengths as shown in the figure. The $(x, y)$ coordinates for joints $A$ and $B$ are given by

$$A(\theta) = (a \cos \theta, a \sin \theta)^T$$
$$B(\psi) = (g + b \cos \psi, b \sin \psi)^T.$$ 

Using the fact that the link connecting $A$ and $B$ is of fixed length $h$, i.e., $||A(\theta) - B(\psi)||^2 = h^2$, we have the constraint

$$b^2 + g^2 + 2gb \cos \psi + a^2 - 2(a \cos \theta(g + b \cos \psi) + ab \sin \theta \sin \psi) = h^2.$$ 

Grouping the coefficients of $\cos \psi$ and $\sin \psi$, the above equation can be expressed in the form

$$\alpha(\theta) \cos \psi + \beta(\theta) \sin \psi = \gamma(\theta), \quad (2.11)$$

Figure 2.30: Planar four-bar linkage.

and slider joint variables.
where
\[
\begin{align*}
\alpha(\theta) &= 2gb - 2ab \cos \theta \\
\beta(\theta) &= -2ab \sin \theta \\
\gamma(\theta) &= h^2 - g^2 - b^2 - a^2 + 2ag \cos \theta.
\end{align*}
\]

We now express \( \psi \) as a function of \( \theta \), by first dividing both sides of Equation (2.11) by \( \sqrt{\alpha^2 + \beta^2} \) to obtain
\[
\frac{\alpha}{\sqrt{\alpha^2 + \beta^2}} \cos \psi + \frac{\beta}{\sqrt{\alpha^2 + \beta^2}} \sin \psi = \frac{\gamma}{\sqrt{\alpha^2 + \beta^2}}
\]
Referring to Figure 2.30(b), the angle \( \phi \) is given by \( \phi = \tan^{-1}(\beta/\alpha) \), so that Equation (22) becomes
\[
\cos(\psi - \phi) = \frac{\gamma}{\sqrt{\alpha^2 + \beta^2}}.
\]

Therefore,
\[
\psi = \tan^{-1}\left(\frac{\beta}{\alpha}\right) \pm \cos^{-1}\left(\frac{\gamma}{\sqrt{\alpha^2 + \beta^2}}\right).
\]

(a) Note that a solution exists only if \( \gamma^2 \leq \alpha^2 + \beta^2 \). What are the physical implications if this constraint is not satisfied?

(b) Note that for each value of input angle \( \theta \), there exists two possible values of the output angle \( \psi \). What do these two solutions look like?

(c) Draw the configuration space of the mechanism in \( \theta-\psi \) space for the following link length values: \( a = b = g = h = 1 \).

(d) Repeat (c) for the following link length values: \( a = 1, b = 2, h = \sqrt{5}, g = 2 \).

(e) Repeat (c) for the following link length values: \( a = 1, \ b = 1, \ h = 1, \ g = \sqrt{3} \).

23. A circular disc robot moves in the plane as shown in Figure 2.31. An L-shaped obstacle is nearby. Using the \((x, y)\) coordinates of the robot’s center as its configuration space coordinates, draw the free configuration space (i.e., the set of all feasible configurations) of the robot.
24. The tip coordinates for the two-link planar 2R robot of Figure 2.32 are given by

\[ x = 2 \cos \theta_1 + \cos(\theta_1 + \theta_2) \]
\[ y = 2 \sin \theta_1 + \sin(\theta_1 + \theta_2). \]

(a) What is the robot’s configuration space?
(b) What is the robot’s task space (i.e., the set of all points reachable by the tip)?
(c) Suppose infinitely long vertical barriers are placed at \( x = 1 \) and \( x = -1 \). What is the free C-space of the robot (i.e., the portion of the C-space that does not result in any collisions with the vertical barriers)?

25. (a) Consider a planar 3R open chain with link lengths (starting from the fixed base joint) 5, 2, and 1, respectively. Considering only the Cartesian point of the tip, draw its workspace.
(b) Now consider a planar 3R open chain with link lengths (starting from the fixed base joint) 1, 2, and 5, respectively. Considering only the Cartesian point of the tip, draw its workspace. Which of these two chains has a larger workspace?
(c) A not-so-clever designer claims that he can make the workspace of any planar open chain larger simply by increasing the length of the last link. Explain the fallacy behind this claim.

26. (a) Describe the task space for a robot arm writing on a blackboard.
(b) Describe the task space for a robot arm twirling a baton.

27. Give a mathematical description of the topologies of the C-spaces of the following systems. Use cross-products, as appropriate, of spaces such as a closed interval of a line \([a, b]\) and \(\mathbb{R}^k\), \(S^m\), and \(T^n\), where \(k\), \(m\), and \(n\) are chosen appropriately.

(i) The chassis of a car-like mobile robot rolling on an infinite plane.
Configuration Space

(ii) The car-like mobile robot, but including a representation of the wheel configurations.

(iii) The car-like mobile robot driving around on a spherical asteroid.

(iv) The car-like mobile robot on an infinite plane with an RRPR robot arm mounted on it. The prismatic joint has joint limits, but the revolute joints do not.

(v) A free-flying spacecraft with a 6R arm mounted on it, no joint limits.

28. Describe an algorithm that drives the rolling coin of Figure 2.11 from any arbitrary initial configuration in its four-dimensional C-space to any arbitrary goal configuration, despite the two nonholonomic constraints. The control inputs are the rolling speed $\dot{\theta}$ and the turning speed $\dot{\phi}$.

29. A differential-drive mobile robot has two wheels which do not steer but whose speeds can be controlled independently. The robot goes forward and backward by spinning the wheels in the same direction at the same speed, and it turns by spinning the wheels at different speeds. The configuration of the robot is given by five variables: the $(x, y)$ location of the point halfway between the wheels, the heading direction $\theta$ of the robot’s chassis relative to the $x$-axis of the world frame, and the rotation angles $\phi_1$ and $\phi_2$ of the two wheels about the axis through the centers of the wheels (Figure 2.33). Assume that the radius of each wheel is $r$ and the distance between the wheels is $2d$.

(i) Let $q = (x, y, \theta, \phi_1, \phi_2)^T$ be the configuration of the robot. If the two control inputs are the angular velocities of the wheels $\omega_1 = \dot{\phi}_1$ and $\omega_2 = \dot{\phi}_2$, write the vector differential equation $\dot{q} = g_1(q)\omega_1 + g_2(q)\omega_2$. The vector fields $g_1(q)$ and $g_2(q)$ are called control vector fields, expressing how the system moves when the respective control is applied.

(ii) Write the corresponding Pfaffian constraints $A(q)\dot{q} = 0$ for this system. How many Pfaffian constraints are there?
(iii) Are the constraints holonomic or nonholonomic? Or how many are holonomic and how many nonholonomic?

30. Determine if the following differential constraints are holonomic or nonholonomic:
   (a) 
   \[ (1 + \cos q_1)\dot{q}_1 + (1 + \cos q_2)\dot{q}_2 + (\cos q_1 + \cos q_2 + 4)\dot{q}_3 = 0. \]
   (b) 
   \[ -\dot{q}_1 \cos q_2 + \dot{q}_3 \sin(q_1 + q_2) - \dot{q}_4 \cos(q_1 + q_2) = 0 \]
   \[ \dot{q}_3 \sin q_1 - \dot{q}_4 \cos q_1 = 0. \]
Chapter 3

Rigid-Body Motions

In the previous chapter, we saw that a minimum of six numbers are needed to specify the position and orientation of a rigid body in three-dimensional physical space. In this chapter we develop a systematic way to describe a rigid body’s position and orientation that relies on attaching a reference frame to the body. The configuration of this frame with respect to a fixed reference frame is then represented as a $4 \times 4$ matrix. This is an example of an implicit representation of the C-space, as discussed in the previous chapter: the actual six-dimensional space of rigid-body configurations is obtained by applying ten constraints to the sixteen-dimensional space of $4 \times 4$ real matrices.

Such a matrix not only represents the configuration of a frame, but it can also be used to (1) translate and rotate a vector or a frame, and (2) change the representation of a vector or a frame from coordinates in one frame (e.g., $\{a\}$) to coordinates in another frame (e.g., $\{b\}$). These operations can be performed by simple linear algebra, which is a major reason we choose to represent a configuration as a $4 \times 4$ matrix.

The non-Euclidean (non-“flat”) nature of the C-space of positions and orientations leads us to use the matrix representation. A rigid body’s velocity, however, can be represented simply as a point in $\mathbb{R}^6$: three angular velocities and three linear velocities, which together we call a spatial velocity or twist. More generally, even though a robot’s C-space may not be Euclidean, the set of feasible velocities at any point in the C-space always forms a Euclidean space. As an example, consider a robot whose C-space is the sphere $S^2$: although the C-space is not flat, the velocity at any configuration can be represented by two real numbers (an element of $\mathbb{R}^2$), such as the rate of change of the latitude and the longitude. At any point on the sphere, the space of velocities can be thought of as the plane (a Euclidean space) tangent to that point on the sphere.

Any rigid-body configuration can be achieved by starting from the fixed (home) reference frame and integrating a constant twist for a specified time. Such a motion resembles the motion of a screw, rotating about and translating along the same fixed axis. The observation that all configurations can be achieved by a screw motion motivates a six-parameter representation of the
configuration called the exponential coordinates. The six parameters can be divided into parameters to describe the direction of the screw axis and a scalar to indicate how far the screw motion must be followed to achieve the desired configuration.

This chapter concludes with a discussion of forces. Just as angular and linear velocities are packaged together into a single vector in $\mathbb{R}^6$, moments (torques) and forces are packaged together into a six-vector called a spatial force or wrench.

To illustrate the concepts and to provide a synopsis of the chapter, we begin with a motivating planar example. Before doing so, we make some remarks about vector notation.

**A Word about Vectors and Reference Frames**

A free vector is a geometric quantity with a length and a direction. Think of it as an arrow in $\mathbb{R}^n$. It is called “free” because it is not necessarily rooted anywhere; only its length and direction matter. A linear velocity can be viewed as a free vector: the length of the arrow is the speed and the direction of the arrow is the direction of the velocity. A free vector is denoted by a regular text symbol, e.g., $v$.

If a reference frame and length scale have been chosen for the underlying space in which $v$ lies, then this free vector can be moved so the base of the arrow is at the origin, without changing the orientation. The free vector $v$ can then be represented as a column vector in the coordinates of the reference frame. This vector is written in italics, $v \in \mathbb{R}^n$, where $v$ is at the “head” of the arrow in the frame’s coordinates. If a different reference frame and length scale are chosen, then the representation $v$ will change, but the underlying free vector $v$ is unchanged.

In other words, we say that $v$ is coordinate free; it refers to a physical quantity in the underlying space, and it does not care how we represent it. On the other hand, $v$ is a representation of $v$ that depends on a choice of a coordinate frame.

A point $p$ in physical space can also be represented as a vector. Given a choice of reference frame and length scale for physical space, the point $p$ can be represented as a vector from the reference frame origin to $p$; its vector representation is denoted in italics by $p \in \mathbb{R}^n$. Here, as before, a different choice of reference frame and length scale for physical space leads to a different representation $p \in \mathbb{R}^n$ for the same point $p$ in physical space. See Figure 3.1.

In the rest of this book, a choice of length scale will always be assumed, but we will be dealing with reference frames at different positions and orientations. A reference frame can be placed anywhere in space, and any reference frame leads to an equally valid representation of the underlying space and objects in it. However, we always assume that exactly one stationary fixed frame, or space frame, denoted $\{s\}$, has been defined. This might be attached to a corner of a room, for example. Similarly, we often assume that at least one frame has been attached to some moving rigid body, such as the body of a quadrotor flying in the room. The body frame, denoted $\{b\}$, is the stationary frame that is coincident with the body-attached frame at any instant.
3.1 Rigid-Body Motions in the Plane

Figure 3.1: The point $p$ exists in physical space, and it does not care how we represent it. If we fix a reference frame $\{a\}$, with unit coordinate axes $\hat{x}_a$ and $\hat{y}_a$, we can represent $p$ as $p_a = (1, 2)$. If we fix a reference frame $\{b\}$ at a different location, a different orientation, and a different length scale, we can represent $p$ as $p_b = (4, -2)$.

Important! All frames in this book are stationary, inertial frames. When we refer to a body frame $\{b\}$, we mean a motionless frame that is instantaneously coincident with a frame that is fixed to a (possibly moving) body. This is important to keep in mind, since you may have had a dynamics course that used non-inertial moving frames. Do not confuse these with the stationary, inertial body frames of this book.

For simplicity, we refer to a body frame as a frame attached to a moving rigid body. Despite this, at any instant, by “body frame” we mean the stationary frame that is coincident with the frame moving along with the body.

It is worth repeating to yourself one more time: all frames are stationary.

While it is common to attach the origin of the $\{b\}$ frame to some important point on the body, such as its center of mass, this is not required. The origin of the $\{b\}$ frame might not even be on the physical body itself, as long as its location relative to the body, viewed from an observer on the body that is stationary relative to the body, is constant.

3.1 Rigid-Body Motions in the Plane

Consider the planar body of Figure 3.2, whose motion is confined to the plane. Suppose a length scale and a fixed reference frame have been chosen as shown. We call the fixed reference frame the fixed frame, or the space frame, denoted $\{s\}$, and label its unit axes $\hat{x}_s$ and $\hat{y}_s$. (Throughout this book, the $\hat{}$ notation indicates a unit vector.) Similarly, we attach a reference frame with unit axes $\hat{x}_b$ and $\hat{y}_b$ to the planar body. Because this frame moves with the body, it is called the body frame, and is denoted $\{b\}$. 
To describe the configuration of the planar body, only the position and orientation of the body frame with respect to the fixed frame needs to be specified. The body frame origin $p$ can be expressed in terms of the coordinate axes of $\{s\}$ as

$$ p = p_x \hat{x}_s + p_y \hat{y}_s. \tag{3.1} $$

You are probably more accustomed to writing this vector as simply $p = (p_x, p_y)$; this is fine when there is no possibility of ambiguity about reference frames, but writing $p$ as in Equation (3.1) clearly indicates the reference frame with respect to which $(p_x, p_y)$ is defined.

The simplest way to describe the orientation of the body frame $\{b\}$ relative to the fixed frame $\{s\}$ is by specifying the angle $\theta$ as shown in Figure 3.2. Another (admittedly less simple) way is to specify the directions of the unit axes $\hat{x}_b$ and $\hat{y}_b$ of $\{b\}$ relative to $\{s\}$, in the form

$$ \hat{x}_b = \cos \theta \hat{x}_s + \sin \theta \hat{y}_s \tag{3.2} $$

$$ \hat{y}_b = -\sin \theta \hat{x}_s + \cos \theta \hat{y}_s. \tag{3.3} $$

At first sight this seems a rather inefficient way to represent the body frame orientation. However, imagine the body were to move arbitrarily in three-dimensional space; a single angle $\theta$ alone clearly would not suffice to describe the orientation of the displaced reference frame. We would in fact need three angles, but it is not yet clear how to define an appropriate set of three angles. On the other hand, expressing the directions of the coordinate axes of $\{b\}$ in terms of coefficients of the coordinate axes of $\{s\}$, as we have done above for the planar case, is straightforward.

Assuming we agree to express everything in terms of $\{s\}$, then just as the
3.1. Rigid-Body Motions in the Plane

Figure 3.3: The frame \{b\} in \{s\} is given by \((P,p)\), and the frame \{c\} in \{b\} is given by \((Q,q)\). From these we can derive the frame \{c\} in \{s\}, described by \((R,r)\). The numerical values of the vectors \(p\), \(q\), and \(r\), and the coordinate axis directions of the three frames, are evident from the grid of unit squares.

Point \(p\) can be represented as a column vector \(p \in \mathbb{R}^2\) of the form

\[
p = \begin{bmatrix}
p_x \\
p_y
\end{bmatrix},
\]

(3.4)

the two vectors \(\hat{x}_b\) and \(\hat{y}_b\) can also be written as column vectors and packaged into the following 2 \times 2 matrix \(P\),

\[
P = [\hat{x}_b \ \hat{y}_b] = \begin{bmatrix}
cos \theta & -\sin \theta \\
\sin \theta & cos \theta
\end{bmatrix}.
\]

(3.5)

The matrix \(P\) is an example of a rotation matrix. Although \(P\) is constructed of four numbers, they are subject to three constraints (each column of \(P\) must be a unit vector, and the two columns must be orthogonal to each other), and the one remaining degree of freedom is parametrized by \(\theta\). Together, the pair \((P,p)\) provides a description of the orientation and position of \{b\} relative to \{s\}.

Now refer to the three frames in Figure 3.3. Repeating the approach above, and expressing \{c\} in \{s\} as the pair \((R,r)\), we can write

\[
r = \begin{bmatrix}
r_x \\
r_y
\end{bmatrix}, \quad R = \begin{bmatrix}
cos \phi & -\sin \phi \\
\sin \phi & cos \phi
\end{bmatrix}.
\]

(3.6)

We could also describe the frame \{c\} relative to \{b\}. Letting \(q\) denote the vector from the origin of \{b\} to the origin of \{c\} expressed in \{b\} coordinates, and letting \(Q\) denote the orientation of \{c\} relative to \{b\}, we can write \{c\}
relative to \{b\} as the pair \((Q,q)\), where
\[
q = \begin{bmatrix} q_x \\ q_y \end{bmatrix}, \quad Q = \begin{bmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{bmatrix}.
\]
(3.7)

If we know \((Q,q)\) (the configuration of \{c\} relative to \{b\}) and \((P,p)\) (the configuration of \{b\} relative to \{s\}), we can compute the configuration of \{c\} relative to \{s\} as follows:
\[
R = PQ \quad \text{(convert } Q \text{ to the } \{s\} \text{ frame)}
\]
(3.8)
\[
r = Pq + p \quad \text{(convert } q \text{ to the } \{s\} \text{ frame and vector sum with } p). \quad (3.9)
\]

Thus \((P,p)\) not only represents a configuration of \{b\} in \{s\}; it can also be used to convert the representation of a point or frame from \{b\} coordinates to \{s\} coordinates.

Now consider a rigid body with two frames attached to it, \{d\} and \{c\}. The frame \{d\} is initially coincident with \{s\}, and \{c\} is initially described by \((R,r)\) in \{s\} (Figure 3.4(a)). Then the body is moved so that \{d\} moves to \{d'\}, coincident with a frame \{b\} described by \((P,p)\) in \{s\}. Where does \{c\} end up after this motion? Denoting \((R',r')\) as the configuration of the new frame \{c'\}, you can verify that
\[
R' = PR \quad \text{(3.10)}
\]
\[
r' = Pr + p, \quad (3.11)
\]

very similar to Equations (3.8) and (3.9). The difference is that \((P,p)\) is expressed in the same frame as \((R,r)\), so the equations are not viewed as a change of coordinates, but instead as a rigid-body displacement (also known as a rigid-body motion) that \(\circlearrowleft\) rotates \{c\} according to \(P\) and \(\circlearrowright\) translates it by \(p\) in \{s\}. See Figure 3.4(a).

Thus we see that a rotation matrix-vector pair such as \((P,p)\) can be used to do three things:

(i) Represent a configuration of a rigid body in \{s\} (Figure 3.2).

(ii) Change the reference frame in which a vector or frame is represented (Figure 3.3).

(iii) Displace a vector or a frame (Figure 3.4(a)).

Referring to Figure 3.4(b), note that the rigid-body motion illustrated in Figure 3.4(a), expressed as a rotation followed by a translation, can be obtained by simply rotating the body about a fixed point \(s\) by an angle \(\beta\). This is a planar example of a screw motion.\(^1\) The displacement can therefore be parametrized by the three screw coordinates \((\beta,s_x,s_y)\), where \((s_x,s_y)\) denote the coordinates for the point \(s\) (i.e., the screw axis) in the fixed-frame \{s\}.

\(^1\)If the displacement is a pure translation without rotation, then \(s\) lies at infinity.
3.1. Rigid-Body Motions in the Plane

Figure 3.4: (a) The frame \{d\}, fixed to an elliptical rigid body and initially coincident with \{s\}, is displaced to \{d’\} (coincident with the stationary frame \{b\}), by first rotating according to \(P\) then translating according to \(p\), where \((P,p)\) is the representation of \{b\} in \{s\}. The same transformation takes the frame \{c\}, also attached to the rigid body, to \{c’\}. The transformation marked \(\mathbf{1}\) rigidly rotates \{c\} about the origin of \{s\}, and then transformation \(\mathbf{2}\) translates the frame by \(p\) expressed in \{s\}. (b) Instead of viewing this displacement as a rotation followed by a translation, both rotation and translation can be performed simultaneously. The displacement can be viewed as a rotation of \(\beta = 90^\circ\) about a fixed point \(s\).

Another way to represent the screw motion is to consider it the displacement obtained by following a simultaneous angular and linear velocity for a given amount of time. Inspecting Figure 3.4(b), we see that rotating about \(s\) with a unit angular velocity \((\omega = 1\ \text{rad/s})\) means that a point at the origin of the \{s\} frame moves at two units per second in the +\(\hat{x}\) direction of the \{s\} frame \((v = (v_x, v_y) = (2, 0))\). Packaging these together in a 3-vector as \(V = (\omega, v_x, v_y) = (1, 2, 0)\), we call this the planar twist (velocity) corresponding to a unit angular velocity rotation about \(s\). Following this planar twist for time (or angle) \(t = \pi/2\) yields the final displacement. We can now express the screw motion \((\beta, s_x, s_y)\) in the alternate form \(Vt = (\pi/2, \pi, 0)\). This form has some advantages, and we call these coordinates the exponential coordinate representation of the planar rigid-body displacement.
Remainder of the chapter. In the remainder of this chapter we generalize the concepts above to three-dimensional rigid-body motions. For this purpose consider a rigid body occupying three-dimensional physical space as shown in Figure 3.5. Assume that a length scale for physical space has been chosen, and that both the fixed frame \( \{s\} \) and body frame \( \{b\} \) have been chosen as shown. Throughout this book all reference frames are right-handed, i.e., the unit axes \( \{\hat{x}, \hat{y}, \hat{z}\} \) always satisfy \( \hat{x} \times \hat{y} = \hat{z} \). Denote the unit axes of the fixed frame by \( \{\hat{x}_s, \hat{y}_s, \hat{z}_s\} \) and the unit axes of the body frame by \( \{\hat{x}_b, \hat{y}_b, \hat{z}_b\} \). Let \( p \) denote the vector from the fixed frame origin to the body frame origin. In terms of the fixed frame coordinates, \( p \) can be expressed as

\[
p = p_1\hat{x}_s + p_2\hat{y}_s + p_3\hat{z}_s.
\] (3.12)

The axes of the body frame can also be expressed as

\[
\hat{x}_b = r_{11}\hat{x}_s + r_{21}\hat{y}_s + r_{31}\hat{z}_s
\] (3.13)

\[
\hat{y}_b = r_{12}\hat{x}_s + r_{22}\hat{y}_s + r_{32}\hat{z}_s
\] (3.14)

\[
\hat{z}_b = r_{13}\hat{x}_s + r_{23}\hat{y}_s + r_{33}\hat{z}_s.
\] (3.15)

Defining \( p \in \mathbb{R}^3 \) and \( R \in \mathbb{R}^{3 \times 3} \) as

\[
p = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}, \quad R = [\hat{x}_b \ \hat{y}_b \ \hat{z}_b] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},
\] (3.16)

the twelve parameters given by \((R, p)\) then provide a description of the position and orientation of the rigid body relative to the fixed frame.

Since the orientation of a rigid body has three degrees of freedom, only three of the nine entries in \( R \) can be chosen independently. One three-parameter representation of rotations are the exponential coordinates, which define an axis of rotation and the distance rotated about that axis. We leave other popular representations of orientations (three-parameter Euler angles and roll-pitch-yaw angles, the Cayley-Rodrigues parameters, and the unit quaternions that use four variables subject to one constraint) to Appendix B.
3.2 Rotations and Angular Velocities

We then examine six-parameter exponential coordinates for the configuration of a rigid body that arise from integrating a six-dimensional twist consisting of the body’s angular and linear velocity. This representation follows from the Chasles-Mozzi theorem that states that every rigid-body displacement can be described as a finite rotation and translation about a fixed screw axis.

We conclude with a discussion of forces and moments. Rather than treat these as separate three-dimensional quantities, we merge the moment and force vectors into a six-dimensional wrench. The twist and wrench, and rules for manipulating them, form the basis for the kinematic and dynamic analyses in the subsequent chapters.

3.2 Rotations and Angular Velocities

3.2.1 Rotation Matrices

We argued earlier that of the nine entries in the rotation matrix $R$, only three can be chosen independently. We begin by expressing a set of six explicit constraints on the entries of $R$. Recall that the three columns of $R$ correspond to the body frame’s unit axes $\{\hat{x}_b, \hat{y}_b, \hat{z}_b\}$. The following conditions must therefore be satisfied:

(i) Unit norm condition: $\hat{x}_b$, $\hat{y}_b$, and $\hat{z}_b$ are all of unit norm, or

$$
\begin{align*}
 r_{11}^2 + r_{21}^2 + r_{31}^2 &= 1 \\
r_{12}^2 + r_{22}^2 + r_{32}^2 &= 1 \\
r_{13}^2 + r_{23}^2 + r_{33}^2 &= 1.
\end{align*}
$$

(ii) Orthogonality condition: $\hat{x}_b \cdot \hat{y}_b = \hat{x}_b \cdot \hat{z}_b = \hat{y}_b \cdot \hat{z}_b = 0$ (here $\cdot$ denotes the inner product), or

$$
\begin{align*}
 r_{11}r_{12} + r_{21}r_{22} + r_{31}r_{32} &= 0 \\
r_{12}r_{13} + r_{22}r_{23} + r_{32}r_{33} &= 0 \\
r_{13}r_{11} + r_{23}r_{21} + r_{33}r_{31} &= 0.
\end{align*}
$$

These six constraints can be expressed more compactly as a single set of constraints on the matrix $R$,

$$
R^T R = I,
$$

where $R^T$ denotes the transpose of $R$ and $I$ denotes the identity matrix.

There is still the matter of accounting for the fact that the frame is right-handed (i.e., $\hat{x}_b \times \hat{y}_b = \hat{z}_b$, where $\times$ denotes the cross-product) rather than left-handed (i.e., $\hat{x}_b \times \hat{y}_b = -\hat{z}_b$); our six equality constraints above do not distinguish between right- and left-handed frames. We recall the following formula for evaluating the determinant of a $3 \times 3$ matrix $M$: denoting the three columns of $M$ by $a$, $b$, and $c$, respectively, its determinant is given by

$$
\det M = a^T (b \times c) = c^T (a \times b) = b^T (c \times a).
$$

(3.20)
Substituting the columns for $R$ into this formula then leads to the constraint
\[
\det R = 1. \tag{3.21}
\]
Note that if the frame had been left-handed, we would have $\det R = -1$. In summary, the six equality constraints represented by (3.19) imply that $\det R = \pm 1$; imposing the additional constraint $\det R = 1$ means that only right-handed frames are allowed. The constraint $\det R = 1$ does not change the number of independent continuous variables needed to parametrize $R$.

The set of $3 \times 3$ rotation matrices forms the **Special Orthogonal Group** $SO(3)$, which we now formally define:

**Definition 3.1.** The **Special Orthogonal Group** $SO(3)$, also known as the group of rotation matrices, is the set of all $3 \times 3$ real matrices $R$ that satisfy (i) $R^T R = I$ and (ii) $\det R = 1$.

The set of $2 \times 2$ rotation matrices is a subgroup of $SO(3)$, and is denoted $SO(2)$.

**Definition 3.2.** The **Special Orthogonal Group** $SO(2)$ is the set of all $2 \times 2$ real matrices $R$ that satisfy (i) $R^T R = I$ and (ii) $\det R = 1$.

From the definition it follows that every $R \in SO(2)$ can be written
\[
R = \begin{bmatrix}
r_{11} & r_{12} \\
r_{21} & r_{22}
\end{bmatrix} = \begin{bmatrix}
\cos \theta & -\sin \theta \\
\sin \theta & \cos \theta
\end{bmatrix},
\]
where $\theta \in [0, 2\pi)$. Elements of $SO(2)$ represent planar orientations and elements of $SO(3)$ represent spatial orientations.

### 3.2.1.1 Properties of Rotation Matrices

The sets of rotation matrices $SO(2)$ and $SO(3)$ are called “groups” because they satisfy the properties required of a mathematical group.\(^2\) Specifically, a group consists of a set of elements and an operation on two elements (matrix multiplication for $SO(n)$) such that, for all $A$, $B$ in the group, the following properties are satisfied:

- **closure**: $AB$ is also in the group.
- **associativity**: $(AB)C = A(BC)$.
- **identity element existence**: There exists an $I$ in the group (the identity matrix for $SO(n)$) such that $AI = IA = A$.
- **inverse element existence**: There exists an $A^{-1}$ in the group such that $AA^{-1} = A^{-1}A = I$.

\(^2\)More specifically, the $SO(n)$ groups are also called *Lie groups*, where “Lie” is pronounced “lee,” because the elements of the group form a differentiable manifold.
3.2. Rotations and Angular Velocities

Proofs of these properties are given below, using the fact that the identity matrix $I$ is a trivial example of a rotation matrix.

**Proposition 3.1.** The inverse of a rotation matrix $R \in SO(3)$ is also a rotation matrix, and it is equal to the transpose of $R$, i.e., $R^{-1} = R^T$.

*Proof.* The condition $R^T R = I$ implies that $R^{-1} = R^T$ and $RR^T = I$. Since $\det R^T = \det R = 1$, $R^T$ is also a rotation matrix. \qed

**Proposition 3.2.** The product of two rotation matrices is a rotation matrix.

*Proof.* Given $R_1, R_2 \in SO(3)$, their product $R_1 R_2$ satisfies $(R_1 R_2)^T (R_1 R_2) = R_2^T R_1^T R_1 R_2 = R_2^T R_2 = I$. Further, $\det R_1 R_2 = \det R_1 \cdot \det R_2 = 1$. Thus $R_1 R_2$ satisfies the conditions for a rotation matrix. \qed

**Proposition 3.3.** Multiplication of rotation matrices is associative, $(R_1 R_2) R_3 = R_1 (R_2 R_3)$, but generally not commutative, $R_1 R_2 \neq R_2 R_1$. For the special case of rotation matrices in $SO(2)$, rotations commute.

*Proof.* Associativity and non-commutativity follows from properties of matrix multiplication in linear algebra. Commutativity for planar rotations follows from a direct calculation. \qed

Another important property is that the action of a rotation matrix on a vector (e.g., rotating the vector) does not change the length of the vector.

**Proposition 3.4.** For any vector $x \in \mathbb{R}^3$ and $R \in SO(3)$, the vector $y = Rx$ is of the same length as $x$.

*Proof.* This follows from $\|y\|^2 = y^T y = (Rx)^T R x = x^T R^T R x = x^T x = \|x\|^2$. \qed

### 3.2.1.2 Uses of Rotation Matrices

Analogous to Section 3.1, there are three major uses for a rotation matrix $R$:

(i) Represent an orientation.

(ii) Change the reference frame in which a vector or a frame is represented.

(iii) Rotate a vector or a frame.

In the first use, $R$ is thought of as representing a frame; in the second and third uses, $R$ is thought of as an operator that acts on a vector or frame (changing its reference frame or rotating it).

To illustrate these uses, refer to Figure 3.6, which shows three different coordinate frames—{$a$}, {$b$}, and {$c$}—representing the same space.\(^3\) These frames have the same origin, since we are only representing orientations, but

\(^3\)In the rest of the book, all coordinate frames will use the same length scale; only their position and orientation may be different.
to make the axes clear, the figure shows the same space drawn three times. A point p in the space is also shown. Not shown is a fixed space frame \{s\}, which is aligned with \{a\}. The orientations of the three frames relative to \{s\} can be written

\[ R_a = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_b = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_c = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}, \]

and the location of the point p in these frames can be written

\[ p_a = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \quad p_b = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}, \quad p_c = \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}. \]

Note that \{b\} is obtained by rotating \{a\} about \(\hat{z}_a\) by 90°, and \{c\} is obtained by rotating \{b\} about \(\hat{y}_b\) by −90°. (The direction of positive rotation about an axis, \(\theta > 0\), is determined by the direction the fingers of your right hand curl about the axis when you point the thumb of your right hand along the axis.)

**Representing an orientation.** When we write \(R_c\), we are implicitly referring to the orientation of frame \{c\} relative to the fixed frame \{s\}. We can be more explicit about this by writing it as \(R_{ac}\): we are representing the frame of the second subscript, \{c\}, relative to the frame of the first subscript, \{s\}. This notation allows us to express a frame relative to a frame that is not \{s\}; for example, \(R_{bc}\) is the orientation of \{c\} relative to \{b\}. If there is no possibility of confusion regarding the frames involved, we may simply write \(R\).

Inspecting Figure 3.6, we see that

\[ R_{ac} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}, \quad R_{ca} = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}. \]

A simple calculation shows that \(R_{ac}R_{ca} = I\), i.e., \(R_{ac} = R_{ca}^{-1}\), or, equivalently, from Proposition 3.1, \(R_{ac} = R_{ca}^T\). In fact, for any two frames \{d\} and \{e\},

\[ R_{de} = R_{ed}^{-1} = R_{ed}^T. \]
3.2. Rotations and Angular Velocities

You can verify this fact using any two frames in Figure 3.6.

**Changing the reference frame.** The rotation matrix \( R_{ab} \) represents the orientation of \( \{b\} \) in \( \{a\} \), and \( R_{bc} \) represents the orientation of \( \{c\} \) in \( \{b\} \). A straightforward calculation shows that the orientation of \( \{c\} \) in \( \{a\} \) can be computed as

\[
R_{ac} = R_{ab}R_{bc}.
\] (3.22)

In the previous equation, \( R_{bc} \) can be viewed as a representation of an orientation, while \( R_{ab} \) can be viewed as a mathematical operator that changes the reference frame from \( \{b\} \) to \( \{a\} \), i.e.,

\[
R_{ac} = R_{ab}R_{bc} = \text{change reference frame from } \{b\} \text{ to } \{a\} (R_{bc}).
\]

A subscript cancellation rule helps to remember this property. When multiplying two rotation matrices, if the second subscript of the first matrix matches the first subscript of the second matrix, the two subscripts cancel and a change of reference frame is achieved:

\[
R_{ab}R_{bc} = R_{ac}.
\]

A rotation matrix is just a collection of three unit vectors, so the reference frame of a vector can also be changed by a rotation matrix using a modified version of the subscript cancellation rule:

\[
R_{ab}p_b = R_{ac}p_a.
\]

You can verify these properties using the frames and points in Figure 3.6.

**Rotating a vector or a frame.** The last use of a rotation matrix is to rotate a vector or a frame. Figure 3.7 shows a frame \( \{c\} \) initially aligned with \( \{s\} \) with axes \( \{\hat{x}, \hat{y}, \hat{z}\} \). If we rotate the \( \{c\} \) frame about a unit axis \( \hat{\omega} \) by an amount \( \theta \), the new \( \{c'\} \) frame has coordinate axes \( \{\hat{x}', \hat{y}', \hat{z}'\} \). The rotation matrix \( R = R_{sc'} \) represents the orientation of \( \{c'\} \) relative to \( \{s\} \), but instead we can think of it as representing the rotation operation itself. Emphasizing our view of \( R \) as a rotation operator, instead of as an orientation, we can write

\[
R = \text{Rot}(\hat{\omega}, \theta),
\]

the operation that rotates the orientation represented by the identity matrix to the orientation represented by \( R \). As we will see in Section 3.2.3.3, for \( \hat{\omega} = (\hat{\omega}_1, \hat{\omega}_2, \hat{\omega}_3) \),

\[
\text{Rot}(\hat{\omega}, \theta) = \\
\begin{bmatrix}
c_\theta + \hat{\omega}_1^2(1 - c_\theta) & \hat{\omega}_1\hat{\omega}_2(1 - c_\theta) - \hat{\omega}_3s_\theta & \hat{\omega}_1\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_2s_\theta \\
\hat{\omega}_1\hat{\omega}_2(1 - c_\theta) + \hat{\omega}_3s_\theta & c_\theta + \hat{\omega}_2^2(1 - c_\theta) & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_1s_\theta \\
\hat{\omega}_1\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_2s_\theta & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_1s_\theta & c_\theta + \hat{\omega}_3^2(1 - c_\theta)
\end{bmatrix},
\]

where \( c_\theta = \cos(\theta) \) and \( s_\theta = \sin(\theta) \).
A coordinate frame with axes \( \{ \hat{x}, \hat{y}, \hat{z} \} \) is rotated an amount \( \theta \) about a unit axis \( \hat{\omega} \) (which is aligned with \(-\hat{y}\) in this figure). The orientation of the final frame, with axes \( \{ \hat{x}', \hat{y}', \hat{z}' \} \), is written as \( R \) relative to the original frame.

where \( s_\theta = \sin \theta \) and \( c_\theta = \cos \theta \). Any \( R \in SO(3) \) can be obtained by rotating from the identity matrix by some \( \theta \) about some \( \hat{\omega} \).

Now say \( R_{sb} \) represents some \( \{ b \} \) relative to \( \{ s \} \), and we want to rotate \( \{ b \} \) by \( \theta \) about a unit axis \( \hat{\omega} \), i.e., \( R = \text{Rot}(\hat{\omega}, \theta) \). To be clear about what we mean, we have to specify whether the axis of rotation \( \hat{\omega} \) is expressed in \( \{ s \} \) coordinates or \( \{ b \} \) coordinates. Depending on our choice, the same numerical \( \hat{\omega} \) (and therefore the same numerical \( R \)) corresponds to different rotation axes in the underlying space, unless the \( \{ s \} \) and \( \{ b \} \) frames are aligned. Letting \( \{ b' \} \) be the new frame after rotating by \( \theta \) about \( \hat{\omega}_s = \hat{\omega} \) (the rotation axis \( \hat{\omega} \) is considered to be in the fixed \( \{ s \} \) frame), and \( \{ b'' \} \) be the new frame after rotating by \( \theta \) about \( \hat{\omega}_b = \hat{\omega} \) (the rotation axis \( \hat{\omega} \) is considered to be in the body \( \{ b \} \) frame), representations of these new frames can be calculated as

\[
R_{sb'} = \text{rotate by}_R \text{in}_\{ s \} \text{frame} \ (R_{sb}) = RR_{sb} \tag{3.23}
\]
\[
R_{sb''} = \text{rotate by}_R \text{in}_\{ b \} \text{frame} \ (R_{sb}) = R_{sb}R. \tag{3.24}
\]

In other words, premultiplying by \( R = \text{Rot}(\hat{\omega}, \theta) \) yields a rotation about \( \hat{\omega} \) considered to be in the fixed frame, and postmultiplying by \( R \) yields a rotation about \( \hat{\omega} \) considered to be in the body frame.

Rotating by \( R \) in the \( \{ s \} \) frame and the \( \{ b \} \) frame is illustrated in Figure 3.8.

To rotate a vector \( v \), note that there is only one frame involved, the frame that \( v \) is represented in, and therefore \( \hat{\omega} \) must be interpreted in this frame. The rotated vector \( v' \), in that same frame, is

\[ v' = Rv. \]

### 3.2.1.3 Other Representations of Orientations

Other popular representations of orientations include Euler angles, roll-pitch-yaw angles, the Cayley-Rodrigues parameters, and unit quaternions. These representations, and their relation to rotation matrices, are discussed in Appendix B.
3.2. Rotations and Angular Velocities

\[ R = \text{Rot}(\hat{z}, 90^\circ) \]

\[ R_{sb} = R_{sb} \]

\[ R_{ab} = R_{sb} R \]

3.2.2 Angular Velocity

Referring to Figure 3.9(a), suppose a body frame with unit axes \( \hat{x}, \hat{y}, \hat{z} \) is attached to a rotating body. Let us determine the time derivatives of these unit axes. Beginning with \( \hat{x} \), first note that \( \hat{x} \) is of unit length; only the direction of \( \hat{x} \) can vary with time (the same goes for \( \hat{y} \) and \( \hat{z} \)). If we examine the body frame at times \( t \) and \( t + \Delta t \), the change in frame orientation can be described as a rotation of angle \( \Delta \theta \) about some unit axis \( \hat{w} \) passing through the origin. The axis \( \hat{w} \) is coordinate-free; it is not yet represented in any particular reference frame.

In the limit as \( \Delta t \) approaches zero, the ratio \( \frac{\Delta \theta}{\Delta t} \) becomes the rate of rotation \( \dot{\theta} \), and \( \hat{w} \) can similarly be regarded as the instantaneous axis of rotation. In fact, \( \hat{w} \) and \( \dot{\theta} \) can be put together to define the **angular velocity** \( \omega \) as follows:

\[ \omega = \hat{w} \dot{\theta}. \]  

(3.25)
Figure 3.9: (Left) The instantaneous angular velocity vector. (Right) Calculating $\dot{\hat{x}}$.

Referring to Figure 3.9(b), it should be evident that

\begin{align*}
\dot{\hat{x}} &= w \times \hat{x} \quad (3.26) \\
\dot{\hat{y}} &= w \times \hat{y} \quad (3.27) \\
\dot{\hat{z}} &= w \times \hat{z}. \quad (3.28)
\end{align*}

To express these equations in coordinates, we have to choose a reference frame in which to represent $w$. We can choose any reference frame, but two natural choices are the fixed frame $\{s\}$ and the body frame $\{b\}$. Let us start with fixed frame $\{s\}$ coordinates. Let $R(t)$ be the rotation matrix describing the orientation of the body frame with respect to the fixed frame at time $t$; $R(t)$ is its time rate of change. The first column of $R(t)$, denoted $r_1(t)$, describes $\hat{x}$ in fixed frame coordinates; similarly, $r_2(t)$ and $r_3(t)$ respectively describe $\hat{y}$ and $\hat{z}$ in fixed frame coordinates. At a specific time $t$, let $\omega_s \in \mathbb{R}^3$ be the angular velocity $w$ expressed in fixed frame coordinates. Then Equations (3.26)–(3.28) can be expressed in fixed frame coordinates as

\begin{align*}
\dot{r}_i &= \omega_s \times r_i, \quad i = 1, 2, 3.
\end{align*}

These three equations can be rearranged into the following single $3 \times 3$ matrix equation:

\begin{equation}
\dot{R} = [\omega_s \times r_1 \mid \omega_s \times r_2 \mid \omega_s \times r_3] = \omega_s \times R. \quad (3.29)
\end{equation}

To eliminate the cross product in Equation (3.29), we introduce some new notation and rewrite $\omega_s \times R$ as $[\omega_s]R$, where $[\omega_s]$ is a $3 \times 3$ skew-symmetric matrix representation of $\omega_s \in \mathbb{R}^3$:

**Definition 3.3.** Given a vector $x = (x_1, x_2, x_3)^T \in \mathbb{R}^3$, define

\begin{equation}
[x] = \begin{bmatrix}
0 & -x_3 & x_2 \\
-x_3 & 0 & -x_1 \\
x_2 & x_1 & 0
\end{bmatrix}. \quad (3.30)
\end{equation}
The matrix \([x]\) is a 3 \times 3 \textbf{skew-symmetric} matrix representation of \(x\); that is,
\[ [x] = -[x]^T. \]
The set of all 3 \times 3 real skew-symmetric matrices is called \(so(3)\).\(^4\)

A useful property involving rotations and skew-symmetric matrices is the following:

**Proposition 3.5.** Given any \(\omega \in \mathbb{R}^3\) and \(R \in SO(3)\), the following always holds:

\[ R[\omega]R^T = [R\omega]. \quad (3.31) \]

**Proof.** Letting \(r_i^T\) be the \(i\)th row of \(R\),
\[
R[\omega]R^T = \begin{bmatrix}
  r_1^T(\omega \times r_1) & r_1^T(\omega \times r_2) & r_1^T(\omega \times r_3) \\
  r_2^T(\omega \times r_1) & r_2^T(\omega \times r_2) & r_2^T(\omega \times r_3) \\
  r_3^T(\omega \times r_1) & r_3^T(\omega \times r_2) & r_3^T(\omega \times r_3)
\end{bmatrix}
\]
\[
= \begin{bmatrix}
  0 & -r_3^T\omega & r_2^T\omega \\
  r_3^T\omega & 0 & -r_1^T\omega \\
  -r_2^T\omega & r_1^T\omega & 0
\end{bmatrix}
= [R\omega].
\]

where the second line makes use of the determinant formula for 3 \times 3 matrices, i.e., if \(M\) is a 3 \times 3 matrix with columns \(\{a, b, c\}\), then \(\det M = a^T(b \times c) = c^T(a \times b) = b^T(c \times a)\).

With the skew-symmetric notation, we can rewrite Equation (3.29) as
\[ [\omega_{s}]R = \dot{R}. \]

We can post-multiply both sides of Equation (3.33) by \(R^{-1}\) to get
\[ [\omega_{s}] = \dot{R}R^{-1}. \quad (3.34) \]

Now let \(\omega_{b}\) be \(w\) expressed in body frame coordinates. To see how to obtain \(\omega_{b}\) from \(\omega_{s}\) and vice versa, we explicitly write \(R\) as \(R_{sb}\). Then \(\omega_{s}\) and \(\omega_{b}\) are two different vector representations of the same angular velocity \(w\), and by our subscript cancellation rule, \(\omega_{s} = R_{sb}\omega_{b}\). Therefore
\[ \omega_{b} = R_{sb}^{-1}\omega_{s} = R^{-1}\omega_{s} = R^T\omega_{s}. \quad (3.35) \]

Let us now express this relation in skew-symmetric matrix form:
\[ [\omega_{b}] = [R^T\omega_{s}] = R^T[\omega_{s}]R \quad \text{(by Proposition 3.5)} \]
\[ = R^T(R^T\dot{R})R \]
\[ = R^T\dot{R} = R^{-1}\dot{R}. \quad (3.36) \]

\(^4\)The set of skew-symmetric matrices \(so(3)\) is called the Lie algebra of the Lie group \(SO(3)\). It consists of all possible \(\dot{R}\) when \(R = I\).
In summary, we have the following two equations that relate $R$ and $\dot{R}$ to the angular velocity $w$:

**Proposition 3.6.** Let $R(t)$ denote the orientation of the rotating frame as seen from the fixed frame. Denote by $w$ the angular velocity of the rotating frame. Then

\[
\dot{RR}^{-1} = [\omega_s] \quad (3.37) \\
R^{-1}\dot{R} = [\omega_b], \quad (3.38)
\]

where $\omega_s \in \mathbb{R}^3$ is the fixed frame vector representation of $w$ and $[\omega_s] \in \text{so}(3)$ is its $3 \times 3$ matrix representation, and $\omega_b \in \mathbb{R}^3$ is the body frame vector representation of $w$ and $[\omega_b] \in \text{so}(3)$ is its $3 \times 3$ matrix representation.

It is important to note that $\omega_b$ is *not* the angular velocity relative to a moving frame. Instead, $\omega_b$ is the angular velocity relative to the *stationary* frame $\{b\}$ that is instantaneously coincident with a frame attached to the moving body.

It is also important to note that the fixed-frame angular velocity $\omega_s$ *does not depend on the choice of the body frame*. Similarly, the body-frame angular velocity $\omega_b$ *does not depend on the choice of the fixed frame*. While Equations (3.37) and (3.38) may appear to depend on both frames (since $R$ and $\dot{R}$ individually depend on both $\{s\}$ and $\{b\}$), the product $\dot{RR}^{-1}$ is independent of $\{b\}$ and the product $R^{-1}\dot{R}$ is independent of $\{s\}$.

Finally, an angular velocity expressed in an arbitrary frame $\{d\}$ can be represented in another frame $\{c\}$ if we know the rotation that takes $\{c\}$ to $\{d\}$, using our now-familiar subscript cancellation rule:

\[\omega_c = R_{cd}\omega_d.\]

### 3.2.3 Exponential Coordinate Representation of Rotation

We now introduce a three-parameter representation for rotations, the *exponential coordinates for rotation*. The exponential coordinates parametrize a rotation matrix in terms of a rotation axis (represented by a unit vector $\hat{\omega}$), together with an angle of rotation $\theta$ about that axis; the vector $\hat{\omega}\theta \in \mathbb{R}^3$ then serves as the three-parameter exponential coordinate representation of the rotation. This representation is also called the *axis-angle* representation of a rotation, but we prefer to use the term “exponential coordinates” to emphasize the connection to the upcoming exponential coordinates for rigid-body transformations.

The exponential coordinates for a rotation can be interpreted equivalently as:

- rotating about the axis $\hat{\omega}$ by $\theta$;
- integrating the angular velocity $\hat{\omega}\theta$ for one second; or
- integrating the angular velocity $\hat{\omega}$ for $\theta$ seconds.
3.2. Rotations and Angular Velocities

The latter two views suggest that we consider exponential coordinates in the setting of linear differential equations. Below we briefly review some key results from linear differential equations.

3.2.3.1 Essential Results from Linear Differential Equations

Let us begin with the simple scalar linear differential equation

$$\dot{x}(t) = ax(t), \quad (3.39)$$

where $x(t) \in \mathbb{R}$, $a \in \mathbb{R}$ is constant, and the initial condition $x(0) = x_0$ is assumed given. Equation (3.39) has solution

$$x(t) = e^{at}x_0.$$ 

It is also useful to remember the series expansion of the exponential function:

$$e^{at} = 1 + at + \frac{(at)^2}{2!} + \frac{(at)^3}{3!} + \ldots$$

Now consider the vector linear differential equation

$$\dot{x}(t) = Ax(t) \quad (3.40)$$

where $x(t) \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$ is constant, and the initial condition $x(0) = x_0$ is given. From the earlier scalar result, one can conjecture a solution of the form

$$x(t) = e^{At}x_0 \quad (3.41)$$

where the matrix exponential $e^{At}$ now needs to be defined in a meaningful way. Again mimicking the scalar case, we define the matrix exponential to be

$$e^{At} = I + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \ldots \quad (3.42)$$

The first question to be addressed is under what conditions this series converges, so that the matrix exponential is well-defined. It can be shown that if $A$ is constant and finite, this series is always guaranteed to converge to a finite limit; the proof can be found in most texts on ordinary linear differential equations and is not covered here.

The second question is whether Equation (3.41), using Equation (3.42), is indeed a solution to Equation (3.40). Taking the time derivative of $x(t) = e^{At}x_0$,

$$\dot{x}(t) = \left(\frac{d}{dt}e^{At}\right)x_0$$

$$= \frac{d}{dt}\left(I + At + \frac{A^2t^2}{2!} + \frac{A^3t^3}{3!} + \ldots\right)x_0$$

$$= \left(A + A^2t + \frac{A^3t^2}{2!} + \ldots\right)x_0$$

$$= Ae^{At}x_0$$

$$= Ax(t), \quad (3.43)$$
which proves that \( x(t) = e^{At}x_0 \) is indeed a solution. That this is a unique solution follows from the basic existence and uniqueness result for linear ordinary differential equations, which we invoke here without proof.

While \( AB \neq BA \) for arbitrary square matrices \( A \) and \( B \), it is always true that

\[
A e^{At} = e^{At} A
\]

for any square \( A \) and scalar \( t \). You can verify this directly using the series expansion for the matrix exponential. Therefore, in line four of Equation (3.43), \( A \) could also have been factored to the right, i.e.,

\[
\dot{x}(t) = e^{At} Ax_0
\]

While the matrix exponential \( e^{At} \) is defined as an infinite series, closed-form expressions are often available. For example, if \( A \) can be expressed as \( A = PD P^{-1} \) for some \( D \in \mathbb{R}^{n \times n} \) and invertible \( P \in \mathbb{R}^{n \times n} \), then

\[
e^{At} = I + At + \frac{(At)^2}{2!} + \ldots
\]

\[
= I + (PDP^{-1})t + (PDP^{-1})(PDP^{-1})\frac{t^2}{2!} + \ldots
\]

\[
= P(I + Dt + \frac{(Dt)^2}{2!} + \ldots)P^{-1}
\]

\[
= Pe^{Dt}P^{-1}.
\]

If moreover \( D \) is diagonal, i.e., \( D = \text{diag}\{d_1, d_2, \ldots, d_n\} \), then its matrix exponential is particularly simple to evaluate:

\[
e^{Dt} = \begin{bmatrix}
    e^{d_1t} & 0 & \ldots & 0 \\
    0 & e^{d_2t} & \ldots & 0 \\
    \vdots & \vdots & \ddots & \vdots \\
    0 & 0 & \ldots & e^{d_nt}
\end{bmatrix}
\]

We summarize the results above in the following proposition.

**Proposition 3.7.** The linear differential equation \( \dot{x}(t) = Ax(t) \) with initial condition \( x(0) = x_0 \), where \( A \in \mathbb{R}^{n \times n} \) is constant and \( x(t) \in \mathbb{R}^n \), has solution

\[
x(t) = e^{At}x_0
\]

where

\[
e^{At} = I + tA + \frac{t^2}{2!}A^2 + \frac{t^3}{3!}A^3 + \ldots
\]

The matrix exponential \( e^{At} \) further satisfies the following properties:

(i) \( \frac{d}{dt}e^{At} = Ae^{At} = e^{At}A \).

(ii) If \( A = PD P^{-1} \) for some \( D \in \mathbb{R}^{n \times n} \) and invertible \( P \in \mathbb{R}^{n \times n} \), then \( e^{At} = Pe^{Dt}P^{-1} \).
3.2. Rotations and Angular Velocities

(iii) If $AB = BA$, then $e^A e^B = e^{A+B}$.

(iv) $(e^A)^{-1} = e^{-A}$.

The third property can be established by expanding the exponentials and comparing terms. The fourth property follows by setting $B = -A$ in the third property.

A final linear algebraic result useful in finding closed-form expressions for $e^{At}$ is the Cayley-Hamilton Theorem, which we state here without proof:

**Proposition 3.8.** Let $A \in \mathbb{R}^{n \times n}$ be constant, with characteristic polynomial

$$p(s) = \det(sI - A) = s^n + c_{n-1}s^{n-1} + \ldots + c_1s + c_0,$$

and define $p(A)$ as

$$p(A) = A^n + c_{n-1}A^{n-1} + \ldots + c_1A + c_0I.$$

Then $p(A) = 0$.

### 3.2.3.2 Exponential Coordinates of Rotations

The exponential coordinates of a rotation can be viewed equivalently as (1) a unit axis of rotation $\hat{\omega}$ ($\hat{\omega} \in \mathbb{R}^3$, $\|\hat{\omega}\| = 1$) together with a rotation angle about the axis $\theta \in \mathbb{R}$, or (2) as the three-vector obtained by multiplying the two together, $\hat{\omega}\theta \in \mathbb{R}^3$. When we represent the motion of a robot joint in the next chapter, the first view has the advantage of separating the description of the joint axis from the motion $\theta$ about the axis.

Referring to Figure 3.10, suppose a three-dimensional vector $p(0)$ is rotated by $\theta$ about $\hat{\omega}$ to $p(\theta)$; here we assume all quantities are expressed in fixed frame coordinates. This rotation can be achieved by imagining that $p(0)$ rotates at a constant rate of 1 rad/s (since $\hat{\omega}$ is unit) from time $t = 0$ to $t = \theta$. Let $p(t)$ denote this path. The velocity of $p(t)$, denoted $\dot{p}$, is then given by

$$\dot{p} = \hat{\omega} \times p.$$  \hspace{1cm} (3.49)

To see why this is true, let $\phi$ be the angle between $p(t)$ and $\hat{\omega}$. Observe that $p$ traces a circle of radius $\|p\| \sin \phi$ about the $\hat{\omega}$-axis. Then $\dot{p} = \hat{\omega} \times p$ is tangent to the path with magnitude $\|p\| \sin \phi$, which is exactly Equation (3.49).

The differential equation (3.49) can be expressed as

$$\dot{p} = [\hat{\omega}] p$$  \hspace{1cm} (3.50)

with initial condition $p(0)$. This is a linear differential equation of the form $\dot{x} = Ax$ that we studied earlier; its solution is given by

$$p(t) = e^{[\hat{\omega}]t} p(0).$$
Rigid-Body Motions

Figure 3.10: The vector \( p(0) \) is rotated by an angle \( \theta \) about the axis \( \hat{\omega} \), to \( p(\theta) \).

Since \( t \) and \( \theta \) are interchangeable, the equation above can also be written

\[ p(\theta) = e^{[\hat{\omega}]\theta} p(0). \]

We now derive a closed-form expression for \( e^{[\hat{\omega}]\theta} \). Here we make use of the Cayley-Hamilton Theorem. First, the characteristic polynomial associated with the \( 3 \times 3 \) matrix \( [\hat{\omega}] \) is given by

\[ p(s) = \det(sI - [\hat{\omega}]) = s^3 + s. \]

The Cayley-Hamilton Theorem then implies \( [\hat{\omega}]^3 + [\hat{\omega}] = 0 \), or \( [\hat{\omega}]^3 = -[\hat{\omega}] \).

Let us now expand the matrix exponential \( e^{[\hat{\omega}]\theta} \) in series form. Replacing \( [\hat{\omega}]^3 \) by \(-[\hat{\omega}]\), \( [\hat{\omega}]^4 \) by \(-[\hat{\omega}]^2\), \( [\hat{\omega}]^5 \) by \(-[\hat{\omega}]^3 = [\hat{\omega}] \), and so on, we obtain

\[
e^{[\hat{\omega}]\theta} = I + [\hat{\omega}]\theta + [\hat{\omega}]^2 \frac{\theta^2}{2!} + [\hat{\omega}]^3 \frac{\theta^3}{3!} + \ldots
\]

\[ = I + \left( \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \cdots \right) [\hat{\omega}] + \left( \frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \cdots \right) [\hat{\omega}]^2. \]

Now recall the series expansions for \( \sin \theta \) and \( \cos \theta \):

\[
\sin \theta = \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \ldots
\]

\[
\cos \theta = 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \ldots
\]

The exponential \( e^{[\hat{\omega}]\theta} \) therefore simplifies to the following:

\textbf{Proposition 3.9.} \textit{Given a vector } \( \hat{\omega} \theta \in \mathbb{R}^3 \), \textit{such that } \( \theta \) \textit{is any scalar and} \( \hat{\omega} \in \mathbb{R}^3 \) \textit{is a unit vector,}

\[ \text{Rot}(\hat{\omega}, \theta) = e^{[\hat{\omega}]\theta} = I + \sin \theta [\hat{\omega}] + (1 - \cos \theta) [\hat{\omega}]^2 \in SO(3). \]  (3.51)

\textit{This formula provides the matrix exponential of } \([\hat{\omega}]\theta \in \mathfrak{so}(3)\).
This formula is also known as Rodrigues’ formula for rotations.

We have shown how to use the matrix exponential to construct a rotation matrix from a rotation axis $\hat{\omega}$ and an angle $\theta$. Further, the quantity $e^{[\hat{\omega}]\theta}p$ amounts to rotating $p \in \mathbb{R}^3$ about the fixed-frame axis $\hat{\omega}$ by an angle $\theta$. Similarly, considering that a rotation matrix $R$ consists of three column vectors, the rotation matrix $R' = e^{[\hat{\omega}]\theta}R = \text{Rot}(\hat{\omega}, \theta)R$ is the orientation achieved by rotating $R$ by $\theta$ about the axis $\hat{\omega}$ in the fixed frame. Reversing the order of matrix multiplication, $R'' = Re^{-[\hat{\omega}]\theta} = R\text{Rot}(\hat{\omega}, \theta)$ is the orientation achieved by rotating $R$ by $\theta$ about $\hat{\omega}$ in the body frame.

**Example 3.1.** The frame $\{b\}$ in Figure 3.11 is obtained by rotating from an initial orientation aligned with the fixed frame $\{s\}$ about a unit axis $\hat{\omega} = (0, 0.866, 0.5)^T$ by an angle of $\theta = 30^\circ = 0.524$ rad. The rotation matrix representation of $\{b\}$ can be calculated as

$$R = e^{[\hat{\omega}]\theta} = \begin{bmatrix} 0 & -0.5 & 0.866 \\ 0.5 & 0 & 0 \\ -0.866 & 0 & 0 \end{bmatrix} + 0.134 \begin{bmatrix} 0 & -0.5 & 0.866 \\ 0.5 & 0 & 0 \\ -0.866 & 0 & 0 \end{bmatrix}^2 = \begin{bmatrix} 0.866 & -0.250 & 0.433 \\ 0.250 & 0.967 & 0.058 \\ -0.433 & 0.058 & 0.900 \end{bmatrix}.$$  

The frame $\{b\}$ can be represented by $R$ or by its exponential coordinates $\hat{\omega} = (0, 0.866, 0.5)^T$ and $\theta = 0.524$ rad, i.e., $\hat{\omega}\theta = (0, 0.453, 0.262)^T$.

If $\{b\}$ is then rotated by $-\theta = -0.524$ rad about the same fixed-frame axis $\hat{\omega}$, i.e.,

$$R' = e^{-[\hat{\omega}]\theta}R,$$

then we would find $R' = I$, as expected; the frame has rotated back to the identity (aligned with the $\{s\}$ frame). On the other hand, if $\{b\}$ were to be rotated by $-\theta$ about $\hat{\omega}$ in the body frame (this axis is different from $\hat{\omega}$ in the fixed frame), the new orientation would not be aligned with $\{s\}$:

$$R'' = Re^{-[\hat{\omega}]\theta} \neq I.$$

Our next task is to show that for any rotation matrix $R \in SO(3)$, one can always find a unit vector $\hat{\omega}$ and scalar $\theta$ such that $R = e^{[\hat{\omega}]\theta}$.

### 3.2.3.3 Matrix Logarithm of Rotations

If $\hat{\omega}\theta \in \mathbb{R}^3$ represents the exponential coordinates of a rotation matrix $R$, then the skew-symmetric matrix $[\hat{\omega}\theta] = [\hat{\omega}]\theta$ is the **matrix logarithm** of the rotation $R$. The matrix logarithm is the inverse of the matrix exponential. Just as the
matrix exponential “integrates” the matrix representation of an angular velocity \([\hat{\omega}]\theta \in so(3)\) for one second to give an orientation \(R \in SO(3)\), the matrix logarithm “differentiates” an \(R \in SO(3)\) to find the matrix representation of a constant angular velocity \([\hat{\omega}]\theta \in so(3)\) which, if integrated for one second, rotates a frame from \(I\) to \(R\). In other words,

\[
\text{exp} : [\hat{\omega}]\theta \in so(3) \rightarrow R \in SO(3) \\
\text{log} : R \in SO(3) \rightarrow [\hat{\omega}]\theta \in so(3)
\]

To derive the matrix logarithm, let us expand each of the entries for \(e^{[\hat{\omega}]\theta}\) in Equation (3.51),

\[
\begin{bmatrix}
c_\theta + \hat{\omega}_1^2(1 - c_\theta) & \hat{\omega}_1\hat{\omega}_2(1 - c_\theta) - \hat{\omega}_3 s_\theta & \hat{\omega}_1\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_2 s_\theta \\
\hat{\omega}_1\hat{\omega}_2(1 - c_\theta) + \hat{\omega}_3 s_\theta & c_\theta + \hat{\omega}_2^2(1 - c_\theta) & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_1 s_\theta \\
\hat{\omega}_1\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_2 s_\theta & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_1 s_\theta & c_\theta + \hat{\omega}_3^2(1 - c_\theta)
\end{bmatrix}
\]

(3.52)

where \(\hat{\omega} = (\hat{\omega}_1, \hat{\omega}_2, \hat{\omega}_3)^T\), and we use the shorthand notation \(s_\theta = \sin \theta\) and \(c_\theta = \cos \theta\). Setting the above equal to the given \(R \in SO(3)\) and subtracting the transpose from both sides leads to the following:

\[
\begin{align*}
r_{32} - r_{23} &= 2\hat{\omega}_1 \sin \theta \\
r_{13} - r_{31} &= 2\hat{\omega}_2 \sin \theta \\
r_{21} - r_{12} &= 2\hat{\omega}_3 \sin \theta.
\end{align*}
\]

Therefore, as long as \(\sin \theta \neq 0\) (or equivalently, \(\theta\) is not an integer multiple of \(\pi\)), we can write

\[
\begin{align*}
\hat{\omega}_1 &= \frac{1}{2 \sin \theta} (r_{32} - r_{23}) \\
\hat{\omega}_2 &= \frac{1}{2 \sin \theta} (r_{13} - r_{31}) \\
\hat{\omega}_3 &= \frac{1}{2 \sin \theta} (r_{21} - r_{12}).
\end{align*}
\]

The above equations can also be expressed in skew-symmetric matrix form as

\[
[\hat{\omega}] = \begin{bmatrix}
0 & -\hat{\omega}_3 & \hat{\omega}_2 \\
\hat{\omega}_3 & 0 & -\hat{\omega}_1 \\
-\hat{\omega}_2 & \hat{\omega}_1 & 0
\end{bmatrix} = \frac{1}{2 \sin \theta} (R - R^T).
\]

(3.53)
3.2. Rotations and Angular Velocities

Recall that $\hat{\omega}$ represents the axis of rotation for the given $R$. Because of the $\sin \theta$ term in the denominator, $[\hat{\omega}]$ is not well defined if $\theta$ is an integer multiple of $\pi$.\(^5\) We address this situation next, but for now let us assume this is not the case and find an expression for $\theta$. Setting $R$ equal to (3.52) and taking the trace of both sides (recall that the trace of a matrix is the sum of its diagonal entries),

$$\text{tr } R = r_{11} + r_{22} + r_{33} = 1 + 2 \cos \theta. \quad (3.54)$$

The above follows since $\hat{\omega}_1^2 + \hat{\omega}_2^2 + \hat{\omega}_3^2 = 1$. For any $\theta$ satisfying $1 + 2 \cos \theta = \text{tr } R$, such that $\theta$ is not an integer multiple of $\pi$, $R$ can be expressed as the exponential $e^{[\hat{\omega}]\theta}$ with $[\hat{\omega}]$ as given in Equation (3.53).

Let us now return to the case $\theta = k\pi$, where $k$ is some integer. When $k$ is an even integer, regardless of $\hat{\omega}$, we have rotated back to $R = I$, so the vector $\hat{\omega}$ is undefined. When $k$ is an odd integer (corresponding to $\theta = \pm \pi, \pm 3\pi, \ldots$), which in turn implies $\text{tr } R = -1$, the exponential formula (3.51) simplifies to

$$R = e^{[\hat{\omega}]\pi} = I + 2[\hat{\omega}]^2. \quad (3.55)$$

The three diagonal terms of Equation (3.55) can be manipulated to

$$\hat{\omega}_i = \pm \sqrt{\frac{r_{ii} + 1}{2}}, \quad i = 1, 2, 3. \quad (3.56)$$

The off-diagonal terms lead to the following three equations:

$$\begin{align*}
2\hat{\omega}_1\hat{\omega}_2 &= r_{12} \\
2\hat{\omega}_2\hat{\omega}_3 &= r_{23} \\
2\hat{\omega}_1\hat{\omega}_3 &= r_{13}.
\end{align*} \quad (3.57)$$

From Equation (3.55) we also know that $R$ must be symmetric: $r_{12} = r_{21}$, $r_{23} = r_{32}$, $r_{13} = r_{31}$. Both Equations (3.56) and (3.57) may be necessary to obtain a solution for $\hat{\omega}$. Once a solution $\hat{\omega}$ has been found, then $R = e^{[\hat{\omega}]\theta}$, where $\theta = \pm \pi, \pm 3\pi, \ldots$.

From the above it can be seen that solutions for $\theta$ exist at $2\pi$ intervals. If we restrict $\theta$ to the interval $[0, \pi]$, then the following algorithm can be used to compute the matrix logarithm of the rotation matrix $R \in SO(3)$.

**Algorithm:** Given $R \in SO(3)$, find a $\theta \in [0, \pi]$ and a unit rotation axis $\hat{\omega} \in \mathbb{R}^3, ||\hat{\omega}|| = 1$, such that $e^{[\hat{\omega}]\theta} = R$. The vector $\hat{\omega}\theta \in \mathbb{R}^3$ comprises the exponential coordinates for $R$ and the skew-symmetric matrix $[\hat{\omega}]\theta \in so(3)$ is a matrix logarithm of $R$.

(i) If $R = I$, then $\theta = 0$ and $\hat{\omega}$ is undefined.

\(^5\)A singularity such as this is unavoidable for any three-parameter representation of rotation. Euler angles and roll-pitch-yaw angles suffer similar singularities.
(ii) If \( \text{tr} \, R = -1 \), then \( \theta = \pi \). Set \( \hat{\omega} \) to any of the following three vectors that is a feasible solution:

\[
\hat{\omega} = \frac{1}{\sqrt{2(1 + r_{33})}} \begin{bmatrix} r_{13} \\ r_{23} \\ 1 + r_{33} \end{bmatrix} \tag{3.58}
\]

or

\[
\hat{\omega} = \frac{1}{\sqrt{2(1 + r_{22})}} \begin{bmatrix} r_{12} \\ 1 + r_{22} \\ r_{32} \end{bmatrix} \tag{3.59}
\]

or

\[
\hat{\omega} = \frac{1}{\sqrt{2(1 + r_{11})}} \begin{bmatrix} 1 + r_{11} \\ r_{21} \\ r_{31} \end{bmatrix} \tag{3.60}
\]

(iii) Otherwise \( \theta = \cos^{-1} \left( \frac{\text{tr} \, R - 1}{2} \right) \in [0, \pi) \) and

\[
[\hat{\omega}] = \frac{1}{2 \sin \theta} (R - R^T). \tag{3.61}
\]

Since every \( R \in SO(3) \) satisfies one of the three cases in the algorithm, for every \( R \) there exists a set of exponential coordinates \( \hat{\omega} \theta \).

The formula for the logarithm suggests a picture of the rotation group \( SO(3) \) as a solid ball of radius \( \pi \) (see Figure 3.12): given a point \( r \in \mathbb{R}^3 \) in this solid ball, let \( \hat{\omega} = r/\|r\| \) be the unit axis in the direction from the origin to \( r \) and and \( \theta = \|r\| \) be the distance from the origin to \( r \), so that \( r = \hat{\omega} \theta \). The rotation
matrix corresponding to \( r \) can then be regarded as a rotation about the axis \( \hat{\omega} \) by an angle \( \theta \). For any \( R \in SO(3) \) such that \( \text{tr} \, R \neq -1 \), there exists a unique \( r \) in the interior of the solid ball such that \( e^{[r]} = R \). In the event that \( \text{tr} \, R = -1 \), \( \log R \) is given by two antipodal points on the surface of this solid ball. That is, if there exists some \( r \) such that \( R = e^{[r]} \) with \( \| r \| = \pi \), then \( R = e^{[-r]} \) also holds; both \( r \) and \( -r \) correspond to the same rotation \( R \).

### 3.3 Rigid-Body Motions and Twists

In this section we derive representations for rigid-body configurations and velocities that extend, but otherwise are analogous to, those in Section 3.2 for rotations and angular velocities. In particular, the homogeneous transformation matrix \( T \) is analogous to the rotation matrix \( R \); a screw axis \( S \) is analogous to a rotation axis \( \hat{\omega} \); a twist \( V \) can be expressed as \( S \dot{\theta} \) and is analogous to an angular velocity \( \omega = \hat{\omega} \dot{\theta} \); and exponential coordinates \( S \theta \in \mathbb{R}^6 \) for rigid-body motions are analogous to exponential coordinates \( \hat{\omega} \theta \in \mathbb{R}^3 \) for rotations.

#### 3.3.1 Homogeneous Transformation Matrices

We now consider representations for the combined orientation and position of a rigid body. A natural choice would be to use a rotation matrix \( R \in SO(3) \) to represent the orientation of \( \{b\} \) in \( \{s\} \) and a vector \( p \in \mathbb{R}^3 \) to represent the origin of \( \{b\} \) in \( \{s\} \). Rather than identifying \( R \) and \( p \) separately, we package them into a single matrix as follows.

**Definition 3.4.** The Special Euclidean Group \( SE(3) \), also known as the group of rigid-body motions or homogeneous transformations in \( \mathbb{R}^3 \), is the set of all \( 4 \times 4 \) real matrices \( T \) of the form

\[
T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix},
\]

where \( R \in SO(3) \) and \( p \in \mathbb{R}^3 \) is a column vector.

An element \( T \in SE(3) \) will sometimes be denoted \( (R,p) \). We begin this section by establishing some basic properties of \( SE(3) \), and explaining why we package \( R \) and \( p \) into this matrix form.

Many of the robotic mechanisms we have encountered thus far are planar. With planar rigid-body motions in mind, we make the following definition:

**Definition 3.5.** The Special Euclidean Group \( SE(2) \) is the set of all \( 3 \times 3 \) real matrices \( T \) of the form

\[
T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix},
\]

where \( R \in SO(2) \), \( p \in \mathbb{R}^2 \), and \( 0 \) denotes a row vector of two zeros.
A matrix $T \in SE(2)$ is always of the form

$$T = \begin{bmatrix} r_{11} & r_{12} & p_1 \\ r_{21} & r_{22} & p_2 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & p_1 \\ \sin \theta & \cos \theta & p_2 \\ 0 & 0 & 1 \end{bmatrix},$$

where $\theta \in [0, 2\pi)$.

### 3.3.1.1 Properties of Transformation Matrices

We now list some basic properties of transformation matrices, which can be proven by calculation. First, the identity $I$ is a trivial example of a transformation matrix. The first three properties confirm that $SE(3)$ is a group.

**Proposition 3.10.** The inverse of a transformation matrix $T \in SE(3)$ is also a transformation matrix, and has the following form

$$T^{-1} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix}. \quad (3.64)$$

**Proposition 3.11.** The product of two transformation matrices is also a transformation matrix.

**Proposition 3.12.** Multiplication of transformation matrices is associative, $(T_1T_2)T_3 = T_1(T_2T_3)$, but generally not commutative, $T_1T_2 \neq T_2T_1$.

Before stating the next proposition, we note that just as in Section 3.1, it is often useful to calculate the quantity $Rx + p$, where $x \in \mathbb{R}^3$ and $(R,p)$ represents $T$. If we append a ‘1’ to $x$, making it a four-dimensional vector, this computation can be performed as a single matrix multiplication:

$$T \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} Rx + p \\ 1 \end{bmatrix}. \quad (3.65)$$

The vector $(x^T, 1)^T$ is the representation of $x$ in **homogeneous coordinates**, and accordingly $T \in SE(3)$ is called a homogenous transformation. When, by an abuse of notation, we write $Tx$, we mean $Rx + p$.

**Proposition 3.13.** Given $T = (R, p) \in SE(3)$ and $x, y \in \mathbb{R}^3$, the following hold:

1. $\|Tx - Ty\| = \|x - y\|$, where $\|\cdot\|$ denotes the standard Euclidean norm in $\mathbb{R}^3$, i.e., $\|x\| = \sqrt{x^T x}$.
2. $\langle Tx - Tz, Ty - Tz \rangle = \langle x - z, y - z \rangle$ for all $z \in \mathbb{R}^3$, where $\langle \cdot, \cdot \rangle$ denotes the standard Euclidean inner product in $\mathbb{R}^3$, i.e., $\langle x, y \rangle = x^T y$. 


3.3. Rigid-Body Motions and Twists

In Proposition 3.13, $T$ is regarded as a transformation on points in $\mathbb{R}^3$, i.e., $T$ transforms a point $x$ to $Tx$. The first property then asserts that $T$ preserves distances, while the second asserts that $T$ preserves angles. Specifically, if $x, y, z \in \mathbb{R}^3$ represent the three vertices of a triangle, then the triangle formed by the transformed vertices $\{Tx, Ty, Tz\}$ has the same set of lengths and angles as those of the triangle $\{x, y, z\}$ (the two triangles are said to be isometric). One can easily imagine taking $\{x, y, z\}$ to be the points on a rigid body, in which case $\{Tx, Ty, Tz\}$ represents a displaced version of the rigid body. It is in this sense that $SE(3)$ can be identified with the rigid-body motions.

3.3.1.2 Uses of Transformation Matrices

As with rotation matrices, there are three major uses for a transformation matrix $T$:

(i) Represent the configuration (position and orientation) of a rigid body.

(ii) Change the reference frame in which a vector or frame is represented.

(iii) Displace a vector or frame.

In the first use, $T$ is thought of as representing the configuration of a frame; in the second and third uses, $T$ is thought of as an operator that acts to change the reference frame or to move a vector or a frame.

To illustrate these uses, we refer to the three reference frames $\{a\}$, $\{b\}$, and $\{c\}$, and the point $v$, in Figure 3.13. These frames are chosen so that the alignment of their axes is clear, allowing visual confirmation of calculations.

Representing a configuration. The fixed frame $\{s\}$ is coincident with $\{a\}$, and the frames $\{a\}$, $\{b\}$, and $\{c\}$, represented by $T_{sa} = (R_{sa}, p_{sa})$, $T_{sb} = (R_{sb}, p_{sb})$, $T_{sc} = (R_{sc}, p_{sc})$, respectively, can be expressed relative to $\{s\}$ by the rotations

$$R_{sa} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_{sb} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad R_{sc} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$
and the location of the origin of each frame relative to \{s\} can be written

\[ \begin{bmatrix} p_{sa} \\ p_{sb} \\ p_{sc} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -2 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}. \]

Since \{a\} is collocated with \{s\}, the transformation matrix \(T_{sa}\) constructed from \((R_{sa}, p_{sa})\) is the identity matrix.

Any frame can be expressed relative to any other frame, not just \{s\}; for example, \(T_{bc} = (R_{bc}, p_{bc})\) represents \{b\} relative to \{c\}:

\[ \begin{bmatrix} R_{bc} \\ p_{bc} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ -1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ -3 \\ -1 \end{bmatrix}. \]

It can also be shown, using Proposition 3.10, that

\[ T_{de} = T_{ed}^{-1} \]

for any two frames \{d\} and \{e\}.

**Changing the reference frame of a vector or a frame.** By a subscript cancellation rule analogous to that for rotations, for any three reference frames \{a\}, \{b\}, and \{c\}, and any vector \(v\) expressed in \{b\} as \(v_b\),

\[ T_{ab}T_{bc} = T_{ac} \]

\[ T_{ab}v_b = T_{ac}v_a, \]

where \(v_a\) is the vector \(v\) expressed in \{a\}.

**Displacing (rotating and translating) a vector or a frame.** A transformation matrix \(T\), viewed as the pair \((R, p) = (\text{Rot}(\hat{\omega}, \theta), p)\), can act on a frame \(T_{sb}\) by rotating it by \(\theta\) about an axis \(\hat{\omega}\) and translating it by \(p\). Whether we pre-multiply or post-multiply \(T_{sb}\) by the operator \(T\) determines whether the \(\hat{\omega}\) axis and \(p\) are interpreted in the fixed frame \{s\} or the body frame \{b\}:

\[ T_{sb'} = TT_{sb} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} RR_{sb} & Rp_{sb} + p \\ 0 & 1 \end{bmatrix} \]

(fixed frame)

(3.66)

\[ T_{sb''} = T_{sb}T = \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{sb}R & R_{sb}p + p_{sb} \\ 0 & 1 \end{bmatrix} \]

(body frame).

(3.67)

The fixed-frame transformation (pre-multiplication by \(T\)) can be interpreted as first rotating the \{b\} frame by \(\theta\) about an axis \(\hat{\omega}\) in the \{s\} frame (this rotation will cause the origin of \{b\} to move if it is not coincident with the origin of \{s\}), then translating it by \(p\) in the \{s\} frame to get the \{b'\} frame. The
3.3. Rigid-Body Motions and Twists

Figure 3.14: Fixed-frame and body-frame transformations corresponding to \( \hat{\omega} = (0, 0, 1)^T \), \( \theta = 90^\circ \), and \( p = (0, 2, 0)^T \). (Left) The frame \( \{b\} \) rotated by 90\(^\circ\) about \( \hat{z}_s \) and then translated by two units in \( \hat{y}_s \), resulting in the new frame \( \{b'\} \). (Right) The frame \( \{b\} \) translated by two units in \( \hat{y}_b \) and then rotated by 90\(^\circ\) about the \( \hat{z} \) axis of the body frame, resulting in the new frame \( \{b''\} \).

A body-frame transformation (post-multiplication by \( T \)) can be interpreted as first translating \( \{b\} \) by \( p \) considered to be in the \( \{b\} \) frame, then rotating about \( \hat{\omega} \) in this new body frame (this does not move the origin of the frame) to get \( \{b''\} \).

Fixed-frame and body-frame transformations are illustrated in Figure 3.14 for a transformation \( T \) with \( \hat{\omega} = (0, 0, 1)^T \), \( \theta = 90^\circ \), and \( p = (0, 2, 0)^T \), yielding

\[
T = (\text{Rot}(\hat{\omega}, \theta), p) = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\]

Beginning with the frame \( \{b\} \) represented by

\[
T_{sb} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -2 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},
\]

the new frame \( \{b'\} \) achieved by a fixed-frame transformation \( TT_{sb} \) and the new frame \( \{b''\} \) achieved by a body-frame transformation \( T_{sb}T \) are

\[
TT_{sb} = T_{sb'} = \begin{bmatrix} 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T_{sb}T = T_{sb''} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & -4 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\]
Example 3.2. Figure 3.15 shows a robot arm mounted on a wheeled mobile platform, and a camera fixed to the ceiling. Frames \( b \) and \( c \) are respectively attached to the wheeled platform and the end-effector of the robot arm, and frame \( d \) is attached to the camera. A fixed frame \( a \) has been established, and the robot must pick up the object with body frame \( e \). Suppose that the transformations \( T_{db} \) and \( T_{de} \) can be calculated from measurements obtained with the camera. The transformation \( T_{bc} \) can be calculated using the arm’s joint angle measurements. The transformation \( T_{ad} \) is assumed to be known in advance. Suppose these known transformations are given as follows:

\[
T_{db} = \begin{bmatrix}
0 & 0 & -1 & 250 \\
0 & -1 & 0 & -150 \\
-1 & 0 & 0 & 200 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
T_{de} = \begin{bmatrix}
0 & 0 & -1 & 300 \\
0 & -1 & 0 & 100 \\
-1 & 0 & 0 & 120 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
T_{ad} = \begin{bmatrix}
0 & 0 & -1 & 400 \\
0 & -1 & 0 & 50 \\
-1 & 0 & 0 & 300 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
T_{bc} = \begin{bmatrix}
0 & -1/\sqrt{2} & -1/\sqrt{2} & 30 \\
0 & 1/\sqrt{2} & -1/\sqrt{2} & -40 \\
1 & 0 & 0 & 25 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

In order to calculate how to move the robot arm to pick up the object, the configuration of the object relative to the robot hand, \( T_{ce} \), must be determined.
We know that
\[ T_{ab}T_{bc}T_{ce} = T_{ad}T_{de}, \]
where the only quantity besides \( T_{ce} \) not given to us directly is \( T_{ad} \). However, since \( T_{ad} = T_{db}T_{ad} \), we can determine \( T_{ce} \) as follows:
\[ T_{ce} = (T_{ad}T_{db}T_{bc})^{-1}T_{ad}T_{dc}. \]

From the given transformations,
\[
T_{ad}T_{de} = \begin{bmatrix}
1 & 0 & 0 & 280 \\
0 & 1 & 0 & -50 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
\[
T_{ad}T_{db}T_{bc} = \begin{bmatrix}
0 & -1/\sqrt{2} & 1/\sqrt{2} & 230 \\
0 & 1/\sqrt{2} & -1/\sqrt{2} & 160 \\
1 & 0 & 0 & 75 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
\[
(T_{ad}T_{db}T_{bc})^{-1} = \begin{bmatrix}
0 & 0 & 1 & -75 \\
-1/\sqrt{2} & 1/\sqrt{2} & 0 & 70/\sqrt{2} \\
-1/\sqrt{2} & -1/\sqrt{2} & 0 & 390/\sqrt{2} \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

from which \( T_{ce} \) is evaluated to be
\[
T_{ce} = \begin{bmatrix}
0 & 0 & 1 & -75 \\
-1/\sqrt{2} & 1/\sqrt{2} & 0 & 260/\sqrt{2} \\
-1/\sqrt{2} & -1/\sqrt{2} & 0 & 130/\sqrt{2} \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

### 3.3.2 Twists

We now consider both the linear and angular velocity of a moving frame. As before, denote by \( \{s\} \) and \( \{b\} \) the fixed (space) and moving (body) frames, respectively, and let
\[
T_{ab}(t) = T(t) = \begin{bmatrix}
R(t) & p(t) \\
0 & 1
\end{bmatrix}
\]

(3.68)
denote the homogeneous transformation of \( \{b\} \) as seen from \( \{s\} \) (to keep the notation uncluttered, for the time being we write \( T \) instead of the usual \( T_{ab} \)).

In Section 3.2.2 we discovered that pre- or post-multiplying \( \dot{R} \) by \( R^{-1} \) results in a skew-symmetric representation of the angular velocity vector, either in fixed or body frame coordinates. One might reasonably ask if a similar property carries over to \( \dot{T} \), i.e., whether \( T^{-1}\dot{T} \) and \( \dot{TT}^{-1} \) carry similar physical interpretations.
Let us first see what happens when we pre-multiply \( \dot{T} \) by \( T^{-1} \):

\[
T^{-1}\dot{T} = \begin{bmatrix} RT & -RT^T p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{R} & \dot{p} \\ 0 & 0 \end{bmatrix}
\]
\[
= \begin{bmatrix} RT\dot{R} & RT\dot{p} \\ 0 & 0 \end{bmatrix}
\]
\[
= \begin{bmatrix} [\omega_b] & v_b \\ 0 & 0 \end{bmatrix}.
\]

Recall that \( RT\dot{R} = [\omega_b] \) is just the skew-symmetric matrix representation of the angular velocity expressed in \( \{b\} \) coordinates. Also, \( \dot{p} \) is the linear velocity of the origin of \( \{b\} \) expressed in the fixed frame \( \{s\} \), and \( RT\dot{p} = v_b \) is this linear velocity expressed in the frame \( \{b\} \). Putting these two observations together, we can conclude that \( T^{-1}\dot{T} \) represents the linear and angular velocity of the moving frame relative to the stationary frame \( \{b\} \) currently aligned with the moving frame.

The previous calculation of \( T^{-1}\dot{T} \) suggests that it is reasonable to merge \( \omega_b \) and \( v_b \) into a single six-dimensional velocity vector. We define the spatial velocity in the body frame, or simply the body twist\(^6\), to be

\[
\mathcal{V}_b = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} \in \mathbb{R}^6.
\]

Just as it is convenient to have a skew-symmetric matrix representation of an angular velocity vector, it is convenient to have a matrix representation of a twist, as shown in Equation (3.70). We overload the \([\cdot]\) notation, writing

\[
T^{-1}\dot{T} = [\mathcal{V}_b] = \begin{bmatrix} [\omega_b] & v_b \\ 0 & 0 \end{bmatrix} \in se(3),
\]

where \( [\omega_b] \in so(3) \) and \( v_b \in \mathbb{R}^3 \). The set of all \( 4 \times 4 \) matrices of this form is called \( se(3) \), the matrix representation of velocities associated with the rigid-body configurations \( SE(3) \).\(^7\)

Now that we have a physical interpretation for \( T^{-1}\dot{T} \), let us evaluate \( \ddot{T}T^{-1} \):

\[
\ddot{T}T^{-1} = \begin{bmatrix} \dot{R} & \dot{p} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} RT & -RT^T p \\ 0 & 1 \end{bmatrix}
\]
\[
= \begin{bmatrix} \dot{R}RT & \dot{p} - \dot{R}RT^T p \\ 0 & 0 \end{bmatrix}
\]
\[
= \begin{bmatrix} [\omega_s] & v_s \\ 0 & 0 \end{bmatrix}.
\]

\(^6\)The term “twist” has been used in different ways in the mechanisms and screw theory literature. In robotics, however, it is common to use the term to refer to a spatial velocity. We adopt this usage to minimize verbiage, e.g., “spatial velocity in the body frame” vs. “body twist.”

\(^7\)\( se(3) \) is called the Lie algebra of the Lie group \( SE(3) \). It consists of all possible \( \dot{T} \) when \( T = I \).
3.3. Rigid-Body Motions and Twists

Figure 3.16: Physical interpretation of \( \mathbf{v}_s \). The initial (solid line) and displaced (dotted line) configurations of a rigid body.

Observe that the skew-symmetric matrix \( [\omega_s] = \dot{R}R^T \) is the angular velocity expressed in fixed frame coordinates, but that \( \mathbf{v}_s = \dot{p} - \dot{R}R^T \mathbf{p} \) is not the linear velocity of the body frame origin expressed in the fixed frame (that quantity would simply be \( \dot{p} \)). If we write \( \mathbf{v}_s \) as
\[
\mathbf{v}_s = \dot{p} - \omega_s \times \mathbf{p} = \dot{p} + \omega_s \times (-\mathbf{p}),
\]
the physical meaning of \( \mathbf{v}_s \) can now be inferred: imagining an infinitely large moving body, \( \mathbf{v}_s \) is the instantaneous velocity of the point on this body currently at the fixed frame origin, expressed in the fixed frame (see Figure 3.16).

As we did for \( \omega_b \) and \( \mathbf{v}_b \), we assemble \( \omega_s \) and \( \mathbf{v}_s \) into a six-dimensional twist,
\[
\mathbf{V}_s = \begin{bmatrix} \omega_s \\ \mathbf{v}_s \end{bmatrix} \in \mathbb{R}^6, \quad [\mathbf{V}_s] = \begin{bmatrix} [\omega_s] & \mathbf{v}_s \\ 0 & 0 \end{bmatrix} = \dot{T}T^{-1} \in se(3),
\]
where \( [\mathbf{V}_s] \) is the \( 4 \times 4 \) matrix representation of \( \mathbf{V}_s \). We call \( \mathbf{V}_s \) the **spatial velocity in the space frame**, or simply the **spatial twist**.

If we regard the moving body as being infinitely large, there is an appealing and natural symmetry between \( \mathbf{V}_s = (\omega_s, \mathbf{v}_s) \) and \( \mathbf{V}_b = (\omega_b, \mathbf{v}_b) \):

(i) \( \omega_b \) is the angular velocity expressed in \( \{b\} \), and \( \omega_s \) is the angular velocity expressed in \( \{s\} \); and

(ii) \( \mathbf{v}_b \) is the linear velocity of a point at the origin of \( \{b\} \) expressed in \( \{b\} \), and \( \mathbf{v}_s \) is the linear velocity of a point at the origin of \( \{s\} \) expressed in \( \{s\} \).

\( \mathbf{V}_b \) can be obtained from \( \mathbf{V}_s \) as follows:
\[
[\mathbf{V}_b] = T^{-1} \dot{T}T
\]
\[
= T^{-1} [\mathbf{V}_s] T.
\]

Going the other way,
\[
[\mathbf{V}_s] = T [\mathbf{V}_b] T^{-1}.
\]
Writing out the terms of Equation (3.77), we get
\[
V_s = \begin{bmatrix}
R[\omega_b]R^T & -R[\omega_b]R^T p + Rv_b \\
0 & 0
\end{bmatrix}
\]
which, using \(R[\omega]R^T = [R\omega]\) (Proposition 3.5) and \([\omega]p = -[p]\omega\) for \(p, \omega \in \mathbb{R}^3\), can be manipulated into the following relation between \(V_b\) and \(V_s\):
\[
\begin{bmatrix}
\omega_s \\
v_s
\end{bmatrix} = \begin{bmatrix}
R & 0 \\
[p]R & R
\end{bmatrix} \begin{bmatrix}
\omega_b \\
v_b
\end{bmatrix}.
\]
Because the \(6 \times 6\) matrix pre-multiplying \(V_b\) is useful for changing the frame of reference for twists and wrenches, as we will see shortly, we give it its own name.

**Definition 3.6.** Given \(T = (R, p) \in SE(3)\), its **adjoint representation** \([\text{Ad}_T]\) is
\[
[\text{Ad}_T] = \begin{bmatrix}
R & 0 \\
[p]R & R
\end{bmatrix} \in \mathbb{R}^{6 \times 6}.
\]
For any \(V \in \mathbb{R}^6\), the **adjoint map** associated with \(T\) is
\[
V' = [\text{Ad}_T]V,
\]
also sometimes written as
\[
V' = \text{Ad}_T(V).
\]
In terms of the matrix form \([V] \in se(3)\) of \(V \in \mathbb{R}^6\),
\[
[V'] = T[V]T^{-1}.
\]

The adjoint map satisfies the following properties, verifiable by direct calculation:

**Proposition 3.14.** Let \(T_1, T_2 \in SE(3)\), and \(V = (\omega, v)\). Then
\[
\text{Ad}_{T_1}(\text{Ad}_{T_2}(V)) = \text{Ad}_{T_1T_2}(V) \quad \text{or} \quad [\text{Ad}_{T_1}][\text{Ad}_{T_2}]V = [\text{Ad}_{T_1T_2}]V. \quad (3.78)
\]
Also, for any \(T \in SE(3)\) the following holds:
\[
[\text{Ad}_T]^{-1} = [\text{Ad}_{T^{-1}}], \quad (3.79)
\]
The second property follows from the first by choosing \(T_1 = T^{-1}\) and \(T_2 = T\), so that
\[
\text{Ad}_{T^{-1}}(\text{Ad}_T(V)) = \text{Ad}_{T^{-1}T}(V) = \text{Ad}_I(V) = V. \quad (3.80)
\]
3.3. Rigid-Body Motions and Twists

3.3.2.1 Summary of Results on Twists

The main results on twists derived thus far are summarized in the following proposition:

**Proposition 3.15.** Given a fixed (space) frame \( \{ s \} \) and a body frame \( \{ b \} \), let \( T_{sb}(t) \in SE(3) \) be differentiable, where

\[
T_{sb}(t) = \begin{bmatrix} R(t) & p(t) \\ 0 & 1 \end{bmatrix}.
\] (3.81)

Then

\[
T_{sb}^{-1} \dot{T}_{sb} = [\mathcal{V}_b] = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} \in se(3) \quad (3.82)
\]

is the matrix representation of the **body twist**, and

\[
\dot{T}_{sb} T_{sb}^{-1} = [\mathcal{V}_s] = \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} \in se(3) \quad (3.83)
\]

is the matrix representation of the **spatial twist**. The twists \( \mathcal{V}_s \) and \( \mathcal{V}_b \) are related by

\[
\mathcal{V}_s = \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = \begin{bmatrix} R \\ p \end{bmatrix} R \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = [\text{Ad}_{T_{sb}}] \mathcal{V}_b \quad (3.84)
\]

\[
\mathcal{V}_b = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = \begin{bmatrix} R^T \\ -R^T p \end{bmatrix} R \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = [\text{Ad}_{T_{bs}}] \mathcal{V}_s. \quad (3.85)
\]

Similarly, for any two frames \( \{ a \} \) and \( \{ b \} \), a twist represented as \( \mathcal{V}_a \) in \( \{ a \} \) is related to the representation \( \mathcal{V}_b \) in \( \{ b \} \) by

\[
\mathcal{V}_a = [\text{Ad}_{T_{ba}}] \mathcal{V}_b, \quad \mathcal{V}_b = [\text{Ad}_{T_{ab}}] \mathcal{V}_a.
\]

Again analogous to angular velocities, it is important to realize that for a given twist, its fixed-frame representation \( \mathcal{V}_s \) **does not depend on the choice of the body frame \( \{ b \} \)**, and its body-frame representation \( \mathcal{V}_b \) **does not depend on the choice of the fixed frame \( \{ s \} \)**.

**Example 3.3.** Figure 3.17 shows a top view of a car with a single front wheel driving on a plane. The \( \hat{z}_b \)-axis of the body frame \( \{ b \} \) is into the page and the \( \hat{z}_s \)-axis of the fixed frame \( \{ s \} \) is out of the page. The angle of the front wheel of the car causes the car’s motion to be a pure angular velocity \( \omega = 2 \text{ rad/s} \) about an axis out of the page, at the point \( r \) in the plane. Inspecting the figure, we can write \( r \) as \( r_s = (2, -1, 0)^T \) or \( r_b = (2, -1.4, 0)^T \); \( w \) as \( \omega_s = (0, 0, 2)^T \) or \( \omega_b = (0, 0, -2)^T \); and \( T_{sb} \) as

\[
T_{sb} = \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 0.4 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\]
Figure 3.17: The twist corresponding to the instantaneous motion of the chassis of a three-wheeled vehicle can be visualized as an angular velocity \( w \) about the point \( r \).

From the figure and simple geometry, we get
\[
\begin{align*}
v_s &= \omega_s \times (-r_s) = r_s \times \omega_s = (-2, -4, 0)^T \\
v_b &= \omega_b \times (-r_b) = r_b \times \omega_b = (2.8, 4, 0)^T
\end{align*}
\]
to get the twists \( V_s \) and \( V_b \):
\[
V_s = \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \\ -2 \\ -4 \\ 0 \end{bmatrix}, \quad V_b = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -2 \\ 2.8 \\ 4 \\ 0 \end{bmatrix}.
\]

To confirm these results, try calculating \( V_s = [\text{Ad}_{T_{bs}}] V_b \).

### The Screw Interpretation of a Twist

Just as an angular velocity \( \omega \) can be viewed as \( \hat{\omega} \hat{\theta} \), where \( \hat{\omega} \) is the unit rotation axis and \( \hat{\theta} \) is the rate of rotation about that axis, a twist \( V \) can be interpreted as a **screw axis** \( S \) and a velocity \( \hat{\theta} \) about the screw axis.

A screw axis represents the familiar motion of a screw: rotation about the axis while also translating along the axis. One representation of a screw axis \( S \) is the collection \( \{q, \hat{s}, h\} \), where \( q \in \mathbb{R}^3 \) is any point on the axis; \( \hat{s} \) is a unit vector in the direction of the axis; and \( h \) is the **screw pitch**, which defines the ratio of the linear velocity along the screw axis to the angular velocity \( \hat{\theta} \) about the screw axis (Figure 3.18).

Using Figure 3.18 and geometry, we can write the twist \( V = (\omega, v) \) corresponding to an angular velocity \( \hat{\theta} \) about \( S \) (represented by \( \{q, \hat{s}, h\} \)) as
\[
V = \begin{bmatrix} \omega \\ v \end{bmatrix} = \begin{bmatrix} \hat{s} \hat{\theta} \\ -\hat{\theta} \times q + h \hat{s} \hat{\theta} \end{bmatrix}.
\]
3.3. Rigid-Body Motions and Twists

Figure 3.18: A screw axis $S$ represented by a point $q$, a unit direction $\hat{s}$, and a pitch $h$.

Note that the linear velocity $v$ is the sum of two terms: one due to translation along the screw axis, $h\hat{s}\dot{\theta}$, and one due to the linear motion at the origin induced by rotation about the axis, $-\hat{s}\dot{\theta} \times q$. The first term is in the direction of $\hat{s}$, while the second term is in the plane orthogonal to $\hat{s}$. It is not hard to show that, for any $V = (\omega, v)$ where $\omega \neq 0$, there exists an equivalent screw axis $\{q, \hat{s}, h\}$ and velocity $\dot{\theta}$, where $\dot{s} = \omega/||\omega||$, $\dot{\theta} = ||\omega||$, $h = \hat{\omega}^T v/\dot{\theta}$, and $q$ is chosen so that the term $-\hat{s}\dot{\theta} \times q$ provides the portion of $v$ orthogonal to the screw axis.

If $\omega = 0$, then the pitch $h$ of the screw is infinite. So $\hat{s}$ is chosen as $v/||v||$, and $\dot{\theta}$ is interpreted as the linear velocity $||v||$ along $\hat{s}$.

Instead of representing the screw axis $S$ using the cumbersome collection $\{q, \hat{s}, h\}$, with the possibility that $h$ may be infinite and the non-uniqueness of $q$ (any $q$ along the screw axis may be used), we instead define the screw axis $S$ using a normalized version of any twist $V = (\omega, v)$ corresponding to motion along the screw:

(i) If $\omega \neq 0$: $S = V/||\omega|| = (\omega/||\omega||, v/||\omega||)$. The screw axis $S$ is simply $V$ normalized by the length of the angular velocity vector. The angular velocity about the screw axis is $\dot{\theta} = ||\omega||$, such that $S\dot{\theta} = V$.

(ii) If $\omega = 0$: $S = V/||v|| = (0, v/||v||)$. The screw axis $S$ is simply $V$ normalized by the length of the linear velocity vector. The linear velocity along the screw axis is $\dot{\theta} = ||v||$, such that $S\dot{\theta} = V$.

This leads to the following definition of a “unit” (normalized) screw axis:

**Definition 3.7.** For a given reference frame, a screw axis $S$ is written

$$S = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6,$$

where either (i) $||\omega|| = 1$ or (ii) $\omega = 0$ and $||v|| = 1$. If (i) $||\omega|| = 1$, then $v = -\omega \times q + h\omega$, where $q$ is a point on the axis of the screw and $h$ is the pitch of the screw ($h = 0$ for a pure rotation about the screw axis). If (ii) $||\omega|| = 0$ and
The $4 \times 4$ matrix representation $[S]$ of $S$ is

$$[S] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} \in se(3), \quad [\omega] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in so(3), \quad (3.86)$$

where the bottom row of $[S]$ consists of all zeros.

**Important**: Although we use the pair $(\omega, v)$ for both normalized screw axes (where one of $|\omega|$ or $|v|$ must be unit) and general twists (where there are no constraints on $\omega$ and $v$), their meaning should be clear from context.

Since a screw axis $S$ is just a normalized twist, a screw axis represented as $S_a$ in a frame $\{a\}$ is related to the representation $S_b$ in a frame $\{b\}$ by

$$S_a = [\text{Ad}_{T_{ab}}]S_b, \quad S_b = [\text{Ad}_{T_{ba}}]S_a.$$
3.3. Rigid-Body Motions and Twists

Noting the similarity between $G(\theta)$ and the series definition for $e^{[\omega]\theta}$, it is tempting to write $I + G(\theta)[\omega] = e^{[\omega]\theta}$, and to conclude that $G(\theta) = (e^{[\omega]\theta} - I)[\omega]^{-1}$. This is wrong: $[\omega]^{-1}$ does not exist (try computing $\det[\omega]$).

Instead we make use of the result $[\omega]^3 = -[\omega]$ that was obtained from the Cayley-Hamilton Theorem. In this case $G(\theta)$ can be simplified to

$$G(\theta) = I\theta + [\omega]\frac{\theta^2}{2!} + [\omega]\frac{\theta^3}{3!} + \ldots$$

$$= I\theta + \left(\frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \ldots\right)[\omega] + \left(\frac{\theta^3}{3!} - \frac{\theta^5}{5!} + \frac{\theta^7}{7!} - \ldots\right)[\omega]^2$$

$$= I\theta + (1 - \cos \theta)[\omega] + (\theta - \sin \theta)[\omega]^2. \quad (3.88)$$

Putting everything together,

**Proposition 3.16.** Let $S = (\omega, v)$ be a screw axis. If $\|\omega\| = 1$, then for any distance $\theta \in \mathbb{R}$ traveled along the axis,

$$e^{[S]\theta} = \begin{bmatrix} e^{[\omega]\theta} & (I\theta + (1 - \cos \theta)[\omega] + (\theta - \sin \theta)[\omega]^2)v \\ 0 & 1 \end{bmatrix}. \quad (3.89)$$

If $\omega = 0$ and $\|v\| = 1$, then

$$e^{[S]\theta} = \begin{bmatrix} I & v\theta \\ 0 & 1 \end{bmatrix}. \quad (3.90)$$

The latter result of the proposition can be verified directly from the series expansion of $e^{[S]\theta}$ with $\omega$ set to zero.

### 3.3.3.2 Matrix Logarithm of Rigid-Body Motions

The above derivation essentially provides a constructive proof of the Chasles-Mozzi Theorem. That is, given an arbitrary $(R, p) \in SE(3)$, one can always find a screw axis $S = (\omega, v)$ and a scalar $\theta$ such that

$$e^{[S]\theta} = e^{[\omega]\theta}.$$ 

In the simplest case, if $R = I$, then $\omega = 0$, and the preferred choice for $v$ is $v = p/\|p\|$ (this makes $\theta = \|p\|$ the translation distance). If $R$ is not the identity matrix and $\operatorname{tr} R \neq -1$, one solution is given by

$$[\omega] = \frac{1}{2\sin \theta}(R - R^T) \quad (3.92)$$

$$v = G^{-1}(\theta)p, \quad (3.93)$$

where $\theta$ satisfies $1 + 2\cos \theta = \operatorname{tr} R$. We leave as an exercise the verification of the following formula for $G^{-1}(\theta)$:

$$G^{-1}(\theta) = \frac{1}{\theta}I - \frac{1}{2}[\omega] + \left(\frac{1}{\theta} - \frac{1}{2} \cot \frac{\theta}{2}\right)[\omega]^2. \quad (3.94)$$
Finally, if \( \text{tr} \, R = -1 \), we choose \( \theta = \pi \), and \([\omega]\) can be obtained via the matrix logarithm formula on \( SO(3) \). Once \([\omega]\) and \( \theta \) have been determined, \( v \) can then be obtained as \( v = G^{-1}(\theta)p \).

**Algorithm:** Given \((R, p)\) written as \(T \in SE(3)\), find a \( \theta \in [0, \pi] \) and a screw axis \( S = (\omega, v) \in \mathbb{R}^6 \) such that \( e^{[S]\theta} = T \). The vector \( S\theta \in \mathbb{R}^6 \) comprises the exponential coordinates for \( T \) and the matrix \([S]\theta \in se(3)\) is a matrix logarithm of \( T \).

1. If \( R = I \), then set \( \omega = 0 \), \( v = p/\|p\| \), and \( \theta = \|p\| \).
2. If \( \text{tr} \, R = -1 \), then set \( \theta = \pi \), and \([\omega] = \log R\) as determined by the matrix logarithm formula on \( SO(3) \) for the case \( \text{tr} \, R = -1 \). \( v \) is then given by \( v = G^{-1}(\theta)p \).
3. Otherwise set \( \theta = \cos^{-1}\left(\frac{\text{tr} \, R - 1}{2}\right) \in [0, \pi) \) and
   
   \[
   [\omega] = \frac{1}{2 \sin \theta} (R - R^T) 
   \]
   \[
   v = G^{-1}(\theta)p, 
   \]
   (3.95) (3.96)

where \( G^{-1}(\theta) \) is given by Equation (3.94).

**Example 3.4.** As an example, we consider the special case of planar rigid-body motions and examine the matrix logarithm formula on \( SE(2) \). Suppose the initial and final configurations of the body are respectively represented by
3.4 Wrenches

Consider a linear force \( f \) acting on a rigid body at a point \( r \). Defining a reference frame \( \{a\} \), the point \( r \) can be represented as \( r_a \in \mathbb{R}^3 \), the force \( f \) can be
Rigid-Body Motions

Figure 3.20: Relation between a wrench represented as $\mathbf{F}_a$ and $\mathbf{F}_b$.

represented as $\mathbf{f}_a \in \mathbb{R}^3$, and this force creates a torque or **moment** $\mathbf{m}_a \in \mathbb{R}^3$ in the $\{a\}$ frame:

$$\mathbf{m}_a = \mathbf{r}_a \times \mathbf{f}_a.$$ 

Note that the point of application of the force along the line of action of the force is immaterial.

Just as with twists, we can merge the moment and force into a single six-dimensional **spatial force**, or **wrench**, expressed in the $\{a\}$ frame, $\mathbf{F}_a$:

$$\mathbf{F}_a = \begin{bmatrix} \mathbf{m}_a \\ \mathbf{f}_a \end{bmatrix} \in \mathbb{R}^6. \quad (3.97)$$

If more than one wrench acts on a rigid body, the total wrench on the body is simply the vector sum of the individual wrenches, provided the wrenches are expressed in the same frame. A wrench with zero linear component is called a **pure moment**.

A wrench in the $\{a\}$ frame can be represented in another frame $\{b\}$ if $\mathbf{T}_{ba}$ is known (Figure 3.20). One way to derive the relationship between $\mathbf{F}_a$ and $\mathbf{F}_b$ is to derive the appropriate transformations between the individual force and moment vectors based on techniques we have already used.

A simpler and more insightful way to derive the relationship between $\mathbf{F}_a$ and $\mathbf{F}_b$ is to (1) use the results we have already derived relating the representations $\mathbf{V}_a$ and $\mathbf{V}_b$ of the same twist, and (2) use the fact that the power generated (or dissipated) by an $(\mathbf{F}, \mathbf{V})$ pair must be the same regardless of the frame they are represented in. (Imagine if we could create power simply by changing our choice of a reference frame!) Recall that the dot product of a force and a velocity is power, and power is a coordinate-independent quantity. Because of this, we know

$$\mathbf{V}_b^T \mathbf{F}_b = \mathbf{V}_a^T \mathbf{F}_a. \quad (3.98)$$

From Proposition 3.15 we know that $\mathbf{V}_a = [\text{Ad}_{\mathbf{R}_{ab}}] \mathbf{V}_b$, and therefore Equation (3.98) can be rewritten as

$$\mathbf{V}_b^T \mathbf{F}_b = ([\text{Ad}_{\mathbf{R}_{ab}}] \mathbf{V}_b)^T \mathbf{F}_a$$

$$\quad = \mathbf{V}_b^T [\text{Ad}_{\mathbf{R}_{ab}}]^T \mathbf{F}_a.$$
3.5. Summary

Since this must hold for all $V_b$, this simplifies to

$$F_b = [Ad_{T_a}]^T F_a.$$  \hspace{1cm} (3.99)

Similarly,

$$F_a = [Ad_{T_b}]^T F_b.$$  \hspace{1cm} (3.100)

**Proposition 3.17.** Given a wrench $F$, represented in $\{a\}$ as $F_a$ and in $\{b\}$ as $F_b$, the two representations are related by

$$F_b = Ad_{T_a}^T (F_a) = [Ad_{T_a}]^T F_a$$  \hspace{1cm} (3.101)

$$F_a = Ad_{T_b}^T (F_b) = [Ad_{T_b}]^T F_b.$$  \hspace{1cm} (3.102)

3.5 Summary

The following table succinctly summarizes some of the key concepts from the chapter, as well as the parallelism between rotations and rigid-body motions. For more details, consult the appropriate section of the chapter.
**Rotations**  
\[ R \in SO(3) : 3 \times 3 \text{ matrices satisfying} \]
\[ R^T R = I, \det R = 1 \]

**Rigid-Body Motions**  
\[ T \in SE(3) : 4 \times 4 \text{ matrices} \]
\[ T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}, \]
where \( R \in SO(3), p \in \mathbb{R}^3 \)

\[ R^{-1} = R^T \]
\[ T^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix} \]

change of coord frame:  
\[ R_{ab} R_{bc} = R_{ac}, \quad R_{ab} p_b = p_a \]

rotating a frame \( \{b\} \):  
\[ R = \text{Rot}(\hat{\omega}, \theta) \]
\[ R_{ab} = R R_{ab} : \text{ rotate } \theta \text{ about } \hat{\omega}_a = \hat{\omega} \]
\[ R_{ab} = R_{ab} R : \text{ rotate } \theta \text{ about } \hat{\omega}_b = \hat{\omega} \]

unit rotation axis is \( \hat{\omega} \in \mathbb{R}^3 \),  
where \( \| \hat{\omega} \| = 1 \)

“unit” screw axis is \( S = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6 \),  
where either (i) \( \| \omega \| = 1 \) or

(ii) \( \omega = 0 \text{ and } \| v \| = 1 \)

for a screw axis \( \{q, \hat{s}, h\} \) with finite \( h \),  
\[ S = \begin{bmatrix} \omega \\ v \end{bmatrix} = \begin{bmatrix} \hat{s} \\ -\hat{s} \times q + h\hat{s} \end{bmatrix} \]

angular velocity can be written \( \omega = \hat{\omega} \dot{\theta} \)  
twist can be written \( \mathcal{V} = S \dot{\theta} \)

for any 3-vector, e.g., \( \omega \in \mathbb{R}^3 \),  
\[ \omega = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in \mathfrak{so}(3) \]

\[ \mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathfrak{se}(3) \]

\[ [\omega] = -[\omega]^T, [\omega] x = -[x] \omega, \quad [\omega][x] = ([x][\omega])^T, R[\omega] R^T = [R \omega] \]

identities: for \( \omega, x \in \mathbb{R}^3, R \in SO(3) \):

\[ [\omega] = -[\omega]^T, [\omega] x = -[x] \omega, \quad [\omega][x] = ([x][\omega])^T, R[\omega] R^T = [R \omega] \]

identities: \[ [\mathcal{T}]^{-1} = [\mathcal{T}^{-1}] \]

\[ [\text{Ad}_T] = \begin{bmatrix} R & 0 \\ p R & R \end{bmatrix} \in \mathbb{R}^{6 \times 6} \]

\[ [\text{Ad}_{T_a}] [\text{Ad}_{T_b}] = [\text{Ad}_{T_a T_b}] \]

change of coord frame:  
\[ \hat{\omega}_a = R_{ab} \hat{\omega}_b, \quad \omega_a = R_{ab} \omega_b \]

\[ S_a = [\text{Ad}_{T_a} | S_b, \quad \mathcal{V}_a = [\text{Ad}_{T_a} | \mathcal{V}_b \]

continued...
3.6 Software

The following functions are included in the software distribution accompanying the book. The code below is in MATLAB format, but the code is available in other languages. For more details on the software, consult the code and its documentation.

```
invR = RotInv(R)
Computes the inverse of the rotation matrix R.

so3mat = VecToSo3(omg)
Returns the 3×3 skew-symmetric matrix corresponding to omg.

omg = so3ToVec(so3mat)
Returns the 3-vector corresponding to the 3×3 skew-symmetric matrix so3mat.

[omghat,theta] = AxisAng3(expc3)
Extracts the rotation axis \( \hat{\omega} \) and rotation amount \( \theta \) from the 3-vector \( \hat{\omega}\theta \) of exponential coordinates for rotation, expc3.

R = MatrixExp3(expc3)
Computes the rotation matrix corresponding to a 3-vector of exponential coordinates for rotation. (Note: This function takes exponential coordinates as input, not an \( so(3) \) matrix as implied by the function’s name.)

expc3 = MatrixLog3(R)
Computes the 3-vector of exponential coordinates corresponding to the matrix logarithm of R. (Note: This function returns exponential coordinates, not an \( so(3) \) matrix as implied by the function’s name.)

T = RpToTrans(R,p)
Builds the homogeneous transformation matrix T corresponding to a rotation
```
matrix $\mathbf{R} \in SO(3)$ and a position vector $\mathbf{p} \in \mathbb{R}^3$.

$$[\mathbf{R}, \mathbf{p}] = \text{TransToRp}(\mathbf{T})$$

Extracts the rotation matrix and position vector from a homogeneous transformation matrix $\mathbf{T}$.

$$\text{invT} = \text{TransInv}(\mathbf{T})$$

Computes the inverse of a homogeneous transformation matrix $\mathbf{T}$.

$$\text{se3mat} = \text{VecTose3}(\mathbf{V})$$

Returns the $se(3)$ matrix corresponding to a 6-vector twist $\mathbf{V}$.

$$\mathbf{V} = \text{se3ToVec}(\text{se3mat})$$

Returns the 6-vector twist corresponding to an $se(3)$ matrix $\text{se3mat}$.

$$\text{AdT} = \text{Adjoint}(\mathbf{T})$$

Computes the $6 \times 6$ adjoint representation $[\text{Ad}_T]$ of the homogeneous transformation matrix $\mathbf{T}$.

$$\mathbf{S} = \text{ScrewToAxis}(\mathbf{q}, \mathbf{s}, h)$$

Returns a normalized screw axis representation $\mathbf{S}$ of a screw described by a unit vector $\mathbf{s}$ in the direction of the screw axis, located at the point $\mathbf{q}$, with pitch $h$.

$$[\mathbf{S}, \theta] = \text{AxisAng}(\text{expc6})$$

Extracts the normalized screw axis $\mathbf{S}$ and distance traveled along the screw $\theta$ from the 6-vector of exponential coordinates $S\theta$.

$$\mathbf{T} = \text{MatrixExp6}(\text{expc6})$$

Computes the homogeneous transformation matrix corresponding to a 6-vector $S\theta$ of exponential coordinates for rigid-body motion. (Note: This function takes exponential coordinates as input, not an $se(3)$ matrix as implied by the function’s name.)

$$\text{expc6} = \text{MatrixLog}(\mathbf{T})$$

Computes the 6-vector of exponential coordinates corresponding to the matrix logarithm of $\mathbf{T}$. (Note: This function returns exponential coordinates, not an $se(3)$ matrix as implied by the function’s name.)

### 3.7 Notes and References

More detailed coverage of the various parametrizations of $SO(3)$ can be found in, e.g., [96] and the references cited there. The treatment of the matrix exponential representation for screw motions is based on the work of Brockett [14]; a more mathematically detailed discussion can be found in [90]. Classical screw theory is presented in its original form in R. Ball’s treatise [4]. More modern (algebraic and geometric) treatments can be found in, e.g., Bottema and Roth [13], Angeles [1], and McCarthy [86].
3.8 Exercises

1. In terms of the $\hat{x}_s$, $\hat{y}_s$, $\hat{z}_s$ coordinates of a fixed space frame $\{s\}$, the frame $\{a\}$ has an $\hat{x}_a$-axis pointing in the direction $(0, 0, 1)$ and a $\hat{y}_a$-axis pointing in the direction $(-1, 0, 0)$, and the frame $\{b\}$ has an $\hat{x}_b$-axis pointing in the direction $(1, 0, 0)$ and a $\hat{y}_b$-axis pointing in the direction $(0, 0, -1)$.
   (a) Give your best hand drawing of the three frames. Draw them at different locations so they are easy to see.
   (b) Write the rotation matrices $R_{sa}$ and $R_{sb}$.
   (c) Given $R_{sb}$, how do you calculate $R_{sb}^{-1}$ without using a matrix inverse? Write $R_{sb}^{-1}$ and verify its correctness with your drawing.
   (d) Given $R_{sa}$ and $R_{sb}$, how do you calculate $R_{ab}$ (again no matrix inverses)? Compute the answer and verify its correctness with your drawing.
   (e) Let $R_{sb}$ be considered as a transformation operator consisting of a rotation about $\hat{x}_b$ by $-90^\circ$. Calculate $R_{1} = R_{sa}R_{1}$, and think of $R_{sa}$ as a representation of an orientation, $R$ as a rotation of $R_{sa}$, and $R_1$ as the new orientation after performing the rotation. Does the new orientation $R_1$ correspond to rotating $R_{sa}$ by $-90^\circ$ about the world-fixed $\hat{x}_s$-axis or the body-fixed $\hat{x}_a$-axis? Now calculate $R_2 = RR_{sa}$. Does the new orientation $R_2$ correspond to rotating $R_{sa}$ by $-90^\circ$ about the world-fixed $\hat{x}_s$-axis or the body-fixed $\hat{x}_a$-axis?
   (f) Use $R_{ab}$ to change the representation of the point $p_b = (1, 2, 3)^T$ (in $\{b\}$ coordinates) to $\{s\}$ coordinates.
   (g) Choose a point $p$ represented by $p_s = (1, 2, 3)^T$ in $\{s\}$ coordinates. Calculate $p' = R_{sb}p_s$ and $p'' = R_{sb}^{T}p_s$. For each operation, should the result be interpreted as changing coordinates (from the $\{s\}$ frame to $\{b\}$) without moving the point $p$, or as moving the location of the point without changing the reference frame of the representation?
   (h) An angular velocity $\omega$ is represented in $\{s\}$ as $\omega_s = (3, 2, 1)^T$. What is its representation $\omega_a$?
   (h) By hand, calculate the matrix logarithm $[\hat{\omega}]\theta$ of $R_{sa}$. (You may verify your answer with software.) Extract the unit angular velocity $\hat{\omega}$ and rotation amount $\theta$. Redraw the fixed frame $\{s\}$ and in it draw $\hat{\omega}$.
   (i) Calculate the matrix exponential corresponding to the exponential coordinates of rotation $\hat{\omega}\theta = (1, 2, 0)^T$. Draw the corresponding frame relative to $\{s\}$, as well as the rotation axis $\hat{\omega}$.

2. Let $p$ be a point whose coordinates are $p = (\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{2}})$ with respect to the fixed frame $\hat{x}$-$\hat{y}$-$\hat{z}$. Suppose $p$ is rotated about the fixed frame $\hat{x}$-axis by 30 degrees, then about the fixed frame $\hat{y}$-axis by 135 degrees, and finally about the fixed frame $\hat{z}$-axis by $-120$ degrees. Denote the coordinates of this newly rotated point by $p'$.
   (a) What are the coordinates of $p'$?
(b) Find the rotation matrix $R$ such that $p' = Rp$ for the $p'$ you obtained in (a).

3. Suppose $p_i \in \mathbb{R}^3$ and $p'_i \in \mathbb{R}^3$ are related by $p'_i = Rp_i$, $i = 1, 2, 3$, for some unknown rotation matrix $R$ to be determined. Find, if it exists, the rotation $R$ for the three input-output pairs $p_i \mapsto p'_i$:

\[
\begin{align*}
p_1 = (\sqrt{2}, 0, 2)^T & \mapsto p'_1 = (0, 2, \sqrt{2})^T \\
p_2 = (1, 1, -1)^T & \mapsto p'_2 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, -\sqrt{2}\right)^T \\
p_3 = (0, 2\sqrt{2}, 0)^T & \mapsto p'_3 = (-\sqrt{2}, \sqrt{2}, -2)^T.
\end{align*}
\]

4. In this exercise you are asked to prove the property $R_{ab}R_{bc} = R_{ac}$ of Equation (3.22). Define the unit axes of frames $\{a\}, \{b\}, \{c\}$ by the triplet of orthogonal unit vectors $\{\hat{x}_a, \hat{y}_a, \hat{z}_a\}$, $\{\hat{x}_b, \hat{y}_b, \hat{z}_b\}$, and $\{\hat{x}_c, \hat{y}_c, \hat{z}_c\}$, respectively. Suppose that the unit axes of frame $\{b\}$ can be expressed in terms of the unit axes of frame $\{a\}$ by

\[
\begin{align*}
\hat{x}_b &= r_{11}\hat{x}_a + r_{21}\hat{y}_a + r_{31}\hat{z}_a \\
\hat{y}_b &= r_{12}\hat{x}_a + r_{22}\hat{y}_a + r_{32}\hat{z}_a \\
\hat{z}_b &= r_{13}\hat{x}_a + r_{23}\hat{y}_a + r_{33}\hat{z}_a.
\end{align*}
\]

Similarly, suppose the unit axes of frame $\{c\}$ can be expressed in terms of the unit axes of frame $\{b\}$ by

\[
\begin{align*}
\hat{x}_c &= s_{11}\hat{x}_b + s_{21}\hat{y}_b + s_{31}\hat{z}_b \\
\hat{y}_c &= s_{12}\hat{x}_b + s_{22}\hat{y}_b + s_{32}\hat{z}_b \\
\hat{z}_c &= s_{13}\hat{x}_b + s_{23}\hat{y}_b + s_{33}\hat{z}_b.
\end{align*}
\]

From the above prove that $R_{ab}R_{bc} = R_{ac}$.

5. Find the exponential coordinates $\hat{\omega}\theta \in \mathbb{R}^3$ for the $SO(3)$ matrix

\[
\begin{bmatrix}
0 & -1 & 0 \\
0 & 0 & -1 \\
1 & 0 & 0
\end{bmatrix}.
\]

6. Given $R = \text{Rot}(\hat{x}, \frac{\pi}{2})\text{Rot}(\hat{z}, \pi)$, find the unit vector $\hat{\omega}$ and angle $\theta$ such that $R = e^{[\hat{\omega}]\theta}$.

7. (a) Given the rotation matrix

\[
R = \begin{bmatrix}
0 & 0 & 1 \\
0 & -1 & 0 \\
1 & 0 & 0
\end{bmatrix},
\]
3.8. Exercises

find all possible values for \( \hat{\omega} \in \mathbb{R}^3, \|\hat{\omega}\| = 1 \), and \( \theta \in [0, 2\pi] \) such that \( e^{[\hat{\omega}]\theta} = R \).

(b) The two vectors \( v_1, v_2 \in \mathbb{R}^3 \) are related by

\[
v_2 = Rv_1 = e^{[\hat{\omega}]\theta}v_1
\]

where \( \hat{\omega} \in \mathbb{R}^3 \) has length one, and \( \theta \in [-\pi, \pi] \). Given \( \hat{\omega} = \left( \frac{2}{3}, \frac{2}{3}, \frac{1}{3} \right)^T, v_1 = (1, 0, 1)^T, v_2 = (0, 1, 1)^T \), find all angles \( \theta \) that satisfy the above equation.

8. (a) Suppose we seek the logarithm of a rotation matrix \( R \) whose trace is \(-1\). From the exponential formula

\[
e^{[\hat{\omega}]\theta} = I + \sin \theta [\hat{\omega}] + (1 - \cos \theta) [\hat{\omega}]^2, \|\hat{\omega}\| = 1,
\]

and recalling that \( \text{tr} R = -1 \) implies \( \theta = \pi \), the above equation simplifies to

\[
R = I + 2[\hat{\omega}]^2 = \begin{bmatrix}
1 - 2(\hat{\omega}_2^2 + \hat{\omega}_3^2) & 2\hat{\omega}_1\hat{\omega}_2 & 2\hat{\omega}_1\hat{\omega}_3 \\
2\hat{\omega}_1\hat{\omega}_2 & 1 - 2(\hat{\omega}_1^2 + \hat{\omega}_3^2) & 2\hat{\omega}_2\hat{\omega}_3 \\
2\hat{\omega}_1\hat{\omega}_3 & 2\hat{\omega}_2\hat{\omega}_3 & 1 - 2(\hat{\omega}_1^2 + \hat{\omega}_2^2)
\end{bmatrix}
\]

Using the fact that \( \hat{\omega}_1^2 + \hat{\omega}_2^2 + \hat{\omega}_3^2 = 1 \), is it correct to conclude that

\[
\hat{\omega}_1 = \sqrt{\frac{r_{11} + 1}{2}}, \quad \hat{\omega}_2 = \sqrt{\frac{r_{22} + 1}{2}}, \quad \hat{\omega}_3 = \sqrt{\frac{r_{33} + 1}{2}}
\]

is also a solution?

(c) Using the fact that \([\hat{\omega}]^3 = -[\hat{\omega}]\), the identity \( R = I + 2[\hat{\omega}]^2 \) can also be written in the alternative form

\[
R - I = 2[\hat{\omega}]^2
\]

\[
[\hat{\omega}] (R - I) = 2 [\hat{\omega}]^3 = -2 [\hat{\omega}]
\]

\[
[\hat{\omega}] (R + I) = 0.
\]

The resulting equation is a system of three linear equations in \((\hat{\omega}_1, \hat{\omega}_2, \hat{\omega}_3)\). What is the relation between the solution to this linear system and the logarithm of \( R \)?

9. Exploiting all of the known properties of rotation matrices, determine the minimum number of arithmetic operations (multiplication and division, addition and subtraction) required to multiply two rotation matrices.

10. Due to finite arithmetic precision, the numerically obtained product of two rotation matrices is not necessarily a rotation matrix; that is, the resulting rotation \( A \) may not exactly satisfy \( A^T A = I \) as desired. Devise an iterative numerical procedure that takes an arbitrary matrix \( A \in \mathbb{R}^{3 \times 3} \), and produces a matrix \( R \in SO(3) \) that minimizes

\[
\|A - R\|_2^2 = \text{tr} (A - R)(A - R)^T.
\]
11. (a) If \( A = [a] \) and \( B = [b] \) for \( a, b \in \mathbb{R}^3 \), then under what conditions on \( a \) and \( b \) does \( e^A e^B = e^{A+B} \)?

(b) If \( A = [\mathcal{V}_a] \) and \( B = [\mathcal{V}_b] \), where \( \mathcal{V}_a = (\omega_a, v_a) \) and \( \mathcal{V}_b = (\omega_b, v_b) \) are arbitrary twists, then under what conditions on \( \mathcal{V}_a \) and \( \mathcal{V}_b \) does \( e^A e^B = e^{A+B} \)? Try to give a physical description of this condition.

(c) Under what conditions on general \( A, B \in \mathbb{R}^{n \times n} \) does \( e^A e^B = e^{A+B} \)?

12. (a) Given a rotation matrix \( A = \text{Rot}(\hat{z}, \alpha) \), where \( \text{Rot}(\hat{z}, \alpha) \) indicates a rotation about the \( \hat{z} \)-axis by an angle \( \alpha \), find all rotation matrices \( R \in SO(3) \) that satisfy \( AR = RA \).

(b) Given rotation matrices \( A = \text{Rot}(\hat{z}, \alpha) \) and \( B = \text{Rot}(\hat{z}, \beta) \), with \( \alpha \neq \beta \), find all rotation matrices \( R \in SO(3) \) that satisfy \( AR = RB \).

(c) Given arbitrary rotation matrices \( A, B \in SO(3) \), find all solutions \( R \in SO(3) \) to the equation \( AR = RB \).

13. (a) Show that the three eigenvalues of a rotation matrix \( R \in SO(3) \) each have unit magnitude, and conclude that they can always be written \( \{\mu + iv, \mu - iv, 1\} \), where \( \mu^2 + v^2 = 1 \).

(b) Show that a rotation matrix \( R \in SO(3) \) can always be factored in the form

\[
R = A \begin{bmatrix} \mu & \nu & 0 \\ -\nu & \mu & 0 \\ 0 & 0 & 1 \end{bmatrix} A^{-1},
\]

where \( A \in SO(3) \) and \( \mu^2 + v^2 = 1 \). (Hint: Denote the eigenvector associated with the eigenvalue \( \mu + iv \) by \( x + iy \), \( x, y \in \mathbb{R}^3 \), and the eigenvector associated with the eigenvalue 1 by \( z \in \mathbb{R}^3 \). For the purposes of this problem you may assume that the set of vectors \( \{x, y, z\} \) can always be chosen to be linearly independent.)

14. Given \( \omega \in \mathbb{R}^3 \), \( \|\omega\| = 1 \), and \( \theta \) a nonzero scalar, show that

\[
(I\theta + (1 - \cos \theta)[\omega] + (\theta - \sin \theta)[\omega]^2)^{-1} = \frac{1}{\theta} I - \frac{1}{2}[\omega] + \left( \frac{1}{\theta} - \frac{1}{2} \cot \frac{\theta}{2} \right) [\omega]^2.
\]

(Hint: From the identity \([\omega]^3 = -[\omega] \), express the inverse expressed as a quadratic matrix polynomial in \([\omega] \).)

15. (a) Given a fixed frame \( \{0\} \), and a moving frame \( \{1\} \) in the identity orientation, perform the following sequence of rotations on \( \{1\} \):

(i) Rotate \( \{1\} \) about the \( \{0\} \) frame \( x \)-axis by \( \alpha \); call this new frame \( \{2\} \).

(ii) Rotate \( \{2\} \) about the \( \{0\} \) frame \( y \)-axis by \( \beta \); call this new frame \( \{3\} \).
What is the final orientation \( R_{04} \)?

(b) Suppose that the third step in (a) is replaced by the following: “Rotate \( \{3\} \) about the frame \( \{3\} \) z-axis by \( \gamma \); call this new frame \( \{4\} \).” What is the final orientation \( R_{04} \)?

(c) From the given transformations, find \( T_{ca} \):

\[
T_{ab} = \begin{bmatrix}
\frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} & 0 & -1 \\
\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad T_{cb} = \begin{bmatrix}
\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\
0 & 1 & 0 & 1 \\
-\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

16. In terms of the \( \hat{x}_s \)-\( \hat{y}_s \)-\( \hat{z}_s \) coordinates of a fixed space frame \( \{s\} \), the frame \( \{a\} \) has an \( \hat{x}_a \)-axis pointing in the direction \((0,0,1)\) and a \( \hat{y}_a \)-axis pointing in the direction \((-1,0,0)\), and the frame \( \{b\} \) has an \( \hat{x}_b \)-axis pointing in the direction \((1,0,0)\) and a \( \hat{y}_b \)-axis pointing in the direction \((0,0,-1)\). The origin of \( \{a\} \) is at \((3,0,0)\) in \( \{s\} \) and the origin of \( \{b\} \) is at \((0,2,0)\).

(a) Give your best hand drawing showing \( \{a\} \) and \( \{b\} \) relative to \( \{s\} \).

(b) Write the rotation matrices \( R_{sa} \) and \( R_{sb} \) and the transformation matrices \( T_{sa} \) and \( T_{sb} \).

(c) Given \( T_{ab} \), how do you calculate \( T_{ab}^{-1} \) without using a matrix inverse? Write \( T_{ab}^{-1} \) and verify its correctness with your drawing.

(d) Given \( T_{sa} \) and \( T_{sb} \), how do you calculate \( T_{ab} \) (again no matrix inverses)? Compute the answer and verify its correctness with your drawing.

(e) Let \( T = T_{sb} \) be considered as a transformation operator consisting of a rotation about \( \hat{x} \) by \(-90^\circ\) and a translation along \( \hat{y} \) by 2 units. Calculate \( T_1 = T_{sa}T \). Does \( T_1 \) correspond to a rotation and translation about \( \hat{x}_a \) and \( \hat{y}_a \), respectively (world-fixed transformation of \( T_{sa} \)), or a rotation and translation about \( \hat{x}_a \) and \( \hat{y}_a \), respectively (body-fixed transformation of \( T_{sa} \))? Now calculate \( T_2 = TT_{sa} \). Does \( T_2 \) correspond to a body-fixed or world-fixed transformation of \( T_{sa} \)?

(f) Use \( T_{sb} \) to change the representation of the point \( p_b = (1,2,3)^T \) (in \( \{b\} \) coordinates) to \( \{s\} \) coordinates.

(g) Choose a point \( p \) represented by \( p_s = (1,2,3)^T \) in \( \{s\} \) coordinates. Calculate \( p' = T_{sb}p_s \) and \( p'' = T_{sb}^{-1}p_s \). For each operation, should the result be interpreted as changing coordinates (from the \( \{s\} \) frame to the \( \{b\} \)) without moving the point \( p \), or as moving the location of the point without changing the reference frame of the representation?

(g) A twist \( V \) is represented in \( \{s\} \) as \( V_s = (3,2,1,-1,-2,-3)^T \). What is its representation \( V_a \)?

(h) By hand, calculate the matrix logarithm \( [S] \theta \) of \( T_{sa} \). (You may verify your answer with software.) Extract the normalized screw axis \( S \) and rotation amount \( \theta \). Get the \( \{q,s,h\} \) representation of the screw axis. Redraw the fixed frame \( \{s\} \) and in it draw \( S \).

(i) Calculate the matrix exponential corresponding to the exponential coordi-
nates of rigid-body motion $S \theta = (0, 1, 2, 3, 0, 0)^T$. Draw the corresponding frame relative to \{s\}, as well as the screw axis $S$.

![Figure 3.21: Four reference frames defined in a robot’s workspace.](image)

17. Four reference frames are shown in the robot workspace of Figure 3.21: the fixed frame \{a\}, the end-effector frame effector \{b\}, camera frame \{c\}, and workpiece frame \{d\}.  
(a) Find $T_{ad}$ and $T_{cd}$ in terms of the dimensions given in the figure.  
(b) Find $T_{ab}$ given that

$$T_{bc} = \begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

18. Consider a robot arm mounted on a spacecraft as shown in Figure 3.22, in which frames are attached to the earth \{e\}, satellite \{s\}, the spacecraft \{a\}, and the robot arm \{r\}, respectively.  
(a) Given $T_{ea}$, $T_{ar}$, and $T_{es}$, find $T_{rs}$.  
(b) Suppose the frame \{s\} origin as seen from \{e\} is $(1, 1, 1)$, and

$$T_{er} = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$
3.8. Exercises

19. Two satellites are circling the earth as shown in Figure 3.23. Frames \( \{1\} \) and \( \{2\} \) are rigidly attached to the satellites such that their \( x \)-axes always point toward the earth. Satellite 1 moves at a constant speed \( v_1 \), while satellite 2 moves at a constant speed \( v_2 \). To simplify matters assume the earth is does not rotate about its own axis. The fixed frame \( \{0\} \) is located at the center of the earth. Figure 3.23 shows the position of the two satellites at \( t = 0 \).

(a) Derive frames \( T_{01} \), \( T_{02} \) as a function of \( t \).

(b) Using your results obtain in (a), find \( T_{21} \) as a function of \( t \).

20. Consider the high-wheel bicycle of Figure 3.24, in which the diameter of the
front wheel is twice that of the rear wheel. Frames \{a\} and \{b\} are attached to the centers of each wheel, and frame \{c\} is attached to the top of the front wheel. Assuming the bike moves forward in the \(\hat{y}\) direction, find \(T_{ac}\) as a function of the front wheel’s rotation angle \(\theta\) (assume \(\theta = 0\) at the instant shown in the figure).

21. The space station of Figure 3.25 moves in circular orbit around the earth, and at the same time rotates about an axis always pointing toward the north star. Due to an instrument malfunction, a spacecraft heading toward the space station is unable to locate the docking port. An earth-based ground station
3.8. Exercises

sends the following information to the spacecraft:

\[
T_{ab} = \begin{bmatrix}
0 & -1 & 0 & -100 \\
1 & 0 & 0 & 300 \\
0 & 0 & 1 & 500 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad p_a = \begin{bmatrix}
0 \\
800 \\
0
\end{bmatrix},
\]

where \( p_a \) is the vector \( \vec{p} \) expressed in \( \{a\} \) frame coordinates.

(a) From the given information, find \( r_b \), the vector \( \vec{r} \) expressed in \( \{b\} \) frame coordinates.

(b) Determine \( T_{bc} \) at the instant shown in the figure. Assume here that the \( \hat{y} \) and \( \hat{z} \) axes of the \( \{a\} \) and \( \{c\} \) frames are coplanar with the docking port.

\[\text{Figure 3.26: A laser tracking a moving target.}\]

22. A target moves along a circular path at constant angular velocity \( \omega \) rad/s as shown in Figure 3.26. The target is tracked by a laser mounted on a moving platform, rising vertically at constant speed \( v \). Assume the laser and the platform start at \( L_1 \) at \( t = 0 \), while the target starts at frame \( T_1 \).

(a) Derive frames \( T_{01}, T_{12}, T_{03} \) as a function of \( t \).

(b) Using your results from part (a), derive \( T_{23} \) as a function of \( t \).

23. Two toy cars are moving on a round table as shown in Figure 3.27. Car 1 moves at a constant speed \( v_1 \) along the circumference of the table, while car 2 moves at a constant speed \( v_2 \) along a radius; the positions of the two vehicles at \( t = 0 \) are shown in the figure.

(a) Find \( T_{01}, T_{02} \) as a function of \( t \).

(b) Find \( T_{12} \) as a function of \( t \).
Figure 3.27: Two toy cars on a round table.

Figure 3.28: A robot arm with a screw joint.

24. Figure 3.28 shows the configuration, at $t = 0$, of a robot arm whose first joint is a screw joint of pitch $h = 2$. The arm’s link lengths are $L_1 = 10$, $L_2 = L_3 = 5$, and $L_4 = 3$. Suppose all joint angular velocities are constant,
3.8. Exercises

with values $\omega_1 = \frac{\pi}{4}$, $\omega_2 = \frac{\pi}{8}$, $w_3 = -\frac{\pi}{4}$ rad/s. Find $T_{sb}(4) \in SE(3)$, i.e., the end-effector frame $\{b\} \in SE(3)$ relative to the fixed frame $\{s\}$, at time $t = 4$.

Figure 3.29: A camera rigidly attached to a robot arm.

25. A camera is rigidly attached to a robot arm as shown in Figure 3.29. The transformation $X \in SE(3)$ is constant. The robot arm moves from pose 1 to pose 2. The transformations $A \in SE(3)$ and $B \in SE(3)$ are measured and assumed known.

(a) Suppose $X$ and $A$ are given as follows:

$$X = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

What is $B$?

(b) Now suppose

$$A = \begin{bmatrix} R_A & p_A \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} R_B & p_B \\ 0 & 1 \end{bmatrix}$$

are known and we wish to find

$$X = \begin{bmatrix} R_X & p_X \\ 0 & 1 \end{bmatrix}.$$

Suppose $R_A = e^{[\alpha]}$ and $R_B = e^{[\beta]}$. What are the conditions on $\alpha \in \mathbb{R}^3$ and $\beta \in \mathbb{R}^3$ for a solution $R_X$ to exist?
(c) Now suppose we have a set of \( k \) equations

\[ A_i X = XB_i \text{ for } i = 1, \ldots, k \]

Assume \( A_i \) and \( B_i \) are all known. What is the minimum number of \( k \) for which a unique solution exists?

26. Draw the screw axis with \( q = (3, 0, 0)^T, \hat{s} = (0, 0, 1)^T \), and \( h = 2 \).

27. Draw the screw axis corresponding to the twist \( \mathcal{V} = (0, 2, 2, 4, 0, 0)^T \).

28. Assume that the space frame angular velocity is \( \omega_s = (1, 2, 3)^T \) for a moving body with frame \{b\} at

\[
R = \begin{bmatrix}
0 & -1 & 0 \\
0 & 0 & -1 \\
1 & 0 & 0
\end{bmatrix}
\]

in the world frame \{s\}. Calculate the angular velocity \( \omega_b \) in \{b\}.

29. Two frames \{a\} and \{b\} are attached to a moving rigid body. Show that the twist of \{a\} in space frame coordinates is the same as the twist of \{b\} in space frame coordinates.

![Diagram of a cube undergoing two different screw motions](image)

Figure 3.30: A cube undergoing two different screw motions.

30. A cube undergoes two different screw motions from frame \{1\} to frame \{2\} as shown in Figure 3.30. In both cases (a) and (b), the initial configuration of the cube is

\[
T_{01} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]
3.8. Exercises

(a) For each case (a) and (b), find the exponential coordinates $S\theta = (\omega, v)$ such that $T_{02} = e^{[S\theta]}T_{01}$, where no constraints are placed on $\omega$ or $v$.
(b) Repeat (a), this time with the constraint that $||\omega|| \in [-\pi, \pi]$.

31. In Example 3.2 and Figure 3.15, the block the robot must pick up weighs 1 kg, which means the robot must provide approximately 10 N of force in the $\hat{z}_e$ direction of the block’s frame $\{e\}$ (which you can assume is at the block’s center of mass). Express this force as a wrench in the $\{e\}$ frame, $F_e$. Given the transformation matrices in Example 3.2, express this same wrench in the end-effector frame $\{c\}$ as $F_c$.

32. Given two reference frames $\{a\}$ and $\{b\}$ in physical space, and a fixed frame $\{o\}$, define the distance between frames $\{a\}$ and $\{b\}$ as

$$\text{dist}(T_{oa}, T_{ob}) \equiv \sqrt{\theta^2 + ||p_{ab}||^2}$$

where $R_{ab} = e^{[\omega]_o}$. Suppose the fixed frame is displaced to another frame $\{o'\}$, and that $T_{o'\alpha} = ST_{oa}, T_{o'\beta} = ST_{ob}$ for some constant $S = (R_s, p_s) \in SE(3)$.
(a) Evaluate $\text{dist}(T_{o'\alpha}, T_{o'\beta})$ using the above distance formula.
(b) Under what conditions on $S$ does $\text{dist}(T_{oa}, T_{ob}) = \text{dist}(T_{o'\alpha}, T_{o'\beta})$?

33. (a) Find the general solution to the differential equation $\dot{x} = Ax$, where

$$A = \begin{bmatrix} -2 & 1 \\ 0 & -1 \end{bmatrix}.$$  

What happens to the solution $x(t)$ as $t \to \infty$?
(b) Do the same for

$$A = \begin{bmatrix} 2 & -1 \\ 1 & 2 \end{bmatrix}.$$  

What happens to the solution $x(t)$ as $t \to \infty$?

34. Let $x \in \mathbb{R}^2$, $A \in \mathbb{R}^{2 \times 2}$, and consider the linear differential equation $\dot{x}(t) = Ax(t)$. Suppose that

$$x(t) = \begin{bmatrix} e^{-3t} \\ -3e^{-3t} \end{bmatrix}$$

is a solution for the initial condition $x(0) = (1, -3)$, and

$$x(t) = \begin{bmatrix} e^t \\ e^t \end{bmatrix}.$$  

is a solution for the initial condition $x(0) = (1, 1)$. Find $A$ and $e^{At}$.

35. Given a differential equation of the form $\dot{x} = Ax + f(t)$, where $x \in \mathbb{R}^n$ and $f(t)$ is a given differentiable function of $t$. Show that the general solution can
be written
\[ x(t) = e^{At}x(0) = \int_0^t e^{A(t-s)} f(s) \, ds. \]

(Hint: Define \( z(t) = e^{-At}x(t) \), and evaluate \( \dot{z}(t) \).)

36. Referring to Appendix B, answer the following questions related to ZXZ Euler angles.
(a) Derive a procedure for finding the ZXZ Euler angles of a rotation matrix.
(b) Using the results of (a), find the ZXZ Euler angles for the following rotation matrix:
\[
\begin{bmatrix}
-\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\
-\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} \\
\frac{1}{2} & \frac{1}{2} & \frac{1}{2}
\end{bmatrix}
\]

37. Consider a wrist mechanism with two revolute joints \( \theta_1 \) and \( \theta_2 \), in which the end-effector frame orientation \( R \in SO(3) \) is given by
\[ R = e^{[\hat{\omega}_1] \theta_1} e^{[\hat{\omega}_2] \theta_2}, \]
with \( \hat{\omega}_1 = (0,0,1) \) and \( \hat{\omega}_2 = (0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}) \). Determine whether the following orientation is reachable (that is, find, if it exists, a solution \( (\theta_1, \theta_2) \) for the following \( R \)):
\[
\begin{bmatrix}
\frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\
0 & 1 & 0 \\
\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}}
\end{bmatrix}
\]

38. Show that rotation matrices of the form
\[
\begin{bmatrix}
r_{11} & r_{12} & 0 \\
r_{21} & r_{22} & r_{23} \\
r_{31} & r_{32} & r_{33}
\end{bmatrix}
\]
can be parametrized using just two parameters \( \theta \) and \( \phi \) as follows:
\[
\begin{bmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & \cos \phi \\
\sin \theta \sin \phi & \cos \theta \sin \phi & \cos \phi
\end{bmatrix}
\]
What should the range of values be for \( \theta \) and \( \phi \)?

39. Figure 3.31 shows a three degree of freedom wrist mechanism in its zero position (that is, with all its joints set to zero).
(a) Express the tool frame orientation \( R_{03} = R(\alpha, \beta, \gamma) \) as a product of three rotation matrices.
(b) Find all possible angles \( (\alpha, \beta, \gamma) \) for the two values of \( R_{03} \) given below. If no solution exists, explain why in terms of the analogy between \( SO(3) \) and the solid ball of radius \( \pi \).
3.8. Exercises

Figure 3.31: A three degree of freedom wrist mechanism.

(i) $R_{03} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$.

(ii) $R_{03} = e^{\hat{\omega} \pi}$, where $\hat{\omega} = (0, \frac{1}{\sqrt{5}}, \frac{2}{\sqrt{5}})$.

40. Refer to Appendix B.
(a) Verify the formula for obtaining the unit quaternion representation of a rotation $R \in SO(3)$.
(b) Verify the formula for obtaining the rotation matrix $R$ given a unit quaternion $q \in S^3$.
(c) Verify the product rule for two unit quaternions. That is, given two unit quaternions $q, p \in S^3$ corresponding respectively to the rotations $R, Q \in SO(3)$, find a formula for the unit quaternion representation of the product $RQ \in SO(3)$.

41. (Refer to Appendix B.) The Cayley transform of Equation (B.18) can be generalized to higher-order as follows:

$$R = (I - [r])^k(I + [r])^{-k}.$$  \hspace{1cm} (3.103)

(a) For the case $k = 2$, show that the rotation $R$ corresponding to $r$ can be
computed from the formula

\[ R = I + 4 \frac{1 - r^T r}{(1 + r^T r)^2} [r] + \frac{8}{(1 + r^T r)^2} [r]^2. \]  

(3.104)

(b) Conversely, given a rotation matrix \( R \), show that a vector \( r \) that satisfies equation (3.104) can be obtained as

\[ r = \hat{\omega} \tan \frac{\theta}{4}. \]  

(3.105)

where as before \( \hat{\omega} \) is the unit vector corresponding to the axis of rotation for \( R \), and \( \theta \) is the corresponding rotation angle. Is this solution unique?

(c) Show that the angular velocity in the body-fixed frame obeys the following relation:

\[ \dot{r} = \frac{1}{4} \left\{ (1 - r^T r) I + 2[r] + 2rr^T \right\} \omega. \]  

(3.106)

(d) Explain what happens to the singularity at \( \pi \) that exists for the standard Cayley-Rodrigues parameters. Discuss the relative advantages and disadvantages of the modified Cayley-Rodrigues parameters, particularly as one goes to order \( k = 4 \) and higher.

(e) Compare the number of arithmetic operations for multiplying two rotation matrices, two unit quaternions, and two Cayley-Rodrigues representations. Which requires fewer arithmetic operations?

42. Among the programming languages for which the software exists, choose your favorite language. If it is not represented, port the Chapter 3 software to your favorite programming language.

43. Write a function that returns “true” if a given \( 3 \times 3 \) matrix is within \( \epsilon \) of being a rotation matrix, and “false” otherwise. It is up to you how to define the “distance” between a random \( 3 \times 3 \) real matrix and the closest member of \( SO(3) \). If the function returns “true,” it should also return the “nearest” matrix in \( SO(3) \). See, for example, Exercise 10.

44. Write a function that returns “true” if a given \( 4 \times 4 \) matrix is within \( \epsilon \) of an element of \( SE(3) \), and “false” otherwise.

45. Write a function that returns “true” if a given \( 3 \times 3 \) matrix is within \( \epsilon \) of an element of \( so(3) \), and “false” otherwise.

46. Write a function that returns “true” if a given \( 4 \times 4 \) matrix is within \( \epsilon \) of an element of \( se(3) \), and “false” otherwise.

47. The primary purpose of the provided software is to be easy to read and educational, reinforcing the concepts in the book. The code is optimized neither for efficiency nor robustness, nor does it do full error-checking on its inputs.
3.8. Exercises

Familiarize yourself with all of the code in your favorite language by reading the functions and their comments. This should help cement your understanding of the material in this chapter. Then:
(a) Rewrite one function to do full error-checking on its input, and have the function return a recognizable error value if the function is called with improper input (e.g., an argument to the function is not an element of $SO(3)$, $SE(3)$, $so(3)$, or $se(3)$, as expected).
(b) Rewrite one function to improve computational efficiency, perhaps by using what you know about properties of rotation or transformation matrices.
(c) Can you reduce the numerical sensitivity of either of the matrix logarithm functions?

48. Use the provided software to write a program that allows the user to specify an initial configuration of a rigid body by $T$, a screw axis specified by $\{q, \hat{s}, h\}$, and a total distance traveled along the screw axis $\theta$. The program should calculate the final configuration $T_1 = e^{[S] \theta} T$ attained when the rigid body follows the screw $S$ a distance $\theta$, as well as the intermediate configurations at $\theta/4$, $\theta/2$, and $3\theta/4$. At the initial, intermediate, and final configurations, the program should plot the $\{b\}$ axes of the rigid body. The program should also calculate the screw axis $S_1$, and the distance $\theta_1$ following $S_1$, that takes the rigid body from $T_1$ to the origin, and plot the screw axis $S_1$. Test the program with $q = (0, 2, 0)^T$, $\hat{s} = (0, 0, 1)^T$, $h = 2$, $\theta = \pi$, and

$$ T = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$
Chapter 4

Forward Kinematics

The forward kinematics of a robot refers to the calculation of the position and orientation of its end-effector frame from its joint values. Figure 4.1 illustrates the forward kinematics problem for a 3R planar open chain. Starting from the base link, the link lengths are $L_1$, $L_2$, and $L_3$. Choose a fixed frame $\{0\}$ with origin located at the base joint as shown, and assume an end-effector frame $\{4\}$ has been attached to the tip of the third link. The Cartesian position $(x,y)$ and orientation $\phi$ of the end-effector frame as a function of the joint angles $(\theta_1, \theta_2, \theta_3)$ are then given by

\[
\begin{align*}
  x &= L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3) \quad (4.1) \\
  y &= L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3) \quad (4.2) \\
  \phi &= \theta_1 + \theta_2 + \theta_3. \quad (4.3)
\end{align*}
\]

If one is only interested in the $(x,y)$ position of the end-effector, the robot’s task space is then taken to be the $x$-$y$ plane, and the forward kinematics would consist of Equations (4.1)-(4.2) only. If the end-effector’s position and orientation both matter, the forward kinematics would consist of the three equations (4.1)-(4.3).

While the above analysis can be done using only basic trigonometry, it is not difficult to imagine that for more general spatial chains, the analysis can become considerably more complicated. A more systematic method of deriving the forward kinematics might involve attaching reference frames to each of the links; in Figure 4.1 the three link reference frames are respectively labeled $\{1\}$, $\{2\}$, and $\{3\}$. The forward kinematics can then be written as a product of four homogeneous transformation matrices,

\[
T_{04} = T_{01}T_{12}T_{23}T_{34}, \quad (4.4)
\]
Forward Kinematics

Figure 4.1: Forward kinematics of a 3R planar open chain. For each frame, the \( \hat{x} \) and \( \hat{y} \) axes are shown, and the \( \hat{z} \) axes are parallel and out of the page.

where

\[
T_{01} = \begin{bmatrix}
\cos \theta_1 & -\sin \theta_1 & 0 & 0 \\
\sin \theta_1 & \cos \theta_1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad T_{12} = \begin{bmatrix}
\cos \theta_2 & -\sin \theta_2 & 0 & L_1 \\
\sin \theta_2 & \cos \theta_2 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]

\[
T_{23} = \begin{bmatrix}
\cos \theta_3 & -\sin \theta_3 & 0 & L_2 \\
\sin \theta_3 & \cos \theta_3 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad T_{34} = \begin{bmatrix}
1 & 0 & 0 & L_3 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

Observe that \( T_{34} \) is constant, and that each remaining \( T_{i-1,i} \) depends only on the joint variable \( \theta_i \).

As an alternative to this approach, let us define \( M \) to be the position and orientation of frame \( \{4\} \) when all joint angles are set to zero (the “home” or “zero” position of the robot). Then

\[
M = \begin{bmatrix}
1 & 0 & 0 & L_1 + L_2 + L_3 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

Now consider each of the revolute joint axes to be a zero-pitch screw axis. If \( \theta_1 \) and \( \theta_2 \) are held at their zero position, then the screw axis corresponding to
rotating about joint three can be expressed in the \( \{0\} \) frame as

\[
S_3 = \begin{bmatrix}
\omega_3 \\
v_3
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
1 \\
0 \\
-(L_1 + L_2) \\
0
\end{bmatrix}.
\]

You should be able to confirm this by simple visual inspection of Figure 4.1. When the arm is stretched out straight to the right at its zero configuration, imagine a turntable rotating with an angular velocity of \( \omega_3 = 1 \text{ rad/s} \) about the axis of joint 3. The linear velocity \( v_3 \) of the point on the turntable at the origin of \( \{0\} \) is in the \(-\hat{y}_0\) direction at a rate of \( L_1 + L_2 \) units/s. Algebraically, \( v_3 = -\omega_3 \times q_3 \), where \( q_3 \) is any point on the axis of joint 3 expressed in \( \{0\} \), e.g., \( q_3 = (L_1 + L_2, 0, 0)^T \).

The screw axis \( S_3 \) can be expressed in \( \text{se}(3) \) matrix form as

\[
[S_3] = \begin{bmatrix}
[\omega] & \begin{bmatrix}v_0 \\
v_0 \\
v_0 \end{bmatrix}
\end{bmatrix} = \begin{bmatrix}
0 & -1 & 0 & 0 \\
1 & 0 & 0 & -(L_1 + L_2) \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\]

Therefore, for any \( \theta_3 \), the matrix exponential representation for screw motions from the previous chapter allows us to write

\[
T_{04} = e^{[S_3]\theta_3} M \quad (\text{for } \theta_1 = \theta_2 = 0).
\] (4.7)

Now, for \( \theta_1 = 0 \) and any fixed (but arbitrary) \( \theta_3 \), rotation about joint two can be viewed as applying a screw motion to the rigid (link two)/(link three) pair, i.e.,

\[
T_{04} = e^{[S_2]\theta_2} e^{[S_3]\theta_3} M \quad (\text{for } \theta_1 = 0),
\] (4.8)

where \([S_3]\) and \(M\) are as defined previously, and

\[
[S_2] = \begin{bmatrix}
0 & -1 & 0 & 0 \\
1 & 0 & 0 & -L_1 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\] (4.9)

Finally, keeping \( \theta_2 \) and \( \theta_3 \) fixed, rotation about joint one can be viewed as applying a screw motion to the entire rigid three-link assembly. We can therefore write, for arbitrary values of \( (\theta_1, \theta_2, \theta_3) \),

\[
T_{04} = e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} M,
\] (4.10)

where

\[
[S_1] = \begin{bmatrix}
0 & -1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}.
\] (4.11)
Thus the forward kinematics can be expressed as a product of matrix exponentials, each corresponding to a screw motion. Note that this latter derivation of the forward kinematics does not make use of any link reference frames; only \( \{0\} \) and \( M \) must be defined.

In this chapter we consider the forward kinematics of general open chains. One widely used representation for the forward kinematics of open chains relies on the Denavit-Hartenberg parameters (D-H parameters), and this representation makes use of Equation (4.4). Another representation relies on the **Product of Exponentials** (PoE) formula, which corresponds to Equation (4.10). The advantage of the D-H representation is that it requires the minimum number of parameters to describe the robot’s kinematics; for an \( n \)-joint robot, it uses \( 3n \) numbers to describe the robot’s structure and \( n \) numbers to describe the joint values. The PoE representation is not minimal (it requires \( 6n \) numbers to describe the \( n \) screw axes, in addition to the \( n \) joint values), but it has advantages over the D-H representation (e.g., no link frames are necessary) and it is our preferred choice of forward kinematics representation. The D-H representation, and its relationship to the PoE representation, is described in Appendix C.

### 4.1 Product of Exponentials Formula

To use the PoE formula, it is only necessary to assign a stationary frame \( \{s\} \) (e.g., at the fixed base of the robot, or really anywhere that is convenient to define a reference frame) and a frame \( \{b\} \) at the end-effector, described by \( M \) when the robot is at its zero position. However, it is not uncommon to define a frame at each link, typically at the joint axis; these are needed for the D-H representation and they are useful for displaying a CAD model of the robot and for defining the mass properties of the link, which we will need starting in Chapter 8. Thus when we are defining the kinematics of an \( n \)-joint robot, we may either (1) minimally use the frames \( \{s\} \) and \( \{b\} \) if we are only interested in kinematics, or (2) refer to \( \{s\} \) as frame \( \{0\} \) when the robot is at its zero position. However, it is not uncommon to define a frame at each link, typically at the joint axis; these are needed for the D-H representation and they are useful for displaying a CAD model of the robot and for defining the mass properties of the link, which we will need starting in Chapter 8. Thus when we are defining the kinematics of an \( n \)-joint robot, we may either (1) minimally use the frames \( \{s\} \) and \( \{b\} \) if we are only interested in kinematics, or (2) refer to \( \{s\} \) as frame \( \{0\} \) when the robot is at its zero position.

#### 4.1.1 First Formulation: Screw Axes Expressed in Base Frame

The key concept behind the PoE formula is to regard each joint as applying a screw motion to all the outward links. To illustrate, consider a general spatial open chain like the one shown in Figure 4.2, consisting of \( n \) one-dof joints that are connected serially. To apply the PoE formula, you must choose a fixed base frame and an end-effector frame attached to the last link. Place the robot in its zero position by setting all joint values to zero, with the direction of positive
4.1. Product of Exponentials Formula

$$e^{[S_n] \theta_n} M$$

(4.12)

where $T \in SE(3)$ is the new configuration of the end-effector frame, and $S_n = (\omega_n, v_n)$ is the screw axis of joint $n$, as expressed in the fixed base frame. If joint $n$ is revolute (corresponding to a screw motion of zero pitch), then $\omega_n \in \mathbb{R}^3$ is a unit vector in the positive direction of joint axis $n$; $v_n = -\omega_n \times q_n$, with $q_n$ any arbitrary point on joint axis $n$ as written in coordinates in the fixed base frame; and $\theta_n$ is the joint angle. If joint $n$ is prismatic, then $\omega_n = 0$; $v_n \in \mathbb{R}^3$ is a unit vector in the direction of positive translation; and $\theta_n$ represents the prismatic extension/retraction.

If we assume joint $n - 1$ is also allowed to vary, then this has the effect of applying a screw motion to link $n - 1$ (and by extension to link $n$, since link $n$ is connected to link $n - 1$ via joint $n$). The end-effector frame thus undergoes a displacement of the form

$$T = e^{[S_{n-1}] \theta_{n-1}} \left( e^{[S_n] \theta_n} M \right)$$

(4.13)

Continuing with this reasoning and now allowing all the joints $(\theta_1, \ldots, \theta_n)$ to vary, it follows that

$$T(\theta) = e^{[S_1] \theta_1} \cdots e^{[S_{n-1}] \theta_{n-1}} e^{[S_n] \theta_n} M.$$  

(4.14)
This is the product of exponentials formula describing the forward kinematics of an n-dof open chain. Specifically, we call Equation (4.14) the **space form** of the product of exponentials formula, referring to the fact that the screw axes are expressed in the fixed space frame.

To summarize, to calculate the forward kinematics of an open chain using the space form of the PoE formula (4.14), we need the following elements:

(i) The end-effector configuration $M \in SE(3)$ when the robot is at its home position.

(ii) The screw axes $S_1 \ldots S_n$, expressed in the fixed base frame, corresponding to the joint motions when the robot is at its home position.

(iii) The joint variables $\theta_1 \ldots \theta_n$.

Unlike with the D-H representation, no link reference frames need to be defined. Further advantages will come to light when we examine the velocity kinematics in the next chapter.

### 4.1.2 Examples

We now derive the forward kinematics for some common spatial open chains using the PoE formula.

**Example: 3R Spatial Open Chain**

Consider the 3R open chain of Figure 4.3, shown in its home position (all joint variables set equal to zero). Choose the fixed frame $\{0\}$ and end-effector frame...
4.1. Product of Exponentials Formula

The screw axis $S_1 = (\omega_1, v_1)$ for joint axis 1 is then given by $\omega_1 = (0, 0, 1)$ and $v_1 = (0, 0, 0)$ (the fixed frame origin (0,0,0) is a convenient choice for the point $q_1$ lying on joint axis 1). To determine the screw axis $S_2$ for joint axis 2, observe that joint axis 2 points in the $-\hat{y}_0$ axis direction, so that $\omega_2 = (0, -1, 0)$. Choose $q_2 = (L_1, 0, 0)$, in which case $v_2 = -\omega_2 \times q_2 = (0, 0, -L_1)$. Finally, to determine the screw axis $S_3$ for joint axis 3, note that $\omega_3 = (1, 0, 0)$. Choosing $q_3 = (0, 0, -L_2)$, it follows that $v_3 = -\omega_3 \times q_3 = (0, -L_2, 0)$.

In summary, we have the following 4×4 matrix representations for the three joint screw axes $S_1$, $S_2$, and $S_3$:

$$
\begin{bmatrix}
S_1 \\
S_2 \\
S_3
\end{bmatrix} =
\begin{bmatrix}
0 & -1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 1 & L_1 \\
0 & 1 & 0 & 0 \\
-1 & 0 & 0 & -L_2 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & -1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & -L_1 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & -1 & -L_2 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
$$

It will be more convenient to list the screw axes in the following tabular form:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\omega_i$</th>
<th>$v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>2</td>
<td>(0, -1, 0)</td>
<td>(0, 0, -L_1)</td>
</tr>
<tr>
<td>3</td>
<td>(1, 0, 0)</td>
<td>(0, L_2, 0)</td>
</tr>
</tbody>
</table>

Example: 6R Spatial Open Chain

We now derive the forward kinematics of the 6R open chain of Figure 4.4. Six-dof arms play an important role in robotics because they have the minimum
number of joints allowing the end-effector to move a rigid body in all of its
degrees of freedom, subject only to limits on the robot’s workspace. For this
reason, six-dof robot arms are sometimes called “general purpose manipulators.”

The zero position and the direction of positive rotation for each joint axis
are as shown in the figure. A fixed frame \( \{0\} \) and end-effector frame \( \{6\} \) are
also assigned as shown. The end-effector frame \( M \) in the zero position is then

\[
M = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 3L \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\] (4.15)

The screw axis for joint 1 is in the direction \( \omega_1 = (0, 0, 1) \). The most convenient
choice for point \( q_1 \) lying on joint axis 1 is the origin, so that \( v_1 = (0, 0, 0) \). The
screw axis for joint 2 is in the \( \hat{y} \) direction of the fixed frame, so \( \omega_2 = (0, 1, 0) \).
Choosing \( q_2 = (0, 0, 0) \), we have \( v_2 = (0, 0, 0) \). The screw axis for joint 3 is
in the direction \( \omega_3 = (-1, 0, 0) \). Choosing \( q_3 = (0, 0, 0) \) leads to \( v_3 = (0, 0, 0) \).
The screw axis for joint 4 is in the direction \( \omega_4 = (-1, 0, 0) \). Choosing \( q_4 = 
(0, L, 0) \) leads to \( v_4 = (0, 0, L) \). The screw axis for joint 5 is in the direction
\( \omega_5 = (-1, 0, 0) \); choosing \( q_5 = (0, 2L, 0) \) leads to \( v_5 = (0, 0, 2L) \). The screw
axis for joint 6 is in the direction \( \omega_6 = (0, 1, 0) \); choosing \( q_6 = (0, 0, 0) \) leads
to \( v_6 = (0, 0, 0) \). In summary, the screw axes \( S_i = (\omega_i, v_i) \), \( i = 1, \ldots, 6 \) are as follows:

<table>
<thead>
<tr>
<th>( i )</th>
<th>( \omega_i )</th>
<th>( v_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>2</td>
<td>(0, 1, 0)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>3</td>
<td>(-1, 0, 0)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>4</td>
<td>(-1, 0, 0)</td>
<td>(0, 0, L)</td>
</tr>
<tr>
<td>5</td>
<td>(-1, 0, 0)</td>
<td>(0, 0, 2L)</td>
</tr>
<tr>
<td>6</td>
<td>(0, 1, 0)</td>
<td>(0, 0, 0)</td>
</tr>
</tbody>
</table>
Example: An RRPRRR Spatial Open Chain

In this example we consider the six degree-of-freedom RRPRRR spatial open chain of Figure 4.5. The end-effector frame in the zero position is given by

\[ M = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & L_1 + L_2 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}. \]

The screw axes \( S_i = (\omega_i, v_i) \) are listed in the following table:

<table>
<thead>
<tr>
<th>( i )</th>
<th>( \omega_i )</th>
<th>( v_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>2</td>
<td>(1, 0, 0)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>3</td>
<td>(0, 0, 0)</td>
<td>(0, 1, 0)</td>
</tr>
<tr>
<td>4</td>
<td>(0, 1, 0)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>5</td>
<td>(1, 0, 0)</td>
<td>(0, 0, -L_1)</td>
</tr>
<tr>
<td>6</td>
<td>(0, 1, 0)</td>
<td>(0, 0, 0)</td>
</tr>
</tbody>
</table>

Note that the third joint is prismatic, so that \( \omega_3 = 0 \) and \( v_3 \) is a unit vector in the direction of positive translation.

Example: Universal Robots’ UR5 6R Robot Arm

The Universal Robots UR5 6R robot arm is shown in Figure 4.6. Each joint is directly driven by a brushless motor combined with 100:1 zero-backlash harmonic drive gearing, which greatly increases the torque available at the joint while reducing its maximum speed.

Figure 4.6 shows the screw axes \( S_1 \ldots S_6 \) when the robot is at its zero posi-
Forward Kinematics

\[ L_1 = 425 \text{ mm} \]
\[ W_1 = 109 \text{ mm} \]
\[ W_2 = 82 \text{ mm} \]
\[ L_2 = 392 \text{ mm} \]
\[ H_2 = 95 \text{ mm} \]
\[ y_b \]
\[ x_b \]
\[ x_s \]
\[ y_s \]
Positive rotation about the axes is by the right-hand rule.

Figure 4.6: Universal Robots’ UR5 6R robot arm (at its zero position, at right).

The end-effector frame \( \{b\} \) in the zero position is given by

\[
M = \begin{bmatrix}
-1 & 0 & 0 & L_1 + L_2 \\
0 & 0 & 1 & W_1 + W_2 \\
0 & 1 & 0 & H_1 - H_2 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

The screw axes \( S_i = (\omega_i, v_i) \) are listed in the following table:

<table>
<thead>
<tr>
<th>( i )</th>
<th>( \omega_i )</th>
<th>( v_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>2</td>
<td>(0, 1, 0)</td>
<td>((-H_1, 0, 0))</td>
</tr>
<tr>
<td>3</td>
<td>(0, 1, 0)</td>
<td>((-H_1, 0, L_1))</td>
</tr>
<tr>
<td>4</td>
<td>(0, 1, 0)</td>
<td>((-H_1, 0, L_1 + L_2))</td>
</tr>
<tr>
<td>5</td>
<td>(0, 0, -1)</td>
<td>((-W_1, L_1 + L_2, 0))</td>
</tr>
<tr>
<td>6</td>
<td>(0, 1, 0)</td>
<td>((H_2 - H_1, 0, L_1 + L_2))</td>
</tr>
</tbody>
</table>

As an example of forward kinematics, set \( \theta_2 = -\pi/2 \) and \( \theta_5 = \pi/2 \), with all other joint angles equal to zero. Then the configuration of the end-effector is

\[
T(\theta) = e^{[S_1] \theta_1} e^{[S_2] \theta_2} e^{[S_3] \theta_3} e^{[S_4] \theta_4} e^{[S_5] \theta_5} e^{[S_6] \theta_6} M
\]

\[
= I e^{-[S_2] \pi/2} e^{[S_3] \pi/2} e^{[S_5] \pi/2} M
\]

\[
= e^{-[S_2] \pi/2} e^{[S_3] \pi/2} M
\]

since \( e^0 = I \). Evaluating, we get

\[
e^{-[S_2] \pi/2} = \begin{bmatrix}
0 & 0 & -1 & 0.089 \\
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0.089 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad e^{[S_3] \pi/2} = \begin{bmatrix}
0 & 1 & 0 & 0.708 \\
-1 & 0 & 0 & 0.926 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]
4.1. Product of Exponentials Formula

where the linear units are meters, and

\[
T(\theta) = e^{-[S_2]_{\pi/2}} e^{-[S_5]_{\pi/2}} M = \begin{bmatrix}
0 & -1 & 0 & 0.095 \\
1 & 0 & 0 & 0.109 \\
0 & 0 & 1 & 0.988 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

as shown in Figure 4.7.

4.1.3 Second Formulation: Screw Axes Expressed in End-Effector Frame

The matrix identity \( e^{M^{-1}P} = M^{-1} e^P M \) (Proposition 3.7) can also be expressed as \( Me^{M^{-1}P} = e^P M \). Beginning with the rightmost term of the previously derived product of exponentials formula, if we repeatedly apply this
identity, after $n$ iterations we obtain

$$T(\theta) = e^{[S_1] \theta_1} \cdots e^{[S_n] \theta_n} M$$

$$= e^{[S_1] \theta_1} \cdots e^{M^{-1} [S_n] M \theta_n}$$

$$= e^{[S_1] \theta_1} \cdots e^{M^{-1} [S_{n-1}] M \theta_{n-1} e^{M^{-1} [S_n] M \theta_n}}$$

$$= Me^{M^{-1} [S_1] M \theta_1} \cdots e^{M^{-1} [S_{n-1}] M \theta_{n-1} e^{M^{-1} [S_n] M \theta_n}}$$

$$= Me^{[B_1] \theta_1} \cdots e^{[B_{n-1}] \theta_{n-1} e^{[B_n] \theta_n}}, \quad (4.16)$$

where each $[B_i] = M^{-1} [S_i] M = [\text{Ad}_{M^{-1}}] S_i$, $i = 1, \ldots, n$. Equation (4.16) is an alternative form of the product of exponentials formula, representing the joint axes as screw axes $B_i$ in the end-effector (body) frame when the robot is at its zero position. We call Equation (4.16) the body form of the product of exponentials formula.

It is worth thinking about the order of the transformations expressed in the space form PoE formula (Equation (4.14)) and in the body form formula (Equation (4.16)). In the space form, $M$ is first transformed by the most distal joint, progressively moving inward to more proximal joints. Note that the fixed space-frame representation of the screw axis for a more proximal joint is not affected by the joint displacement at a distal joint (e.g., joint three’s displacement does not affect joint two’s screw axis representation in the space frame). In the body form, $M$ is first transformed by the first joint, progressively moving outward to more distal joints. The body-frame representation of the screw axis for a more distal joint is not affected by the joint displacement at a proximal joint (e.g., joint two’s displacement does not affect joint three’s screw axis representation in the body frame.) Therefore, it makes sense that we need only determine the screw axes at the robot’s zero position: any $S_i$ is unaffected by more distal transformations, and any $B_i$ is unaffected by more proximal transformations.

**Example: 6R Spatial Open Chain**

We now express the forward kinematics for the same 6R open chain of Figure 4.4 in the second form

$$T(\theta) = Me^{[B_1] \theta_1} e^{[B_2] \theta_2} \cdots e^{[B_6] \theta_6}.$$

Assume the same fixed and end-effector frames and zero position as the previous example. $M$ is still the same as in Equation (4.15), obtained as the end-effector frame as seen from the fixed frame with the chain in its zero position. The screw axis for each joint axis, however, is now expressed with respect to the end-effector frame in its zero position:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\omega_i$</th>
<th>$v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0, 0, 1)</td>
<td>(-3L, 0, 0)</td>
</tr>
<tr>
<td>2</td>
<td>(0, 1, 0)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>3</td>
<td>(-1, 0, 0)</td>
<td>(0, 0, -3L)</td>
</tr>
<tr>
<td>4</td>
<td>(-1, 0, 0)</td>
<td>(0, 0, -2L)</td>
</tr>
<tr>
<td>5</td>
<td>(-1, 0, 0)</td>
<td>(0, 0, -L)</td>
</tr>
<tr>
<td>6</td>
<td>(0, 1, 0)</td>
<td>(0, 0, 0)</td>
</tr>
</tbody>
</table>
4.1. Product of Exponentials Formula

\[ W_1 = 45 \text{ mm} \]
\[ L_3 = 60 \text{ mm} \]
\[ L_2 = 300 \text{ mm} \]
\[ L_1 = 550 \text{ mm} \]

Axes 1, 2, and 3 intersect at the origin of \( \{s\} \). Axes 5, 6, and 7 intersect at a common point 60 mm from the origin of \( \{b\} \). Axes 1, 3, 5, and 7 are aligned with \( \hat{z}_s \), and axes 2, 4, and 6 are out of the page at the zero configuration. Positive rotation about the axes is by the right-hand rule.

\[ \hat{y}_s \] and \( \hat{y}_b \) axes are aligned and out of the page.

Figure 4.8: Barrett Technology’s WAM 7R robot arm at its zero configuration (right).

**Example: Barrett Technology’s WAM 7R Robot Arm**

Barrett Technology’s WAM 7R robot arm is shown in Figure 4.8. The extra (seventh) joint means that the robot is redundant for the task of positioning its end-effector frame in \( SE(3) \)—in general, for a given end-effector configuration in the robot’s workspace, there is a one-dimensional set of joint variables in the robot’s seven-dimensional joint space that achieves that configuration. This extra degree of freedom can be used for obstacle avoidance or to optimize some objective function, such as minimizing the motor power needed to hold the robot at that configuration.

Also, some of the joints of the WAM are driven by motors placed at the base of the robot, reducing the robot’s moving mass. Torques are transferred
from the motors to the joints by cables winding around drums at the joints and motors. Because the moving mass is reduced, the motor torque requirements are decreased, allowing lower (cable) gearing ratios and high speeds. This design is in contrast to the design of the UR5, where the motor and harmonic drive gearing for each joint are directly at the joint.

Figure 4.8 illustrates the WAM’s end-effector frame screw axes $B_1 \ldots B_7$ when the robot is at its zero position. The end-effector frame $\{b\}$ in the zero position is given by

$$M = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & L_1 + L_2 + L_3 \\
0 & 0 & 0 & 1
\end{bmatrix}.$$

The screw axes $B_i = (\omega_i, v_i)$ are listed in the following table:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\omega_i$</th>
<th>$v_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>2</td>
<td>(0, 1, 0)</td>
<td>($L_1 + L_2 + L_3$, 0, 0)</td>
</tr>
<tr>
<td>3</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>4</td>
<td>(0, 1, 0)</td>
<td>($L_2 + L_3$, 0, 0)</td>
</tr>
<tr>
<td>5</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
<tr>
<td>6</td>
<td>(0, 1, 0)</td>
<td>($L_3$, 0, 0)</td>
</tr>
<tr>
<td>7</td>
<td>(0, 0, 1)</td>
<td>(0, 0, 0)</td>
</tr>
</tbody>
</table>

Figure 4.9 shows the WAM arm with $\theta_2 = 45^\circ$, $\theta_4 = -45^\circ$, $\theta_6 = -90^\circ$, and all other joint angles equal to zero, giving

$$T(\theta) = Me^{[B_2]\pi/4}e^{-[B_4]\pi/4}e^{-[B_6]\pi/2} = \begin{bmatrix}
0 & 0 & -1 & 0.3157 \\
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0.6571 \\
0 & 0 & 0 & 1
\end{bmatrix}.$$

### 4.2 The Universal Robot Description Format

The **Universal Robot Description Format** (URDF) is an XML (eXtensible Markup Language) file format used by the Robot Operating System (ROS) to describe the kinematics, inertial properties, and link geometry of robots. A URDF file describes the joints and links of a robot:

- **Joints.** Joints connect two links: a parent link and a child link. A few of the possible joint types include prismatic, revolute (including joint limits), continuous (revolute without joint limits), and fixed (a virtual joint that does not permit any motion). Each joint has an origin frame that defines the position and orientation of the child link frame relative to the parent link frame when the joint variable is zero. The origin is
on the joint’s axis. Each joint has an \textit{axis} \textit{x-y-z} unit vector, expressed in the \textit{child} link’s frame, in the direction of positive rotation for a revolute joint or positive translation for a prismatic joint.

- \textbf{Links}. While joints fully describe the kinematics of a robot, links define the mass properties of the links. These are necessary starting in Chapter 8, when we begin to study the dynamics of robots. The elements of a link include its \textit{mass}; an \textit{origin} frame that defines the position and orientation of a frame at the link’s center of mass relative to the link’s joint frame described above; and an \textit{inertia} matrix, relative to the link’s center of mass frame, specified by the six elements of the inertia matrix on or above the diagonal. (As we will see in Chapter 8, the inertia matrix for a rigid body is a \(3 \times 3\) symmetric positive-definite matrix. Since the inertia matrix is symmetric, it is only necessary to define the terms on and above the diagonal.)

Note that most links have two frames rigidly attached: a first frame at the joint (defined by the joint element that connects the link to its parent) and a second frame at the link’s center of mass (defined by the link element, relative to the first frame).

A URDF file can represent any robot with a tree structure. This includes
serial chain robot arms and robot hands, but not a Stewart platform or other mechanisms with closed loops. An example robot with a tree structure is shown in Figure 4.10.

The orientation of a frame \( \{b\} \) relative to a frame \( \{a\} \) is represented using roll-pitch-yaw coordinates: first, roll about the fixed \( \hat{x}_a \)-axis; then pitch about the fixed \( \hat{y}_a \)-axis; then yaw about the fixed \( \hat{z}_a \)-axis.

The kinematics and mass properties of the UR5 robot arm (Figure 4.11) are defined in the URDF file below, which demonstrates the syntax of the joint’s elements parent, child, origin, and axis, and the link’s elements mass, origin, and inertia. A URDF requires a frame defined at every joint, so we define frames \( \{1\} \) to \( \{6\} \), in addition to the fixed base frame \( \{0\} \) (i.e., \( \{s\} \)) and the end-effector frame \( \{7\} \) (i.e., \( \{b\} \)). Figure 4.11 gives the extra information needed to fully write the URDF.

Although the joint types in the URDF are defined as “continuous,” the UR5 joints do in fact have joint limits; they are omitted here for simplicity.

The UR5 URDF file (kinematics and inertial properties only).

```xml
<?xml version="1.0" ?>
```
4.2. The Universal Robot Description Format

Figure 4.11: The orientations of the frames \{s\} (also written \{0\}), \{b\} (also written \{7\}), and \{1\} through \{6\} are illustrated on the translucent UR5. The frames \{s\} and \{1\} are aligned with each other; frames \{2\} and \{3\} are aligned with each other; and frames \{4\}, \{5\}, and \{6\} are aligned with each other. Therefore, only the axes of frames \{s\}, \{2\}, \{4\}, and \{b\} are labeled. Just below the image of the robot is a skeleton indicating how the frames are offset from each other, including the distance and the direction (expressed in the \{s\} frame).

```xml
<robot name="ur5">

<!-- ********** KINEMATIC PROPERTIES (JOINTS) ********** -->

<!-- joint name="world_joint" type="fixed" -->
  <parent link="world"/>
  <child link="base_link"/>
  <origin rpy="0.0 0.0 0.0" xyz="0.0 0.0 0.0"/>
</joint>

<!-- joint name="joint1" type="continuous" -->
  <parent link="base_link"/>
  <child link="link1"/>
  <origin rpy="0.0 0.0 0.0" xyz="0.0 0.0 0.089159"/>
  <axis xyz="0 0 1"/>
</joint>
```


4.3. Summary

Beyond the properties described above, a URDF can describe other properties of a robot, such as its visual appearance (including CAD models of the links) as well as simplified representations of link geometries that can be used for collision detection in motion planning algorithms.

4.3 Summary

- Given an open chain with a fixed reference frame \(\{s\}\) and a reference frame \(\{b\}\) attached to some point on its last link—this frame is denoted the **end-effector frame**—the forward kinematics is the mapping \(T(\theta)\) from the joint values \(\theta\) to the position and orientation of \(\{b\}\) in \(\{s\}\).
Forward Kinematics

- In the Denavit-Hartenberg representation, the forward kinematics of an open chain is described in terms of relative displacements between reference frames attached to each link. If the link frames are sequentially labeled \{0\}, ..., \{n + 1\}, where \{0\} is the fixed frame \{s\}, \{i\} is a frame attached to link \(i\) at joint \(i\) \((i = 1 \ldots n)\), and \{n + 1\} is the end-effector frame \{b\}, then the forward kinematics is expressed as

\[
T_{0,n+1}(\theta) = T_{01}(\theta_1) \cdots T_{n-1,n}(\theta_n)T_{n,n+1}
\]

where \(\theta_i\) denotes the joint \(i\) variable and \(T_{n,n+1}\) indicates the (fixed) configuration of the end-effector frame in \{n\}. If the end-effector frame \{b\} is chosen to be coincident with \{n\}, then we can dispense with the frame \{n + 1\}.

- The Denavit-Hartenberg convention requires that reference frames assigned to each link obey a strict convention (see Appendix C). Following this convention, the link frame transformation \(T_{i-1,i}\) between link frames \{i − 1\} and \{i\} can be parametrized using only four parameters, the Denavit-Hartenberg parameters. Three of these describe the kinematic structure, while the fourth is the joint value. Four numbers is the minimum needed to represent the displacement between two link frames.

- The forward kinematics can also be expressed as the following product of exponentials (the space form),

\[
T(\theta) = e^{[S_1] \theta_1} \cdots e^{[S_n] \theta_n} M,
\]

where \(S_i = (\omega_i, v_i)\) denotes the screw axis associated with positive motion along joint \(i\) expressed in fixed frame coordinates \{s\}, \(\theta_i\) is the joint \(i\) variable, and \(M \in SE(3)\) denotes the position and orientation of the end-effector frame \{b\} when the robot is in its zero position. It is not necessary to define individual link frames; it is only necessary to define \(M\) and the screw axes \(S_1, \ldots, S_n\).

- The product of exponentials formula can also be written in the equivalent body form,

\[
T(\theta) = Me^{[B_1] \theta_1} \cdots e^{[B_n] \theta_n},
\]

where \([B_i] = [\text{Ad}_M^{-1}]S_i\), \(i = 1, \ldots, n\). \(B_i = (\omega_i, v_i)\) is the screw axis corresponding to joint axis \(i\), expressed in \{b\}, with the robot in its zero position.

- The Universal Robot Description Format (URDF) is a file format, used by the Robot Operating System and other software, for representing the kinematics, inertial properties, visual properties, and other information for general tree-like robot mechanisms, including serial chains. A URDF file includes descriptions of joints, which connect a parent link and a child link and fully specify the kinematics of the robot, as well as descriptions of links, which specify inertial properties.
4.4 Software

\[ T = \text{FKinBody}(M,Blist,\text{thetalist}) \]
Computes the end-effector frame given the zero position of the end-effector \( M \),
the list of joint screws expressed in the end-effector frame \( Blist \), and the list of
joint values \( \text{thetalist} \).

\[ T = \text{FKinSpace}(M,Slist,\text{thetalist}) \]
Computes the end-effector frame given the zero position of the end-effector \( M \),
the list of joint screws expressed in the fixed space frame \( Slist \), and the list of
joint values \( \text{thetalist} \).

Notes and References
The product of exponentials formula is first presented by Brockett in [14]. Compu-
tational aspects of the product of exponentials formula are also discussed in
[95].
4.5 Exercises

1. Familiarize yourself with the functions FKinBody and FKinFixed in your favorite programming language. Can you make these functions more computationally efficient? If so, indicate how. If not, explain why not.

2. The RRRP SCARA robot of Figure 4.12 is shown in its zero position. Determine the end-effector zero position configuration \( M \), the screw axes \( S_i \) in \( \{0\} \), and the screw axes \( B_i \) in \( \{b\} \). For \( l_0 = l_1 = l_2 = 1 \) and the joint variable values \( \theta = (0, \pi/2, -\pi/2, 1) \), use both the FKinFixed and the FKinBody functions to find the end-effector configuration \( T \in SE(3) \). Confirm that they agree with each other.

3. Determine the end-effector frame screw axes \( B_i \) for the 3R robot in Figure 4.3.

4. Determine the end-effector frame screw axes \( B_i \) for the RRPRRR robot in Figure 4.5.

5. Determine the end-effector frame screw axes \( B_i \) for the UR5 robot in Figure 4.6.

6. Determine the space frame screw axes \( S_i \) for the WAM robot in Figure 4.8.

Figure 4.12: An RRRP SCARA robot for performing pick-and-place operations.
7. The PRRRRR spatial open chain of Figure 4.13 is shown in its zero position. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

8. The spatial RRRRPR open chain of Figure 4.14 is shown in its zero position, with fixed and end-effector frames chosen as shown. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

9. The spatial RRPPRR open chain of Figure 4.15 is shown in its zero position. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

10. The URRPR spatial open chain of Figure 4.16 is shown in its zero position.
Figure 4.15: A spatial RRPPRR open chain with prescribed fixed and end-effector frames.

Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

11. The spatial RPRRR open chain of Figure 4.17 is shown in its zero position. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

12. The RRPRRR spatial open chain of Figure 4.18 is shown in its zero position (all joints lie on the same plane). Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$. Setting $\theta_5 = \pi$ and all other joint variables to zero, find $T_{06}$ and $T_{60}$.

13. The spatial RRRPRR open chain of Figure 4.19 is shown in its zero position. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

14. The RPH robot of Figure 4.20 is shown in its zero position. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{s\}$, and the screw axes $B_i$ in $\{b\}$. Use both the FKinFixed and the FKinBody functions to find the end-effector configuration $T \in SE(3)$ when $\theta = (\pi/2, 3, \pi)$. Confirm that they agree with each other.

15. The HRR robot in Figure 4.21 is shown in its zero position. Determine the end-effector zero position configuration $M$, the screw axes $S_i$ in $\{0\}$, and the screw axes $B_i$ in $\{b\}$.

16. The forward kinematics of a four-dof open chain in its zero position is
written in the following exponential form:

\[ T(\theta) = e^{[A_1]\theta_1}e^{[A_2]\theta_2}M_e^{[A_3]\theta_3}e^{[A_4]\theta_4}. \]

Suppose the manipulator’s zero position is redefined to

\[ (\theta_1, \theta_2, \theta_3, \theta_4) = (\alpha_1, \alpha_2, \alpha_3, \alpha_4). \]

Defining \(\theta'_i = \theta_i - \alpha_i, i = 1, ..., 4\), the forward kinematics can then be written

\[ T_{04}(\theta'_1, \theta'_2, \theta'_3, \theta'_4) = e^{[A'_1]\theta'_1}e^{[A'_2]\theta'_2}M'_e^{[A'_3]\theta'_3}e^{[A'_4]\theta'_4}. \]

Find \(M'_e\) and each of the \(A'_i\).

**17.** Figure 4.22 shows a snake robot with end-effectors at each end. Reference frames \(\{b_1\}\) and \(\{b_2\}\) are attached to the two end-effectors as shown.

(a) Suppose the end-effector 1 is grasping a tree (which can be thought of as ground), and end effector 2 is free to move. Assume the robot is in its zero
configuration. Then $T_{b_1 b_2} \in SE(3)$ can be expressed in the following product of exponentials form:

$$T_{b_1 b_2} = e^{[S_1] \theta_1} e^{[S_2] \theta_2} \cdots e^{[S_5] \theta_5} M.$$ 

Find $S_3, S_5,$ and $M.$

(b) Now suppose end-effector 2 is rigidly grasping a tree, and end-effector 1 is free to move. Then $T_{b_2 b_1} \in SE(3)$ can be expressed in the following product of exponentials form:

$$T_{b_2 b_1} = e^{[A_2] \theta_6} e^{[A_4] \theta_4} e^{[A_5] \theta_3} N e^{[A_2] \theta_2} e^{[A_1] \theta_1}.$$ 

Find $A_2, A_4,$ and $N.$
4.5. Exercises 143

Figure 4.19: A spatial RRRPRR open chain with prescribed fixed and end-effector frames.

Figure 4.20: An RPH open chain shown at its zero position. All arrows along/about the joint axes are drawn in the positive direction (i.e., in the direction of increasing joint value). The pitch of the screw joint is 0.1 m/rad, i.e., it advances linearly by 0.1 m for every radian rotated. The link lengths are $L_0 = 4$, $L_1 = 3$, $L_2 = 2$, and $L_3 = 1$ (figure not drawn to scale).

18. The two identical PUPR open chains of Figure 4.23 are shown in their zero position.
(a) In terms of the given fixed frame $\{A\}$ and end-effector frame $\{a\}$, the forward kinematics for the robot on the left (robot A) can be expressed in the following product of exponentials form:

$$T_{Aa} = e^{[S_1]^{\theta_1}} e^{[S_2]^{\theta_2}} \cdots e^{[S_5]^{\theta_5}} M_a.$$  

Find $S_2$ and $S_4$.
(b) Suppose the end-effector of robot A is inserted into the end-effector of robot B (so that the origins of the end-effectors coincide); the two robots then form a single-loop closed chain. Then the configuration space of the single-loop closed
Figure 4.21: HRR robot. Denote the pitch of the screw joint by $h$.

Figure 4.22: Snake robot
4.5. Exercises

A chain can be expressed in the form
\[ e^{-[S_5] \phi_5} e^{-[S_4] \phi_4} e^{-[S_3] \phi_3} e^{-[S_2] \phi_2} e^{-[S_1] \phi_1} e[S_1] \theta_1 e[S_2] \theta_2 e[S_3] \theta_3 e[S_4] \theta_4 e[S_5] \theta_5 = M, \]
for some constant \( M \in SE(3) \) and \( B_i = (\omega_i, v_i), i = 1, \ldots, 5 \). Find \( B_3, B_5 \) and \( M \). (Hint: given any \( A \in \mathbb{R}^{n \times n}, (e^A)^{-1} = e^{-A} \).)

Figure 4.23: Two PUPR open chains.

19. The RRPRR spatial open chain of Figure 4.24 is shown in its zero position.
(a) The forward kinematics can be expressed in the form
\[ T_{sb} = M_1 e^{[A_1] \theta_1} M_2 e^{[A_2] \theta_2} \cdots M_5 e^{[A_5] \theta_5}. \]
Find \( M_2, M_3, A_2, \) and \( A_3 \).
(b) Expressing the forward kinematics in the form
\[ T_{sb} = e^{[S_1] \theta_1} e^{[S_2] \theta_2} \cdots e^{[S_5] \theta_5} M, \]
find \( M \) and \( S_1, \ldots, S_5 \) in terms of \( M_1, \ldots, M_5, A_1, \ldots, A_5 \) appearing in (a).

20. The spatial PRRPRR open chain of Figure 4.25 is shown in its zero position, with space and end-effector frames chosen as shown. Derive its forward kinematics in the form
\[ T_{0n} = e^{[S_1] \theta_1} e^{[S_2] \theta_2} e^{[S_3] \theta_3} e^{[S_4] \theta_4} e^{[S_5] \theta_5} M e^{[S_6] \theta_6} \]
where \( M \in SE(3) \).
21. (Refer to Appendix C.) For each $T \in SE(3)$ below, find, if they exist, values for the four parameters $(\alpha, a, d, \phi)$ that satisfy

$$T = \text{Rot}(\hat{x}, \alpha) \text{Trans} (\hat{x}, a) \text{Trans}(\hat{z}, d) \text{Rot}(\hat{z}, \phi).$$
4.5. Exercises

(a) \( T = \begin{bmatrix} 0 & 1 & 1 & 3 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \).

(b) \( T = \begin{bmatrix} \cos \beta & \sin \beta & 0 & 1 \\ \sin \beta & -\cos \beta & 0 & 0 \\ 0 & 0 & -1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \).

(c) \( T = \begin{bmatrix} 0 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \).
Chapter 5

Velocity Kinematics and Statics

In the previous chapter we saw how to calculate the robot end-effector frame’s position and orientation for a given set of joint positions. In this chapter we examine the related problem of calculating the end-effector’s twist from a given set of joint positions and velocities.

Before we get to a representation of the end-effector twist as $V \in \mathbb{R}^6$ starting in Section 5.1, let us consider the case where the end-effector configuration is represented by a minimal set of coordinates $x \in \mathbb{R}^m$ and the velocity is given by $\dot{x} \in \mathbb{R}^m$. In this case, the forward kinematics can be written as

$$x(t) = f(\theta(t)),$$

where $\theta \in \mathbb{R}^n$ is a set of joint variables. By the chain rule, the time derivative at time $t$ is

$$\dot{x} = \frac{\partial f(\theta)}{\partial \theta} \dot{\theta} = J(\theta) \dot{\theta},$$

where $J(\theta) \in \mathbb{R}^{m \times n}$ is called the Jacobian. The Jacobian matrix represents the linear sensitivity of the end-effector velocity $\dot{x}$ to the joint velocity $\dot{\theta}$, and it is a function of the joint variables $\theta$.

To provide a concrete example, consider a 2R planar open chain (left side of Figure 5.1) with forward kinematics given by

$$x_1 = L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2)$$
$$x_2 = L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2).$$

Differentiating both sides with respect to time yields

$$\dot{x}_1 = -L_1 \dot{\theta}_1 \sin \theta_1 - L_2 (\dot{\theta}_1 + \dot{\theta}_2) \sin(\theta_1 + \theta_2)$$
$$\dot{x}_2 = L_1 \dot{\theta}_1 \cos \theta_1 + L_2 (\dot{\theta}_1 + \dot{\theta}_2) \cos(\theta_1 + \theta_2),$$
which can be rearranged into an equation of the form \( \dot{x} = J(\theta)\dot{\theta} \):

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 
\end{bmatrix} =
\begin{bmatrix}
-L_1 \sin \theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\
L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2)
\end{bmatrix}
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}.
\] (5.1)

Writing the two columns of \( J(\theta) \) as \( J_1(\theta) \) and \( J_2(\theta) \), and the tip velocity \( \dot{x} \) as \( v_{\text{tip}} \), Equation (5.1) can be written

\[ v_{\text{tip}} = J_1(\theta)\dot{\theta}_1 + J_2(\theta)\dot{\theta}_2. \]

As long as \( J_1(\theta) \) and \( J_2(\theta) \) are not collinear, it is possible to generate a tip velocity \( v_{\text{tip}} \) in any arbitrary direction in the \( x_1-x_2 \) plane by choosing appropriate joint velocities \( \dot{\theta}_1 \) and \( \dot{\theta}_2 \). Since \( J_1(\theta) \) and \( J_2(\theta) \) depend on the joint values \( \theta_1 \) and \( \theta_2 \), one may ask if there are any configurations at which \( J_1(\theta) \) and \( J_2(\theta) \) become collinear. For our example the answer is yes: if \( \theta_2 \) is 0° or 180°, then regardless of the value of \( \theta_1 \), \( J_1(\theta) \) and \( J_2(\theta) \) will be collinear, and the Jacobian \( J(\theta) \) becomes a singular matrix. Such configurations are called **singularities**; they are characterized by the robot tip being unable to generate velocities in certain directions.

Now let’s plug in \( L_1 = L_2 = 1 \) and consider the robot at two different nonsingular postures: \( \theta = (0, \pi/4) \) and \( \theta = (0, 3\pi/4) \). The Jacobians \( J(\theta) \) at these two configurations are

\[
J\left(\begin{bmatrix}
0 \\
\pi/4
\end{bmatrix}\right) = \begin{bmatrix}
-0.71 \\
1.71
\end{bmatrix}, \quad
J\left(\begin{bmatrix}
0 \\
3\pi/4
\end{bmatrix}\right) = \begin{bmatrix}
-0.71 \\
0.29
\end{bmatrix}.
\]

The right side of Figure 5.1 illustrates the robot at the \( \theta_2 = \pi/4 \) configuration. Column \( i \) of the Jacobian matrix, \( J_i(\theta) \), corresponds to the tip velocity when \( \dot{\theta}_i = 1 \) and the other joint velocity is zero. These tip velocities (and therefore Jacobian columns) are indicated in Figure 5.1.

The Jacobian can be used to map bounds on the rotational speed of the joints to bounds on \( v_{\text{tip}} \), as illustrated in Figure 5.2. Instead of mapping a polygon of joint velocities through the Jacobian as in Figure 5.2, we could instead map...
a unit circle of joint velocities in the $\theta_1$-$\theta_2$ plane. This circle represents an “iso-effort" contour in the joint velocity space, where total actuator effort is considered to be the sum of squares of the joint velocities. This circle maps through the Jacobian to an ellipse in the space of tip velocities, and this ellipse is referred to as the manipulability ellipsoid.\footnote{A two-dimensional ellipsoid, as in our example, is commonly referred to as an ellipse.} Figure 5.3 shows examples of this mapping for the two different postures of the 2R arm. As the manipulator configuration approaches a singularity, the ellipse collapses to a line segment, since the ability of the tip to move in one direction is lost.

Using the manipulability ellipsoid, one can quantify how close a given posture is to a singularity. For example, we can compare the lengths of the major and minor principal semi-axes of the manipulability ellipsoid, respectively denoted $\ell_{\text{max}}$ and $\ell_{\text{min}}$. The closer the ellipsoid is to a circle—meaning the ratio $\ell_{\text{max}}/\ell_{\text{min}}$ is small, close to 1—the more easily the tip can move in arbitrary directions, and thus the more removed it is from a singularity.

The Jacobian also plays a central role in static analysis. Suppose an external force is applied to the robot tip. What are the joint torques required to resist this external force?

This question can be answered via a conservation of power argument. Assuming there is no power loss due to joint friction, and that the robot is at equilibrium (no power is used to move the robot), the power measured at the robot’s tip must equal the power generated at the joints. Denoting the tip force vector generated by the robot as $\mathbf{f}_{\text{tip}}$ and the joint torque vector by $\mathbf{\tau}$,
conservation of power then requires that
\[ f_{\text{tip}}^T v_{\text{tip}} = \tau^T \dot{\theta}, \]
for all arbitrary joint velocities \( \dot{\theta} \). Since \( v_{\text{tip}} = J(\theta) \dot{\theta} \), the equality
\[ f_{\text{tip}}^T J(\theta) \dot{\theta} = \tau^T \dot{\theta} \]
must hold for all possible \( \dot{\theta} \).\(^2\) This can only be true if
\[ \tau = J^T(\theta) f_{\text{tip}}. \]  
\(^{(5.2)}\)

The joint torque \( \tau \) needed to create the tip force \( f_{\text{tip}} \) is calculated from the equation above.

\(^2\)Since the robot is at equilibrium, the joint velocity \( \dot{\theta} \) is technically zero. This can be considered the limiting case as \( \dot{\theta} \) approaches zero. To be more formal, we could invoke the “principle of virtual work” which deals with infinitesimal joint displacements instead of joint velocities.
Figure 5.4: Mapping joint torque bounds to tip force bounds.

For our two-link planar chain example, $J(\theta)$ is a square matrix dependent on $\theta$. If the configuration $\theta$ is not a singularity, then both $J(\theta)$ and its transpose are invertible, and Equation (5.2) can be written

$$f_{\text{tip}} = \left( (J(\theta))^T \right)^{-1} \tau = J^{-T}(\theta) \tau. \quad (5.3)$$

Using the equation above, one can now determine, under the same static equilibrium assumption, what input torques are needed to generate a desired tip force, e.g., to determine the joint torques needed for the robot tip to push against a wall with a specified normal force. For a given posture $\theta$ of the robot at equilibrium, and given a set of joint torque limits such as

$$-1 \text{ Nm} \leq \tau_1 \leq 1 \text{ Nm}$$
$$-1 \text{ Nm} \leq \tau_2 \leq 1 \text{ Nm},$$

Equation (5.3) can be used to generate the set of all possible tip forces as shown in Figure 5.4.

Similar to the manipulability ellipsoid, a force ellipsoid can be drawn by mapping a unit circle “iso-effort” contour in the $\tau_1$-$\tau_2$ plane to an ellipsoid in the $f_1$-$f_2$ tip force plane via the Jacobian transpose inverse $J^{-T}(\theta)$ (see Figure 5.5). This illustrates how easily the robot can generate forces in different directions. As evident from the manipulability and force ellipsoids, if it is easy to generate a tip velocity in a given direction, then it is difficult to generate force in that same direction, and vice-versa (Figure 5.6). In fact, for a given robot configuration, the principal axes of the manipulability ellipsoid and force ellipsoid are identical, and the lengths of the principal semi-axes of the force ellipsoid are the reciprocals of the lengths of the principal semi-axes of the manipulability ellipsoid.

At a singularity, the manipulability ellipsoid collapses to a line segment. The force ellipsoid, on the other hand, becomes infinitely long in a direction orthogonal to the manipulability ellipsoid line segment (i.e., the direction of the
aligned links) and skinny in the orthogonal direction. Consider, for example, carrying a heavy suitcase with your arm. It is much easier if your arm hangs straight down in gravity (with your elbow fully straightened, at a singularity), because the force you must support passes directly through your joints, therefore requiring no torques about the joints. Only the joint structure bears the load, not muscles generating torques. Unlike the manipulability ellipsoid, which loses dimension at a singularity and therefore its area drops to zero, the force ellipsoid’s area goes to infinity. (Assuming the joints can support the load!)

In this chapter we present methods for deriving the Jacobian for general open chains, where the configuration of the end-effector is expressed as $T \in SE(3)$ and its velocity is expressed as a twist $\mathbf{v}$ in the fixed base frame or the end-effector body frame. We also examine how the Jacobian can be used for velocity and static analysis, including identifying kinematic singularities and determining the manipulability and force ellipsoids. Later chapters on inverse kinematics, motion planning, dynamics, and control make extensive use of the Jacobian and related notions introduced in this chapter.
5.1. Manipulator Jacobian

In the 2R planar open chain example, we saw that for any joint configuration $\theta$, the tip velocity vector $v_{\text{tip}}$ and joint velocity vector $\dot{\theta}$ are linearly related via the Jacobian matrix $J(\theta)$, i.e., $v_{\text{tip}} = J(\theta)\dot{\theta}$. The tip velocity $v_{\text{tip}}$ depends on the coordinates of interest for the tip, which in turn determines the specific form of the Jacobian. For example, in the most general case, $v_{\text{tip}}$ can be taken to be the six-dimensional twist of the end-effector frame, while for pure orienting devices like a wrist, $v_{\text{tip}}$ is usually taken to be the angular velocity of the end-effector frame. Other choices for $v_{\text{tip}}$ lead to different formulations for the Jacobian. We begin with the general case where $v_{\text{tip}}$ is taken to be the six-dimensional end-effector twist expressed in the fixed frame.

5.1.1 Space Jacobian

In this section we derive the relationship between an open chain’s joint velocity vector $\dot{\theta}$ and the end-effector’s spatial twist $\mathcal{V}_a$. We first recall a few basic properties from linear algebra and linear differential equations: (i) if $A, B \in$
Calculating $\dot{\theta}$ for $\theta(t)$ is a scalar function of $t$, then $\frac{d}{dt}e^{A\theta} = Ae^{A\theta}\dot{\theta} = e^{A\theta}A\dot{\theta}$; (ii) if $A \in \mathbb{R}^{n \times n}$ is constant and $\theta(t)$ is a scalar function of $t$, then $\frac{d}{dt}e^{A\theta} = Ae^{A\theta}\dot{\theta} = e^{A\theta}A\dot{\theta}$; (iii) $(e^{A\theta})^{-1} = e^{-A\theta}$.

Consider an $n$-link open chain whose forward kinematics is expressed in the following product of exponentials form:

$$T(\theta_1, \ldots, \theta_n) = e^{[S_1]_\theta_1}e^{[S_2]_\theta_2} \cdots e^{[S_n]_\theta_n} M.$$  \hspace{1cm} (5.4)

The spatial twist $\mathcal{V}_s$ is given by $[\mathcal{V}_s] = \dot{T}T^{-1}$, where

$$\dot{T} = \left( \frac{d}{dt}e^{[S_1]_\theta_1} \right) \cdots e^{[S_n]_\theta_n}M + e^{[S_1]_\theta_1} \left( \frac{d}{dt}e^{[S_2]_\theta_2} \right) \cdots e^{[S_n]_\theta_n}M + \cdots$$

$$= [S_1]\dot{\theta}_1 e^{[S_1]_\theta_1} \cdots e^{[S_n]_\theta_n}M + e^{[S_1]_\theta_1} [S_2]\dot{\theta}_2 e^{[S_2]_\theta_2} \cdots e^{[S_n]_\theta_n}M + \cdots$$

Also,

$$T^{-1} = M^{-1}e^{-[S_n]_\theta_n} \cdots e^{-[S_1]_\theta_1}.$$ 

Calculating $\dot{T}T^{-1}$,

$$[\mathcal{V}_s] = [S_1]\dot{\theta}_1 + e^{[S_1]_\theta_1} [S_2]e^{-[S_1]_\theta_1}\dot{\theta}_2 + e^{[S_1]_\theta_1} [S_2]e^{-[S_1]_\theta_1}[S_3]e^{-[S_2]_\theta_2} \dot{\theta}_3 + \cdots$$

The above can also be expressed in vector form by means of the adjoint mapping:

$$\mathcal{V}_s = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} = \begin{bmatrix} \mathcal{V}_{s1}(\theta) & \mathcal{V}_{s2}(\theta) & \cdots & \mathcal{V}_{sn}(\theta) \end{bmatrix} = J_s(\theta)\dot{\theta}.$$ \hspace{1cm} (5.5)

Observe that $\mathcal{V}_s$ is a sum of $n$ spatial twists of the form

$$\mathcal{V}_s = \mathcal{V}_{s1}(\theta)\dot{\theta}_1 + \cdots + \mathcal{V}_{sn}(\theta)\dot{\theta}_n,$$ \hspace{1cm} (5.6)

where each $\mathcal{V}_{si}(\theta) = (\omega_{si}(\theta), v_{si}(\theta))$ depends explicitly on the joint values $\theta \in \mathbb{R}^n$ for $i = 2, \ldots, n$. In matrix form,

$$\mathcal{V}_s = \begin{bmatrix} \mathcal{V}_{s1}(\theta) & \mathcal{V}_{s2}(\theta) & \cdots & \mathcal{V}_{sn}(\theta) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{bmatrix}.$$ \hspace{1cm} (5.7)

The matrix $J_s(\theta)$ is said to be the Jacobian in fixed (space) frame coordinates, or more simply the space Jacobian.

**Definition 5.1.** Let the forward kinematics of an $n$-link open chain be expressed in the following product of exponentials form:

$$T = e^{[S_1]_\theta_1} \cdots e^{[S_n]_\theta_n} M.$$ \hspace{1cm} (5.8)

The space Jacobian $J_s(\theta) \in \mathbb{R}^{6 \times n}$ relates the joint rate vector $\dot{\theta} \in \mathbb{R}^n$ to the end-effector spatial twist $\mathcal{V}_s$ via

$$\mathcal{V}_s = J_s(\theta)\dot{\theta}.$$ \hspace{1cm} (5.9)
5.1. Manipulator Jacobian

The $i$th column of $J_s(\theta)$ is

$$V_{si}(\theta) = \text{Ad}_{T_{i-1}}(S_i),$$

for $i = 2, \ldots, n$, with the first column $V_{s1}(\theta) = S_1$. □

To understand the physical meaning behind the columns of $J_s(\theta)$, observe that the $i$th column is of the form $\text{Ad}_{T_{i-1}}(S_i)$, where $T_{i-1} = e^{[S_1] \theta_1} \cdots e^{[S_{i-1}] \theta_{i-1}}$; recall that $S_i$ is the screw axis describing the $i$th joint axis in terms of the fixed frame with the robot in its zero position. $\text{Ad}_{T_{i-1}}(S_i)$ is therefore the screw axis describing the $i$th joint axis after it undergoes the rigid body displacement $T_{i-1}$. Physically this is the same as moving the first $i-1$ joints from their zero position to the current values $\theta_1, \ldots, \theta_{i-1}$. Therefore, the $i$th column $V_{si}(\theta)$ of $J_s(\theta)$ is simply the screw vector describing joint axis $i$, expressed in fixed frame coordinates, as a function of the joint variables $\theta_1, \ldots, \theta_{i-1}$.

In summary, the procedure for determining the columns $V_{si}$ of $J_s(\theta)$ is similar to the procedure for deriving the joint screws $S_i$ in the product of exponentials formula $e^{[S_1] \theta_1} \cdots e^{[S_n] \theta_n} M$: each column $V_{si}(\theta)$ is the screw vector describing joint axis $i$, expressed in fixed frame coordinates, but for arbitrary $\theta$ rather than $\theta = 0$.

Example: Space Jacobian for a Spatial RRRP Chain

We now illustrate the procedure for finding the space Jacobian for the spatial RRRP chain of Figure 5.7. Denote the $i$th column of $J_s(\theta)$ by $V_{si} = (\omega_{si}, v_{si})$. The $\text{Ad}_{T_{i-1}}$ matrices are implicit in our calculations of the screw axes of the displaced joint axes.

- Observe that $\omega_{s1}$ is constant and in the $\hat{z}_s$-direction: $\omega_{s1} = (0, 0, 1)$. Picking $q_1$ to be the origin, $v_{s1} = (0, 0, 0)$.
Velocity Kinematics and Statics

Figure 5.8: Space Jacobian for the spatial RRPRRR chain.

- \( \omega_{s2} \) is also constant in the \( \hat{z}_s \)-direction, so \( \omega_{s2} = (0, 0, 1) \). Pick \( q_2 \) to be the point \( (L_1 c_1, L_1 s_1, 0) \), where \( c_1 = \cos \theta_1 \), \( s_1 = \sin \theta_1 \). Then \( v_{s2} = -\omega_2 \times q_2 = (L_1 s_1, -L_1 c_1, 0) \).

- The direction of \( \omega_{s3} \) is always fixed in the \( \hat{z}_s \)-direction regardless of the values of \( \theta_1 \) and \( \theta_2 \), so \( \omega_{s3} = (0, 0, 1) \). Picking \( q_3 = (L_1 c_1 + L_2 c_{12}, L_1 s_1 + L_2 s_{12}, 0) \), where \( c_{12} = \cos(\theta_1 + \theta_2) \), \( s_{12} = \sin(\theta_1 + \theta_2) \), it follows that \( v_{s3} = (L_1 s_1 + L_2 s_{12}, -L_1 c_1 - L_2 c_{12}, 0) \).

- Since the final joint is prismatic, \( \omega_{s4} = (0, 0, 0) \), and the joint axis direction is given by \( v_{s4} = (0, 0, 1) \).

The space Jacobian is therefore

\[
J_s(\theta) = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 \\
0 & L_1 s_1 & L_1 s_1 + L_2 s_{12} & 0 \\
0 & -L_1 c_1 & -L_1 c_1 - L_2 c_{12} & 0 \\
0 & 0 & 0 & 1 
\end{bmatrix}
\]

Example: Space Jacobian for Spatial RRPRRR Chain

We now derive the space Jacobian for the spatial RRPRRR chain of Figure 5.8. The base frame is chosen as shown in the figure.

- The first joint axis is in the direction \( \omega_{s1} = (0, 0, 1) \). Picking \( q_1 = (0, 0, L_1) \), we get \( v_{s1} = -\omega_1 \times q_1 = (0, 0, 0) \).

- The second joint axis is in the direction \( \omega_{s2} = (-c_1, -s_1, 0) \). Picking \( q_2 = (0, 0, L_1) \), we get \( v_{s2} = -\omega_2 \times q_2 = (L_1 s_1, -L_1 c_1, 0) \).
5.1. Manipulator Jacobian

- The third joint is prismatic, so \( \omega_{s3} = (0, 0, 0) \). The direction of the prismatic joint axis is given by

\[
v_{s3} = \text{Rot}(\dot{z}, \theta_1)\text{Rot}(\dot{x}, -\theta_2) \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -s_1 c_2 \\ c_1 c_2 \\ -s_2 \end{bmatrix}.
\]

- Now consider the wrist portion of the chain. The wrist center is located at the point

\[
q_w = \begin{bmatrix} 0 \\ 0 \\ L_1 \end{bmatrix} + \text{Rot}(\dot{z}, \theta_1)\text{Rot}(\dot{x}, -\theta_2) \begin{bmatrix} 0 \\ L_1 + \theta_3 \\ 0 \end{bmatrix} = \begin{bmatrix} -(L_2 + \theta_3)s_1 c_2 \\ (L_2 + \theta_3)c_1 c_2 \\ L_1 - (L_2 + \theta_3)s_2 \end{bmatrix}.
\]

Observe that the directions of the wrist axes depend on \( \theta_1, \theta_2, \) and the preceding wrist axes. These are

\[
\omega_{s4} = \text{Rot}(\dot{z}, \theta_1)\text{Rot}(\dot{x}, -\theta_2) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -s_1 s_2 \\ c_1 s_2 \\ c_2 \end{bmatrix}
\]

\[
\omega_{s5} = \text{Rot}(\dot{z}, \theta_1)\text{Rot}(\dot{x}, -\theta_2)\text{Rot}(\dot{z}, \theta_4) \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -c_1 c_4 + s_1 c_2 s_4 \\ -s_1 c_4 - c_1 c_2 s_4 \\ s_2 s_4 \end{bmatrix}
\]

\[
\omega_{s6} = \text{Rot}(\dot{z}, \theta_1)\text{Rot}(\dot{x}, -\theta_2)\text{Rot}(\dot{z}, \theta_4)\text{Rot}(\dot{x}, -\theta_5) \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -c_5 (s_1 c_2 c_4 + c_1 s_4) + s_1 s_2 s_5 \\ c_5 (c_1 c_2 c_4 - s_1 s_4) - c_1 s_2 s_5 \\ -s_2 c_4 c_5 - c_2 s_5 \end{bmatrix}.
\]

The space Jacobian can now be computed and written in matrix form as follows:

\[
J_s(\theta) = \begin{bmatrix} \omega_{s1} & \omega_{s2} & 0 & \omega_{s4} & \omega_{s5} & \omega_{s6} \\ 0 & -\omega_{s2} x q_2 & v_{s3} & -\omega_{s4} x q_w & -\omega_{s5} x q_w & -\omega_{s6} x q_w \end{bmatrix}.
\]

Note that we were able to obtain the entire Jacobian directly, without having to explicitly differentiate the forward kinematic map.

5.1.2 Body Jacobian

In the previous section we derived the relationship between the joint rates and \( \mathcal{V}_s = \dot{T}T^{-1} \), the end-effector’s twist expressed in fixed frame coordinates. Here we derive the relationship between the joint rates and \( \mathcal{V}_b = T^{-1}\dot{T} \), the end-effector twist in end-effector frame coordinates. For this purpose it will be more convenient to express the forward kinematics in the alternate product of exponentials form:

\[
T(\theta) = M e^{[B_1] \theta_1} e^{[B_2] \theta_2} \cdots e^{[B_n] \theta_n}.
\]
Computing $\dot{T}$,

$$
\dot{T} = Me^{[B_1]q_1} \cdots e^{[B_n-1]q_{n-1}} \left( \frac{d}{dt} e^{[B_n]q_n} \right) + Me^{[B_1]q_1} \cdots \left( \frac{d}{dt} e^{[B_n-1]q_{n-1}} \right) e^{[B_n]q_n} + \ldots 
$$

$$
= Me^{[B_1]q_1} \cdots e^{[B_n]q_n} [B_n] \dot{q}_n + Me^{[B_1]q_1} \cdots e^{[B_n-1]q_{n-1}} [B_n] \dot{q}_{n-1} + \ldots + Me^{[B_1]q_1} [B_1] e^{[B_2]q_2} \cdots e^{[B_n]q_n} \dot{q}_1.
$$

Also,

$$
T^{-1} = e^{-[B_n]q_n} \cdots e^{-[B_1]q_1} M^{-1}.
$$

Evaluating $T^{-1} \dot{T}$,

$$
[\mathcal{V}_b] = \left[ B_n \right] \dot{q}_n + Ad_{e^{-[B_n]q_n}} \left( B_{n-1} \right) \dot{q}_{n-1} + \ldots + Ad_{e^{-[B_1]q_1}} \left( B_1 \right) \dot{q}_1. \quad (5.12)
$$

or in vector form,

$$
\mathcal{V}_b = \sum_{\mathbb{B}_n} \dot{q}_n + Ad_{e^{-[B_n]q_n}} \left( B_{n-1} \right) \dot{q}_{n-1} + \ldots + Ad_{e^{-[B_1]q_1}} \left( B_1 \right) \dot{q}_1. \quad (5.12)
$$

$\mathcal{V}_b$ can therefore be expressed as a sum of $n$ body twists, i.e.,

$$
\mathcal{V}_b = \mathcal{V}_{b1}(\theta) \dot{q}_1 + \ldots + \mathcal{V}_{bn}(\theta) \dot{q}_n, \quad (5.13)
$$

where each $\mathcal{V}_{bi}(\theta) = (\omega_{bi}(\theta), v_{bi}(\theta))$ depends explicitly on the joint values $\theta$ for $i = 1, \ldots, n - 1$. In matrix form,

$$
\mathcal{V}_b = \begin{bmatrix} \mathcal{V}_{b1}(\theta) & \mathcal{V}_{b2}(\theta) & \cdots & \mathcal{V}_{bn}(\theta) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix} = J_b(\theta) \dot{q}. \quad (5.14)
$$

The matrix $J_b(\theta)$ is the Jacobian in the end-effector (or body) frame coordinates, or more simply the **body Jacobian**.

**Definition 5.2.** Let the forward kinematics of an $n$-link open chain be expressed in the following product of exponentials form:

$$
T = Me^{[B_1]q_1} \cdots e^{[B_n]q_n}. \quad (5.15)
$$

The **body Jacobian** $J_b(\theta) \in \mathbb{R}^{6 \times n}$ relates the joint rate vector $\dot{\theta} \in \mathbb{R}^n$ to the end-effector twist $\mathcal{V}_b = (\omega_b, v_b)$ via

$$
\mathcal{V}_b = J_b(\theta) \dot{\theta}. \quad (5.16)
$$

The $i$th column of $J_b(\theta)$ is

$$
\mathcal{V}_{bi}(\theta) = Ad_{e^{-[B_i]q_i} \cdots e^{-[B_{i+1}]q_{i+1}} \left( B_i \right), \quad (5.17)
$$

for $i = n - 1, \ldots, 1$, with $\mathcal{V}_{bn}(\theta) = B_n$. \(\square\)
5.1. Manipulator Jacobian

Analogous to the columns of the space Jacobian, a similar physical interpretation can be given to the columns of $J_b(\theta)$: each column $V_{bi}(\theta) = (\omega_{bi}(\theta), v_{bi}(\theta))$ of $J_b(\theta)$ is the screw vector for joint axis $i$, expressed in coordinates of the end-effector frame rather than the fixed frame. The procedure for determining the columns of $J_b(\theta)$ is similar to the procedure for deriving the forward kinematics in the product of exponentials form $M e^{\dot{\theta}_1} \cdots e^{\dot{\theta}_n}$, the only difference being that each of the end-effector frame joint screws $V_{bi}(\theta)$ are expressed for arbitrary $\theta$ rather than $\theta = 0$.

5.1.3 Relationship between the Space and Body Jacobian

Denoting the fixed frame by $\{s\}$ and the end-effector frame by $\{b\}$, the forward kinematics can be written $T_{sb}(\theta)$. The twist of the end-effector frame can be written in terms of the fixed and end-effector frame coordinates as

$$ [V_s] = T_{sb} T_{sb}^{-1} [V_b], $$

with $V_s$ and $V_b$ related by $V_s = \text{Ad}_{T_{sb}}(V_b)$ and $V_b = \text{Ad}_{T_{bs}}(V_s)$. $V_s$ and $V_b$ are also related to their respective Jacobians via

$$ V_s = J_s(\theta) \dot{\theta}, $$
$$ V_b = J_b(\theta) \dot{\theta}. $$

Equation (5.18) can therefore be written

$$ \text{Ad}_{T_{sb}}(V_b) = J_s(\theta) \dot{\theta}. $$

Applying $[\text{Ad}_{T_{sa}}]$ to both sides of Equation (5.20), and using the general property $[\text{Ad}_X][\text{Ad}_Y] = [\text{Ad}_{XY}]$ of the adjoint map, we obtain

$$ \text{Ad}_{T_{sa}}(\text{Ad}_{T_{sb}}(V_b)) = \text{Ad}_{T_{sa}T_{sb}}(V_b) = V_b = \text{Ad}_{T_{sb}}(J_s(q) \dot{\theta}). $$

Since we also have $V_b = J_b(\theta) \dot{\theta}$ for all $\dot{\theta}$, it follows that $J_s(\theta)$ and $J_b(\theta)$ are related by

$$ J_b(\theta) = \text{Ad}_{T_{sa}}(J_s(\theta)) = [\text{Ad}_{T_{sa}}] J_s(\theta). $$

The space Jacobian can in turn be obtained from the body Jacobian via

$$ J_s(\theta) = \text{Ad}_{T_{sb}}(J_b(\theta)) = [\text{Ad}_{T_{sb}}] J_b(\theta). $$

The fact that the space and body Jacobians, and space and body twists, are similarly related by the adjoint map should not be surprising, since each column of the space and body Jacobian corresponds to a twist.

One of the important implications of Equations (5.21) and (5.22) is that $J_b(\theta)$ and $J_s(\theta)$ always have the same rank; this is shown explicitly in the later section on singularity analysis.
5.1.4 Alternative Notions of the Jacobian

The space and body Jacobians derived above are matrices that relate joint rates to the twist of the end-effector. There exist alternative notions of the Jacobian that are based on a representation of the end-effector configuration using a minimum set of coordinates \( q \). Such representations are particularly relevant when the task space is considered to be a subspace of \( SE(3) \). For example, the configuration of the end-effector of a planar robot could be treated as \( q = (x, y, \theta) \in \mathbb{R}^3 \) instead of as an element of \( SE(2) \).

When using a minimum set of coordinates, the end-effector velocity is not given by a twist \( \dot{V} \) but by the time derivative of the coordinates \( \dot{q} \), and the Jacobian \( J_a \) in the velocity kinematics \( \dot{q} = J_a(\theta) \dot{\theta} \) is sometimes called the analytic Jacobian, as opposed to the geometric Jacobian in space and body form, as described above.

For an \( SE(3) \) task space, a typical choice of the minimal coordinates \( q \in \mathbb{R}^6 \) includes three coordinates for the origin of the end-effector frame in the fixed base frame, and three coordinates for the orientation of the end-effector frame in the fixed base frame. Example coordinates for the orientation include Euler angles (see Appendix B) and exponential coordinates for rotation.

Example: Analytic Jacobian with Exponential Coordinates for Rotation

In this example, we find the relationship between the geometric Jacobian \( J_b \) in the body frame and an analytic Jacobian \( J_a \) that uses exponential coordinates \( r = \hat{\omega} \theta \) to represent orientation. (Recall that \( \|\hat{\omega}\| = 1 \) and \( \theta \in [0, \pi] \).

First, consider an open chain with \( n \) joints with body Jacobian

\[
V_b = J_b(\theta) \dot{\theta},
\]

where \( J_b(\theta) \in \mathbb{R}^{6\times n} \). The angular and linear velocity components of \( V_b = (\omega_b, v_b) \) can be written explicitly as

\[
V_b = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = J_b(\theta) \dot{\theta} = \begin{bmatrix} J_\omega(\theta) \\ J_v(\theta) \end{bmatrix} \dot{\theta},
\]

where \( J_\omega \) is the \( 3 \times n \) matrix corresponding to the top three rows of \( J_b \) and \( J_v \) is the \( 3 \times n \) matrix corresponding to the bottom three rows of \( J_b \).

Now suppose our minimal set of coordinates \( q \in \mathbb{R}^6 \) is given by \( q = (r, x) \), where \( x \in \mathbb{R}^3 \) is the position of the origin of the end-effector frame and \( r = \hat{\omega} \theta \in \mathbb{R}^3 \) is the exponential coordinate representation for the rotation. The coordinate time-derivative \( \dot{x} \) is related to \( v_b \) by a rotation to get \( v_b \) in the fixed coordinates,

\[
\dot{x} = R_{sb}(\theta)v_b = R_{sb}(\theta)J_v(\theta) \dot{\theta},
\]

where \( R_{sb}(\theta) = e^{\|r\|} = e^{[\hat{\omega}]\theta} \).

The time-derivative \( \dot{r} \) is related to the body angular velocity \( \omega_b \) by

\[
\begin{align*}
\omega_b &= A(r) \dot{r} \\
A(r) &= I - \frac{1 - \cos \|r\|}{\|r\|^2} [r] + \frac{\|r\| - \sin \|r\|}{\|r\|^3} [r]^2.
\end{align*}
\]
5.2 Statics of Open Chains

(The derivation of this formula is explored in Exercise 10.) Provided the matrix \( A(r) \) is invertible, \( \dot{r} \) can be obtained from \( \omega_b \) as

\[
\dot{r} = A^{-1}(r)\omega_b = A^{-1}(r)J_\omega(\theta)\dot{\theta}.
\]

Putting these together,

\[
\dot{\theta} = \left[ \begin{array}{c} \dot{r} \\ \dot{x} \end{array} \right] = \left[ \begin{array}{cc} A^{-1}(r) & 0 \\ 0 & R_{sb} \end{array} \right] \left[ \begin{array}{c} \omega_b \\ v_b \end{array} \right],
\]

i.e., the analytic Jacobian \( J_a \) is related to the body Jacobian \( J_b \) by

\[
J_a(\theta) = \left[ \begin{array}{cc} A^{-1}(r) & 0 \\ 0 & R_{sb}(\theta) \end{array} \right] J_b(\theta).
\]

5.1.5 Inverse Velocity Kinematics

The sections above answer the question “What twist results from a given set of joint velocities?” The answer, written independently of the frame in which twists are represented, is given by

\[
V = J(\theta)\dot{\theta}.
\]

Often we are interested in the inverse question: given a desired twist \( V \), what joint velocities \( \dot{\theta} \) are needed? This is a question of inverse velocity kinematics, which is discussed in more detail in Chapter 6.3. In brief, if \( J(\theta) \) is square (the number of joints \( n \) is equal to six, the number of elements of a twist) and full rank, then \( \dot{\theta} = J^{-1}(\theta)V \). If \( n \neq 6 \) or the robot is at a singularity, however, then \( J(\theta) \) is not invertible. In the case \( n < 6 \), arbitrary twists \( V \) cannot be achieved—the robot does not have enough joints. If \( n > 6 \), then we call the robot redundant. In this case, a desired twist \( V \) places six constraints on the joint rates, and the remaining \( n - 6 \) freedoms correspond to internal motions of the robot that are not evident in the motion of the end-effector. As an example, if you consider your arm, from your shoulder to your palm, as a seven-joint open chain, when you place your palm at a fixed configuration in space (e.g., on the surface of a table), you still have one internal degree of freedom, corresponding to the position of your elbow.

5.2 Statics of Open Chains

Using our familiar principle of conservation of power,

\[
\text{power at the joints} = (\text{power to move the robot}) + (\text{power at end-effector}),
\]

and considering the robot to be at static equilibrium (no power used to move the robot), we can equate the power at the joints to the power at the end-effector\(^3\),

\[
\tau^T \dot{\theta} = F_b^T V_b,
\]

\(^3\)We are considering the limiting case as \( \dot{\theta} \) goes to zero, consistent with our assumption that the robot is at equilibrium.
where $\tau$ is the set of joint torques. Using the identity $V_b = J_b(\theta)\dot{\theta}$, we get
\[
\tau = J_b^T(\theta)F_b
\]
relating the joint torques to the wrench written in the end-effector frame. Similarly,
\[
\tau = J_s^T(\theta)F_s
\]
in the fixed space frame. Leaving out the choice of the frame, we can simply write
\[
\tau = J^T(\theta)F. \tag{5.25}
\]
Thus if we are given a desired end-effector wrench $F$ and the joint angles $\theta$, and we want the robot to resist an externally-applied wrench $-F$ in order to stay at equilibrium, we can calculate the joint torques $\tau$ needed to generate the opposing wrench $F$.\textsuperscript{4} This is important in force control of a robot, for example.

One could also ask the opposite question, namely, what is the end-effector wrench generated by a given set of joint torques? If $J^T$ is a $6 \times 6$ invertible matrix, then clearly $F = J^{-T}(\theta)\tau$. However, if the number of joints $n$ is not equal to six, then $J^T$ is not invertible, and the question is not well posed.

If the robot is redundant ($n > 6$), then even if the end-effector is embedded in concrete, the robot is not immobilized and the joint torques may cause internal motions of the links. The static equilibrium assumption is no longer satisfied, and we need dynamics to know what will happen to the robot.

If $n \leq 6$ and $J^T \in \mathbb{R}^{n \times 6}$ has rank $n$, then embedding the end-effector in concrete will immobilize the robot. If $n < 6$, however, no matter what $\tau$ we choose, the robot does not actively generate forces in the $6-n$ wrench directions defined by the null space of $J^T$,
\[
\text{Null}(J^T(\theta)) = \{F \mid J^T(\theta)F = 0\},
\]
since no actuators act in these directions. The robot can, however, resist arbitrary externally-applied wrenches in the space $\text{Null}(J^T(\theta))$ without moving, due to the lack of joints that would allow motions due to these forces. As an example, consider a motorized rotating door with a single revolute joint ($n = 1$) and an end-effector frame at the doorknob. The door can only actively generate a force at the knob that is tangent to the allowed circle of motion of the knob (defining a single direction in the wrench space), but it can resist arbitrary wrenches in the orthogonal five-dimensional wrench space without moving.

### 5.3 Singularity Analysis

The Jacobian allows us to identify postures at which the robot’s end-effector loses the ability to move instantaneously in one or more directions. Such a

\textsuperscript{4}If the robot has to support itself against gravity to maintain static equilibrium, these torques $\tau$ must be added to the torques that offset gravity.
5.3. Singularity Analysis

posture is called a **kinematic singularity**, or simply a **singularity**. Mathematically a singular posture is one in which the Jacobian $J(\theta)$ fails to be of maximal rank. To understand why, consider the body Jacobian $J_b(\theta)$, whose columns are denoted $\mathcal{V}_{bi}$, $i = 1, \ldots, n$. Then

\[
\mathcal{V}_b = \begin{bmatrix}
\mathcal{V}_{b1}(\theta) & \mathcal{V}_{b2}(\theta) & \cdots & \mathcal{V}_{bn}(\theta)
\end{bmatrix}
\begin{bmatrix}
\dot{\theta}_1 \\
\vdots \\
\dot{\theta}_n
\end{bmatrix}
\]

Thus, the tip frame can achieve twists that are linear combinations of the $\mathcal{V}_{bi}$. As long as $n \geq 6$, the maximum rank that $J_b(\theta)$ can attain is six. Singular postures correspond to those values of $\theta$ at which the rank of $J_b(\theta)$ drops below the maximum possible value; at such postures the tip frame loses the ability to generate instantaneous spatial velocities in in one or more dimensions. The loss of mobility at a singularity is accompanied by the ability to resist arbitrary wrenches in the direction corresponding to the lost mobility.

The mathematical definition of a kinematic singularity is independent of the choice of body or space Jacobian. To see why, recall the relationship between $J_s(\theta)$ and $J_b(\theta)$: $J_s(\theta) = \text{Ad}_{\mathcal{T}_{sb}}(J_b(\theta)) = [\text{Ad}_{\mathcal{T}_{sb}}]J_b(\theta)$, or more explicitly,

\[
J_s(\theta) = \begin{bmatrix}
R_{sb} & 0 \\
[p_{sb}] & R_{sb}
\end{bmatrix} J_b(\theta).
\]

We now claim that the matrix $[\text{Ad}_{\mathcal{T}_{sb}}]$ is always invertible. This can be established by examining the linear equation

\[
\begin{bmatrix}
R_{sb} & 0 \\
[p_{sb}] & R_{sb}
\end{bmatrix} \begin{bmatrix}
x \\
y
\end{bmatrix} = 0.
\]

Its unique solution is $x = y = 0$, implying that the matrix $[\text{Ad}_{\mathcal{T}_{sb}}]$ is invertible. Since multiplying any matrix by an invertible matrix does not change its rank, it follows that

\[
\text{rank } J_s(\theta) = \text{rank } J_b(\theta),
\]

as claimed; singularities of the space and body Jacobian are one and the same.

Kinematic singularities are also independent of the choice of the fixed frame and the end-effector frame. Choosing a different fixed frame is equivalent to simply relocating the robot arm, which should have absolutely no effect on whether a particular posture is singular or not. This obvious fact can be verified by referring to Figure 5.9(a). The forward kinematics with respect to the original fixed frame is denoted $\mathcal{T}(\theta)$, while the forward kinematics with respect to the relocated fixed frame is denoted $\mathcal{T}'(\theta) = \mathcal{P}\mathcal{T}(\theta)$, where $\mathcal{P} \in SE(3)$ is constant.
Then the body Jacobian of $T'(\theta)$, denoted $J'_b(\theta)$, is obtained from $T'^{-1}\dot{T}'$. A simple calculation reveals that

$$T'^{-1}\dot{T}' = (T^{-1}P^{-1})(P\dot{T}) = T^{-1}\dot{T},$$

i.e., $J'_b(\theta) = J_b(\theta)$, so that the singularities of the original and relocated robot arms are the same.

To see that singularities are independent of the end-effector frame, refer to Figure 5.9(b), and suppose the forward kinematics for the original end-effector frame is given by $T(\theta)$, while the forward kinematics for the relocated end-effector frame is $T'(\theta) = T(\theta)Q$, where $Q \in SE(3)$ is constant. This time looking at the space Jacobian—recall that singularities of $J_b(\theta)$ coincide with those of $J_s(\theta)$—let $J'_s(\theta)$ denote the space Jacobian of $T'(\theta)$. A simple calculation reveals that

$$\dot{T}'T'^{-1} = (\dot{T}Q)(Q^{-1}T^{-1}) = \dot{T}T^{-1},$$

i.e., $J'_s(\theta) = J_s(\theta)$, so that kinematic singularities are invariant with respect to choice of end-effector frame.

In the remainder of this section we consider some common kinematic singularities that occur in six-dof open chains with revolute and prismatic joints. We now know that either the space or body Jacobian can be used for our analysis; we use the space Jacobian in the examples below.

**Case I: Two Collinear Revolute Joint Axes**

The first case we consider is one in which two revolute joint axes are collinear (see Figure 5.10(a)). Without loss of generality these joint axes can be labeled
5.3. Singularity Analysis

![Figure 5.10: (a) A kinematic singularity in which two joint axes are collinear; (b) A kinematic singularity in which three revolute joint axes are parallel and coplanar.](image)

1 and 2. The corresponding columns of the Jacobian are

\[
\begin{align*}
V_{s1}(\theta) &= \begin{bmatrix} \omega_{s1} \\ -\omega_{s1} \times q_1 \end{bmatrix}, \\
V_{s2}(\theta) &= \begin{bmatrix} \omega_{s2} \\ -\omega_{s2} \times q_2 \end{bmatrix}
\end{align*}
\]

Since the two joint axes are collinear, we must have \(\omega_{s1} = \pm \omega_{s2}\); let us assume the positive sign. Also, \(\omega_{s1} \times (q_1 - q_2) = 0\) for \(i = 1, 2\). Then \(V_{s1} = V_{s2}\), which implies that \(V_{s1}\) and \(V_{s2}\) lie on the same line in six-dimensional space. Therefore, the set \(\{V_{s1}, V_{s2}, \ldots, V_{s6}\}\) cannot be linearly independent, and the rank of \(J_s(\theta)\) must be less than six.

**Case II: Three Coplanar and Parallel Revolute Joint Axes**

The second case we consider is one in which three revolute joint axes are parallel, and also lie on the same plane (three coplanar axes—see Figure 5.10(b)). Without loss of generality we label these as joint axes 1, 2, and 3. In this case we choose the fixed frame as shown in the figure; then

\[
J_s(\theta) = \begin{bmatrix} \omega_{s1} & \omega_{s1} & \omega_{s1} \\ 0 & -\omega_{s1} \times q_2 & -\omega_{s1} \times q_3 & \cdots \end{bmatrix}
\]

and since \(q_2\) and \(q_3\) are points on the same unit axis, it is not difficult to verify that the above three vectors cannot be linearly independent.

**Case III: Four Revolute Joint Axes Intersecting at a Common Point**

Here we consider the case where four revolute joint axes intersect at a common point (Figure 5.11). Again, without loss of generality label these axes from 1 to 4. In this case we choose the fixed frame origin to be the common point of
intersection, so that $q_1 = \ldots = q_4 = 0$. In this case

$$J_s(\theta) = \begin{bmatrix}
\omega_{s1} & \omega_{s2} & \omega_{s3} & \omega_{s4} & \cdots \\
0 & 0 & 0 & 0 & \cdots
\end{bmatrix}.$$  

The first four columns clearly cannot be linearly independent; one can be written as a linear combination of the other three. Such a singularity occurs, for example, when the wrist center of an elbow-type robot arm is directly above the shoulder.

**Case IV: Four Coplanar Revolute Joints**

Here we consider the case in which four revolute joint axes are coplanar. Again, without loss of generality label these axes from 1 to 4. Choose a fixed frame such that the joint axes all lie on the $x$-$y$ plane; in this case the unit vector $\omega_{si} \in \mathbb{R}^3$ in the direction of joint axis $i$ is of the form

$$\omega_{si} = \begin{bmatrix}
\omega_{six} \\
\omega_{siy} \\
0
\end{bmatrix}.$$  

Similarly, any reference point $q_i \in \mathbb{R}^3$ lying on joint axis $i$ is of the form

$$q_i = \begin{bmatrix}
q_{ix} \\
q_{iy} \\
0
\end{bmatrix},$$  

and subsequently

$$v_{si} = -\omega_{si} \times q_i = \begin{bmatrix}
0 \\
0 \\
\omega_{siy} q_{ix} - \omega_{six} q_{iy}
\end{bmatrix}.$$
5.4. Manipulability

The first four columns of the space Jacobian $J_s(\theta)$ are

$$
\begin{bmatrix}
\omega_{s1x} & \omega_{s2x} & \omega_{s3x} & \omega_{s4x} \\
\omega_{s1y} & \omega_{s2y} & \omega_{s3y} & \omega_{s4y} \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\omega_{s1y}q_{1x} - \omega_{s1x}q_{1y} & \omega_{s2y}q_{2x} - \omega_{s2x}q_{2y} & \omega_{s3y}q_{3x} - \omega_{s3x}q_{3y} & \omega_{s4y}q_{4x} - \omega_{s4x}q_{4y}
\end{bmatrix}
$$

which clearly cannot be linearly independent.

Case V: Six Revolute Joints Intersecting a Common Line

The final case we consider is six revolute joint axes intersecting a common line. Choose a fixed frame such that the common line lies along the $\hat{z}$-axis, and select the intersection between this common line and joint axis $i$ as the reference point $q_i \in \mathbb{R}^3$ for axis $i$; each $q_i$ is thus of the form $q_i = (0, 0, q_{iz})$, and

$$
v_{si} = -\omega_{si} \times q_i = (\omega_{sify_i} z, -\omega_{siz} q_{iz}, 0)
$$

$i = 1, \ldots, 6$. The space Jacobian $J_s(\theta)$ thus becomes

$$
J_s(\theta) =
\begin{bmatrix}
\omega_{s1x} & \omega_{s2x} & \omega_{s3x} & \omega_{s4x} & \omega_{s5x} & \omega_{s6x} \\
\omega_{s1y} & \omega_{s2y} & \omega_{s3y} & \omega_{s4y} & \omega_{s5y} & \omega_{s6y} \\
\omega_{s1z} & \omega_{s2z} & \omega_{s3z} & \omega_{s4z} & \omega_{s5z} & \omega_{s6z} \\
\omega_{s1y}q_{1z} & \omega_{s2y}q_{2z} & \omega_{s3y}q_{3z} & \omega_{s4y}q_{4z} & \omega_{s5y}q_{5z} & \omega_{s6y}q_{6z} \\
-\omega_{s1x}q_{1z} & -\omega_{s2x}q_{2z} & -\omega_{s3x}q_{3z} & -\omega_{s4x}q_{4z} & -\omega_{s5x}q_{5z} & -\omega_{s6x}q_{6z} \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

which is clearly singular.

5.4 Manipulability

In the previous section we saw that at a kinematic singularity, a robot’s end-effector loses the ability to move or rotate in one or more directions. A kinematic singularity is a binary proposition—a particular configuration is either kinematically singular, or it is not—and it is reasonable to ask whether it is possible to quantify the proximity of a particular configuration to a singularity. The answer is yes; in fact, one can even do better and quantify not only the proximity to a singularity, but also determine the directions in which the end-effector’s ability to move is diminished, and to what extent. The manipulability ellipsoid allows one to geometrically visualize the directions in which the end-effector moves with least effort and with greatest effort.

Manipulability ellipsoids are illustrated for a 2R planar arm in Figure 5.3. The Jacobian is given by Equation (5.1).
velocities for joint rates $\dot{\theta}$ satisfying $\|\dot{\theta}\| = 1$, a unit sphere in the $n$-dimensional joint velocity space.\footnote{A two-dimensional ellipsoid is often referred to as an “ellipse,” and an ellipsoid in more than three dimensions is often referred to as a “hyperellipsoid,” but here we use the term “ellipsoid” independent of the dimension. Similarly, we refer to a “sphere” independent of the dimension, instead of using “circle” for two dimensions and “hypersphere” for more than three dimensions.} Assuming $J$ is invertible, the unit joint velocity condition can be written

$$1 = \dot{\theta}^T \dot{\theta} = (J^{-1} \dot{q})^T (J^{-1} \dot{q}) = \dot{q}^T J^{-T} J^{-1} \dot{q} = \dot{q}^T (J J^T)^{-1} \dot{q} = \dot{q}^T A^{-1} \dot{q}.$$  \hspace{1cm} (5.26)

If $J$ is full rank (rank $m$), the matrix $A = J J^T \in \mathbb{R}^{m \times m}$ is square, symmetric, and positive definite, as is $A^{-1}$.

Consulting a textbook on linear algebra, we see that for any symmetric positive-definite $A^{-1} \in \mathbb{R}^{m \times m}$, the set of vectors $\dot{q} \in \mathbb{R}^m$ satisfying

$$\dot{q}^T A^{-1} \dot{q} = 1$$

defines an ellipsoid in the $m$-dimensional space. Letting $v_i$ and $\lambda_i$ be the eigenvectors and eigenvalues of $A$, the directions of the principal axes of the ellipsoid are $v_i$ and the lengths of the principal semi-axes are $\sqrt{\lambda_i}$, as illustrated in Figure 5.12. Furthermore, the volume $V$ of the ellipsoid is proportional to the product of the semi-axis lengths,

$$V \text{ is proportional to } \sqrt{\lambda_1 \lambda_2 \ldots \lambda_m} = \sqrt{\det(A)} = \sqrt{\det(J J^T)}.$$  

For the geometric Jacobian $J$ (either $J_b$ in the end-effector frame or $J_s$ in the fixed frame), we can express the $6 \times n$ Jacobian as

$$J(\theta) = \begin{bmatrix} J_\omega(\theta) \\ J_s(\theta) \end{bmatrix},$$
where $J_\omega$ is the top three rows of $J$ and $J_v$ is the bottom three rows of $J$. It makes sense to separate the two, because the units of angular velocity and linear velocity are different. This leads to two three-dimensional manipulability ellipsoids, one for angular velocities and one for linear velocities. The manipulability ellipsoids are given by principal semi-axes aligned with the eigenvectors of $A$ with lengths given by the square roots of the eigenvalues, where $A = J_\omega J_\omega^T$ for the angular velocity manipulability ellipsoid and $A = J_v J_v^T$ for the linear velocity manipulability ellipsoid.

When calculating the linear velocity manipulability ellipsoid, it generally makes more sense to use the body Jacobian $J_b$ instead of the space Jacobian $J_s$, since we are usually interested in the linear velocity of a point at the origin of the end-effector frame, rather than the linear velocity of a point at the origin of the fixed space frame.

Apart from the geometry of the manipulability ellipsoid, it can be useful to assign a single scalar measure defining how easily the robot can move at a given posture, or how close it is to a singularity. One measure is the ratio between the longest and shortest semi-axes of the manipulability ellipsoid,

$$\mu_1(A) = \frac{\sqrt{\lambda_{\text{max}}(A)}}{\sqrt{\lambda_{\text{min}}(A)}} = \frac{\lambda_{\text{max}}(A)}{\lambda_{\text{min}}(A)} \geq 1,$$

where $A = J J^T$. When $\mu_1(A)$ is low, close to one, then the manipulability ellipsoid is nearly spherical or isotropic, meaning that it is equally easy to move in any direction. This is generally a desirable situation. As the robot approaches a singularity, $\mu_1(A)$ goes to infinity.

A similar measure $\mu_2(A)$ is just the square of $\mu_1(A)$, also known as the condition number of the matrix $A = J J^T$,

$$\mu_2(A) = \frac{\lambda_{\text{max}}(A)}{\lambda_{\text{min}}(A)} \geq 1.$$

Again, smaller values, close to one, are preferred. The condition number of a matrix is commonly used to characterize the sensitivity of the result of a matrix multiplication by a vector to small errors in the vector.

A final measure is simply proportional to the volume of the manipulability ellipsoid,

$$\mu_3(A) = \sqrt{\lambda_1 \lambda_2 \ldots} = \sqrt{\det(A)}.$$

In this case, unlike the first two measures, a larger value is better.

Just like the manipulability ellipsoid, a force ellipsoid can be drawn for joint torques $\tau$ satisfying $\|\tau\| = 1$. Beginning from $\tau = J^T(\theta)F$, we arrive at similar results as above, except now the ellipsoid satisfies

$$1 = f^T J J^T f = f^T B^{-1} f,$$

where $B = (J J^T)^{-1} = A^{-1}$. For the force ellipsoid, the matrix $B$ plays the same role as $A$ in the manipulability ellipsoid, and it is the eigenvectors and the square roots of eigenvalues of $B$ that define the shape of the force ellipsoid.
Since eigenvectors of any invertible matrix \( A \) are also eigenvectors of \( B = A^{-1} \), the principal axes of the force ellipsoid are aligned with the principal axes of the manipulability ellipsoid. Furthermore, since the eigenvalues of \( B = A^{-1} \) associated with each principal axis are the reciprocals of the corresponding eigenvalues of \( A \), the lengths of the principal semi-axes of the force ellipsoid are given by \( 1/\sqrt{\lambda_i} \), where \( \lambda_i \) are the eigenvalues of \( A \). Thus the force ellipsoid is obtained from the manipulability ellipsoid simply by stretching the manipulability ellipsoid along each principal axis \( i \) by a factor of \( 1/\lambda_i \). Furthermore, since the volume \( V_A \) of the manipulability ellipsoid is proportional to the product of the semi-axes, \( \sqrt{\lambda_1 \lambda_2 \ldots} \), and the volume \( V_B \) of the force ellipsoid is proportional to \( 1/\sqrt{\lambda_1 \lambda_2 \ldots} \), the product of the two volumes \( V_A V_B \) is constant independent of the joint variables \( \theta \). Therefore, positioning the robot to increase the manipulability ellipsoid volume measure \( \mu_3(A) \) simultaneously reduces the force ellipsoid volume measure \( \mu_3(B) \). This also explains the observation in the beginning of the chapter that, as the robot approaches a singularity, \( V_A \) goes to zero while \( V_B \) goes to infinity.

5.5 Summary

- Let the forward kinematics of an \( n \)-link open chain be expressed in the following product of exponentials form:

\[
T(\theta) = e^{[S_1]\theta_1} \ldots e^{[S_n]\theta_n} M.
\]

The **space Jacobian** \( J_s(\theta) \in \mathbb{R}^{6 \times n} \) relates the joint rate vector \( \dot{\theta} \in \mathbb{R}^n \) to the end-effector spatial twist \( \mathcal{V}_s \) via \( \mathcal{V}_s = J_s(\theta) \dot{\theta} \). The \( i \)th column of \( J_s(\theta) \) is

\[
\mathcal{V}_{si}(\theta) = \text{Ad}_{e^{[S_1]\theta_1} \ldots e^{[S_{i-1}]\theta_{i-1}}} ([S_i]_i),
\]

for \( i = 2, \ldots, n \), with the first column \( \mathcal{V}_{s1}(\theta) = [S_1]_1 \). \( \mathcal{V}_{si} \) is the screw vector for joint \( i \) expressed in space frame coordinates, with the joint values \( \theta \) assumed to be arbitrary rather than zero.

- Let the forward kinematics of an \( n \)-link open chain be expressed in the following product of exponentials form:

\[
T(\theta) = M e^{[B_1]\theta_1} \ldots e^{[B_n]\theta_n}.
\]

The **body Jacobian** \( J_b(\theta) \in \mathbb{R}^{6 \times n} \) relates the joint rate vector \( \dot{\theta} \in \mathbb{R}^n \) to the end-effector body twist \( \mathcal{V}_b = (\omega_b, v_b) \) via \( \mathcal{V}_b = J_b(\theta) \dot{\theta} \). The \( i \)th column of \( J_b(\theta) \) is

\[
\mathcal{V}_{bi}(\theta) = \text{Ad}_{e^{-[[S_n]\theta_n]_n} \ldots e^{-[[S_{i+1}]\theta_{i+1}]_{i+1}}} ([B_i]_i),
\]

for \( i = n - 1, \ldots, 1 \), with \( \mathcal{V}_{in}(\theta) = [B_n]_n \). \( \mathcal{V}_{bi} \) is the screw vector for joint \( i \) expressed in body frame coordinates, with the joint values \( \theta \) assumed to be arbitrary rather than zero.
5.6. Software

- The body Jacobian is related to the space Jacobian via the relation

\[ J_s(\theta) = [\text{Ad}_{T_{sb}}] J_b(\theta) \]
\[ J_b(\theta) = [\text{Ad}_{T_{sb}}] J_s(\theta) \]

where \( T_{sb} = T(\theta) \).

- Consider a spatial open chain with \( n \) one-dof joints that is also assumed to be in static equilibrium. Let \( \tau \in \mathbb{R}^n \) denote the vector of joint torques and forces, and \( F \in \mathbb{R}^6 \) be the wrench applied at the end-effector, in either space or body frame coordinates. Then \( \tau \) and \( F \) are related by

\[ \tau = J^T_b(\theta) F_b = J^T_s(\theta) F_s. \]

- A kinematically singular configuration for an open chain, or more simply a **kinematic singularity**, is any configuration \( \theta \in \mathbb{R}^n \) at which the rank of the Jacobian is not maximal. For six-dof spatial open chains consisting of revolute and prismatic joints, some common singularities include (i) two collinear revolute joint axes; (ii) three coplanar and parallel revolute joint axes; (iii) four revolute joint axes intersecting at a common point; (iv) four coplanar revolute joints, and (v) six revolute joints intersecting a common line.

- The manipulability ellipsoid describes how easily the robot can move in different directions. For a Jacobian \( J \), the principal axes of the manipulability ellipsoid are defined by the eigenvectors of \( JJ^T \), and the corresponding lengths of the principal semi-axes are the square roots of the eigenvalues.

- The force ellipsoid describes how easily the robot can generate forces in different directions. For a Jacobian \( J \), the principal axes of the force ellipsoid are defined by the eigenvectors of \( (JJ^T)^{-1} \), and the corresponding lengths of the principal semi-axes are the square roots of the eigenvalues.

- Measures of the manipulability and force ellipsoid include the ratio of the longest principal semi-axis to the shortest; the square of this measure; and the volume of the ellipsoid. The first two measures indicate that the robot is far from a singularity if they are small (close to one).

5.6 Software

\( \text{Jb} = \text{JacobianBody(Blist,thetalist)} \)

Computes the body Jacobian \( J_b(\theta) \in \mathbb{R}^{6 \times n} \) given a list of joint screws \( B_i \) expressed in the body frame and a list of joint angles.

\( \text{Js} = \text{JacobianSpace(Blist,thetalist)} \)

Computes the space Jacobian \( J_s(\theta) \in \mathbb{R}^{6 \times n} \) given a list of joint screws \( S_i \) expressed in the fixed space frame and a list of joint angles.
5.7 Notes and References

The terms spatial velocity and spatial force were first coined by Roy Featherstone [32], and are also referred to in the literature as twists and wrenches, respectively. There is a well developed calculus of twists and wrenches that is covered in treatments of classical screw theory, e.g., [13], [4]. Singularities of closed chains are discussed in the later chapter on closed chain kinematics. Manipulability ellipsoids and their dual, force ellipsoids, are discussed in greater detail in [97].
5.8 Exercises

1. A wheel of unit radius is rolling to the right at a rate of 1 rad/sec (see Figure 5.13; the dotted circle shows the wheel at $t = 0$).
   (a) Find $V_s(t)$, the space velocity of the $\{b\}$ frame, as a function of $t$.
   (b) Find the linear velocity of the $\{b\}$ frame origin expressed in $\{s\}$ frame coordinates.

2. The 3R planar open chain of Figure 5.14(a) is shown in its zero position.
   (a) Suppose the tip must apply a force of 5 N in the $\hat{x}$-direction. What torques should be applied at each of the joints?
   (b) Suppose the tip must now apply a force of 5 N in the $\hat{y}$-direction. What torques should be applied at each of the joints?

3. Answer the following questions for the 4R planar open chain of Figure 5.14(b).
   (a) Derive the forward kinematics in the form
      \[
      T(\theta) = e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} e^{[S_4]\theta_4} M,
      \]

Figure 5.13: A rolling wheel.

Figure 5.14: (a) A 3R planar open chain. (b) A 4R planar open chain.
where each $S_i \in \mathbb{R}^3$ and $M \in SE(2)$.
(b) Derive the body Jacobian.
(c) Suppose the chain is in static equilibrium at the configuration $\theta_1 = \theta_2 = 0, \theta_3 = \frac{\pi}{2}, \theta_4 = -\frac{\pi}{2}$, and a force $f = (10, 10, 0)$ and moment $m = (0, 0, 10)$ are applied to the tip (both $f$ and $m$ are expressed with respect to the fixed frame). What are the torques experienced at each of the joints?
(d) Under the same conditions as (c), suppose that a force $f = (-10, 10, 0)$ and moment $m = (0, 0, -10)$ are applied to the tip. What are the torques experienced at each of the joints?
(e) Find all kinematic singularities for this chain.

4. Figure 5.15 shows two fingers grasping a can. Assume that all contacts are point contacts with sufficiently large friction coefficient. Frame \{b\} is attached to the center of the can. Frames \{b_1\} and \{b_2\} are attached to the can at the two contact points as shown. $f_1 = (f_{1,x}, f_{1,y}, f_{1,z})^T$ is the force applied by fingertip 1 to the can, expressed in \{b_1\} coordinates. Similarly, $f_2 = (f_{2,x}, f_{2,y}, f_{2,z})^T$ is the force applied by fingertip 2 to the can, expressed in \{b_2\} coordinates.
(a) Assume the system is in static equilibrium, and find the total spatial force $F_b$ applied by the two fingers to the can; express your result in \{b\} coordinates.
(b) Suppose $F_{\text{ext}}$ is an arbitrary external spatial force applied to the can ($F_{\text{ext}}$ is also expressed in frame \{b\} coordinates). Find all $F_{\text{ext}}$ that cannot be resisted by the fingertip forces.

5. Referring to Figure 5.16, the rigid body rotates about the point $(L, L)$ with angular velocity $\dot{\theta} = 1$.
(a) Find the position of point $P$ on the moving body with respect to the fixed reference frame in terms of $\theta$.
(b) Find the velocity of point $P$ in terms of the fixed frame.
(c) What is $T_{sb}$, the displacement of frame \{s\} as seen from the fixed frame \{s\}?
(d) Find the twist of $T_{sb}$ in body coordinates.
(e) Find the twist of $T_{sb}$ in space coordinates.
(f) What is the relation between the twists from (d) and (e)?
(g) What is the relation between the twist from (d) and $\dot{P}$ from (b)?
(h) What is the relation between the twist from (e) and $\dot{P}$ from (b)?

6. Figure 5.17 shows a design for a new amusement park ride. The rider sits at the location indicated by the moving frame \{b\}. The fixed frame \{s\} is attached to the top shaft as shown. $R = 1m$ and $L = 2m$, and the two joints $\theta_1$ and $\theta_2$ rotate at a constant angular velocity of 1 rad/sec.
(a) Suppose $t = 0$ at the instant shown in the figure. Find the linear velocity
velocity \( v_b \) and angular velocity \( \omega_b \) of the rider as a function of time \( t \). Express your answer in moving frame \( \{b\} \) coordinates.

(b) Find the linear velocity of the rider as a function of time \( t \). Express your answer in fixed frame \( \{s\} \) coordinates.

7. The RRP robot in Figure 5.18 is shown in its zero position.
(a) Write the screw axes in the space frame. Evaluate the forward kinematics when \( \theta = (90^\circ, 90^\circ, 1) \). Hand-draw or use a computer to show the arm and the end-effector frame in this configuration. Provide the space Jacobian \( J_s \) at this configuration.
(b) Write the screw axes in the end-effector body frame. Evaluate the forward kinematics when \( \theta = (90^\circ, 90^\circ, 1) \) and confirm that you get the same result. Provide the body Jacobian \( J_b \) at this configuration.

8. The RPR robot of Figure 5.19 is shown in its zero position. The fixed and end-effector frames are respectively denoted \( \{s\} \) and \( \{b\} \).
(a) Find the space Jacobian \( J_s(\theta) \) for arbitrary configurations \( \theta \in \mathbb{R}^3 \).
(b) Assume the manipulator is in its zero position. Suppose an external force \( f \in \mathbb{R}^3 \) is applied to the \( \{b\} \) frame origin. Find all directions of \( f \) that can be
5.8. Exercises

Figure 5.20: A spatial 3R open chain.

resisted by the manipulator with zero torques.

9. Find all kinematic singularities of the 3R wrist with the forward kinematics

\[ R = e^{[\omega_1]_{\theta_1}} e^{[\omega_2]_{\theta_2}} e^{[\omega_3]_{\theta_3}}, \]

where \( \omega_1 = (0, 0, 1)^T, \omega_2 = (1/\sqrt{2}, 0, 1/\sqrt{2})^T, \) and \( \omega_3 = (1, 0, 0)^T. \)

10. In this exercise we derive the analytic Jacobian for an \( n \)-link open chain corresponding to the exponential coordinates on \( SO(3) \).

(a) Given an \( n \times n \) matrix \( A(t) \) parametrized by \( t \) that is also differentiable with respect to \( t \), its exponential \( X(t) = e^{A(t)} \) is then an \( n \times n \) matrix that is always nonsingular. Prove the following:

\[
X^{-1} \dot{X} = \int_0^1 e^{-A(t)s} \dot{A}(t) e^{A(t)s} ds
\]

\[
\dot{X} X^{-1} = \int_0^1 e^{A(t)s} \dot{A}(t) e^{-A(t)s} ds.
\]

(b) Use the above result to show that for \( r(t) \in \mathbb{R}^3 \) and \( R(t) = e^{[r(t)]} \), the angular velocity in the body frame, \( [\omega_b] = R \dot{R}^T \), is related to \( \dot{r} \) by

\[
\omega_b = A(r) \dot{r}
\]

\[
A(r) = I - \frac{1 - \cos \|r\|}{\|r\|^2} [r] + \frac{\|r\| - \sin \|r\|}{\|r\|^3} [r]^2.
\]

(c) Derive the corresponding formula relating the angular velocity in the space frame, \( [\omega_s] = \dot{R} R^T \), with \( \dot{r} \).

11. The spatial 3R open chain of Figure 5.20 is shown in its zero position.

(a) In its zero position, suppose we wish to make the end-effector move with
linear velocity \( v_{\text{tip}} = (10, 0, 0) \), where \( v_{\text{tip}} \) is expressed with respect to the space frame \( \{s\} \). What are the required input joint velocities \( \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3 \)?

(b) Suppose the robot is in the configuration \( \theta_1 = 0, \theta_2 = 45^\circ, \theta_3 = -45^\circ \). Assuming static equilibrium, suppose we wish to generate an end-effector force \( f_b = (10, 0, 0) \), where \( f_b \) is expressed with respect to the end-effector frame \( \{b\} \). What are the required input joint torques \( \tau_1, \tau_2, \tau_3 \)?

(c) Under the same conditions as in (b), suppose we now seek to generate an end-effector moment \( m_b = (10, 0, 0) \), where \( m_b \) is expressed with respect to the end-effector frame \( \{b\} \). What are the required input joint torques \( \tau_1, \tau_2, \tau_3 \)?

(d) Suppose the maximum allowable torques for each joint motor are \( \|\tau_1\| \leq 10, \|\tau_2\| \leq 20, \|\tau_3\| \leq 5 \).

In the zero position, what is the maximum force that can be applied by the tip in the end-effector frame \( \hat{x} \)-direction?

12. The RRRP chain of Figure 5.21 is shown in its zero position.
(a) Determine the body Jacobian \( J_b(\theta) \) when \( \theta_1 = \theta_2 = 0, \theta_3 = \pi/2, \theta_4 = L \).
(b) Determine the linear velocity of the end-effector frame in fixed frame coordinates when \( \theta_1 = \theta_2 = 0, \theta_3 = \pi/2, \theta_4 = L \) and \( \dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_3 = \dot{\theta}_4 = 1 \).

13. For the 6R spatial open chain of Figure 5.22,
14. Show that a six-dof spatial open chain is in a kinematic singularity when any two of its revolute joint axes are parallel, and any prismatic joint axis is normal to the plane spanned by the two parallel revolute joint axes (see Figure 5.23).

15. The spatial PRRRRP open chain of Figure 5.24 is shown in its zero position.
   (a) At the zero position, find the first three columns of the space Jacobian.
   (b) Find all configurations at which the first three columns of the space Jacobian become linearly dependent.
   (c) Suppose the chain is in the configuration $\theta_1 = \theta_2 = \theta_3 = \theta_5 = \theta_6 = 0, \theta_4 = 90^\circ$. Assuming static equilibrium, suppose a pure force $f_b = (10, 0, 10)$ is applied to the origin of the end-effector frame, where $f_b$ is expressed in terms of the end-effector frame. Find the joint torques $\tau_1, \tau_2, \tau_3$ experienced at the
Consider the PRRRRR spatial open chain of Figure 5.25 shown in its zero position. The distance from the origin of the fixed frame to the origin of the end-effector frame at the home position is \( L \).

(a) Determine the first three columns of the space Jacobian \( J_s \).

(b) Determine the last two columns of the body Jacobian \( J_b \).

(c) For what value of \( L \) is the home position a singularity?

(d) In the zero position, what joint torques must be applied in order to generate a pure end-effector force of 100 N in the \( -\hat{z} \) direction?

17. The PRRRRP robot of Figure 5.26 is shown in its zero position.

(a) Find the first three columns of the space Jacobian \( J_s(\theta) \)

(b) Assuming the robot is in its zero position and \( \dot{\theta} = (1, 0, 1, -1, 2, 0)^T \), find the spatial velocity \( \mathcal{V}_s \) of the end-effector frame.
5.8. Exercises

(c) Is the zero position a kinematic singularity? Explain your answer.

18. The six-dof RRPRPR open chain of Figure 5.27 has fixed frame \{s\} and end-effector frame \{b\} attached as shown. At its zero position, joint axes 1, 2 and 6 lie on the \(\hat{y}-\hat{z}\) plane of the fixed frame, and joint axis 4 is aligned along the fixed frame \(\hat{x}\)-axis.
(a) Find the first three columns of the space Jacobian \(J_s(\theta)\).
(b) At the zero position, let \(\dot{\theta} = (1, 0, 1, -1, 2, 0)^T\). Find the spatial velocity \(V_s\) of the end-effector frame at the instant shown.
(c) Is the zero position a kinematic singularity? Explain your answer.

19. The spatial PRRRRP open chain of Figure 5.28 is shown in its zero position.
(a) Determine the first four columns of the space Jacobian \(J_s(\theta)\).
(b) Determine whether the zero position is a kinematic singularity.
(c) Calculate the joint torques required for the tip to apply the following end-effector wrenches:

Figure 5.27: An RRPRPR open chain shown at its zero position.

Figure 5.28: A spatial PRRRRP open chain with a skewed joint axis.
20. The spatial RRPRRR open chain of Figure 5.29 is shown in its zero position.
(a) For the fixed frame \{0\} and tool frame \{t\} as shown, express the forward
kinematics in the product of exponentials form
\[ T(\theta) = e^{[S_1] \theta_1} e^{[S_2] \theta_2} e^{[S_3] \theta_3} e^{[S_4] \theta_4} e^{[S_5] \theta_5} e^{[S_6] \theta_6} M. \]
(b) Find the first three columns of the space Jacobian \( J_s(\theta) \).
(c) Suppose that the fixed frame \{0\} is moved to another location \{0\}' as shown
in the figure. Find the first three columns of the space Jacobian \( J_s(\theta) \) with
respect to this new fixed frame.
(d) Determine if the zero position is a kinematic singularity, and if so, provide
a geometric description in terms of the joint screw axes.

21. The construction robot of Figure 5.30 is used for applications that require
large, high impact forces. Kinematically the robot can be modelled as a RPRRR
open chain. \( \theta_2 \) is a prismatic joint, and \( \theta_4 \) is defined to be the angle between
links 3 and 4 as shown in the figure. The robot is shown in its zero position
in Figures 5.30(a) and 5.30(b); the origin of the tool frame \{t\} is located at
coordinates \((\sqrt{3L_1} + L_2, L_1 - \sqrt{3L_3}, L_3)\) in terms of the fixed \{s\}-frame coordinates.

(a) The forward kinematics \( T_{st} \) can be expressed as
\[ T_{st} = e^{[S_1] \theta_1} e^{[S_2] \theta_2} e^{[S_3] \theta_3} e^{[S_4] \theta_4} e^{[S_5] \theta_5} M. \]
Find \( S_1 \) and \( S_4 \).
(b) In reality, joints \((\theta_1, \theta_2, \theta_3, \theta_5)\) are actuated. At the zero position, suppose
5.8. Exercises

the joint velocities are set to \((\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_5) = (1, 0, 1, 2)\). Find the spatial velocity \(V_s\) of the tool frame.

(c) Now suppose joints \(\theta_1\) and \(\theta_2\) are fixed at zero, and only joints \(\theta_3\) and \(\theta_5\) are actuated. Suppose \(L_1 = L_2\), and find the relation between \(\theta_3\) and \(\theta_5\) at which the manipulability ellipsoid becomes a circle.

(d) Under the same conditions as (c), but now with \(\theta_3 = \pi/6\) and \(\theta_5 = 0\), in what direction of \((f_x, f_y)\) can the tip exert a maximum force? Assume the maximum exertable torques at joint 3 and 5 are the same.

22. Figure 5.31 shows an RRPRRR arm exercise robot used for stroke patient rehabilitation\(^6\).

(a) Assume the manipulator is in its zero position, and suppose that \(M_{0c} \in SE(3)\) is the displacement from frame \(\{0\}\) to frame \(\{c\}\), and \(M_{ct} \in SE(3)\) is the displacement from frame \(\{c\}\) to frame \(\{t\}\). Express the forward kinematics \(T_{0t}\) in the form

\[
T_{0t} = e^{[A_1][\theta_1]}e^{[A_2][\theta_2]}M_{0c}e^{[A_3][\theta_3]}e^{[A_4][\theta_4]}M_{ct}e^{[A_5][\theta_5]}e^{[A_6][\theta_6]}.
\]

Find \(A_2, A_4, A_5\).

(b) Suppose that \(\theta_2 = 90^\circ\) and all the other joint variables are fixed at zero. Set the joint velocities to \((\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5) = (1, 0, 1, 0, 0, 1)\), and find the spatial velocity \(V_s\) of the tool frame in frame \(\{0\}\) coordinates.

(c) Is the configuration described in part (b) a kinematic singularity? Explain your answer.
(d) Suppose a person now operates the rehabilitation robot. At the configuration described in part (b), a spatial force $F_{\text{elbow}}$ is applied to the elbow link, and a spatial force $F_{\text{tip}}$ is applied to the last link. Both $F_{\text{elbow}}$ and $F_{\text{tip}}$ are
expressed in frame \{0\} coordinates and are given by

\[
\mathcal{F}_\text{elbow} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T
\]

\[
\mathcal{F}_\text{tip} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}^T.
\]

Find the joint torques that must be applied in order for the robot to maintain static equilibrium.

**23.** Consider an n-link open chain, with reference frames attached to each link. Let

\[
T_{0k} = e^{[S]_{\theta_1}} \ldots e^{[S]_{\theta_k}} M_k, \quad k = 1, \ldots, n
\]

be the forward kinematics up to link frame \{k\}. Let \( J_s(\theta) \) be the space Jacobian for \( T_{0n} \). The columns of \( J_s(\theta) \) are denoted

\[
J_s(\theta) = \begin{bmatrix} V_{s1}(\theta) & \cdots & V_{sn}(\theta) \end{bmatrix}.
\]

Let \( \dot{V}_k = \dot{T}T_{0k}^{-1} \) be the spatial velocity of link frame \{k\} in frame \{k\} coordinates.

(a) Derive explicit expressions for \( V_2 \) and \( V_3 \).

(b) Based on your results from (a), derive a recursive formula for \( V_{k+1} \) in terms of \( V_k, V_{s1}, \ldots, V_{s,k+1}, \) and \( \dot{\theta} \).

**24.** Write a program that allows the user to enter the link lengths \( L_1 \) and \( L_2 \) of a 2R planar robot (Figure 5.32), and a list of configurations of the robot (each as the joint angles \((\theta_1, \theta_2)\)), and plots the manipulability ellipse at each of those configurations. The program should plot the arm at each configuration (two line segments) and the manipulability ellipse centered at the endpoint of the arm. Choose the same scaling for all the ellipses so that they are easily visualized (e.g., the ellipse should usually be shorter than the arm, but not so small that you can’t easily see it). The program should also print the three manipulability measures \( \mu_1, \mu_2, \mu_3 \) for each configuration.

(i) Choose \( L_1 = L_2 = 1 \) and plot the arm and its manipulability ellipse at the four configurations \((-10^\circ, 20^\circ), (60^\circ, 60^\circ), (135^\circ, 90^\circ), (190^\circ, 160^\circ)\). At which of these configurations does the arm appear most isotropic? Does this agree with the manipulability measures calculated by the program?

(ii) Does the eccentricity of the ellipse depend on \( \theta_1 \)? On \( \theta_2 \)? Explain your answers.

(iii) Choose \( L_1 = L_2 = 1 \). Hand-draw the arm at \((-45^\circ, 90^\circ)\). Hand-draw the endpoint linear velocity vector arising from \( \dot{\theta}_1 = 1 \text{ rad/s} \) and \( \dot{\theta}_2 = 0 \). Hand-draw the endpoint linear velocity vector arising from \( \dot{\theta}_1 = 0 \) and \( \dot{\theta}_2 = 1 \text{ rad/s} \). Hand-draw the vector sum of these two to get the endpoint linear velocity when \( \dot{\theta}_1 = 1 \text{ rad/s} \) and \( \dot{\theta}_2 = 1 \text{ rad/s} \).
25. Modify the program in the previous exercise to plot the force ellipse. Demonstrate it at the same four configurations as in the first part of the previous exercise.

26. The kinematics of the 6R UR5 robot are given in Chapter 4.1.2.
   (i) Give the numerical space Jacobian $J_s$ when all joint angles are $\pi/2$. Divide the Jacobian matrix into an angular velocity portion $J_\omega$ (joint rates act on angular velocity) and a linear velocity portion $J_v$ (joint rates act on linear velocity).

   (ii) At this configuration, calculate the directions and lengths of the principal semi-axes of the three-dimensional angular velocity manipulability ellipsoid (based on $J_\omega$) and the directions and lengths of the principal semi-axes of the three-dimensional linear velocity manipulability ellipsoid (based on $J_v$).

   (iii) At this configuration, calculate the directions and lengths of the principal semi-axes of the three-dimensional moment (torque) force ellipsoid (based on $J_\omega$) and the directions and lengths of the principal semi-axes of the three-dimensional linear force ellipsoid (based on $J_v$).

27. The kinematics of the 7R WAM robot are given in Chapter 4.1.3.
   (i) Give the numerical body Jacobian $J_b$ when all joint angles are $\pi/2$. Divide the Jacobian matrix into an angular velocity portion $J_\omega$ (joint rates act on angular velocity) and a linear velocity portion $J_v$ (joint rates act on linear velocity).

   (ii) At this configuration, calculate the directions and lengths of the principal semi-axes of the three-dimensional angular velocity manipulability ellipsoid (based on $J_\omega$) and the directions and lengths of the principal semi-axes of the three-dimensional linear velocity manipulability ellipsoid (based on $J_v$).
(iii) At this configuration, calculate the directions and lengths of the principal semi-axes of the three-dimensional moment (torque) force ellipsoid (based on $J_\omega$) and the directions and lengths of the principal semi-axes of the three-dimensional linear force ellipsoid (based on $J_e$).

28. Examine the software functions for this chapter in your favorite programming language. Verify that they work the way you expect. Can you make them more computationally efficient?
Chapter 6

Inverse Kinematics

For a general $n$ degree-of-freedom open chain with forward kinematics $T(\theta)$, $\theta \in \mathbb{R}^n$, the inverse kinematics problem can be stated as follows: given a homogeneous transform $X \in SE(3)$, find solutions $\theta$ that satisfy $T(\theta) = X$. To highlight the main features of the inverse kinematics problem, let us consider the two-link planar open chain of Figure 6.1(a) as a motivational example. Considering only the position of the end-effector and ignoring its orientation, the forward kinematics can be expressed as

$$
\begin{bmatrix}
x \\
y
\end{bmatrix} =
\begin{bmatrix}
L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) \\
L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2)
\end{bmatrix}.
$$

(6.1)

Assuming $L_1 > L_2$, the set of reachable points, or the workspace, is an annulus of inner radius $L_1 - L_2$ and outer radius $L_1 + L_2$. Given some end-effector position $(x, y)$, it is not hard to see that there will be either zero, one, or two solutions depending on whether $(x, y)$ lies in the exterior, boundary, or interior of this annulus, respectively. When there are two solutions, the angle at the second joint (the “elbow” joint) may be positive or negative. These two solutions are sometimes called “lefty” and “righty” solutions, or “elbow-up” and “elbow-down” solutions.

Finding an explicit solution $(\theta_1, \theta_2)$ for a given $(x, y)$ is also not difficult. For this purpose, we will find it useful to introduce the two-argument arctangent function atan2($y, x$), which returns the angle from the origin to a point $(x, y)$ in the plane. It is similar to the inverse tangent $\tan^{-1}(y/x)$, but whereas $\tan^{-1}$ only returns angles in the range $(-\pi/2, \pi/2)$, since points $(x, y)$ are indistinguishable from points $(-x, -y)$, the atan2 function returns angles in the range $(-\pi, \pi]$. For this reason, atan2 is sometimes called the four-quadrant arctangent.

We also recall the law of cosines,

$$
c^2 = a^2 + b^2 - 2ab \cos C,
$$

where $a$, $b$, and $c$ are the lengths of the three sides of a triangle and $C$ is the interior angle of the triangle opposite the side of length $c$. 

191
Inverse Kinematics

Referring to Figure 6.1(b), angle $\beta$, restricted to lie in the interval $[0, \pi]$, can be determined from the law of cosines,

$$L_1^2 + L_2^2 - 2L_1L_2 \cos \beta = x^2 + y^2,$$

from which it follows that

$$\beta = \cos^{-1} \left( \frac{L_1^2 + L_2^2 - x^2 - y^2}{2L_1L_2} \right).$$

Also from the law of cosines,

$$\alpha = \cos^{-1} \left( \frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 \sqrt{x^2 + y^2}} \right).$$

The angle $\gamma$ is determined using the two-argument arctangent function, $\gamma = \text{atan2}(y, x)$. With these, the righty solution to the inverse kinematics is

$$\theta_1 = \gamma - \alpha, \quad \theta_2 = \pi - \beta$$

and the lefty solution is

$$\theta_1 = \gamma + \alpha, \quad \theta_2 = \beta - \pi.$$

If $x^2 + y^2$ lies outside the range $[L_1 - L_2, L_1 + L_2]$, then no solution exists.

This simple motivational example illustrates that for open chains, the inverse kinematics problem may have multiple solutions; this is in contrast to the forward kinematics, where a unique end-effector displacement $T$ exists for given
6.1. Analytic Inverse Kinematics

We begin by writing the forward kinematics of a spatial six-dof open chain in the following product of exponentials form:

\[ T(\theta) = e^{[S_1] \theta_1} e^{[S_2] \theta_2} e^{[S_3] \theta_3} e^{[S_4] \theta_4} e^{[S_5] \theta_5} e^{[S_6] \theta_6} M. \]

Given some end-effector frame \( X \in SE(3) \), in the inverse kinematics problem we seek solutions \( \theta \in \mathbb{R}^6 \) that satisfy \( T(\theta) = X \). In the following subsections we derive analytic inverse kinematic solutions for the PUMA and Stanford arms.

6.1.1 6R PUMA-Type Arm

We first consider a 6R arm of the PUMA type. Referring to Figure 6.2, when the arm is placed in its zero position, (i) the two shoulder joint axes intersect orthogonally at a common point, with joint axis 1 aligned in the \( \hat{z}_0 \) direction and joint axis 2 aligned in the \( \hat{x}_0 \) direction; (ii) joint axis 3 (the elbow joint) lies in the \( \hat{x}_0-\hat{y}_0 \) plane and is aligned parallel with joint axis 2; (iii) joint axes 4, 5, and 6...
(the wrist joints) intersect orthogonally at a common point (the wrist center) to form an orthogonal wrist; for the purposes of this example we assume joint axes 4, 5, and 6 are aligned in the $\hat{z}_0$, $\hat{y}_0$, and $\hat{x}_0$ directions, respectively. The arm may also have an offset at the shoulder (see Figure 6.3). The inverse kinematics problem for PUMA-type arms can be decoupled into an inverse position and inverse orientation subproblem, as we now show.

We first consider the simple case of a zero-offset PUMA-type arm. Referring to Figure 6.2 and expressing all vectors in terms of fixed frame coordinates, denote components of the wrist center $p \in \mathbb{R}^3$ by $p = (p_x, p_y, p_z)$. Projecting $p$ onto the $x$-$y$ plane, it can be seen that

$$\theta_1 = \tan^{-1}\left(\frac{p_y}{p_x}\right),$$

where the $\text{atan2}$ function can be used instead of $\tan^{-1}$. Note that a second valid
6.1. Analytic Inverse Kinematics

Figure 6.5: Four possible inverse kinematics solutions for the 6R PUMA type arm with shoulder offset.

solution for $\theta_1$ is given by

$$\theta_1 = \tan^{-1} \left( \frac{p_y}{p_x} \right) + \pi,$$

provided that the original solution for $\theta_2$ is replaced by $\pi - \theta_2$. As long as $p_x, p_y \neq 0$, both these solutions are valid. When $p_x = p_y = 0$ the arm is in a singular configuration (see Figure 6.4), and there are infinitely many possible solutions for $\theta_1$.

If there is an offset $d_1 \neq 0$ as shown in Figure 6.3, then in general there will be two solutions for $\theta_1$, denoted the **right** and **left arm** solutions (Figure 6.3). As seen from the figure, $\theta_1 = \phi - \alpha$, where $\phi = \tan^{-1} \left( \frac{p_y}{p_x} \right)$ and $\alpha = \tan^{-1} \left( \frac{\sqrt{r^2 - d_1^2}}{d_1} \right) = \tan^{-1} \left( \frac{\sqrt{r^2 + p_y^2 - d_1^2}}{d_1} \right)$. The second solution is given by

$$\theta_1 = \tan^{-1} \left( \frac{p_y}{p_x} \right) - \tan^{-1} \left( \frac{-\sqrt{p_x^2 + p_y^2 - d_1^2}}{d_1} \right).$$

Determining angles $\theta_2$ and $\theta_3$ for the PUMA-type arm now reduces to the inverse kinematics problem for a planar two-link chain:

$$\cos \theta_3 = \frac{r^2 + s^2 - a_2^2 - a_3^2}{2a_2a_3}$$

$$= \frac{p_x^2 + p_y^2 + (p_x^2 - d_1)^2 - a_2^2 - a_3^2}{2a_2a_3}.$$
If we let \( \cos \theta_3 = D \), then \( \theta_3 \) is given by
\[
\theta_3 = \tan^{-1} \left( \pm \frac{\sqrt{1 - D^2}}{D} \right)
\]

\( \theta_2 \) can be obtained in a similar fashion as
\[
\theta_2 = \tan^{-1} \left( \frac{s_2}{r} \right) - \tan^{-1} \left( \frac{a_3 s_3}{a_2 + a_3 c_3} \right)
\]
\[
= \tan^{-1} \left( \frac{p_z - d_1}{\sqrt{p_x^2 + p_y^2}} \right) - \tan^{-1} \left( \frac{a_3 s_3}{a_2 + a_3 c_3} \right)
\]
The two solutions for \( \theta_3 \) correspond to the well-known elbow-up and elbow-down configurations for the two-link planar arm. In general, a PUMA-type arm with an offset will have four solutions to the inverse position problem, as shown in Figure 6.5; the upper postures are called left-arm solutions (elbow-up and elbow-down), while the lower postures are called right-arm solutions (elbow-up and elbow-down).

We now solve the inverse orientation problem, i.e., finding \((\theta_4, \theta_5, \theta_6)\) given the end-effector frame orientation. This is completely straightforward: having found \((\theta_1, \theta_2, \theta_3)\), the forward kinematics can be manipulated into the form
\[
e^{[S_4] \theta_4} e^{[S_5] \theta_5} e^{[S_6] \theta_6} = e^{-[S_3] \theta_3} e^{-[S_2] \theta_2} e^{-[S_1] \theta_1} X M^{-1},
\]
where the right-hand side is now known, and the \( \omega_i \) components of \( S_4, S_5, S_6 \) are
\[
\omega_4 = (0, 0, 1) \\
\omega_5 = (0, 1, 0) \\
\omega_6 = (1, 0, 0).
\]
Denoting the \( SO(3) \) component of the right-hand side of Equation (6.2) by \( R \), the wrist joint angles \((\theta_4, \theta_5, \theta_6)\) can be determined as the solution to
\[
\text{Rot}(\hat{z}, \theta_4) \text{Rot}(\hat{y}, \theta_5) \text{Rot}(\hat{x}, \theta_6) = R,
\]
which corresponds exactly to the ZYX Euler angles as derived in Appendix B.

### 6.1.2 Stanford-Type Arms
If the elbow joint in a 6R PUMA-type arm is replaced by a prismatic joint as shown in Figure 6.6, we then have a Stanford-type arm. Here we consider the inverse position kinematics for the arm of Figure 6.6; the inverse orientation kinematics is identical to that for the PUMA-type arm and is not repeated here.
The first joint variable $\theta_1$ can be found in a similar fashion to the PUMA-type arm: $\theta_1 = \tan^{-1}\left(\frac{p_y}{p_x}\right)$ (provided that $p_x$ and $p_y$ are not both zero). $\theta_2$ is then found from Figure 6.6 to be

$$\theta_2 = \tan^{-1}\left(\frac{s}{r}\right)$$

where $r^2 = p_x^2 + p_y^2$ and $s = p_z - d_1$. Similar to the PUMA-type arm, a second solution for $\theta_1$, $\theta_2$ is given by

$$\theta_1 = \pi + \tan^{-1}\left(\frac{p_y}{p_x}\right)$$
$$\theta_2 = \pi - \tan^{-1}\left(\frac{p_y}{p_x}\right)$$

The translation distance $d_3$ is found from the relation

$$(d_3 + a_2)^2 = r^2 + s^2$$

as

$$d_3 = \sqrt{r^2 + s^2}$$
$$= \sqrt{p_x^2 + p_y^2 + (p_z - d_1)^2 - a_2}$$

Ignoring the negative square root solution for $d_3$, we obtain two solutions to the inverse position kinematics as long as the wrist center $p$ does not intersect
the z-axis of the fixed frame. If there is an offset, then as in the case of the
PUMA-type arm there will be a left and right arm solution.

If the elbow joint in the generalized 6R PUMA-type arm is replaced by a
prismatic joint, the resulting arm is then referred to as a generalized Stanford-
type arm. The inverse kinematics proceeds in the same way as for the general-
ized PUMA-type arm; the only difference occurs in the first step (obtaining $\theta_3$).
The screw vector for the third joint now becomes $S_3 = (0, v_3)$, with $\|v_3\| = 1$,
and $\theta_3$ is found by solving the equation

$$\|e^{[S_3]0_3} p\| = c,$$

for some given $p \in \mathbb{R}^3$ and nonnegative positive scalar $c$. The above equation
reduces to solving the following quadratic in $\theta_3$:

$$\theta_3^2 + 2(p^T v_3) \theta_3 + (\|p\|^2 - c^2) = 0.$$

Imaginary, as well as negative solutions, naturally should be excluded.

### 6.2 Numerical Inverse Kinematics

Iterative numerical methods can be applied if the inverse kinematics equations
do not admit analytic solutions. Even in cases where an analytic solution does
exist, numerical methods are often used to improve the accuracy of these solu-
tions. For example, in a PUMA-type arm, the last three axes may not exactly
intersect at a common point, and the shoulder joint axes may not be exactly or-
thogonal. In such cases, rather than throw away any analytic inverse kinematic
solutions that are available, such solutions can be used as the initial guess in an
iterative numerical procedure for solving the inverse kinematics.

There exist a variety of iterative methods for finding the roots of a nonlinear
equation, and our aim is not to discuss these in detail—any text on numerical
analysis will cover these methods in depth—but rather to develop ways in which
to transform the inverse kinematics equations such that they are amenable to
existing numerical methods. However, it is useful to review one fundamental
approach to nonlinear root-finding, the Newton-Raphson method. Also, in situ-
ations where an exact solution may not exist and we seek the closest approximate
solution, or conversely, an infinity of inverse kinematics solutions exists (i.e., if
the robot is kinematically redundant) and we seek a solution that is optimal
with respect to some criterion, methods of optimization are needed. We be-

### 6.2.1 Newton-Raphson Method

To numerically solve the equation $f(\theta) = 0$ for a given differentiable function
$f : \mathbb{R} \to \mathbb{R}$, assume $\theta_0$ is an initial guess for the solution, and expand $f(\theta)$ to
6.2. Numerical Inverse Kinematics

first-order about $\theta_0$:

$$f(\theta) = f(\theta_0) + \frac{\partial f}{\partial \theta}(\theta_0)(\theta - \theta_0) + \text{higher-order terms.}$$

Keeping only the terms up to first-order, set $f(\theta) = 0$ and solve for $\theta$ to obtain

$$\theta = \theta_0 - \left(\frac{\partial f}{\partial \theta}(\theta_0)\right)^{-1} \theta_0.$$

Using this value of $\theta$ as the new guess for the solution and repeating the above, we get the following iteration:

$$\theta_{k+1} = \theta_k - \left(\frac{\partial f}{\partial \theta}(\theta_k)\right)^{-1} \theta_k.$$

The above iteration is repeated until some stopping criterion is satisfied, e.g.,

$$\frac{|f(\theta_k) - f(\theta_{k+1})|}{|f(\theta_k)|} \leq \epsilon$$

for some user-prescribed threshold value $\epsilon$.

The same formula applies for the case when $f$ is multi-dimensional, i.e., $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$, in which case

$$\frac{\partial f}{\partial \theta}(\theta) = \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1}(\theta) & \cdots & \frac{\partial f_1}{\partial \theta_n}(\theta) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial \theta_1}(\theta) & \cdots & \frac{\partial f_n}{\partial \theta_n}(\theta) \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

The case when the above matrix fails to be invertible is discussed further in the later section describing the numerical inverse kinematics algorithm.

6.2.2 Optimization Basics

Suppose $x^* \in \mathbb{R}$ is a local minimum of a twice-differentiable objective function $f(x)$, $f : \mathbb{R} \rightarrow \mathbb{R}$, in the sense that for all $x$ near $x^*$, we have $f(x) \geq f(x^*)$. We can then expect that the slope of $f(x)$ at $x^*$ is zero, i.e.,

$$\frac{\partial f}{\partial x}(x^*) = 0,$$

and also that

$$\frac{\partial^2 f}{\partial x^2}(x^*) \geq 0.$$

If $f$ is multi-dimensional, i.e., $f : \mathbb{R}^n \rightarrow \mathbb{R}$, and all partial derivatives of $f$ exist up to second-order, then a necessary condition for $x^* \in \mathbb{R}^n$ to be a local minimum is that its gradient be zero:

$$\nabla f(x^*) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x^*) & \cdots & \frac{\partial f_n}{\partial x_n}(x^*) \end{bmatrix}^T = 0.$$

As an example, consider the linear equation $Ax = b$, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ ($m > n$) are given. Because there are more constraints ($m$) than
Inverse Kinematics

variables \((n)\), in general a solution to \(Ax = b\) will not exist. Suppose we seek the \(x\) that best approximates a solution, in the sense of minimizing

\[
\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} \|Ax - b\|^2 = \frac{1}{2}(Ax - b)^T(Ax - b) = \frac{1}{2}x^TA^T Ax - 2b^TAx + b^Tb.
\]

The first-order necessary condition is given by

\[
A^T Ax - A^T b = 0. \tag{6.3}
\]

If \(\text{rank}(A) = n\), then \(A^T A \in \mathbb{R}^{n \times n}\) is invertible, and the solution to (6.3) is

\[
x^* = (A^T A)^{-1} A^T b.
\]

Now suppose we wish to find, among all \(x \in \mathbb{R}^n\) that satisfy \(g(x) = 0\) for some differentiable \(g : \mathbb{R}^n \to \mathbb{R}^m\) (typically \(m \leq n\) to ensure that there exists an infinity of solutions to \(g(x) = 0\)), the \(x^*\) that minimizes the objective function \(f(x)\). Suppose \(x^*\) is a local minimum of \(f\) that is also a regular point of the surface parametrized implicitly by \(g(x) = 0\), i.e., \(x^*\) satisfies \(g(x^*) = 0\) and \(\text{rank} \left( \frac{\partial g}{\partial x}(x^*) \right) = m\).

Then from the fundamental theorem of linear algebra, it can be shown that there exists some \(\lambda^* \in \mathbb{R}^m\) (called the Lagrange multiplier) that satisfies

\[
\nabla f(x^*) + \frac{\partial g}{\partial x}^T(x^*) \lambda^* = 0 \tag{6.4}
\]

Equation (6.4) together with \(g(x^*) = 0\) constitute the first-order necessary conditions for \(x^*\) to be a feasible local minimum of \(f(x)\). Note that these two equations represent \(n + m\) equations in the \(n + m\) unknowns \(x\) and \(\lambda\).

As an example, consider the following quadratic objective function \(f(x)\):

\[
\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2}x^TQx + c^Tx
\]

subject to the linear constraint \(Ax = b\), where \(Q \in \mathbb{R}^n\) is symmetric positive-definite (that is, \(x^TQx > 0\) for all \(x \in \mathbb{R}^n\)), and the matrix \(A \in \mathbb{R}^{m \times n}\), \(m \leq n\), is of maximal rank \(m\). The first-order necessary conditions for this equality-constrained optimization problem are

\[
Qx + A^T \lambda = -c
\]

\[
Ax = b.
\]

Since \(A\) is of maximal rank and \(Q\) is invertible, the solutions to the first-order necessary conditions can be obtained after some manipulation as

\[
x = Gb + (I - GA)Q^{-1}c
\]

\[
\lambda = Bb + BAQ^{-1}c
\]
6.2. Numerical Inverse Kinematics

where \( G \in \mathbb{R}^{n \times m} \) and \( B \in \mathbb{R}^{m \times m} \) are defined as

\[
G = Q^{-1}A^TB, \quad B = (AQ^{-1}A^T)^{-1}.
\]

Both the proof of the existence of Lagrange multipliers and the derivation of the solution to the first-order necessary conditions are further explored in the exercises at the end of this chapter.

6.2.3 Numerical Inverse Kinematics Algorithm

Suppose we express the end-effector frame using a coordinate vector \( x \) governed by the forward kinematics \( x = f(\theta) \), a nonlinear vector equation mapping the \( n \) joint coordinates to the \( m \) end-effector coordinates. Assume \( f : \mathbb{R}^n \to \mathbb{R}^m \) is differentiable, and let \( x_d \) be the desired end-effector coordinates. The goal is to find joint coordinates \( \theta_d \) such that

\[
x_d - f(\theta_d) = 0.
\]

Given an initial guess \( \theta_0 \) which is “close by” a solution \( \theta_d \), the kinematics can be expressed as the Taylor expansion

\[
x_d = f(\theta_d) = f(\theta_0) + \left. \frac{\partial f}{\partial x} \right|_{\theta_0} (\theta_d - \theta_0) + \text{h.o.t.}, \quad (6.5)
\]

where \( J(\theta_0) \in \mathbb{R}^{m \times n} \) is the coordinate Jacobian evaluated at \( \theta_0 \). Truncating the Taylor expansion at first-order, we can approximate Equation (6.5) as

\[
J(\theta_0)\Delta \theta = x_d - f(\theta_0). \quad (6.6)
\]

Assuming \( J(\theta_0) \) is square \((m = n)\) and invertible, we can solve for \( \Delta \theta \) as

\[
\Delta \theta = J^{-1}(\theta_0) (x_d - f(\theta_0)). \quad (6.7)
\]

If the kinematics are linear, i.e., the higher-order terms in Equation (6.5) are zero, then the new guess \( \theta_1 = \theta_0 + \Delta \theta \) would exactly satisfy \( x_d = f(\theta_1) \). If not, the new guess \( \theta_1 \) should be closer to the root than \( \theta_0 \), and the process is then repeated, with the sequence \( \{\theta_0, \theta_1, \theta_2, \ldots\} \) converging to \( \theta_d \) (Figure 6.7).

As indicated in Figure 6.7, if there are multiple inverse kinematics solutions, the iterative process tends to converge to the solution that is “closest” to the initial guess \( \theta_0 \). You can think of each solution as having its own basin of attraction. If the initial guess is not in one of these basins (e.g., the initial guess is not sufficiently close to a solution), the iterative process may not converge.

In practice, for computational efficiency reasons, Equation (6.6) is often solved without directly calculating the inverse \( J^{-1}(\theta_0) \). More efficient techniques exist for solving a set of linear equations \( Ax = b \) for \( x \). For example, for invertible square matrices \( A \), the LU decomposition of \( A \) can be used to solve for \( x \) with fewer operations. In MATLAB, for example, the syntax
Inverse Kinematics

\[ x_d - f(\theta) = x - f(e) \]

\[ \text{slope} = -\frac{\partial f}{\partial \theta}(\theta_0) \]

\[ \Delta \theta = \left( \frac{\partial f}{\partial \theta}(\theta_0) \right)^{-1} (x_d - f(\theta_0)) \]

Figure 6.7: The first step of the Newton-Raphson method for nonlinear root-finding for a scalar \( x \) and \( \theta \). In the first step, the slope \(-\frac{\partial f}{\partial \theta}\) is evaluated at the point \((\theta_0, x_d - f(\theta_0))\). In the second step, the slope would be evaluated at the point \((\theta_1, x_d - f(\theta_1))\), and eventually the process would converge to \( \theta_d \).

Note that an initial guess to the left of the plateau of \( x_d - f(\theta) \) would likely result in convergence to the other root of \( x_d - f(\theta) \), and an initial guess at or near the plateau would result in a large initial \( |\Delta \theta| \) and the iterative process may not converge at all.

\[ x = A\backslash b \]

solves for \( x \) without computing \( A^{-1} \).

If \( J \) is not invertible, either because it is not square or because it is singular, then \( J^{-1} \) in Equation (6.7) does not exist. Equation (6.6) can still be solved (or approximately solved) for \( \Delta \theta \) by replacing \( J^{-1} \) in Equation (6.7) with the Moore-Penrose pseudoinverse \( J^\dagger \). For any equation of the form \( Jy = z \), where \( J \in \mathbb{R}^{m \times n}, y \in \mathbb{R}^n \), and \( z \in \mathbb{R}^m \), the solution

\[ y^* = J^\dagger z \]

falls into one of two categories:

- The solution \( y^* \) exactly satisfies \( Jy^* = z \), and for any solution \( y \) exactly satisfying \( Jy = z \), \( \|y^*\| \leq \|y\| \). In other words, among all solutions, the solution \( y^* \) minimizes the two-norm. There can be an infinite number of solutions \( y \) to \( Jy = z \) if the robot has more joints \( n \) than end-effector coordinates \( m \), i.e., the Jacobian \( J \) is “fat.”

- If there is no \( y \) that exactly satisfies \( Jy = z \), then \( y^* \) minimizes the two-norm of the error, i.e., \( \|Jy^* - z\| \leq \|Jy - z\| \) for any \( y \in \mathbb{R}^n \). This case corresponds to \( \text{rank}(J) < m \), i.e., the robot has fewer joints \( n \) than end-effector coordinates \( m \) (a “tall” Jacobian \( J \)) or is at a singularity.

Many programming languages provide functions to calculate the pseudoinverse; for example, the usage in MATLAB is
6.2. Numerical Inverse Kinematics

\[ y = \text{pinv}(J) \ast z \]

In the case that \( J \) is full rank (rank \( m \) for \( n > m \) or rank \( n \) for \( n < m \)), i.e., the robot is not at a singularity, the pseudoinverse can be calculated as

\[
J^\dagger = J^T (JJ^T)^{-1} \quad \text{if } J \text{ is fat, } n > m \quad \text{(called a right inverse, since } JJ^\dagger = I) \\
J^\dagger = (J^T J)^{-1} J^T \quad \text{if } J \text{ is tall, } n < m \quad \text{(called a left inverse, since } J^\dagger J = I). 
\]

Replacing the Jacobian inverse with the Jacobian pseudoinverse, Equation (6.7) becomes

\[
\Delta \theta = J^\dagger(\theta_0)(x_d - f(\theta_0)) . \tag{6.8}
\]

If rank(\( J \)) < \( m \), then the solution \( \Delta \theta \) calculated in Equation (6.8) may not exactly satisfy Equation (6.6), but it satisfies this condition as closely as possible in a least-squares sense. If \( n > m \), then the solution is the smallest joint variable change (in the two-norm sense) that exactly satisfies Equation (6.6).

Equation (6.8) suggests the Newton-Raphson iterative algorithm for finding \( \theta_d \):

(i) **Initialization:** Given \( x_d \in \mathbb{R}^m \) and an initial guess \( \theta_0 \in \mathbb{R}^n \). Set \( i = 0 \).

(ii) Set \( e = x_d - f(\theta_i) \). While ||\( e \)|| > \( \epsilon \) for some small \( \epsilon \):

- Set \( \theta_{i+1} = \theta_i + J^\dagger(\theta_i)e \).
- Increment \( i \).

To modify this algorithm to work with a desired end-effector configuration represented as \( T_{sd} \in SE(3) \) instead of as a coordinate vector \( x_d \), we can replace the coordinate Jacobian \( J \) with the end-effector body Jacobian \( J_b \in \mathbb{R}^{6 \times n} \). Note, however, that the vector \( e = x_d - f(\theta_i) \), representing the direction from the current guess (evaluated through the forward kinematics) to the desired end-effector configuration, cannot simply be replaced by \( T_{sd} - T_{sb}(\theta_i) \); the pseudoinverse of \( J_b \) should act on a body twist \( V_b \in \mathbb{R}^6 \). To find the right analogy, we should think of \( e = x_d - f(\theta_i) \) as a velocity vector which, if followed for unit time, would cause a motion from \( f(\theta_i) \) to \( x_d \). Similarly, we should look for a body twist \( V_b \) which, if followed for unit time, would cause a motion from \( T_{sb}(\theta_i) \) to the desired configuration \( T_{sd} \).

To find this \( V_b \), we first calculate the desired configuration in the body frame,

\[
T_{bd}(\theta_i) = T_{sb}^{-1}(\theta_i)T_{sd} = T_{bs}(\theta_i)T_{sd}.
\]

Then \( V_b \) is determined using the matrix logarithm,

\[
[V_b] = \log T_{bd}(\theta_i).
\]

This leads to the following inverse kinematics algorithm, analogous to the coordinate vector algorithm:

(i) **Initialization:** Given \( T_{sd} \) and an initial guess \( \theta_0 \in \mathbb{R}^n \). Set \( i = 0 \).
(ii) Set $[V_b] = \log \left(T_{sb}^{-1}(\theta_i)T_{sd}\right)$. While $\|\omega_b\| > \epsilon_\omega$ or $\|v_b\| > \epsilon_v$ for small $\epsilon_\omega, \epsilon_v$:

- Set $\theta_{i+1} = \theta_i + J_b^\dagger(\theta_i)V_b$.
- Increment $i$.

An equivalent form can be derived in the space frame, using the space Jacobian $J_s(\theta)$ and the spatial twist $V_s = [\text{Ad}_{T_{sd}}]V_b$.

For this numerical inverse kinematics method to converge, the initial guess $\theta_0$ should be sufficiently close to a solution $\theta_d$. This condition can be satisfied by starting the robot from an initial home configuration where both the actual end-effector configuration and the joint angles are known, and ensuring that the requested end-effector position $T_{sd}$ change slowly relative to the frequency of the calculation of the inverse kinematics. Then, for the rest of the robot’s run, the calculated $\theta_d$ at the previous timestep serves as the initial guess $\theta_0$ for the new $T_{sd}$ at the next timestep.

**Example: Planar 2R Robot**

The body Jacobian Newton-Raphson inverse kinematics algorithm is applied to the 2R robot in Figure 6.8. Each link is 1 m in length, and we would like to find the joint angles that place the tip of the robot at $(x, y) = (0.366 \text{ m}, 1.366 \text{ m})$, which corresponds to

$$T_{sd} = \begin{bmatrix}
-0.5 & -0.866 & 0 & 0.366 \\
0.866 & -0.5 & 0 & 1.366 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}$$
as shown by the frame \{goal\} in Figure 6.8. The forward kinematics, expressed in the end-effector frame, are given by

\[
M = \begin{bmatrix}
1 & 0 & 0 & 2 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}, \quad \mathcal{B}_1 = \begin{bmatrix}
0 \\
0 \\
1 \\
0
\end{bmatrix}, \quad \mathcal{B}_2 = \begin{bmatrix}
2 \\
0 \\
1 \\
0
\end{bmatrix}.
\]

Our initial guess to the solution is \(\theta_0 = (0, 30^\circ)\), and we specify an error tolerance of \(\epsilon_\omega = 0.001\) rad (or 0.057\(^\circ\)) and \(\epsilon_v = 10^{-4}\) m (100 microns). The progress of the Newton-Raphson method is illustrated in the table below, where only the \((\omega_{zb}, v_{xb}, v_{yb})\) components of the body twist \(\mathcal{V}_b\) are given, since the robot’s motion is restricted to the \(x\)-\(y\) plane:

<table>
<thead>
<tr>
<th>(i)</th>
<th>(\theta_i) (in degrees)</th>
<th>((x, y))</th>
<th>(\mathcal{V}<em>b = (\omega</em>{zb}, v_{xb}, v_{yb}))</th>
<th>(|\omega_b|)</th>
<th>(|v_b|)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>((0.00, 30.00^\circ))</td>
<td>((1.866, 0.500))</td>
<td>((1.571, 0.498, 1.858))</td>
<td>1.571</td>
<td>1.924</td>
</tr>
<tr>
<td>1</td>
<td>((34.23^\circ, 79.18^\circ))</td>
<td>((0.429, 1.480))</td>
<td>((-0.115, -0.074, 0.108))</td>
<td>0.115</td>
<td>0.131</td>
</tr>
<tr>
<td>2</td>
<td>((29.98^\circ, 90.22^\circ))</td>
<td>((0.363, 1.364))</td>
<td>((-0.004, 0.000, -0.004))</td>
<td>0.004</td>
<td>0.004</td>
</tr>
<tr>
<td>3</td>
<td>((30.00^\circ, 90.00^\circ))</td>
<td>((0.366, 1.366))</td>
<td>((0.000, 0.000, 0.000))</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

The iterative procedure converges to within the tolerances after three iterations. Figure 6.8 shows the initial guess, the goal configuration, and the configuration after one iteration. Notice that the first \(v_{xb}\) calculated is positive, even though the origin of the goal frame is in the \(-\hat{x}_b\) direction of the initial guess. This is because the constant body velocity \(\mathcal{V}_b\) that takes the initial guess to \{goal\} in one second is a rotation about the screw axis indicated in the figure.

### 6.3 Inverse Velocity Kinematics

To control a robot to follow a desired end-effector trajectory \(T_{sd}(t)\), one solution is to calculate the inverse kinematics \(\theta_d(k\Delta t)\) at each discrete time step \(k\), then control the joint velocities to be

\[
\dot{\theta} = (\theta_d(k\Delta t) - \theta((k - 1)\Delta t)) / \Delta t
\]

during the time interval \([((k - 1)\Delta t, k\Delta t]\. This is a type of feedback controller, since the desired new joint angles \(\theta_d(k\Delta t)\) are being compared to the most recently measured actual joint angles \(\theta((k - 1)\Delta t)\) to calculate the commanded joint velocities.

Another option that avoids the computation of inverse kinematics is to calculate the required joint velocities \(\dot{\theta}\) directly from the relationship \(J\dot{\theta} = \mathcal{V}_d\), where the desired end-effector twist \(\mathcal{V}_d\) and \(J\) are expressed with respect to the same frame:

\[
\dot{\theta} = J^\dagger(\theta)\mathcal{V}_d. \quad (6.9)
\]
The desired twist $V_d(t)$ can be chosen to be $T^{-1}_{sd}(t)\dot{T}_{sd}(t)$ (the body twist of the desired trajectory at time $t$) or $\dot{T}_{sd}(t)T^{-1}_{sd}(t)$ (the spatial twist), depending on whether the body Jacobian or space Jacobian is used, but small velocity errors are likely to accumulate over time, resulting in increasing position error. Preferably a position feedback controller would be designed to choose $V_d(t)$ to keep the end-effector following $T_{sd}(t)$ with little position error. Feedback control is discussed in Chapter 11.

In the case of a redundant robot with $n > 6$ joints, of the $(n-6)$-dimensional set of solutions satisfying Equation (6.9), the use of the pseudoinverse $J^T(\theta)$ returns the $\dot{\theta}$ minimizing the two-norm $||\dot{\theta}|| = \sqrt{\dot{\theta}^T \dot{\theta}}$. This is appealing; the motion is the minimum joint velocity motion that achieves the desired end-effector twist.

We could also choose to give the individual joint velocities different weighting. For example, as we will see later, the kinetic energy of a robot can be written

$$\frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta},$$

where $M(\theta)$ is the symmetric, positive-definite, configuration-dependent inertia matrix of the robot. We may also seek a path that minimizes some configuration-dependent potential energy function $h(\theta)$ (for example, $h(\theta)$ could be the gravitational potential energy, or an external elastic potential energy that may arise from mechanical springs connected between a robot’s links or at the joints). The time derivative of $h(\theta)$ is then the rate of change of the potential, or power:

$$\frac{d}{dt} h(\theta) = \nabla h(\theta)^T \dot{\theta}.$$

The following optimization problem can therefore be formulated:

$$\min_{\dot{\theta}} \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta} + \nabla h(\theta)^T \dot{\theta}$$

subject to the constraint $J(\theta) \dot{\theta} = V_d$. From the first-order necessary conditions for optimality, i.e.,

$$J^T \lambda = M \dot{\theta} + \nabla h$$

$$V_d = J \dot{\theta},$$

the optimal $\dot{\theta}$ and $\lambda$ can be derived as follows:

$$\dot{\theta} = G V_d + (I - GJ) M^{-1} \nabla h$$

$$\lambda = B V_d + BJ M^{-1} \nabla h,$$

where $B \in \mathbb{R}^{m \times m}$ and $G \in \mathbb{R}^{n \times m}$ are defined as

$$B = (JM^{-1}J^T)^{-1}$$

$$G = M^{-1}J^T(JM^{-1}J^T)^{-1} = M^{-1}J^T B.$$
Recalling the static relation \( \tau = J^T F \) from the previous chapter, the Lagrange multiplier \( \lambda \) can be interpreted as a spatial force in task space. Moreover, from the expression \( \lambda = BV_d + BJM^{-1}\nabla h \), the first term \( BV_d \) can be interpreted as a dynamic force generating the end-effector velocity \( V_d \), while the second term \( BJM^{-1}\nabla h \) can be interpreted as a static force required to keep the end-effector stationary.

If a potential function \( h(q) \) is not specified and we wish to choose the joint velocities \( \dot{\theta} \) to minimize the kinetic energy of the robot (intuitively, minimize the velocities of the joints moving a lot of mass), while still satisfying the desired end-effector twist, the solution

\[
\dot{\theta} = M^{-1}J^T(JM^{-1}J^T)^{-1}V_d,
\]

corresponding to the weighted Moore-Penrose pseudoinverse \( J^\dagger = M^{-1}J^T(JM^{-1}J^T)^{-1} \), minimizes the weighted norm \( \dot{\theta}^T M \dot{\theta} \).

### 6.4 A Note on Closed Loops

A desired end-effector trajectory over a time-interval \([0, t_f]\) is a closed loop if \( T_{sd}(0) = T_{sd}(t_f) \). It should be noted that numerical methods for calculating inverse kinematics for redundant robots, at either the configuration or velocity levels, are likely to yield motions that are not closed loops in the joint space, i.e., \( \theta(0) \neq \theta(t_f) \). If closed-loop motions in joint space are required, an extra set of conditions on the inverse kinematics must be satisfied.

### 6.5 Summary

- Given a spatial open chain with forward kinematics \( T(\theta), \theta \in \mathbb{R}^n \), in the inverse kinematics problem one seeks to find, for some given \( X \in SE(3) \), solutions \( \theta \) that satisfy \( X = T(\theta) \). Unlike the forward kinematics, the inverse kinematics problem can possess multiple solutions, or no solutions in the event that \( X \) lies outside the workspace. For a spatial open chain with \( n \) joints and an \( X \) in the workspace, \( n = 6 \) typically leads to a finite number of inverse kinematic solutions, while \( n > 6 \) leads to an infinite number of solutions.

- The inverse kinematics can be solved analytically for the six-dof PUMA-type robot arm, a popular 6R design consisting of a 3R orthogonal axis wrist connected to a 2R orthogonal axis shoulder by an elbow joint.

- Another class of open chains that admit analytic inverse kinematic solutions are Stanford-type arms; these arms are obtained by replacing the elbow joint in the generalized 6R PUMA-type arm by a prismatic joint. Geometric inverse kinematic algorithms similar to those for PUMA-type arms have been developed.
• Iterative numerical methods are used in cases where analytic inverse kinematic solutions are not available. These typically involve solving the inverse kinematics equations through an iterative procedure like the Newton-Raphson method, and they require an initial guess at the joint variables. The performance of the iterative procedure depends to a large extent on the quality of the initial guess, and in the case that there are several possible inverse kinematic solutions, the method finds the solution that is “closest” to the initial guess. Each iteration is of the form

$$\dot{\theta}_{i+1} = \theta_i + J^\dagger(\theta_i)\nu,$$

where $J^\dagger(\theta)$ is the pseudoinverse of the Jacobian $J(\theta)$ and $\nu$ is the twist that takes $T(\theta_i)$ to $T_{sd}$ in one second.

### 6.6 Software

```
[thetalist,success] = IKinBody(Blist,M,T,thetalist0,eomg,ev)
```
Uses iterative Newton-Raphson to calculate the inverse kinematics given the list of joint screws $B_i$ expressed in the end-effector frame, the end-effector home configuration $M$, the desired end-effector configuration $T$, an initial guess at the joint angles $\theta_0$, and the tolerances $\epsilon_\omega$ and $\epsilon_\nu$ on the final error. If a solution is not found within a maximum number of iterations, then `success` is false.

```
[thetalist,success] = IKinSpace(Slist,M,T,thetalist0,eomg,ev)
```
Similar to `IKinBody`, except the joint screws $S_i$ are expressed in the space frame, and the tolerances are interpreted in the space frame.

### 6.7 Notes and References

The inverse kinematics of the most general $6R$ open chain is known to have up to 16 solutions; this result was first proved by Lee and Liang [63], and also by Raghavan and Roth [101]. Iterative numerical procedures for finding all sixteen solutions of a general $6R$ open chain are reported in [77]. A summary of inverse kinematics methods for kinematically redundant robot arms are discussed in [123]. The repeatability conditions for kinematically redundant inverse kinematics schemes are examined in [116].
6.8 Exercises

1. Write a program that solves the analytical inverse kinematics for a planar 3R robot with link lengths $L_1 = 3$, $L_2 = 2$, and $L_1 = 1$, given the desired position $(x, y)$ and orientation $\theta$ of a frame fixed to the tip of the robot. Each joint has no joint limits. Your program should find all of the solutions (how many are there in the general case?), give the joint angles for each, and draw the robot in these configurations. Test the code for the case of $(x, y, \theta) = (4, 2, 0)$.

2. Solve the inverse position kinematics (you do not need to solve the orientation kinematics) of the 6R open chain robot shown in Figure 6.9.

3. Find the inverse kinematics solutions when the end-effector frame $\{T\}$ of the 6R open chain robot shown in Figure 6.10 is set to $\{T'\}$ as shown. The orientation of $\{T\}$ at the zero position is the same as that of the fixed frame $\{s\}$, and $\{T'\}$ is the result of a pure translation of $\{T\}$ along the $\hat{y}_s$-axis.

4. The RRP open chain of Figure 6.11 is shown in its zero position. Joint axes 1 and 2 intersect at the fixed frame origin, and the end-effector frame origin $p$ is located at $(0, 1, 0)$ when the robot is in its zero position.

   (a) Suppose $\theta_1 = 0$. Solve for $\theta_2$ and $\theta_3$ when the end-effector frame origin $p$ is...
Inverse Kinematics

Figure 6.11: An RRP open chain.

at \((-6, 5, \sqrt{3})\).

(b) If joint 1 is not fixed to zero but instead allowed to vary, find all inverse
kinematic solutions \((\theta_1, \theta_2, \theta_3)\) for the same \(p\) given in (a).

5. The four-dof robot of Figure 6.12 is shown in its zero position. Joint 1
is a screw joint of pitch \(h\). Given the end-effector position \(p = (p_x, p_y, p_z)\)
and orientation \(R = e^{[\hat{z}]\alpha}\), where \(\hat{z} = (0, 0, 1)\) and \(\alpha \in [0, 2\pi]\), find the inverse
kinematics solution \((\theta_1, \theta_2, \theta_3, \theta_4)\) as a function of \(p\) and \(\alpha\).

Figure 6.12: Open chain with screw joint
6. Figure 6.13(a) shows a surgical robot, which can be modelled as an RRRPPRP open chain as shown in Figure 6.13(b).
(a) In the general case, how many inverse kinematic solutions will exist for a given end-effector frame?
(b) Suppose coordinates for points A and B in the fixed frame are respectively given as \((x_A, y_A, z_A)\) and \((x_B, y_B, z_B)\). Solve the inverse kinematics for \(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5\): find an explicit formula for \((\theta_1, \theta_2, \theta_3)\), while for \((\theta_4, \theta_5)\), just describe the procedure.

(a) Surgical robot

(b) RRRPPRP robot at zero position

Figure 6.13: Surgical robot and kinematic model

7. In this exercise you will draw a plot of a scalar \(x_d - f(\theta)\) vs. a scalar \(\theta\) (similar to Figure 6.7) with two roots. Draw it so that for some initial guess \(\theta_0\), the iterative process actually jumps over the closest root and eventually converges to the further root. Hand-draw the plot and show the iteration process that results in convergence to the further root. Comment on the basins of attraction.
of the two roots in your plot.

8. Use Newton-Raphson iterative numerical root finding to perform two steps of finding the root of

\[ f(x, y) = \begin{bmatrix} x^2 - 4 \\ y^2 - 9 \end{bmatrix} \]

when your initial guess is \((x_1, y_1) = (1, 1)\). Write the general form of the gradient (for any guess \((x, y)\)) and compute the results of the first two iterations. You can do this by hand or write a program. If you write a program, submit your code and the printout of the results. Also, give all of the correct roots, not just the one that would be found from your initial guess. How many are there?

9. Modify the function \texttt{IKinBody} to print out the results of each Newton-Raphson iteration, similar to the table for the 2R robot example in Section 6.2. Show the table produced when the initial guess for the 2R robot of Figure 6.8 is \((0, 30^\circ)\) and the goal configuration corresponds to \((90^\circ, 120^\circ)\). Turn in your code.

10. The 3R orthogonal axis wrist mechanism of Figure 6.14 is shown in its zero position, with joint axes 1 and 3 collinear.
(a) Given a desired wrist orientation \(R \in SO(3)\), derive an iterative numerical procedure for solving its inverse kinematics.
(b) Perform a single iteration of Newton-Raphson root-finding using body-frame
6.8. Exercises

Figure 6.15: A 3R nonorthogonal chain.

numerical inverse kinematics. First write the forward kinematics and Jacobian for general configurations of the wrist. Then apply your results for the specific case of an initial guess of \( \theta_1 = \theta_3 = 0, \theta_2 = \pi/6 \), with a desired end-effector frame at

\[
R = \begin{bmatrix}
\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\
\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\
0 & 0 & 1
\end{bmatrix} \in SO(3).
\]

If you write code, submit the code together with the results.

11. The 3R nonorthogonal chain of Figure 6.15 is shown in its zero position.
   (a) Derive a numerical procedure for solving the inverse position kinematics numerically; that is, given some end-effector position \( p \) as indicated in the figure, find \( (\theta_1, \theta_2, \theta_3) \).
   (b) Given an end-effector orientation \( R \in SO(3) \), find all inverse kinematic solutions \( (\theta_1, \theta_2, \theta_3) \).

12. Use the function \texttt{IKinSpace} to find joint variables \( \theta_d \) of the UR5 (Chapter 4.1.2) satisfying

\[
T(\theta_d) = T_{sd} = \begin{bmatrix}
0 & 1 & 0 & -0.5 \\
0 & 0 & -1 & 0.1 \\
-1 & 0 & 0 & 0.1 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

Distances are in meters. Use \( \epsilon_\omega = 0.001 \) (0.057 degrees) and \( \epsilon_v = 0.0001 \) (0.1 mm). For your initial guess \( \theta_0 \), choose all joint angles as 0.1 rad. If the configuration is outside the workspace, or if you find that the zero configuration is too far from a final answer to converge, you may demonstrate \texttt{IKinBody} using another \( T_{sd} \). Note that the numerical inverse kinematics routines do not
Inverse Kinematics

respect joint limits, so it is possible for your routine to find solutions that are not achievable by the actual robot.

13. Use the function \texttt{IKinBody} to find joint variables \( \theta_d \) of the WAM (Chapter 4.1.3) satisfying

\[
T(\theta_d) = T_{sd} = \begin{bmatrix}
1 & 0 & 0 & 0.5 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0.4 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

Distances are in meters. Use \( \epsilon_\omega = 0.001 \) (0.057 degrees) and \( \epsilon_v = 0.0001 \) (0.1 mm). For your initial guess \( \theta_0 \), choose all joint angles as 0.1 rad. If the configuration is outside the workspace, or if you find that the zero configuration is too far from a final answer to converge, you may demonstrate \texttt{IKinBody} using another \( T_{sd} \).

14. The Fundamental Theorem of Linear Algebra (FTLA) states that given a matrix \( A \in \mathbb{R}^{m \times n} \),

\[
\text{null}(A) = \text{range}(A^T)^\perp \quad \text{null}(A^T) = \text{range}(A)^\perp,
\]

where \( \text{null}(A) \) denotes the null space of \( A \) (i.e., the subspace of \( \mathbb{R}^n \) of vectors that satisfy \( Ax = 0 \)), \( \text{range}(A) \) denotes the range or column space of \( A \) (i.e., the subspace of \( \mathbb{R}^m \) spanned by the columns of \( A \)), and \( \text{range}(A)^\perp \) denotes the orthogonal complement to \( \text{range}(A) \) (i.e., the set of all vectors in \( \mathbb{R}^m \) that are orthogonal to every vector in \( \text{range}(A) \)). In this problem you are asked to use FTLA to prove the existence of Lagrange multipliers for the equality constrained optimization problem. Let \( f(x), f : \mathbb{R}^n \rightarrow \mathbb{R} \) differentiable, be the objective function to be minimized. \( x \) must satisfy the equality constraint \( g(x) = 0 \) for given differentiable \( g : \mathbb{R}^n \rightarrow \mathbb{R}^m \).

(a) Suppose \( x^* \) is a local minimum. Let \( x(t) \) be any arbitrary curve on the surface parametrized implicitly by \( g(x) = 0 \) (implying that \( g(x(t)) = 0 \) for all \( t \)) such that \( x(0) = x^* \). Further assume that \( x^* \) is a regular point of the surface. Taking the time derivative of both sides of \( g(x(t)) = 0 \) at \( t = 0 \) then leads to

\[
\frac{\partial g}{\partial x}(x^*) \dot{x}(0) = 0. \quad (6.10)
\]

At the same time, because \( x(0) = x^* \) is a local minimum, it follows that \( f(x(t)) \) (viewed as an objective function in \( t \)) has a local minimum at \( t = 0 \), implying that

\[
\left. \frac{d}{dt} f(x(t)) \right|_{t=0} = \frac{\partial f}{\partial x}(x^*) \dot{x}(0) = 0. \quad (6.11)
\]

Since (6.10) and (6.11) must hold for all arbitrary curves \( x(t) \) on the surface defined by \( g(x) = 0 \), use FTLA to prove the existence of a Lagrange multiplier...
\( \lambda^* \in \mathbb{R}^m \) such that the first-order necessary condition
\[
\nabla f(x^*) + \frac{\partial g}{\partial x}(x^*)^T \lambda^* = 0.
\]
holds.

15. (a) If \( A^{-1} \) exists, show that
\[
\begin{bmatrix}
A & D \\
C & B
\end{bmatrix}^{-1} = \begin{bmatrix}
A^{-1} + EG^{-1}F & -EG^{-1} \\
-G^{-1}F & G^{-1}
\end{bmatrix},
\]
where \( G = B - CA^{-1}D, \ E = A^{-1}D, \) and \( F = CA^{-1} \).

(b) Use the above result to solve the first-order necessary conditions for the equality constrained optimization problem
\[
\min_{x \in \mathbb{R}^n} \frac{1}{2} x^T Q x + c^T x
\]
subject to \( Ax = b \), where \( Q \in \mathbb{R}^{n \times n} \) is symmetric positive-definite and \( A \in \mathbb{R}^{m \times n} \) is of maximal rank \( m \).
Inverse Kinematics
Chapter 7

Kinematics of Closed Chains

Any kinematic chain that contains one or more loops is called a closed chain. Several examples of closed chains were encountered in Chapter 2, from the planar four-bar linkage to spatial mechanisms like the Stewart-Gough platform and Delta robot (Figure 7.1). These mechanisms are examples of parallel mechanisms; these are closed chains consisting of a fixed and moving platform connected by a set of “legs.” The legs themselves are typically open chains, but sometimes can also be other closed chains (like the Delta robot depicted in the figure). In this chapter we analyze the kinematics of closed chains, paying special attention to parallel mechanisms.

The Stewart-Gough Platform is used widely as both a motion simulator and six-axis force-torque sensor. When used as a force-torque sensor, the six prismatic joints experience internal linear forces whenever any external force is applied to the moving platform; by measuring these internal linear forces one can estimate the applied external force. The Delta robot is a three-dof mechanism whose moving platform moves such that it always remains parallel to the fixed platform. Because the three actuators are all attached to the three revolute joints of the fixed platform, the moving parts are relatively light; this allows the Delta to achieve very fast motions.

Closed chains admit a much greater variety of designs than open chains, and their kinematic and static analysis is consequently more complicated. This can be traced to two defining features of closed chains: (i) not all joints are actuated, and (ii) the joint variables must satisfy a number of loop-closure constraint equations, which may or may not be independent depending on the configuration of the mechanism. The presence of unactuated (or passive) joints, together with the fact that the number of actuated joints may deliberately exceed the mechanism’s kinematic degrees of freedom—such mechanisms are said to be redundantly actuated—makes not only the kinematics analysis more challenging, but also introduces new types of singularities not present in
Recall also that for open chains, the kinematic analysis proceeds in a more or less straightforward fashion with the formulation of the forward kinematics (e.g., via the product of exponentials formalism) followed by that of the inverse kinematics. For general closed chains it is usually difficult to obtain an explicit set of equations for the forward kinematics in the form $X = T(\theta)$, where $X \in SE(3)$ is the end-effector frame and $\theta \in \mathbb{R}^n$ are the joint coordinates. The more effective approaches exploit as much as possible any kinematic symmetries and other special features of the mechanism.

In this chapter we begin with a series of case studies involving some well-known parallel mechanisms, and eventually build up a repertoire of kinematic analysis tools and methodologies for handling more general closed chains. Our focus will be on parallel mechanisms that are exactly actuated, i.e., the number of actuated degrees of freedom is equal to the number of degrees of freedom of the mechanism. Methods for the forward and inverse position kinematics of parallel mechanisms are discussed, followed by the characterization and derivation of the constraint Jacobian, and the Jacobians of both the inverse and forward kinematics. The chapter concludes with a discussion of the different types of kinematic singularities that arise in closed chains.

### 7.1 Inverse and Forward Kinematics

One general observation can be made for serial mechanisms vs. parallel mechanisms: For serial chains, forward kinematics is generally straightforward while inverse kinematics may be complex (e.g., multiple solutions or no solutions). For parallel mechanisms, the inverse kinematics is often relatively straightforward (e.g., given the configuration of a platform, it may not be hard to determine the joint variables), while the forward kinematics may be quite complex: an
arbitrarily chosen set of joint variables may be infeasible, or it may correspond to multiple possible configurations of the platform.

In this section we begin with two case studies, the 3×RPR planar parallel mechanism, and its spatial counterpart, the 3×SPS Stewart-Gough platform. The analysis of these two mechanisms draws upon some simplification techniques that result in a reduced form of the governing kinematic equations, which in turn can be generalized to the analysis of more general parallel mechanisms.

### 3×RPR Planar Parallel Mechanism

The first example we consider is the 3-dof planar 3×RPR parallel mechanism shown in Figure 7.2. A fixed frame \( \{s\} \) and body frame \( \{b\} \) are assigned to the platform as shown. The three prismatic joints are typically actuated while the six revolute joints are passive. Denote the lengths of each of the three legs by \( s_i, \ i = 1, 2, 3 \). The forward kinematics problem is to determine, from given values of \( s = (s_1, s_2, s_3) \), the body frame’s position and orientation. Conversely, the inverse kinematics problem is to determine \( s \) from \( T_{sb} \in SE(2) \).

Let \( p \) be the vector from the origin of the \( \{s\} \) frame to the origin of the \( \{b\} \) frame. Let \( \phi \) denote the angle measured from the \( \hat{x}_s \) axis of the \( \{s\} \) frame to the \( \hat{x}_b \) axis of the \( \{b\} \) frame. Further define the vectors \( a_i, b_i, d_i, i = 1, 2, 3 \), as shown in the figure. From these definitions, clearly

\[
d_i = p + b_i - a_i,
\]  

(7.1)
for $i = 1, 2, 3$. Let

$$
\begin{bmatrix}
p_x \\
p_y
\end{bmatrix} = p \text{ in } \{s\}-\text{frame coordinates}
$$

$$
\begin{bmatrix}
a_{ix} \\
a_{iy}
\end{bmatrix} = a_i \text{ in } \{s\}-\text{frame coordinates}
$$

$$
\begin{bmatrix}
d_{ix} \\
d_{iy}
\end{bmatrix} = d_i \text{ in } \{s\}-\text{frame coordinates}
$$

$$
\begin{bmatrix}
b_{ix} \\
b_{iy}
\end{bmatrix} = b_i \text{ in } \{b\}-\text{frame coordinates}
$$

Note that the $(a_{ix}, a_{iy}), (b_{ix}, b_{iy}), i = 1, 2, 3$ are all constant, and that with the exception of the $(b_{ix}, b_{iy})$, all other vectors are expressed in $\{s\}$-frame coordinates. To express Equation (7.1) in terms of $\{s\}$-frame coordinates, $b_{ix}$ must be expressed in $\{s\}$-frame coordinates. This is straightforward: defining

$$
R_{sb} = \begin{bmatrix}
\cos \phi & -\sin \phi \\
\sin \phi & \cos \phi
\end{bmatrix},
$$

it follows that

$$
\begin{bmatrix}
d_{ix} \\
d_{iy}
\end{bmatrix} = \begin{bmatrix}
p_x \\
p_y
\end{bmatrix} + R_{sb} \begin{bmatrix}
b_{ix} \\
b_{iy}
\end{bmatrix} - \begin{bmatrix}
a_{ix} \\
a_{iy}
\end{bmatrix},
$$

for $i = 1, 2, 3$. Also, since $s_i^2 = d_{ix}^2 + d_{iy}^2$, we have

$$
s_i^2 = (p_x + b_{ix} \cos \phi - b_{iy} \sin \phi - a_{ix})^2 + (p_y + b_{ix} \sin \phi + b_{iy} \cos \phi - a_{iy})^2 ,
$$

(7.2)

for $i = 1, 2, 3$.

Formulated as above, the inverse kinematics is trivial to compute: given values for $(p_x, p_y, \phi)$, the leg lengths $(s_1, s_2, s_3)$ can be directly calculated from the above equations (negative values of $s_i$ will not be physically realizable in most cases and can be ignored). In contrast, the forward kinematics problem of determining the body frame’s position and orientation $(p_x, p_y, \phi)$ from the leg lengths $(s_1, s_2, s_3)$ is not trivial. The following tangent half-angle substitution transforms the three equations in (7.2) into a system of polynomials in the newly defined scalar variable $t$:

$$
t = \tan \frac{\phi}{2}
$$

$$
\sin \phi = \frac{2t}{1 + t^2}
$$

$$
\cos \phi = \frac{1 - t^2}{1 + t^2}.
$$

After some algebraic manipulation, the system of polynomials (7.2) can eventually be reduced to a single sixth-order polynomial in $t$; this effectively shows
7.1. Inverse and Forward Kinematics

Figure 7.3: (a) The 3×RPR at a singular configuration. From this configuration, extending the legs may cause the platform to snap to a counterclockwise rotation or to a clockwise rotation. (b) Two solutions to the forward kinematics when all prismatic joint extensions are identical.

that the 3×RPR mechanism may have up to six forward kinematics solutions. Showing that all six mathematical solutions are physically realizable requires further verification.

Figure 7.3(a) shows the mechanism at a singular configuration, where each leg length is identical and as short as possible. This configuration is a singularity, because extending the legs from this symmetric configuration causes the platform to rotate either clockwise or counterclockwise; we cannot predict which.\(^1\) Singularities are covered in greater detail in Section 7.3. Figure 7.3(b) shows two solutions to the forward kinematics when all leg lengths are identical.

7.1.2 Stewart-Gough Platform

We now examine the inverse and forward kinematics of the 6×SPS Stewart-Gough platform of Figure 7.1(a). In this design, the fixed and moving platforms are connected by six serial SPS structures, with the spherical joints passive and the prismatic joints actuated. The derivation of the kinematic equations closely parallels that of the earlier 3×RPR planar mechanism. Let \{s\} and \{b\} denote the fixed and body frames, respectively, and let \(d_i\) be the vector directed from joint \(A_i\) to joint \(B_i, \ i = 1, \ldots, 6\). Referring to Figure 7.1(a), we make the following definitions:

\[
\begin{align*}
p &\in \mathbb{R}^3 = p \text{ in } \{s\}-\text{frame coordinates}; \\
a_i &\in \mathbb{R}^3 = a_i \text{ in } \{s\}-\text{frame coordinates}; \\
b_i &\in \mathbb{R}^3 = b_i \text{ in } \{b\}-\text{frame coordinates}; \\
d_i &\in \mathbb{R}^3 = d_i \text{ in } \{s\}-\text{frame coordinates}; \\
R &\in SO(3) = \text{orientation of } \{b\} \text{ as seen from } \{s\}.
\end{align*}
\]

In order to derive the kinematic constraint equations, note that vectorially,

\[
d_i = p + b_i - a_i, \ i = 1, \ldots, 6.
\]

\(^1\)A third possibility is that the extending legs crush the platform!
Writing the above equations explicitly in $\{s\}$-frame coordinates,

$$d_i = p + Rb_i - a_i, \quad i = 1, \ldots, 6.$$  

Denoting the length of leg $i$ by $s_i$, we have

$$s_i^2 = d_i^T d_i = (p + Rb_i - a_i)^T (p + Rb_i - a_i),$$

for $i = 1, \ldots, 6$. Note that $a_i$ and $b_i$ are all known constant vectors. Writing the constraint equations in this form, the inverse kinematics becomes straightforward: given $p$ and $R$, the six leg lengths $s_i, \ i = 1, \ldots, 6$ can be evaluated directly from the above equations (negative values of $s_i$ in most cases will not be physically realizable and can be ignored).

The forward kinematics is not as straightforward: given each of the leg lengths $s_i, \ i = 1, \ldots, 6$, we must solve for $p \in \mathbb{R}^3$ and $R \in SO(3)$. These six constraint equations, together with six further constraints imposed by the condition $R^T R = I$, constitute a set of twelve equations in twelve unknowns (three for $p$, nine for $R$).

### 7.1.3 General Parallel Mechanisms

For both the $3 \times RPR$ mechanism and Stewart-Gough Platform, we were able to exploit certain features of the mechanism that resulted in a reduced set of equations; for example, the fact that the legs of the Stewart-Gough Platform can be modelled as straight lines considerably simplified the analysis. In this section we briefly consider the case when the legs are general open chains.
Consider such a parallel mechanism as shown in Figure 7.4; here the fixed and moving platforms are connected by three open chains, and the configuration of the moving platform is given by $T_{sb}$. Denote the forward kinematics of the three chains by $T_1(\theta)$, $T_2(\phi)$, and $T_3(\psi)$, respectively, where $\theta \in \mathbb{R}^m$, $\phi \in \mathbb{R}^n$, and $\psi \in \mathbb{R}^p$. The loop-closure conditions can be written $T_{sb} = T_1(\theta) = T_2(\phi) = T_3(\psi)$. Eliminating $T_{sb}$ we get
\begin{align}
T_1(\theta) &= T_2(\phi) \\
T_2(\phi) &= T_3(\psi).
\end{align}

Equations (7.3) and (7.4) each consist of twelve equations (nine for the rotation component and three for the position component), six of which are independent (from the rotation matrix constraint $R^T R = I$, the nine equations for the rotation component can be reduced to a set of three independent equations). Thus there are 24 equations, twelve of which are independent, with $n + m + p$ unknown variables. The mechanism therefore has $d = n + m + p - 12$ degrees of freedom.

In the forward kinematics problem, given values for $d$ of the joint variables $(\theta, \phi, \psi)$, Equations (7.3) and (7.4) can be solved for the remaining joint variables. Generally this is not trivial and multiple solutions are likely. Once the joint values for any one of the open chain legs are known, the forward kinematics of that leg can then be evaluated to determine the forward kinematics of the closed chain.

In the inverse kinematics problem, given the body frame displacement $T_{sb} \in SE(3)$, we set $T = T_1 = T_2 = T_3$ and solve Equations (7.3) and (7.4) for the joint variables $(\theta, \phi, \psi)$. As suggested by the case studies, for most parallel mechanisms there are often features of the mechanism that can be exploited to eliminate some of these equations and simplify them to a more computationally amenable form.

### 7.2 Differential Kinematics

We now consider the differential kinematics of parallel mechanisms. Unlike the case for open chains, in which the objective is to relate the input joint velocities to the twist of the end-effector frame, the analysis for closed chains is complicated by the fact that not all of the joints are actuated. Only the actuated joints can be prescribed input velocities; the velocities of the remaining passive joints must then be determined from the kinematic constraint equations. These passive joint velocities are usually required to eventually determine the twist of the closed chain’s end-effector frame.

For open chains, the Jacobian of the forward kinematics is central to velocity and static analysis. For closed chains, in addition to the forward kinematics Jacobian, the Jacobian defined by the kinematic constraint equations—we will call this the **constraint Jacobian**—also plays a central role in velocity and static analysis. Usually there are features of the mechanism that can be exploited to simplify and reduce the procedure for obtaining the two Jacobians.
We illustrate with a case study of the Stewart-Gough platform, and show that the Jacobian of the inverse kinematics can be obtained straightforwardly via static analysis. Velocity analysis for more general parallel mechanisms is then detailed.

### 7.2.1 Stewart-Gough Platform

Earlier we saw that the inverse kinematics for the Stewart-Gough platform can be solved analytically. That is, given the body frame orientation \( R \in SO(3) \) and position \( p \in \mathbb{R}^3 \), the leg lengths \( s \in \mathbb{R}^6 \) can be obtained analytically in the functional form \( s = g(R, p) \). In principle one could differentiate this equation and manipulate it into the form

\[
\dot{s} = G(R, p)\mathcal{V}_s, \tag{7.5}
\]

where \( \dot{s} \in \mathbb{R}^6 \) denotes the leg velocities, \( \mathcal{V}_s \in \mathbb{R}^6 \) is the body frame’s spatial twist, and \( G(R, p) \in \mathbb{R}^{6 \times 6} \) is the Jacobian of the inverse kinematics. In most cases this procedure will require considerable algebraic manipulation.

Here we take a different approach based on static analysis. Based on the same conservation of power principles that were used to determine the static relationship \( \tau = J^T \mathcal{F} \) for open chains, the static relationship for closed chains can also be expressed in exactly the same form. We illustrate with an analysis of the Stewart-Gough platform.

In the absence of external forces, the only forces applied to the moving platform occur at the spherical joints. In what follows, all vectors are expressed in \( \{s\} \)-frame coordinates. Let

\[ f_i = \hat{n}_i \tau_i \]

be the three-dimensional linear force applied by leg \( i \), where \( \hat{n}_i \in \mathbb{R}^3 \) is a unit vector indicating the direction of the applied force, and \( \tau_i \in \mathbb{R} \) is the magnitude of the linear force. The moment generated \( m_i \) generated by \( f_i \) is

\[ m_i = r_i \times f_i, \]

where \( r_i \in \mathbb{R}^3 \) denotes the vector from the \( \{s\} \)-frame origin to the point of application of the force (the location of spherical joint \( i \) in this case). Since neither the spherical joint at the moving platform nor the spherical joint at the fixed platform can resist any torques about the joints, the force \( f_i \) must be along the line of the leg. Therefore, instead of calculating the moment \( m_i \) using the spherical joint at the moving platform, we can calculate the moment using the spherical joint at the fixed platform:

\[ m_i = q_i \times f_i, \]

where \( q_i \in \mathbb{R}^3 \) denotes the vector from the fixed-frame origin to the base joint of leg \( i \). Since \( q_i \) is constant, expressing the moment as \( q_i \times f_i \) is preferred.
7.2. Differential Kinematics

Combining \( f_i \) and \( m_i \) into the six-dimensional wrench \( \mathcal{F}_i = (m_i, f_i) \), the resultant wrench \( \mathcal{F}_s \) on the moving platform is given by

\[
\mathcal{F}_s = \sum_{i=1}^{6} \mathcal{F}_i = \sum_{i=1}^{6} \left[ \frac{r_i \times \hat{n}_i}{\hat{n}_i} \right] \tau_i
\]

\[
= \left[ \begin{array}{cccc} -\hat{n}_1 \times q_1 & \cdots & -\hat{n}_6 \times q_6 \\ \frac{\hat{n}_1}{\hat{n}_1} & \cdots & \frac{\hat{n}_6}{\hat{n}_6} \end{array} \right] \left[ \begin{array}{c} \tau_1 \\ \vdots \\ \tau_6 \end{array} \right]
\]

\[
= J_s^{-T} \tau,
\]

where \( J_s \) is the spatial Jacobian of the forward kinematics, with its inverse given by

\[
J_s^{-1} = \left[ \begin{array}{cccc} -\hat{n}_1 \times q_1 & \cdots & -\hat{n}_6 \times q_6 \\ \frac{\hat{n}_1}{\hat{n}_1} & \cdots & \frac{\hat{n}_6}{\hat{n}_6} \end{array} \right]^T.
\]

7.2.2 General Parallel Mechanisms

Because of its kinematic structure, the Stewart-Gough platform lends itself particularly well to a static analysis, as each of the six joint forces are directed along their respective legs. The Jacobian (or more precisely, the inverse Jacobian) can therefore be derived in terms of the screws associated with each line. In this section we consider more general parallel mechanisms where the static analysis is less straightforward. Using the previous three-legged spatial parallel mechanism of Figure 7.4 as an illustrative example, we derive a general procedure for determining the forward kinematics Jacobian that can also be generalized to parallel mechanisms.

The mechanism of Figure 7.4 consists of two platforms connected by three legs with \( m, n, \) and \( p \) joints, respectively. To keep things simple, we assume \( m = n = p = 5 \), so the mechanism has \( d = n + m + p - 12 = 15 \) degrees of freedom (generalizing what follows to different types and numbers of legs is completely straightforward). For the fixed and body frames indicated in the figure, write the forward kinematics for the three chains as follows:

\[
T_1(\theta_1, \theta_2, \ldots, \theta_5) = e^{[S_1]_{\theta_1} e^{[S_2]_{\theta_2}} \cdots e^{[S_5]_{\theta_5}}} M_1
\]

\[
T_2(\phi_1, \phi_2, \ldots, \phi_5) = e^{[P_1]_{\phi_1} e^{[P_2]_{\phi_2}} \cdots e^{[P_5]_{\phi_5}}} M_2
\]

\[
T_3(\psi_1, \psi_2, \ldots, \psi_5) = e^{[Q_1]_{\psi_1} e^{[Q_2]_{\psi_2}} \cdots e^{[Q_5]_{\psi_5}}} M_3.
\]

The kinematic loop constraints can be expressed as

\[
T_1(\theta) = T_2(\phi), \quad T_2(\phi) = T_3(\psi).
\]

Since these constraints must be satisfied at all times, we can express their time derivatives in terms of their spatial twists, i.e.,

\[
\dot{T}_1 T_1^{-1} = \dot{T}_2 T_2^{-1}, \quad \dot{T}_2 T_2^{-1} = \dot{T}_3 T_3^{-1}.
\]
Since $\dot{T}_1T_i^{-1} = [V_i]$, where $V_i$ is the spatial twist of chain $i$’s end-effector frame, the above identities can also be expressed in terms of the forward kinematics Jacobian for each chain:

$$J_1(\theta)\dot{\theta} = J_2(\phi)\dot{\phi}$$  \hspace{1cm} (7.10)

$$J_2(\phi)\dot{\phi} = J_3(\psi)\dot{\psi},$$  \hspace{1cm} (7.11)

which can be rearranged as

$$\begin{bmatrix}
J_1(\theta) & -J_2(\phi) & 0 \\
0 & -J_2(\phi) & J_3(\psi)
\end{bmatrix}
\begin{bmatrix}
\dot{\theta} \\
\dot{\phi} \\
\dot{\psi}
\end{bmatrix} = 0.$$  \hspace{1cm} (7.12)

Now rearrange the fifteen joints into those that are actuated and those that are passive. Assume without loss of generality that the three actuated joints are $(\theta_1, \phi_1, \psi_1)$. Define the vector of actuated joints $q_a \in \mathbb{R}^3$ and passive joints $q_p \in \mathbb{R}^{12}$ as

$$q_a = \begin{bmatrix}
\theta_1 \\
\phi_1 \\
\psi_1
\end{bmatrix}, \quad q_p = \begin{bmatrix}
\theta_2 \\
\vdots \\
\phi_5
\end{bmatrix},$$

and $q = (q_a, q_p) \in \mathbb{R}^{15}$. Equation (7.12) can now be rearranged into the form

$$\begin{bmatrix}
H_a(q) & H_p(q)
\end{bmatrix}
\begin{bmatrix}
\dot{q}_a \\
\dot{q}_p
\end{bmatrix} = 0,$$  \hspace{1cm} (7.13)

or equivalently

$$H_a\dot{q}_a + H_p\dot{q}_p = 0,$$  \hspace{1cm} (7.14)

where $H_a \in \mathbb{R}^{12 \times 3}$ and $H_p \in \mathbb{R}^{12 \times 12}$. If $H_p$ is invertible, we have

$$\dot{q}_p = -H_p^{-1}H_a\dot{q}_a.$$  \hspace{1cm} (7.15)

Assuming $H_p$ is invertible, once the velocities of the actuated joints are given, the velocities of the remaining passive joints can be obtained uniquely via Equation (7.15).

It still remains to derive the forward kinematics Jacobian with respect to the actuated joints, i.e., to find $J_a(q) \in \mathbb{R}^{6 \times 3}$ satisfying $V_s = J_a(q)\dot{q}_a$, where $V_s$ is the spatial twist of the end-effector frame. For this purpose we can use the forward kinematics for any of the three open chains; for example, in terms of chain 1, $J_1(\theta)\dot{\theta} = V_s$, and from Equation (7.15) we can write

$$\begin{align*}
\dot{\theta}_2 &= g_2^T \dot{q}_a \\
\dot{\theta}_3 &= g_3^T \dot{q}_a \\
\dot{\theta}_4 &= g_4^T \dot{q}_a \\
\dot{\theta}_5 &= g_5^T \dot{q}_a
\end{align*}$$

$$\begin{align*}
\dot{\theta}_2 &= g_2^T \dot{q}_a \\
\dot{\theta}_3 &= g_3^T \dot{q}_a \\
\dot{\theta}_4 &= g_4^T \dot{q}_a \\
\dot{\theta}_5 &= g_5^T \dot{q}_a
\end{align*}$$

\hspace{1cm} (7.16) \hspace{1cm} (7.17) \hspace{1cm} (7.18) \hspace{1cm} (7.19)
7.3. Singularities

where each \( g_i(q) \in \mathbb{R}^3, \ i = 2, \ldots, 5, \) can be obtained from Equation (7.15). Defining the row vector \( e_1^T = (1, 0, 0) \), the differential forward kinematics for chain 1 can now be written

\[
V_s = J_1(\theta) \begin{bmatrix} e_1^T \\ g_2^T \\ g_3^T \\ g_4^T \\ g_5^T \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \phi_1 \\ \psi_1 \end{bmatrix}.
\]

(7.20)

Since we are seeking \( J_a(q) \) in \( V_s = J_a(q)\dot{q}_a \), and \( \dot{q}_a^T = (\dot{\theta}_1, \dot{\phi}_1, \dot{\psi}_1) \), from the above it now follows that

\[
J_a(q) = J_1(q_1, \ldots, q_5) \begin{bmatrix} e_1^T \\ g_2(q)^T \\ g_3(q)^T \\ g_4(q)^T \\ g_5(q)^T \end{bmatrix}.
\]

(7.21)

The above could also have been derived using either chain 2 or chain 3.

Given values for the actuated joints \( q_a \), it still remains to solve for the passive joints \( q_p \) from the loop constraint equations. Eliminating as many elements of \( q_p \) in advance will obviously simplify matters. The second point to note is that \( H_p(q) \) may become singular, in which case \( \dot{q}_p \) cannot be obtained from \( \dot{q}_a \). Configurations in which \( H_p(q) \) becomes singular correspond to actuator singularities, which are discussed in the next section.

7.3 Singularities

Characterizing the singularities of closed chains involves many more subtleties than for open chains. In this section we highlight the essential features of closed-chain singularities via two planar examples: a four-bar linkage (see Figure 7.5), and a five-bar linkage (see Figure 7.6). Based on these examples we classify closed chain singularities into three basic types: actuator singularities, configuration space singularities, and end-effector singularities.

We begin with the four-bar linkage of Figure 7.5. Recall from Chapter 2 that its C-space is a one-dimensional curve embedded in a four-dimensional ambient space (each dimension is parametrized by one of the four joints). Projecting the C-space onto the joint angles \((\theta, \phi)\) leads to the curve shown in Figure 7.5. In terms of \( \theta \) and \( \phi \), the kinematic loop constraint equations for the four-bar can be expressed as

\[
\phi = \tan^{-1} \left( \frac{\beta}{\alpha} \right) \pm \cos^{-1} \left( \frac{\gamma}{\sqrt{\alpha^2 + \beta^2}} \right),
\]

(7.22)
where
\[ \alpha = 2L_3L_4 - 2L_1L_3 \cos \theta \] (7.23)
\[ \beta = -2L_1L_3 \sin \theta \] (7.24)
\[ \gamma = L_2^2 - L_4^2 - L_3^2 - L_1^2 + 2L_1L_4 \cos \theta. \] (7.25)

The existence and uniqueness of solutions to the above depends on the link lengths \( L_1, \ldots, L_4 \). In particular, a solution will fail to exist if \( \gamma^2 \leq \alpha^2 + \beta^2 \).

The figure depicts the input-output graph for the choice of link lengths \( L_1 = 4 \), \( L_2 = 4 \), \( L_3 = 3 \), \( L_4 = 2 \). For this set of link lengths, \( \theta \) and \( \phi \) both range from 0 to \( 2\pi \).

One of the distinctive features of this graph is the bifurcation point \( P \)
as indicated in the figure. Here two branches of the curve meet, resulting in a self-intersection of the curve with itself. If the four-bar is in the configuration indicated by $P$, it has the choice of following either branch. At no other point in the four-bar’s $C$-space does such branching occur.

We now turn to the five-bar linkage. The kinematic loop constraint equations can be written

$$L_1 \cos \theta_1 + \ldots + L_4 \cos(\theta_1 + \theta_2 + \theta_3 + \theta_4) = L_5 \quad (7.26)$$

$$L_1 \sin \theta_1 + \ldots + L_4 \sin(\theta_1 + \theta_2 + \theta_3 + \theta_4) = 0 \quad (7.27)$$

where we have eliminated in advance the joint variable $\theta_5$ from the loop closure conditions. Writing these two equations in the form $f(\theta_1, \ldots, \theta_4) = 0$, where $f : \mathbb{R}^4 \to \mathbb{R}^2$, the configuration space can be regarded as a two-dimensional surface in $\mathbb{R}^4$. Like the bifurcation point of the four-bar linkage, self-intersections of the surface can also occur. At such points the constraint Jacobian loses rank. For the five-bar, any point $\theta$ at which

$$\text{rank} \left( \frac{\partial f}{\partial \theta} (\theta) \right) < 2 \quad (7.28)$$

corresponds to what we call a **configuration space singularity**. Figure 7.7 illustrates the possible configuration space singularities of the five-bar. Notice that so far we have made no mention of which joints of the five-bar are actuated, or where the end-effector frame is placed. The notion of a configuration space singularity is completely independent of the choice of actuated joints, or where the end-effector frame is placed.

We now consider the case when two joints of the five-bar are actuated. Referring to Figure 7.8, the actuated joints are indicated by filled circles. Under normal operating conditions, the motions of the actuated joints can be independently controlled. Alternatively, locking the actuated joints should immobilize the five-bar and turn it into a rigid structure.

For the **nondegenerate actuator singularity** shown on the left of Figure 7.8, rotating the two actuated joints oppositely and outward will pull the mechanism apart, and rotating them oppositely and inward would either crush
Figure 7.8: Actuator singularities of the planar five-bar linkage: the left is nondegenerate, while the right is degenerate.

the inner two links, or cause the center joint to unpredictably buckle upward or downward. For the degenerate actuator singularity shown on the right, even when the actuated joints are locked in place, the inner two links are free to rotate.

The reason for classifying these singularities as actuator singularities is that, by relocating the actuators to a different set of joints, such singularities can be eliminated. For both the degenerate and nondegenerate actuator singularities of the five-bar, relocating one of the actuators to one of the three passive joints eliminates the singularity.

Visualizing the actuator singularities of the planar five-bar is straightforward enough, but for more complex spatial closed chains this may be difficult. Actuator singularities can be characterized mathematically by the rank of the constraint Jacobian. As before, write the kinematic loop constraints in differential form,

\[ H(q)\dot{q} = \begin{bmatrix} H_a(q) & H_p(q) \end{bmatrix} \begin{bmatrix} \dot{q}_a \\ \dot{q}_p \end{bmatrix} = 0, \]

(7.29)

where \( q_a \in \mathbb{R}^a \) is the vector of actuated joints, and \( q_p \in \mathbb{R}^p \) is the vector of passive joints. It follows that \( H(q) \in \mathbb{R}^{p \times (a+p)} \) and that \( H_p(q) \) is a \( p \times p \) matrix.

With the above definitions, we have the following:

- If rank \( H_p(q) < p \), then \( q \) is an actuator singularity. Distinguishing between degenerate and nondegenerate singularities involves additional mathematical subtleties, and relies on second-order derivative information that we do not pursue further here.

- If rank \( H(q) < p \), then \( q \) is a configuration space singularity. Note that under this condition \( H_p(q) \) is also singular (the converse is not true, however). The configuration space singularities can therefore be regarded as the intersection of all possible actuator singularities obtained over all possible combinations of actuated joints.
7.4. Summary

The final class of singularities depends on the choice of end-effector frame. For the five-bar, ignore the orientation of the end-effector frame, and focus exclusively on its $x$-$y$ location. Figure 7.9 shows the five-bar in an end-effector singularity for the given choice of end-effector location. Note that velocities along the indicated line are not possible in this configuration, similar to the case for singularities for open chains. To see this, consider the effective 2R open chain created by the rightmost joint, the link connecting it to the platform, the joint on the platform, and the effective link connecting the platform joint to the end-effector frame. Since the two links of the 2R robot are aligned, the end-effector frame can have no component of motion along the direction of the links.

End-effector singularities are independent of the choice of actuated joints. They can be mathematically characterized as follows. Choose any valid set of actuated joints $q_a$ such that the mechanism is not at an actuator singularity. Write the forward kinematics in the form

$$f(q_a) = T_{sb}. \quad (7.30)$$

One can then check for rank deficiencies in the Jacobian of $f$, as was done for open chains, to determine the presence of an end-effector singularity.

7.4 Summary

- Any kinematic chain that contains one or more loops is called a closed chain. Parallel mechanisms are a class of closed chain that are characterized by two platforms—one moving and one stationary—connected by several legs; the legs are typically open chains, but can themselves be closed chains. Compared to open chains, the kinematic analysis of closed chains is complicated by the fact that only a subset of joints are actuated, and the joint variables must satisfy a number of loop-closure constraint equations, which may or may not be independent depending on the configuration of the mechanism.
• For a parallel mechanism with equal numbers of actuators and degrees of freedom, the inverse kinematics problem involves finding, from the given position and orientation of the moving platform, the joint coordinates of all the actuated joints. For well-known parallel mechanisms like the planar $3 \times RPR$ and spatial Stewart-Gough platform, the inverse kinematics admits unique solutions.

• For a parallel mechanism with equal numbers of actuators and degrees of freedom, the forward kinematics problem involves finding the position and orientation of the moving platform given coordinates for all the actuated joints. For well-known parallel mechanisms like the $3 \times RPR$ and the spatial Stewart-Gough platform, the forward kinematics usually admits multiple solutions. In the case of the most general Stewart-Gough platform, a maximum of 40 solutions are possible.

• The differential kinematics of a closed chain relates velocities of the actuated joints to the linear and angular velocities of the moving platform. For a closed chain consisting of $n$ one-dof joints, with $m$ actuators and degrees of freedom, let $\theta_a \in \mathbb{R}^m$ denote the vector of actuated joints and $\theta_p \in \mathbb{R}^{n-m}$ denote the vector of passive joints. The kinematic loop-closure constraints are described by an equation of the form $h(\theta_a, \theta_p) = 0$, where $h: \mathbb{R}^n \to \mathbb{R}^{n-m}$. The forward kinematics can be expressed in the form $f(\theta_a) = T$, where $f: \mathbb{R}^m \to SE(3)$. The differential kinematics involves derivatives of both $f$ and $g$ with respect to $\theta_a$ and $\theta_p$. For platforms like the Stewart-Gough platform, the differential kinematics can also be obtained from a static analysis, by exploiting the fact that just as for open chains, the external forces $F$ at the end-effector are related to the joint forces or torques $\tau$ by $\tau = J^T F$.

• Singularities for closed chains can be classified into three types: (i) configuration space singularities at self-intersections of the configuration space surface (also called bifurcation points for one-dimensional configuration spaces); (ii) nondegenerate actuator singularities when the actuated joints cannot be independently actuated, and degenerate actuator singularities when locking all joints fails to make the mechanism a rigid structure; and (iii) end-effector singularities when the end-effector loses one or more degrees of freedom of motion. Configuration space singularities are independent of the choice of actuated joints, while actuator singularities depend on which joints are actuated. End-effector singularities depend on the placement of the end-effector frame, but do not depend on the choice of actuated joints.

7.5 Notes and References

Several methods exist for finding all solutions to systems polynomial equations, e.g., methods based on dialytic elimination, Gröbner bases, etc. Of particular
note is the work of Raghavan and Roth [101], who show that there can be at most forty solutions to the forward kinematics of the general 6-6 Stewart-Gough Platform, and Husty [44], who develops a computational algorithm to find all forty solutions.
7.6 Exercises

![3×RPR planar parallel mechanism](image)

**Figure 7.10: 3×RPR planar parallel mechanism.**

1. In the 3×RPR planar parallel mechanism of Figure 7.10, the prismatic joints are actuated. Define $a_i \in \mathbb{R}^2$ to be the vector from the fixed frame origin $O$ to joint $A_i$, $i = 1, 2, 3$, expressed in fixed frame coordinates. Define $b_i \in \mathbb{R}^2$ to be the vector from the moving platform frame origin $P$ to joint $B_i$, $i = 1, 2, 3$, defined in terms of the moving platform frame coordinates.

   (a) Solve the inverse kinematics.
   (b) Derive a procedure to solve the forward kinematics.
   (c) Is the configuration shown an end-effector singularity? Explain your answer by examining the inverse kinematics Jacobian. Is this also an actuator singularity?

2. For the 3×RPR planar parallel mechanism shown in Figure 7.11(a), let $\phi$ be the angle measured from the $\{b\}$-frame $\hat{x}$-axis to the $\{s\}$-frame $\hat{x}$-axis, and $p \in \mathbb{R}^2$ be the vector from the $\{s\}$-frame origin to the $\{b\}$-frame origin, expressed in $\{s\}$-frame coordinates. Let $a_i \in \mathbb{R}^2$ be the vector from the $\{s\}$-frame origin to the three joints fixed to ground, $i = 1, 2, 3$ (note that two of the joints are overlapping), expressed in $\{s\}$-frame coordinates. Let $b_i \in \mathbb{R}^2$ be the vector from the $\{b\}$-frame origin to the three joints attached to the moving platform, $i = 1, 2, 3$ (note that two of the joints are overlapping), expressed in $\{b\}$-frame coordinates. Denote the leg lengths by $\theta_1$, $\theta_2$, $\theta_3$ as shown.

   (a) Derive a set of independent equations relating $(\phi, p)$ and $(\theta_1, \theta_2, \theta_3)$.
   (b) What is the maximum possible number of forward kinematics solutions?
   (c) Assuming static equilibrium, given joint torques $\tau = (1, 0, -1)$ applied at joints $(\theta_1, \theta_2, \theta_3)$, find the force $(f_x, f_y)$ and the moment $m_z \in \mathbb{R}$ generated at
7.6. Exercises

235

the end-effector frame \{b\}. Express your \((f_x, f_y)\) in fixed frame coordinates.

(d) Now construct a mechanism with three connected 3×RPR parallel mechanisms as shown in Figure 7.11(b). What is the dof of this mechanism?

\[
\begin{align*}
\theta_1 & \quad \theta_2 & \quad \theta_3 \\
x & \quad \psi & \quad \phi
\end{align*}
\]

\[
\begin{align*}
\theta_1 & \quad \theta_2 & \quad \theta_3 \\
x & \quad \psi & \quad \phi
\end{align*}
\]

Figure 7.11: 3XRPR planar parallel mechanism and truss structure.

3. For the 3×RRR planar parallel mechanism shown in Figure 7.12, let \(\phi\) be the orientation of the end-effector frame and \(p \in \mathbb{R}^2\) be the vector \(\vec{p}\) expressed in fixed frame coordinates. Let \(a_i \in \mathbb{R}^2\) be the vector \(\vec{a}_i\) expressed in fixed frame coordinates and \(b_i \in \mathbb{R}^2\) be the vector \(\vec{b}_i\) expressed in the moving body frame coordinates.

(a) Derive a set of independent equations relating \((\phi, p)\) and \((\theta_1, \theta_2, \theta_3)\).

(b) What is the maximum possible number of inverse and forward kinematic solutions for this mechanism?

4. Figure 7.13 shows a six-bar linkage in its zero position. Let \((p_x, p_y)\) be the position of the \{b\}-frame origin expressed in \{s\}-frame coordinates, and \(\phi\) be the orientation of the \{b\} frame. The inverse kinematics problem is defined as finding the joint variables \((\theta, \psi)\) given \((p_x, p_y, \phi)\).

(a) In order to solve the inverse kinematics problem, how many equations are needed? Derive these equations.
(b) Assume joints A, D, and E are actuated. Determine if the configuration shown in Figure 7.13 is an actuator singularity by analyzing an equation of the form
\[
\begin{bmatrix} H_a & H_p \end{bmatrix} \begin{bmatrix} q_a \\ q_p \end{bmatrix} = 0,
\]
where \( q_a \) is the vector of actuated joints and \( q_p \) is the vector of passive joints.

(c) Suppose instead that joints A, B, and D are actuated. Find the forward kinematics Jacobian \( J_a \) in \( V_s = J_a \dot{q}_a \), where \( V_s \) is the velocity twist of the \{b\}-frame expressed in \{s\}-frame coordinates, and \( \dot{q}_a \) is the vector of actuated joint rates.

5. Consider the 3×PSP spatial parallel mechanism of Figure 7.14.
(a) What is the degrees of freedom of this mechanism?
(b) Let \( R_{sb} = \text{Rot}(\hat{z}, \theta) \text{Rot}(\hat{y}, \phi) \text{Rot}(\hat{x}, \psi) \) be the orientation of the moving body frame \( \{b\} \), and \( p_{sb} = (x, y, z) \in \mathbb{R}^3 \) be the vector from the \{s\}-frame origin to the \{b\}-frame origin (both \( R_{sb} \) and \( p_{sb} \) are expressed in \{s\}-frame coordinates).
The vectors \( \vec{a}_i, \vec{b}_i, \vec{d}_i, i = 1, 2, 3 \), are defined as shown in the figure. Derive a set of independent kinematic constraint equations relating \( (\theta, \phi, \psi, x, y, z) \) and the vectors defined.
(c) Given values for \( (x, y, z) \), is it possible to solve for the vertical prismatic joint values \( s_i \), where \( s_i = ||\vec{d}_i||, i = 1, 2, 3 \)? If so, derive an algorithm for doing so.
6. The Eclipse mechanism of Figure 7.15 is a six-dof parallel mechanism whose moving platform is capable of $\pm 90^\circ$ orientations with respect to ground, and also of rotating $360^\circ$ about the vertical axis.

(a) Derive the forward and inverse kinematics. How many forward kinematic solutions are there for general nonsingular configurations?
(b) Find and classify all singularities of this mechanism.

7. For the Delta robot of Figure 2.8, derive the following:
(a) The forward kinematics.
(b) The inverse kinematics.
(c) The Jacobian $J_a$ (assume the revolute joints at the fixed base are actuated).
(d) Identify all actuator singularities of the Delta robot.
8. In the 3×UPU platform of Figure 7.16, the axes of the universal joints are attached to the fixed and moving platforms in the sequence indicated, i.e., the first axis is attached orthogonally to the fixed base, while the fourth axis is attached orthogonally to the moving platform. Derive the following:
(a) The forward kinematics.
(b) The inverse kinematics.
(c) The Jacobian $J_a$ (assume the revolute joints at the fixed base are actuated).
(d) Identify all actuator singularities of this robot.
(e) If you can, build a mechanical prototype and see if the mechanism behaves as predicted by your analysis.
Chapter 8

Dynamics of Open Chains

In this chapter we study once again the motion of open-chain robots, but this time taking into account the forces and torques that cause it; this is the subject of robot dynamics. The associated dynamic equations—also referred to as the equations of motion—are a set of second-order differential equations of the form

\[ \tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}), \]  

(8.1)

where \( \theta \in \mathbb{R}^n \) is the vector of joint variables, \( \tau \in \mathbb{R}^n \) is the vector of joint forces and torques, \( M(\theta) \in \mathbb{R}^{n \times n} \) is a symmetric positive-definite matrix called the mass matrix, and \( h(\theta, \dot{\theta}) \in \mathbb{R}^n \) are forces that lump together centripetal and Coriolis, gravity, and friction terms that depend on \( \theta \) and \( \dot{\theta} \). One should not be deceived by the apparent simplicity of these equations; even for “simple” open chains, e.g., those with joint axes either orthogonal or parallel to each other, \( M(\theta) \) and \( h(\theta, \dot{\theta}) \) can be extraordinarily complex.

Just as a distinction was made between a robot’s forward and inverse kinematics, it is also customary to distinguish between a robot’s forward and inverse dynamics. The forward problem is the problem of determining the robot’s acceleration \( \ddot{\theta} \) given the state \( (\theta, \dot{\theta}) \) and the joint forces and torques,

\[ \ddot{\theta} = M^{-1}(\theta) \left( \tau - h(\theta, \dot{\theta}) \right), \]  

(8.2)

and the inverse problem is finding the joint forces and torques \( \tau \) corresponding to the robot’s state and a desired acceleration, i.e., Equation (8.1).

A robot’s dynamic equations are typically derived in one of two ways: by a direct application of Newton and Euler’s dynamic equations for a rigid body (often called the Newton-Euler formulation), or by the Lagrangian dynamics formulation deriving from the kinetic and potential energy of the robot. The Lagrangian formalism is conceptually elegant and quite effective for robots with simple structures, e.g., with three or fewer degrees of freedom. However, the calculations can quickly become cumbersome for robots with more degrees of freedom. For general open chains, the Newton-Euler formulation leads to efficient recursive algorithms for both the inverse and forward dynamics that can
also be assembled into closed-form analytic expressions for, e.g., the mass matrix \( M(\theta) \) and other terms in the dynamics equation (8.1). The Newton-Euler formulation also takes advantage of tools we have developed so far in this book.

In this chapter we study both the Lagrangian and Newton-Euler dynamics formulations for an open-chain robot.

**8.1 Lagrangian Formulation**

**8.1.1 Basic Concepts and Motivating Examples**

The first step in the Lagrangian formulation of dynamics is to choose a set of independent coordinates \( q \in \mathbb{R}^n \) that describes the system’s configuration, similar to what was done in the analysis of a robot’s configuration space. The coordinates \( q \) are called **generalized coordinates**. Once generalized coordinates have been chosen, these then define another set of coordinates \( f \in \mathbb{R}^n \) called **generalized forces**. The forces \( f \) and the coordinates \( q \) are dual to each other in the sense that the inner product \( f^T \dot{q} \) corresponds to power. A Lagrangian function \( \mathcal{L}(q, \dot{q}) \) is then defined as the overall system’s kinetic energy \( K(q, \dot{q}) \) minus the potential energy \( \mathcal{P}(q) \). The equations of motion can now be expressed in terms of the Lagrangian as follows:

\[
\dot{f} = \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q}, \tag{8.3}
\]

These equations are also referred to as the **Euler-Lagrange equations with external forces**. The derivation can be found in dynamics texts.

We illustrate the Lagrangian dynamics formulation through two examples. In the first example, consider a particle of mass \( m \) constrained to move on a vertical line. The particle’s configuration space is this vertical line, and a natural choice for generalized coordinate is the height of the particle, which we denote by the scalar variable \( x \in \mathbb{R} \). Suppose the gravitational force \( mg \) acts downward, and an external force \( f \) is applied upward. By Newton’s second law, the equation of motion for the particle is

\[
f - mg = m\ddot{x}. \tag{8.4}\]

We now apply the Lagrangian formalism to derive the same result. The kinetic energy is \( \frac{1}{2} m \dot{x}^2 \), the potential energy is \( mgx \), and the Lagrangian is

\[
\mathcal{L}(x, \dot{x}) = K(x, \dot{x}) - \mathcal{P}(x) = \frac{1}{2} m \dot{x}^2 - mgx. \tag{8.5}
\]

The equation of motion is then given by

\[
f = \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{x}} - \frac{\partial \mathcal{L}}{\partial x} = m\ddot{x} + mg, \tag{8.6}
\]

which matches Equation (8.4).

\[^1\text{The external force } f \text{ is zero in the standard form of the Euler-Lagrange equations.}\]
8.1. Lagrangian Formulation

Figure 8.1: (Left) A 2R open chain in gravity. (Right) At $\theta = (0, \pi/2)$.

We now derive the dynamic equations for a planar 2R open chain moving in the presence of gravity (Figure 8.1). The chain moves in the $x$-$y$ plane, with gravity $g$ acting in the $-y$ direction. Before the dynamics can be derived, the mass and inertial properties of all the links must be specified. To keep things simple the two links are modeled as point masses $m_1$ and $m_2$ concentrated at the ends of each link. The position and velocity of the mass of link 1 are then given by

$$
\begin{bmatrix}
x_1 \\
y_1
\end{bmatrix} =
\begin{bmatrix}
L_1 \cos \theta_1 \\
L_1 \sin \theta_1
\end{bmatrix}
\begin{bmatrix}
x_1' \\
y_1'
\end{bmatrix} =
\begin{bmatrix}
-L_1 \sin \theta_1 \\
L_1 \cos \theta_1
\end{bmatrix} \dot{\theta}_1,
$$

while those of the link 2 mass are given by

$$
\begin{bmatrix}
x_2 \\
y_2
\end{bmatrix} =
\begin{bmatrix}
L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) \\
L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2)
\end{bmatrix}
\begin{bmatrix}
x_2' \\
y_2'
\end{bmatrix} =
\begin{bmatrix}
-L_1 \sin \theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\
L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2)
\end{bmatrix}
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}.
$$

We choose the joint coordinates $\theta = (\theta_1, \theta_2)$ to be the generalized coordinates. The generalized forces $\tau = (\tau_1, \tau_2)$ then correspond to joint torques (since $\tau^T \dot{\theta}$ corresponds to power). The Lagrangian $\mathcal{L}(\theta, \dot{\theta})$ is of the form

$$
\mathcal{L}(\theta, \dot{\theta}) = \sum_{i=1}^{2} (\mathcal{K}_i - \mathcal{P}_i),
$$

where the link kinetic energy terms $\mathcal{K}_1$ and $\mathcal{K}_2$ are

$$
\mathcal{K}_1 = \frac{1}{2} m_1 (x_1'^2 + y_1'^2) = \frac{1}{2} m_1 L_1^2 \dot{\theta}_1^2
$$

$$
\mathcal{K}_2 = \frac{1}{2} m_2 (x_2'^2 + y_2'^2)
$$

$$
= \frac{m_2}{2} \left( (L_1^2 + 2L_1L_2 \cos \theta_2 + L_2^2) \dot{\theta}_1^2 + 2(L_2^2 + L_1L_2 \cos \theta_2) \dot{\theta}_1 \dot{\theta}_2 + L_2^2 \dot{\theta}_2^2 \right),
$$

with $L_1$ and $L_2$ being the lengths of the links.
and the link potential energy terms $\mathcal{P}_1$ and $\mathcal{P}_2$ are

$$\begin{align*}
\mathcal{P}_1 &= m_1 g y_1 = m_1 g L_1 \sin \theta_1 \\
\mathcal{P}_2 &= m_2 g y_2 = m_2 g (L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2)).
\end{align*}$$

The Euler-Lagrange equations (8.3) for this example are of the form

$$\tau_i = \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{\theta}_i} - \frac{\partial \mathcal{L}}{\partial \theta_i}, \quad i = 1, 2. \tag{8.8}$$

The dynamic equations for the 2R planar chain follow from explicit evaluation of the right-hand side of (8.8) (we omit the detailed calculations, which are straightforward but tedious):

$$\begin{align*}
\tau_1 &= (m_1 L_1^2 + m_2 (L_1^2 + 2L_1 L_2 \cos \theta_2 + L_2^2)) \ddot{\theta}_1 + m_2 (L_1 L_2 \cos \theta_2 + L_2^2) \ddot{\theta}_2 - m_2 L_1 L_2 \sin \theta_2 (2\dot{\theta}_1 \dot{\theta}_2 + \dot{\theta}_2^2) + (m_1 + m_2) L_1 g \cos \theta_1 + m_2 g L_2 \cos(\theta_1 + \theta_2) \\
\tau_2 &= m_2 (L_1 L_2 \cos \theta_2 + L_2^2) \ddot{\theta}_1 + m_2 L_2^2 \ddot{\theta}_2 + m_2 L_1 L_2 \dot{\theta}_1^2 \sin \theta_2 + m_2 g L_2 \cos(\theta_1 + \theta_2).
\end{align*}$$

We can gather terms together into an equation of the form

$$\tau = M(\theta) \ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta), \quad h(\theta, \dot{\theta}), \tag{8.9}$$

with

$$M(\theta) = \begin{bmatrix}
m_1 L_1^2 + m_2 (L_1^2 + 2L_1 L_2 \cos \theta_2 + L_2^2) & m_2 (L_1 L_2 \cos \theta_2 + L_2^2) \\
m_2 (L_1 L_2 \cos \theta_2 + L_2^2) & m_2 L_2^2
\end{bmatrix}$$

$$c(\theta, \dot{\theta}) = \begin{bmatrix}
-m_2 L_1 L_2 \sin \theta_2 (2\dot{\theta}_1 \dot{\theta}_2 + \dot{\theta}_2^2) \\
m_2 L_1 L_2 \dot{\theta}_1^2 \sin \theta_2
\end{bmatrix}$$

$$g(\theta) = \begin{bmatrix}
(m_1 + m_2) L_1 g \cos \theta_1 + m_2 g L_2 \cos(\theta_1 + \theta_2) \\
m_2 g L_2 \cos(\theta_1 + \theta_2)
\end{bmatrix},$$

where $M(\theta)$ is the symmetric positive-definite mass matrix, $c(\theta, \dot{\theta})$ is a vector of Coriolis and centripetal torques, and $g(\theta)$ is a vector of gravitational torques. These reveal that the equations of motion are linear in $\ddot{\theta}$, quadratic in $\dot{\theta}$, and trigonometric in $\theta$. This is true in general for serial chains containing revolute joints, not just the 2R robot.

The $M(\theta) \ddot{\theta} + c(\theta, \dot{\theta})$ portion of Equation (8.9) could have been derived by writing $f_i = m_i a_i$ for each point mass, where the accelerations $a_i$ are written in terms of $\theta$ by differentiating the expressions for $(\dot{x}_i, \dot{y}_i)$ and $(\ddot{x}_i, \ddot{y}_i)$ given...
8.1. Lagrangian Formulation

245

above:

\[
\begin{bmatrix}
    f_{x1} \\
    f_{y1} \\
    f_{z1}
\end{bmatrix} = m_1 \begin{bmatrix}
    \dot{x}_1 \\
    \dot{y}_1 \\
    \dot{z}_1
\end{bmatrix} = m_1 \begin{bmatrix}
    -L_1 \dot{\theta}_1^2 c_1 - L_1 \dot{\theta}_1 s_1 \\
    -L_1 \dot{\theta}_1^2 s_1 + L_1 \dot{\theta}_1 c_1 \\
    0
\end{bmatrix}
\]

\[
(8.10)
\]

\[
\begin{bmatrix}
    -L_1 \dot{\theta}_1^2 c_1 - L_2 (\dot{\theta}_1 + \dot{\theta}_2)^2 c_{12} - L_1 \dot{\theta}_1 s_1 - L_2 (\dot{\theta}_1 + \dot{\theta}_2) s_{12} \\
    -L_1 \dot{\theta}_1^2 s_1 - L_2 (\dot{\theta}_1 + \dot{\theta}_2)^2 s_{12} + L_1 \dot{\theta}_1 c_1 + L_2 (\dot{\theta}_1 + \dot{\theta}_2) c_{12}
\end{bmatrix}
\]

\[
\begin{bmatrix}
    0
\end{bmatrix},
\]

\[
(8.11)
\]

where \(s_{12}\) indicates \(\sin(\theta_1 + \theta_2)\), etc. Defining \(r_{11}\) as the vector from joint 1 to \(m_1\), \(r_{12}\) as the vector from joint 1 to \(m_2\), and \(r_{22}\) as the vector from joint 2 to \(m_2\), the moments in world-aligned frames \(\{i\}\) attached to joints 1 and 2 can be expressed as \(m_1 = r_{11} \times f_1 + r_{12} \times f_2\) and \(m_2 = r_{22} \times f_2\). (Note that joint 1 must provide torques to move both \(m_1\) and \(m_2\), since both masses are outboard of joint 1, but joint 2 only needs to provide torque to move \(m_2\).) The joint torques \(\tau_1\) and \(\tau_2\) are just the third elements of \(m_1\) and \(m_2\), i.e., the moments about the \(\hat{z}_i\) axes out of the page, respectively.

In \((x, y)\) coordinates, the accelerations of the masses are written simply as second time-derivatives of the coordinates, e.g., \((\dot{x}_2, \dot{y}_2)\). This is because the \(\hat{x}-\hat{y}\) frame is an inertial frame. The joint coordinates \((\theta_1, \theta_2)\) are not in an inertial frame, however, so accelerations are expressed as a sum of terms that are linear in the second derivatives of joint variables, \(\dot{\theta}\), and quadratic of the first derivatives of joint variables, \(\dot{\theta}^T \dot{\theta}\), as seen in Equations (8.10) and (8.11). Quadratic terms containing \(\dot{\theta}_i^2\) are called centripetal terms, and quadratic terms containing \(\dot{\theta}_i \dot{\theta}_j\) are called Coriolis terms. In other words, \(\dot{\theta} = 0\) does not mean zero acceleration of the masses, due to the centripetal and Coriolis terms.

To better understand the centripetal and Coriolis terms, consider the arm at the configuration \((\theta_1, \theta_2) = (0, \pi/2)\), i.e., \(\cos\theta_1 = \sin(\theta_1 + \theta_2) = 1\), \(\sin\theta_1 = \cos(\theta_1 + \theta_2) = 0\). Assuming \(\dot{\theta} = 0\), the acceleration \((\ddot{x}_2, \ddot{y}_2)\) of \(m_2\) from Equation (8.11) can be written

\[
\begin{bmatrix}
    \ddot{x}_2 \\
    \ddot{y}_2
\end{bmatrix} = \begin{bmatrix}
    -L_1 \dot{\theta}_1^2 \\
    -L_2 \dot{\theta}_1^2 - L_2 \dot{\theta}_2^2
\end{bmatrix} + \begin{bmatrix}
    0 \\
    -2L_2 \dot{\theta}_1 \dot{\theta}_2
\end{bmatrix}.
\]

Figure 8.2 shows the centripetal acceleration \(a_{\text{cent1}} = (-L_1 \dot{\theta}_1^2, -L_2 \dot{\theta}_2^2)\) when \(\theta_2 = 0\); the centripetal acceleration \(a_{\text{cent2}} = (0, -L_2 \dot{\theta}_2^2)\) when \(\theta_1 = 0\); and the Coriolis acceleration \(a_{\text{cor}} = (0, -2L_2 \dot{\theta}_1 \dot{\theta}_2)\) when both \(\dot{\theta}_1\) and \(\dot{\theta}_2\) are positive. As illustrated in Figure 8.2, each of the centripetal accelerations \(a_{\text{centi}}\) pulls \(m_2\) toward joint \(i\) to keep \(m_2\) rotating about about the center of the circle defined by joint \(i\).\(^2\) Therefore \(a_{\text{centi}}\) creates zero torque about joint \(i\). The Coriolis acceleration \(a_{\text{cor}}\) in this example passes through joint 2, so it creates zero torque about joint 2, but it creates negative torque about joint 1. Intuitively, the

\(^2\) Without this centripetal acceleration, and therefore centripetal force, the mass \(m_2\) would fly off along a tangent to the circle.
Figure 8.2: Accelerations of \( m_2 \) when \( \theta = (0, \pi/2) \) and \( \ddot{\theta} = 0 \). (Left) The centripetal acceleration \( a_{\text{cent}1} = (-L_1\dot{\theta}_1^2, -L_2\dot{\theta}_1^2) \) of \( m_2 \) when \( \dot{\theta}_2 = 0 \). (Middle) The centripetal acceleration \( a_{\text{cent}2} = (0, -L_2\dot{\theta}_2^2) \) of \( m_2 \) when \( \dot{\theta}_1 = 0 \). (Right) When both joints are rotating with \( \dot{\theta}_i > 0 \), the acceleration is the vector sum of \( a_{\text{cent}1} \) and \( a_{\text{cent}2} \) and the Coriolis acceleration \( a_{\text{cor}} = (0, -2L_2\dot{\theta}_1\dot{\theta}_2) \).

Torque about joint 1 is negative because \( m_2 \) gets closer to joint 1 (due to joint 2’s motion). Therefore the inertia due to \( m_2 \) about the \( \hat{z}_1 \)-axis is dropping, and therefore the positive momentum about joint 1 drops while joint 1’s speed is constant, and therefore joint 1 must apply a negative torque, since torque is defined as the rate of change of momentum. Otherwise \( \dot{\theta}_1 \) would increase as \( m_2 \) gets closer to joint 1, just as a skater’s rotation speed increases as she pulls in her outstretched arms while doing a spin.

8.1.2 General Formulation

We now describe the Lagrangian dynamics formulation for general \( n \)-link open chains. The first step is to select a set of generalized coordinates \( \theta \in \mathbb{R}^n \) for the configuration space of the system. For open chains all of whose joints are actuated, it is convenient and always possible to choose \( \theta \) to be the vector of joint values. The generalized forces will then be denoted \( \tau \in \mathbb{R}^n \). If \( \theta_i \) is a revolute joint, \( \tau_i \) will correspond to a torque, while if \( \theta_i \) is a prismatic joint, \( \tau_i \) will correspond to a force.

Once \( \theta \) has been chosen and the generalized forces \( \tau \) identified, the next step is to formulate the Lagrangian \( \mathcal{L} (\theta, \dot{\theta}) \) as

\[
\mathcal{L} (\theta, \dot{\theta}) = K(\theta, \dot{\theta}) - P(\theta),
\]

where \( K(\theta, \dot{\theta}) \) is the kinetic energy and \( P(\theta) \) the potential energy of the overall system. For rigid-link robots the kinetic energy can always be written in the form

\[
K(\theta) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} m_{ij}(\theta) \dot{\theta}_i \dot{\theta}_j = \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta},
\]

where \( m_{ij}(\theta) \) is the \((i, j)\) element of the \( n \times n \) mass matrix \( M(\theta) \); a constructive proof of this assertion is provided when we examine the Newton-Euler formulation.
8.1. Lagrangian Formulation

The dynamic equations are analytically obtained by evaluating the right-hand side of
\[
\tau_i = \frac{d}{dt} \frac{\partial L}{\partial \dot{\theta}_i} - \frac{\partial L}{\partial \theta_i}, \quad i = 1, \ldots, n. \tag{8.14}
\]
With the kinetic energy expressed as in Equation (8.13), the dynamics can be written explicitly as
\[
\tau_i = \sum_{j=1}^{n} m_{ij}(\theta) \ddot{\theta}_j + \sum_{j=1}^{n} \sum_{k=1}^{n} \Gamma_{ijk}(\theta) \dot{\theta}_j \dot{\theta}_k + \frac{\partial P}{\partial \theta_i}, \quad i = 1, \ldots, n, \tag{8.15}
\]
where the \( \Gamma_{ijk}(\theta) \), known as the Christoffel symbols of the first kind, are defined as
\[
\Gamma_{ijk}(\theta) = \frac{1}{2} \left( \frac{\partial m_{ij}}{\partial \theta_k} + \frac{\partial m_{ik}}{\partial \theta_j} - \frac{\partial m_{jk}}{\partial \theta_i} \right). \tag{8.16}
\]
This shows that the Christoffel symbols, which generate the Coriolis and centripetal terms \( c(\theta, \dot{\theta}) \), are derived from the mass matrix \( M(\theta) \).

As we have already seen, the equations (8.15) are often gathered together in the form
\[
\tau = M(\theta) \ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) \quad \text{or} \quad M(\theta) \ddot{\theta} + h(\theta, \dot{\theta}),
\]
where \( g(\theta) \) is simply \( \partial P/\partial \theta \).

To be more explicit that the Coriolis and centripetal terms are quadratic in the velocity, we could instead use the form
\[
\tau = M(\theta) \ddot{\theta} + \dot{\theta}^T \Gamma(\theta) \dot{\theta} + g(\theta), \tag{8.17}
\]
where \( \Gamma(\theta) \) is an \( n \times n \times n \) matrix, and the product \( \dot{\theta}^T \Gamma(\theta) \dot{\theta} \) should be interpreted as
\[
\dot{\theta}^T \Gamma(\theta) \dot{\theta} = \begin{bmatrix}
\dot{\theta}^T \Gamma_1(\theta) \dot{\theta} \\
\dot{\theta}^T \Gamma_2(\theta) \dot{\theta} \\
\vdots \\
\dot{\theta}^T \Gamma_n(\theta) \dot{\theta}
\end{bmatrix},
\]
where \( \Gamma_i(\theta) \) is an \( n \times n \) matrix with \( (j,k) \) entries \( \Gamma_{ijk} \).

It is also common to see the dynamics written as
\[
\tau = M(\theta) \ddot{\theta} + C(\theta, \dot{\theta}) \dot{\theta} + g(\theta),
\]
where \( C(\theta, \dot{\theta}) \in \mathbb{R}^{n \times n} \) is called the Coriolis matrix, with \( (i,j) \) entries
\[
c_{ij}(\theta, \dot{\theta}) = \sum_{k=1}^{n} \Gamma_{ijk}(\theta) \dot{\theta}_k. \tag{8.18}
\]

The following property, referred to as the passivity property, turns out to have important ramifications in proving the stability of certain robot control laws, as we will see in Chapter 11.
Proposition 8.1. The matrix $\dot{M}(\theta) - 2C(\theta, \dot{\theta}) \in \mathbb{R}^{n \times n}$, where $M(\theta) \in \mathbb{R}^{n \times n}$ is the mass matrix and $\dot{M}(\theta)$ its time derivative, and $C(\theta, \dot{\theta}) \in \mathbb{R}^{n \times n}$ is the Coriolis matrix as defined in Equation (8.18), is skew symmetric.

Proof. The $(i, j)$ component of $\dot{M} - 2C$ is

$$\dot{m}_{ij}(\theta) - 2c_{ij}(\theta, \dot{\theta}) = \sum_{k=1}^{n} \frac{\partial m_{ij}}{\partial \theta_k} \dot{\theta}_k - \frac{\partial m_{ij}}{\partial \theta_j} \dot{\theta}_i + \frac{\partial m_{kj}}{\partial \theta_i} \dot{\theta}_j - \frac{\partial m_{kj}}{\partial \theta_j} \dot{\theta}_i.$$

By switching the indices $i$ and $j$, it can be seen that

$$\dot{m}_{ji}(\theta) - 2c_{ji}(\theta, \dot{\theta}) = - (\dot{m}_{ij}(\theta) - 2c_{ij}(\theta, \dot{\theta})), $$

thus proving that $(\dot{M} - 2C)^T = -(\dot{M} - 2C)$ as claimed. \qed

8.1.3 Understanding the Mass Matrix

The kinetic energy $\frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta}$ is a generalization of the familiar $\frac{1}{2} m v^T v$ for a point mass. The fact that the mass matrix $M(\theta)$ is positive definite, meaning that $\dot{\theta}^T M(\theta) \dot{\theta} > 0$ for all $\dot{\theta} \neq 0$, is a generalization of the fact that the mass of a point mass is always positive, $m > 0$. In both cases, if the velocity is nonzero, the kinetic energy must be positive.

For a point mass with dynamics expressed in Cartesian coordinates as $f = m \ddot{x}$, the mass is independent of the direction of acceleration, and the acceleration $\ddot{x}$ is always “parallel” to the force, in the sense that $\ddot{x}$ is a scalar multiple of $f$. A mass matrix $M(\theta)$, on the other hand, presents different effective mass in different acceleration directions, and $\dot{\theta}$ is not generally a scalar multiple of $\tau$, even when $\dot{\theta} = 0$. To visualize the direction dependence of the effective mass, we can map a unit ball of joint accelerations $\{\ddot{\theta} \mid \ddot{\theta}^T \ddot{\theta} = 1\}$ through the mass matrix $M(\theta)$ to generate a joint force/torque ellipsoid when the mechanism is at rest ($\dot{\theta} = 0$). An example is shown in Figure 8.3 for the 2R arm of Figure 8.1, where $L_1 = L_2 = m_1 = m_2 = 1$. This torque ellipsoid can be interpreted as a direction-dependent mass ellipsoid: to get the same joint acceleration magnitude $\|\ddot{\theta}\|$, you must apply a larger torque to joint 1 than you would have to apply to joint 2. This effect is reduced as link 2 folds back on link 1, reducing the distance between joint 1 and $m_2$, as shown in Figure 8.3. The direction of the principal axes of the mass ellipsoid are given by the eigenvectors $v_i$ of $M(\theta)$ and the lengths of the principal semi-axes are given by the corresponding eigenvalues $\lambda_i$. The acceleration $\ddot{\theta}$ is only a scalar multiple of $\tau$ when $\tau$ is along one of the principal axes of the ellipsoid.

It is easier to visualize the mass matrix if it is represented as an effective mass at the end-effector. In other words, if you grabbed the endpoint of the 2R robot, how massy would it feel depending on the direction you applied force to it? Let us denote the effective mass matrix at the end-effector as $\Lambda(\theta)$, and the
8.1. Lagrangian Formulation

Figure 8.3: (Bold lines) A unit ball of accelerations in $\ddot{\theta}$ maps through the mass matrix $M(\theta)$ to a torque ellipsoid that depends on the configuration of the 2R arm. These torque ellipsoids may be interpreted as mass ellipsoids. (Dotted lines) A unit ball in $\tau$ maps through $M^{-1}(\theta)$ to an acceleration ellipsoid.

velocity of the end-effector as $V = (\dot{x}, \dot{y})^T$. We know that the kinetic energy of the robot must be the same regardless of the coordinates we use, so

$$\frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta} = \frac{1}{2} V^T \Lambda(\theta) V. \tag{8.19}$$

Assuming the Jacobian $J(\theta)$ satisfying $V = J(\theta) \dot{\theta}$ is invertible, then Equation (8.19) can be rewritten as

$$V^T \Lambda V = (J^{-1}V)^T M(J^{-1}V) = V^T (J^{-T}MJ^{-1})V.$$

In other words, the end-effector mass matrix is

$$\Lambda(\theta) = J^{-T}(\theta) M(\theta) J^{-1}(\theta). \tag{8.20}$$

Figure 8.4 shows the end-effector mass ellipsoids, with principal axes given by the eigenvectors of $\Lambda(\theta)$ and principal semi-axis lengths given by the eigenvalues, for the same two 2R robot configurations as in Figure 8.3. The endpoint
acceleration is only a scalar multiple of the force if the force is along one of the principal axes of the ellipsoid.

The change in apparent endpoint mass as a function of the configuration of the robot is an issue for robots used as haptic displays. One way to mitigate this issue is to make the mass of the links as small as possible.

Note that these ellipsoidal interpretations of the relationship between forces and accelerations are only relevant at zero velocity, where there are no Coriolis or centripetal terms.

### 8.1.4 Lagrangian Dynamics vs. Newton-Euler Dynamics

In the rest of this chapter, we focus on the Newton-Euler recursive method for calculating robot dynamics. Using the tools we have developed so far, the Newton-Euler formulation allows computationally efficient computer implementation, particularly for robots with many degrees of freedom, without the need for differentiation. The resulting equations of motion are, and must be, identical to those derived using the energy-based Lagrangian method.
The Newton-Euler method builds on the dynamics of a single rigid body, so we begin there.

8.2 Dynamics of a Single Rigid Body

8.2.1 Classical Formulation

Consider a rigid body consisting of a number of rigidly connected point masses, where point mass $i$ has mass $m_i$ and the total mass is $m = \sum_i m_i$. Let $r_i = (x_i, y_i, z_i)^T$ be the fixed location of mass $i$ in a body frame \(\{b\}\), where the origin of this frame is the unique point such that

$$\sum_i m_i r_i = 0.$$ 

This point is known as the center of mass. If some other point is erroneously chosen as the origin, then the frame \(\{b\}\) should be moved to the center of mass at $\frac{1}{m} \sum_i m_i r_i$ and the $r_i$'s recalculated in this center-of-mass frame.

Now assume that the body is moving with a body twist $\mathbf{V}_b = (\omega, \mathbf{v}_b)$, and let $\mathbf{p}_i(t)$ be the time-varying position of $m_i$, initially located at $r_i$, in the inertial frame \(\{\mathbf{b}\}\). Then

$$\dot{p}_i = v_b + \omega \times p_i$$

$$\ddot{p}_i = \dot{v}_b + \frac{d}{dt} \omega \times p_i + \omega \times \frac{d}{dt} p_i$$

$$= \dot{v}_b + \omega \times p_i + \omega \times (v_b + \omega \times p_i).$$

Plugging in $p_i = r_i$ and using our skew-symmetric notation, we get

$$\ddot{p}_i = \dot{v}_b + [\omega]r_i + [\omega]v_b + [\omega]^2 r_i,$$

Taking as a given that $f_i = m_i \ddot{p}_i$ for a point mass, the force acting on $m_i$ is

$$f_i = m_i (\dot{v}_b + [\omega]r_i + [\omega]v_b + [\omega]^2 r_i),$$

which implies a moment

$$m_i = [r_i] f_i.$$

The total force and moment acting on the body is expressed as the wrench $\mathbf{F}_b$:

$$\mathbf{F}_b = \begin{bmatrix} m_b \\ \mathbf{f}_b \end{bmatrix} = \begin{bmatrix} \sum_i m_i \\ \sum_i f_i \end{bmatrix}.$$

To simplify the expressions for $\mathbf{f}_b$ and $m_b$, keep in mind that $\sum_i m_i r_i = 0$ (and therefore $\sum_i m_i [r_i] = 0$), and for $a, b \in \mathbb{R}^3$, $[a] = -[a]^T$, $[a]b = -[b]a$, and
\[ [a] [b] = ([b] [a])^T. \] Focusing on the linear dynamics,

\[ f_b = \sum_i m_i (\dot{v}_b + [\dot{\omega}_b] r_i + [\omega_b] v_b + [\omega_b]^2 r_i) \]

\[ = \sum_i m_i (\dot{v}_b + [\omega_b] v_b) - \sum_i m_i [r_i] \dot{\omega}_b + \sum_i m_i [r_i] [\omega_b] \dot{\omega}_b \]

\[ = \sum_i m_i (\dot{v}_b + [\omega_b] v_b) \]

\[ = m(\dot{v}_b + [\omega_b] v_b). \quad (8.21) \]

Now focusing on the rotational dynamics,

\[ m_b = \sum_i m_i [r_i] (\dot{v}_b + [\omega_b] r_i + [\omega_b] v_b + [\omega_b]^2 r_i) \]

\[ = \sum_i m_i [r_i] \dot{v}_b - \sum_i m_i [r_i] [\omega_b] \dot{\omega}_b \]

\[ + \sum_i m_i [r_i] ([\dot{\omega}_b] r_i + [\omega_b]^2 r_i) \]

\[ = \sum_i m_i (-[r_i]^2 \dot{\omega}_b - [r_i]^T [\omega_b]^T r_i \omega_b) \]

\[ = \sum_i m_i (-[r_i]^2 \dot{\omega}_b - [\omega_b] [r_i]^2 \omega_b) \]

\[ = \left( -\sum_i m_i [r_i]^2 \right) \dot{\omega}_b + [\omega_b] \left( -\sum_i m_i [r_i]^2 \right) \omega_b \]

\[ = I_b \dot{\omega}_b + [\omega_b] I_b \omega_b , \quad (8.22) \]

where \( I_b = -\sum_i m_i [r_i]^2 \in \mathbb{R}^{3 \times 3} \) is the body’s rotational inertia matrix. In Equation \((8.22)\), note the presence of a term linear in the angular acceleration, \( I_b \dot{\omega}_b \), and a term quadratic in the angular velocities, \([\omega_b] I_b \omega_b \), just as we saw for mechanisms in Section 8.1. Also, \( I_b \) is symmetric and positive definite, just like the mass matrix for a mechanism, and the rotational kinetic energy is given by the quadratic

\[ K = \frac{1}{2} \omega^T I_b \omega. \]

One difference is that \( I_b \) is constant, whereas the mass matrix \( M(\theta) \) changes with the configuration of the mechanism.

Writing out the individual entries of \( I_b \), we get

\[ I_b = \begin{bmatrix}
\sum_i m_i y_i^2 + z_i^2 & -\sum_i m_i x_i y_i & -\sum_i m_i x_i z_i \\
-\sum_i m_i x_i y_i & \sum_i m_i (x_i^2 + z_i^2) & -\sum_i m_i y_i z_i \\
-\sum_i m_i x_i z_i & -\sum_i m_i y_i z_i & \sum_i m_i (x_i^2 + y_i^2)
\end{bmatrix} \]

\[ = \begin{bmatrix}
I_{xx} & I_{xy} & I_{xz} \\
I_{xy} & I_{yy} & I_{yz} \\
I_{xz} & I_{yz} & I_{zz}
\end{bmatrix}. \]
The summations can be replaced by volume integrals over the body $B$, with the point masses $m_i$ replaced by a mass density function $\rho(x, y, z)$:

\[
I_{xx} = \iiint_B (y^2 + z^2)\rho(x, y, z)\,dx\,dy\,dz \\
I_{yy} = \iiint_B (x^2 + z^2)\rho(x, y, z)\,dx\,dy\,dz \\
I_{zz} = \iiint_B (x^2 + y^2)\rho(x, y, z)\,dx\,dy\,dz \\
I_{xy} = -\iiint_B xy\rho(x, y, z)\,dx\,dy\,dz \\
I_{xz} = -\iiint_B xz\rho(x, y, z)\,dx\,dy\,dz \\
I_{yz} = -\iiint_B yz\rho(x, y, z)\,dx\,dy\,dz.
\] (8.23)

If the body has uniform density, $I_b$ is determined exclusively by the shape of the rigid body (see Figure 8.5).

Given an inertia matrix $I_b$, the **principal axes of inertia** are given by the eigenvectors and eigenvalues of $I_b$. Let $v_1, v_2, v_3$ be the eigenvectors of $I_b$ and $\lambda_1, \lambda_2, \lambda_3$ be the corresponding eigenvalues. Then the principal axes of inertia are in the directions of $v_1, v_2, v_3$, and the scalar moments of inertia about these axes, the **principal moments of inertia**, are $\lambda_1, \lambda_2, \lambda_3 > 0$. One of the principal axes maximizes the moment of inertia among all axes passing through the center of mass, and one of the principal axes minimizes the moment of inertia. For bodies with symmetry, often the principal axes of inertia are apparent. The principal axes may not be unique, however; for a uniform-density solid sphere, for example, any three orthogonal axes intersecting at the center of mass is a set of principal axes, and the minimum principal moment of inertia is equal to the maximum principal moment of inertia.

If the principal axes of inertia are aligned with the axes of $\{b\}$, the off-diagonal terms of $I_b$ are all zero, and the eigenvalues are the scalar moments of inertia $I_{xx}, I_{yy}, I_{zz}$ about the $\hat{x}, \hat{y}, \hat{z}$ axes, respectively. In this case, the equations of motion (8.22) simplify to

\[
m_b = \begin{bmatrix}
I_{xx}\dot{\omega}_x + (I_{zz} - I_{yy})\omega_y\omega_z \\
I_{yy}\dot{\omega}_y + (I_{xx} - I_{zz})\omega_x\omega_z \\
I_{zz}\dot{\omega}_z + (I_{yy} - I_{xx})\omega_x\omega_y
\end{bmatrix},
\] (8.24)

where $\omega_b = (\omega_x, \omega_y, \omega_z)$. When possible, we choose the axes of $\{b\}$ to be aligned with the principal axes of inertia, to reduce the number of nonzero entries in $I_b$ and to simplify the equations of motion.

Examples of common uniform-density solid bodies, their principal axes of inertia, and the principal moments of inertia obtained by solving the integrals (8.23), are given in Figure 8.5.
Dynamics of Open Chains

rectangular parallelepiped
circular cylinder
ellipsoid

volume = \(abc\)
volume = \(\pi r^2 h\)
volume = \(4\pi abc/3\)

\[
\begin{align*}
I_{xx} &= m(w^2 + h^2)/12 \\
I_{yy} &= m(\ell^2 + h^2)/12 \\
I_{zz} &= m(\ell^2 + w^2)/12
\end{align*}
\]

\[
\begin{align*}
I_{xx} &= m(3r^2 + h^2)/12 \\
I_{yy} &= m(3r^2 + h^2)/12 \\
I_{zz} &= m\ell^2/3
\end{align*}
\]

\[
\begin{align*}
I_{xx} &= m(b^2 + c^2)/5 \\
I_{yy} &= m(a^2 + c^2)/5 \\
I_{zz} &= m(a^2 + b^2)/5
\end{align*}
\]

Figure 8.5: The principal axes and the inertia about the principal axes for uniform density bodies of mass \(m\). Note that the \(\hat{x}\) and \(\hat{y}\) principal axes of the cylinder are not unique.

An inertia matrix \(I_b\) can be expressed in a rotated frame \(\{c\}\) described by the rotation matrix \(R_{bc}\). Denoting this inertia matrix as \(I_c\), and knowing that the kinetic energy of the rotating body must be identical regardless of the chosen frame, we have

\[
\frac{1}{2} \omega_c^T I_c \omega_c = \frac{1}{2} \omega_b^T I_b \omega_b
\]

\[
= \frac{1}{2} (R_{bc} \omega_c)^T I_b (R_{bc} \omega_c)
\]

\[
= \frac{1}{2} \omega_c^T (R_{bc}^T I_b R_{bc}) \omega_c.
\]

In other words,

\[
I_c = R_{bc}^T I_b R_{bc}. \tag{8.25}
\]

If the axes of \(\{b\}\) are not aligned with the principal axes of inertia, then we can diagonalize the inertia matrix by expressing it instead in the rotated frame \(\{c\}\), where the columns of \(R_{bc}\) correspond to eigenvalues of \(I_b\).

Sometimes it is convenient to represent the inertia matrix in a frame at a point not at the center of mass of the body, such as at a joint. **Steiner’s theorem** can be stated as follows.

**Theorem 8.2.** The inertia matrix \(I_q\) about a frame aligned with \(\{b\}\), but at a point \(q = (q_x, q_y, q_z)^T\) in \(\{b\}\), is related to the inertia matrix \(I_b\) calculated at the center of mass by

\[
I_q = I_b + m(q^T q I_3 - q q^T), \tag{8.26}
\]

where \(I\) is the 3 \times 3 identity matrix and \(m\) is the mass of the body.

Steiner’s theorem is a more general statement of the parallel-axis theorem, which states that the scalar inertia \(I_d\) about an axis parallel to, but a distance
8.2. Dynamics of a Single Rigid Body

$d$ from, an axis through the center of mass, is related to the scalar inertia $I_{cm}$ about the axis through the center of mass by

$$I_d = I_{cm} + md^2. \quad (8.27)$$

Proofs of these are left to the exercises.

Equations (8.25) and (8.26) are useful for calculating the inertia of a rigid body consisting of multiple component rigid bodies. First calculate the inertia matrices of the $n$ component bodies in terms of frames at their individual centers of mass. Then choose a common frame $\{\text{common}\}$ and use Equations (8.25) and (8.26) to express each of the inertia matrices in this common frame. Once the individual inertia matrices are expressed in $\{\text{common}\}$, they can be summed to get the inertia matrix $I_{\text{common}}$ for the composite rigid body.

In the case of motion confined to the $\hat{x}$-$\hat{y}$ plane, where $\omega_b = (0, 0, \omega_z)^T$ and the inertia of the body about the $\hat{z}$-axis through the center of mass is given by the scalar $I_{zz}$, the spatial rotational dynamics (8.22) reduce to the planar rotational dynamics

$$m_z = I_{zz} \dot{\omega}_z,$$

and the rotational kinetic energy is

$$K = \frac{1}{2} I_{zz} \omega_z^2.$$

8.2.2 Twist-Wrench Formulation

The linear dynamics (8.21) and the rotational dynamics (8.22) can be written in the following combined form:

$$\begin{bmatrix} m_b \\ f_b \end{bmatrix} = \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix} \begin{bmatrix} \dot{\omega}_b \\ \dot{v}_b \end{bmatrix} + \begin{bmatrix} [\omega_b] & 0 \\ 0 & [\omega_b] \end{bmatrix} \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}, \quad (8.28)$$

where $I$ is the $3 \times 3$ identity matrix. With the benefit of hindsight, and also making use of the fact that $[v]v = v \times v = 0$ and $[v]^T = -[v]$, we write Equation (8.28) in the following equivalent form:

$$\begin{bmatrix} m_b \\ f_b \end{bmatrix} = \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix} \begin{bmatrix} \dot{\omega}_b \\ \dot{v}_b \end{bmatrix} + \begin{bmatrix} [\omega_b] & 0 \\ 0 & [\omega_b] \end{bmatrix} \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} - \begin{bmatrix} [\omega_b] & 0 \\ 0 & [\omega_b] \end{bmatrix}^T \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}, \quad (8.29)$$

Written this way, each of the terms can now be identified with six-dimensional spatial quantities as follows:

(i) $(\omega_b, v_b)$ and $(m_b, f_b)$ can be respectively identified with the body twist $V_b$ and body wrench $F_b$,

$$V_b = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}, \quad F_b = \begin{bmatrix} m_b \\ f_b \end{bmatrix}. \quad (8.30)$$
(ii) The spatial inertia matrix $G_b \in \mathbb{R}^{6 \times 6}$ is defined as

$$G_b = \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix},$$  

(8.31)

where $I$ denotes the $3 \times 3$ identity matrix. As an aside, the kinetic energy of the rigid body can be expressed in terms of the spatial inertia matrix as

$$\text{Kinetic Energy} = \frac{1}{2} \omega_b^T I_b \omega_b + \frac{1}{2} m v_b^T v_b = \frac{1}{2} V_b^T G_b V_b.$$  

(8.32)

(iii) The spatial momentum $P_b \in \mathbb{R}^6$ is defined as

$$P_b = \begin{bmatrix} I_b \omega_b \\ m \omega_b \end{bmatrix} = \begin{bmatrix} I_b & 0 \\ 0 & mI \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = G_b V_b.$$  

(8.33)

Observe that the $P_b$ term in Equation (8.29) is left-multiplied by the matrix

$$- \begin{bmatrix} \omega_b \\ v_b \\ \omega_b \end{bmatrix}^T.$$  

(8.34)

We now explain the origin and geometric significance of this matrix. First, recall that the cross product of two vectors $\omega_1, \omega_2 \in \mathbb{R}^3$ can be calculated using skew-symmetric matrix notation as follows:

$$[\omega_1 \times \omega_2] = [\omega_1][\omega_2] - [\omega_2][\omega_1].$$  

(8.35)

The matrix in (8.34) can be thought of as a generalization of the cross-product operation to six-dimensional twists. Specifically, given two twists $V_1 = (\omega_1, v_1)$ and $V_2 = (\omega_2, v_2)$, we perform a calculation analogous to (8.35):

$$\begin{bmatrix} \omega_1 \\ v_1 \end{bmatrix} \begin{bmatrix} \omega_2 \\ v_2 \end{bmatrix} - \begin{bmatrix} \omega_2 \\ v_2 \end{bmatrix} \begin{bmatrix} \omega_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} [\omega_1][\omega_2] - [\omega_2][\omega_1] \\ [\omega_1]v_2 - [\omega_2]v_1 \end{bmatrix} = \begin{bmatrix} [\omega''] \\ v' \end{bmatrix}.$$  

which can be written more compactly in vector form as

$$\begin{bmatrix} \omega' \\ v' \end{bmatrix} = \begin{bmatrix} [\omega_1] \\ [v_1] \end{bmatrix} \begin{bmatrix} \omega_2 \\ v_2 \end{bmatrix}.$$  

This generalization of the cross product to two twists $V_1$ and $V_2$ is called the Lie bracket of $V_1$ and $V_2$.

**Definition 8.1.** Given two twists $V_1 = (\omega_1, v_1)$ and $V_2 = (\omega_2, v_2)$, the Lie bracket of $V_1$ and $V_2$, written either as $[V_1, V_2]$ or $\text{ad}_{V_1}(V_2)$, is defined as follows:

$$[V_1, V_2] = \begin{bmatrix} [\omega_1] \\ [v_1] \end{bmatrix} \begin{bmatrix} \omega_2 \\ v_2 \end{bmatrix} = \text{ad}_{V_1}(V_2) \in se(3).$$  

(8.36)
8.3. Newton-Euler Inverse Dynamics

Given \( V = (\omega, v) \), we further define the following notation for the \( 6 \times 6 \) matrix representation \([adV]\):

\[
[adV] = \begin{bmatrix} \omega & 0 \\ v & [\omega] \end{bmatrix} \in \mathbb{R}^{6\times 6}.
\] (8.37)

With this notation the Lie bracket \([V_1, V_2]\) can also be expressed as

\[
[V_1, V_2] = [adV_1](V_2) = [adV_1]V_2.
\] (8.38)

**Definition 8.2.** Given a twist \( V = (\omega, v) \) and wrench \( F = (m, f) \), define the mapping

\[
[adT V](F) = [adV]T F = \begin{bmatrix} \omega & 0 \\ v & [\omega] \end{bmatrix}^T \begin{bmatrix} m \\ f \end{bmatrix} = \begin{bmatrix} -[\omega]m - [v]f \\ -[\omega]f \end{bmatrix}
\] (8.39)

Using the above notation and definitions, the dynamic equations for a single rigid body can now be written as

\[
F_b = \mathcal{G}_b \dot{V}_b - [ad_{V_1}]P_b
\]
\[
= \mathcal{G}_b \dot{V}_b - [ad_{V_1}]^T \mathcal{G}_b V_b.
\] (8.40)

Note the analogy between Equation (8.40) and the moment equation for a rotating rigid body:

\[
m_b = I_b \ddot{\omega}_b - [\omega_b] I_b \dot{\omega}_b.
\] (8.41)

Equation (8.41) is simply the rotational component of (8.40).

### 8.3 Newton-Euler Inverse Dynamics

We now consider the inverse dynamics problem for an \( n \)-link open chain connected by one-degree-of-freedom joints. Given the joint positions \( \theta \in \mathbb{R}^n \), velocities \( \dot{\theta} \in \mathbb{R}^n \), and accelerations \( \ddot{\theta} \in \mathbb{R}^n \), the objective is to calculate the right-hand side of the dynamics equation

\[
\tau = M(\theta) \ddot{\theta} + h(\theta, \dot{\theta}).
\]

The main result is a recursive inverse dynamics algorithm consisting of a forward and backward iteration stage. In the forward iteration, the velocities and accelerations of each link are propagated from the base to the tip, while in the backward iteration, the forces and moments experienced by each link are propagated from the tip to the base.

#### 8.3.1 Derivation

A body-fixed reference frame \( \{i\} \) is attached to the center of mass of each link \( i, i = 1, \ldots, n \). The base frame is denoted \( \{0\} \), and a frame at the end-effector is denoted \( \{n+1\} \). This frame is fixed in \( \{n\} \).
When the manipulator is at the home position, with all joint variables zero, we denote the configuration of frame \{j\} in \{i\} as \(M_{i,j} \in SE(3)\), and the configuration of \{i\} in the base frame \{0\} using the shorthand \(M_i = M_{0,i}\). With these definitions, \(M_{i−1,i}\) can be calculated as

\[
M_{i−1,i} = M_{i−1}^{-1}M_i. \tag{8.42}
\]

The screw axis for joint \(i\), expressed in the link frame \{i\}, is \(A_i\). This same screw axis is expressed in the space frame \{0\} as \(S_i\), where the two are related by

\[
A_i = Ad_{M_{i−1}}(S_i).
\]

Defining \(T_{i,j} \in SE(3)\) to be the configuration of frame \{j\} in \{i\} for arbitrary joint variables \(\theta\), then the configuration of \{i\} relative to \{i−1\} given the joint variable \(\theta_i\) is

\[
T_{i−1,i}(\theta_i) = M_{i−1,i}e^{[A_i] \theta_i}.
\]

We further define the following notation:

(i) Denote the twist of link frame \{i\}, expressed in frame \{i\} coordinates, by \(V_i = (\omega_i, v_i)\).

(ii) Denote the wrench transmitted through joint \(i\) to link frame \{i\}, expressed in frame \{i\} coordinates, by \(F_i = (m_i, f_i)\).

(iii) Let \(G_i \in \mathbb{R}^{6 \times 6}\) denote the spatial inertia matrix of link \(i\), expressed relative to link frame \{i\}. Since we assume that all link frames are situated at the link center of mass, \(G_i\) has the block-diagonal form

\[
G_i = \begin{bmatrix} I_i & 0 \\ 0 & m_i I \end{bmatrix}, \tag{8.43}
\]

where \(I_i\) denotes the 3 \times 3 rotational inertia matrix of link \(i\) and \(m_i\) is the link mass.

With these definitions, we can recursively calculate the twist and acceleration of each link, moving from the base to the tip. The twist of link \(i\), expressed in \{i\}, is the sum of the twist of link \(i−1\), \(V_{i−1}\), but expressed in \{i\}, and the added twist due to the joint rate \(\dot{\theta}_i\):

\[
V_i = Ad_{T_{i−1,i}}(V_{i−1}) + A_i \dot{\theta}_i. \tag{8.44}
\]

The accelerations \(\ddot{V}_i\) can also be found recursively. Noting that

\[
[V_i] = \frac{d}{dt}T_{i−1,i}[V_i]T_{i−1,i}^{-1} + T_{i−1,i}[\ddot{V}_i]T_{i−1,i}^{-1} + T_{i−1,i}[\dot{V}_i] \frac{d}{dt}T_{i−1,i}^{-1} + [A_i] \dot{\theta}_i,
\]

and

\[
\frac{d}{dt}T_{i−1,i} = M_{i−1,i}[A_i]e^{[A_i] \theta_i} \dot{\theta}_i = M_{i−1,i}e^{[A_i] \theta_i} [A_i] \dot{\theta}_i
\]

\[
\frac{d}{dt}T_{i−1,i}^{-1} = -T_{i−1,i}^{-1} \dot{T}_{i−1,i} T_{i−1,i}^{-1},
\]
it can be shown that
\[ \dot{\mathbf{V}}_i = \mathbf{A}_i \ddot{\mathbf{V}}_i + \text{Ad}_{T_{i-1},i} (\dot{\mathbf{V}}_{i-1}) + [\text{Ad}_{T_{i-1},i} (\mathbf{V}_{i-1}), \mathbf{A}_i \dot{\mathbf{	heta}}_i], \]  
where \([\text{Ad}_{T_{i-1},i} (\mathbf{V}_{i-1}), \mathbf{A}_i \dot{\mathbf{	heta}}_i]\) denotes the Lie bracket of \(\text{Ad}_{T_{i-1},i} (\mathbf{V}_{i-1})\) with \(\mathbf{A}_i \dot{\mathbf{	heta}}_i\). Note that since \([\mathbf{A}_i, \mathbf{A}_i] = 0\) and \(\text{Ad}_{T_{i-1},i} (\mathbf{V}_{i-1}) = \mathbf{V}_i - \mathbf{A}_i \dot{\mathbf{	heta}}_i\), we obtain the alternative but equivalent formula
\[ \dot{\mathbf{V}}_i = \mathbf{A}_i \ddot{\mathbf{V}}_i + \text{Ad}_{T_{i-1},i} (\dot{\mathbf{V}}_{i-1}) + [\mathbf{V}_i, \mathbf{A}_i \dot{\mathbf{	heta}}_i]. \]  

Once we have determined all the link twists and accelerations moving outward from the base, we can calculate the joint torques or forces by moving inward from the tip. The rigid-body dynamics (8.40) tells us the total wrench that acts on link \(i\) given \(\mathbf{V}_i\) and \(\dot{\mathbf{V}}_i\), and the total wrench acting on link \(i\) is the sum of the wrench \(\mathbf{F}_i\) transmitted through joint \(i\) and the wrench applied to the link through joint \(i+1\) (or, for link \(n\), the wrench applied to the link by the environment at the end-effector frame \(\{n+1\}\)) expressed in the frame \(i\):
\[ \mathbf{G}_i \dot{\mathbf{V}}_i - \text{ad}_{\dot{\mathbf{V}}_i}^T (\mathbf{G}_i \mathbf{V}_i) = \mathbf{F}_i - \text{Ad}_{T_{i+1},i}^T (\mathbf{F}_{i+1}). \]  
See Figure 8.6. Solving from the tip toward the base, at each link we solve for the only unknown in Equation (8.47): \(\mathbf{F}_i\). Since joint \(i\) has only one degree of freedom, five dimensions of the six-vector \(\mathbf{F}_i\) are provided “for free” by the structure of the joint, and the actuator only has to provide the force or torque in the direction of the joint’s screw axis:
\[ \mathbf{\tau}_i = \mathbf{F}_i^T \mathbf{A}_i. \]  
These equations provide the torques required at each joint, solving the inverse dynamics problem.
8.3.2 Newton-Euler Inverse Dynamics Algorithm

**Initialization:** Attach a frame \{0\} to the base, frames \{1\} to \{n\} to the centers of mass of links \{1\} to \{n\}, and a frame \{n + 1\} at the end-effector, fixed in the frame \{n\}. Define \(M_{i-1,i}\) to be the configuration of \{i\} in \{i - 1\} when \(\theta_i = 0\). Let \(A_i\) be the screw axis of joint \(i\) expressed in \{i\}, and \(\mathcal{G}_i\) be the \(6 \times 6\) spatial inertia matrix of link \(i\). Define \(\mathcal{V}_0\) to be the twist of the base frame \{0\} expressed in base frame coordinates. (This quantity is typically zero.) Let \(g \in \mathbb{R}^3\) be the gravity vector expressed in base frame \{0\} coordinates, and define \(\dot{V}_0 = (0, -g)\). (Gravity is treated as an acceleration of the base in the opposite direction.) Define \(F_{n+1} = F_{\text{tip}} = (m_{\text{tip}}, f_{\text{tip}})\) to be the wrench applied to the environment by the end-effector expressed in the end-effector frame \{n + 1\}.

**Forward iterations:** Given \(\theta, \dot{\theta}, \ddot{\theta}\), for \(i = 1\) to \(n\) do

\[
T_{i-1,i} = M_{i-1,i}e^{[A_i]^{\theta_i}}, \quad (8.49)
\]

\[
V_i = \text{Ad}_{T_{i-1,i}}(V_{i-1}) + A_i \dot{\theta}_i \quad (8.50)
\]

\[
\dot{V}_i = \text{Ad}_{T_{i-1,i}}(\dot{V}_{i-1}) + [V_i, A_i] \ddot{\theta}_i + A_i \dddot{\theta}_i. \quad (8.51)
\]

**Backward iterations:** For \(i = n\) to 1 do

\[
F_i = \text{Ad}^T_{T_{i+1,i}}(F_{i+1}) + \mathcal{G}_i \dot{V}_i - \text{ad}^T_{V_i}(\mathcal{G}_i V_i) \quad (8.52)
\]

\[
\tau_i = F_i^T A_i. \quad (8.53)
\]

8.4 Dynamic Equations in Closed Form

In this section we show how the equations in the recursive inverse dynamics algorithm can be organized into a closed-form set of dynamics equations of the form \(\tau = M(\theta) \ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta)\).

Before doing so, we prove our earlier assertion that the total kinetic energy \(K\) of the robot can be expressed as \(K = \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta}\). We do so by noting that \(K\) can be expressed as the sum of the kinetic energies of each link:

\[
K = \frac{1}{2} \sum_{i=1}^{n} V_i^T \mathcal{G}_i V_i, \quad (8.54)
\]

where \(V_i\) is the twist of link frame \{i\}, and \(\mathcal{G}_i\) is the spatial inertia matrix of link \(i\) as defined by Equation (8.31) (both are expressed in link frame \{i\} coordinates). Let \(T_{0i}(\theta_1, \ldots, \theta_i)\) denote the forward kinematics from the base frame \{0\} to link frame \{i\}, and let \(J_{ib}(\theta)\) denote the body Jacobian obtained from \(T_{0i}^{-1} T_{0i}\). Note that \(J_{ib}\) as defined is a \(6 \times i\) matrix; we turn it into a \(6 \times n\) matrix by filling in all entries of the last \(n - i\) columns with zeros. With this definition of \(J_{ib}\), we can write

\[
V_i = J_{ib}(\theta) \dot{\theta}, \quad i = 1, \ldots, n.
\]
The kinetic energy can then be written
\[ K = \frac{1}{2} \theta^T \left( \sum_{i=1}^{n} J_i^T(\theta) G_i J_i(\theta) \right) \dot{\theta}. \tag{8.55} \]

The term inside the parentheses is precisely the mass matrix \( M(\theta) \):
\[ M(\theta) = \sum_{i=1}^{n} J_i^T(\theta) G_i J_i(\theta). \tag{8.56} \]

We now return to the original task of deriving a closed-form set of dynamic equations. We start by defining the following stacked vectors:
\[ \mathcal{V} = \begin{bmatrix} \mathcal{V}_1 \\ \vdots \\ \mathcal{V}_n \end{bmatrix} \in \mathbb{R}^{6n} \tag{8.57} \]
\[ \mathcal{F} = \begin{bmatrix} \mathcal{F}_1 \\ \vdots \\ \mathcal{F}_n \end{bmatrix} \in \mathbb{R}^{6n} \tag{8.58} \]

Further define the following matrices:
\[ \mathcal{A} = \begin{bmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & A_n \end{bmatrix} \in \mathbb{R}^{6n \times n} \tag{8.59} \]
\[ \mathcal{G} = \begin{bmatrix} G_1 & 0 & \cdots & 0 \\ 0 & G_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & G_n \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.60} \]
\[ [\text{ad}\mathcal{V}] = \begin{bmatrix} [\text{ad}\mathcal{V}_1] & 0 & \cdots & 0 \\ 0 & [\text{ad}\mathcal{V}_2] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & [\text{ad}\mathcal{V}_n] \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.61} \]
\[ [\text{ad}\mathcal{A}] = \begin{bmatrix} [\text{ad}\mathcal{A}_{1,\theta_1}] & 0 & \cdots & 0 \\ 0 & [\text{ad}\mathcal{A}_{2,\theta_2}] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & [\text{ad}\mathcal{A}_{n,\theta_n}] \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.62} \]
\[ \mathcal{S}(\theta) = \begin{bmatrix} [\text{Ad}\mathcal{T}_{21}] & 0 & \cdots & 0 \\ 0 & [\text{Ad}\mathcal{T}_{32}] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & [\text{Ad}\mathcal{T}_{n,n-1}] \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.63} \]
We write $S(\theta)$ to emphasize the dependence of $S$ on $\theta$. Finally, define the following stacked vectors:

$$
V_{\text{base}} = \begin{bmatrix}
Ad_{T_{10}}(V_0) \\
0 \\
\vdots \\
0
\end{bmatrix} \in \mathbb{R}^{6n} \tag{8.64}
$$

$$
\dot{V}_{\text{base}} = \begin{bmatrix}
Ad_{T_{10}}(\dot{V}_0) \\
0 \\
\vdots \\
0
\end{bmatrix} \in \mathbb{R}^{6n} \tag{8.65}
$$

$$
F_{\text{tip}} = \begin{bmatrix}
0 \\
\vdots \\
0 \\
Ad_{T_{n+1,n}}(F_{n+1})
\end{bmatrix} \in \mathbb{R}^{6n}. \tag{8.66}
$$

Note that $A \in \mathbb{R}^{6n \times n}$ and $G \in \mathbb{R}^{6n \times 6n}$ are constant block-diagonal matrices, in which $A$ contains only the kinematic parameters, while $G$ contains only the mass and inertial parameters for each link.

With the above definitions, our earlier recursive inverse dynamics algorithm can be assembled into the following set of matrix equations:

$$
V = S(\theta)V + A\dot{\theta} + V_{\text{base}} \tag{8.67}
$$

$$
\dot{V} = S(\theta)\dot{V} + A\ddot{\theta} - [\text{ad}_A](S(\theta)V + V_{\text{base}}) + \dot{V}_{\text{base}} \tag{8.68}
$$

$$
F = S^T(\theta)F + G\dot{V} - [\text{ad}_V]^TG\dot{V} + F_{\text{tip}} \tag{8.69}
$$

$$
\tau = A^TF. \tag{8.70}
$$

$S(\theta)$ has the property that $S^n(\theta) = 0$ (such a matrix is said to be nilpotent of order $n$), and one consequence verifiable through direct calculation is that $(I - S(\theta))^{-1} = I + S(\theta) + \ldots + S^{n-1}(\theta)$. Defining $L(\theta) = (I - S(\theta))^{-1}$, it can further be verified via direct calculation that

$$
L(\theta) = \begin{bmatrix}
I & 0 & 0 & \cdots & 0 \\
[\text{Ad}_{T_{21}}] & I & 0 & \cdots & 0 \\
[\text{Ad}_{T_{31}}] & [\text{Ad}_{T_{32}}] & I & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
[\text{Ad}_{T_{n1}}] & [\text{Ad}_{T_{n2}}] & [\text{Ad}_{T_{n3}}] & \cdots & I
\end{bmatrix} \in \mathbb{R}^{6n \times 6n}. \tag{8.71}
$$

We write $L(\theta)$ to emphasize the dependence of $L$ on $\theta$. The earlier matrix
equations can now be reorganized as

\[
\begin{align*}
\mathcal{V} &= \mathcal{L}(\theta) \left( A \dot{\theta} + \mathcal{V}_{\text{base}} \right) \quad (8.72) \\
\dot{\mathcal{V}} &= \mathcal{L}(\theta) \left( A \ddot{\theta} + [\text{ad}_{A \dot{\theta}}] S(\theta) \mathcal{V} + [\text{ad}_{A \dot{\theta}}] \mathcal{V}_{\text{base}} + \dot{\mathcal{V}}_{\text{base}} \right) \quad (8.73) \\
\mathcal{F} &= \mathcal{L}^T(\theta) \left( \mathcal{G} \mathcal{V} - [\text{ad}_{\mathcal{V}}]^T \mathcal{G} \mathcal{V} + \mathcal{F}_{\text{tip}} \right) \quad (8.74) \\
\tau &= \mathcal{A}^T \mathcal{F}. \quad (8.75)
\end{align*}
\]

If the robot applies an external wrench \( \mathcal{F}_{\text{tip}} \) at the end-effector, this can be included into the following dynamics equation,

\[
\tau = M(\theta) \ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) + J^T(\theta) \mathcal{F}_{\text{tip}}, \quad (8.76)
\]

where \( J(\theta) \) denotes the Jacobian of the forward kinematics expressed in the same reference frame as \( \mathcal{F}_{\text{tip}} \), and

\[
\begin{align*}
M(\theta) &= \mathcal{A}^T \mathcal{L}^T(\theta) \mathcal{G} \mathcal{L}(\theta) A \quad (8.77) \\
c(\theta, \dot{\theta}) &= -\mathcal{A}^T \mathcal{L}^T(\theta) \left( \mathcal{G} \mathcal{L}(\theta) [\text{ad}_{A \dot{\theta}}] S(\theta) + [\text{ad}_{\mathcal{V}}]^T \mathcal{G} \right) \mathcal{L}(\theta) A \dot{\theta} \quad (8.78) \\
g(\theta) &= \mathcal{A}^T \mathcal{L}^T(\theta) \mathcal{G} \mathcal{L}(\theta) \mathcal{V}_{\text{base}}. \quad (8.79)
\end{align*}
\]

### 8.5 Forward Dynamics of Open Chains

The forward dynamics problem is to solve

\[
M(\theta) \ddot{\theta} = \tau(t) - h(\theta, \dot{\theta}) - J^T(\theta) \mathcal{F}_{\text{tip}} \quad (8.80)
\]

for \( \ddot{\theta} \), given \( \theta, \dot{\theta}, \tau \), and the wrench \( \mathcal{F}_{\text{tip}} \) applied by the end-effector (if applicable). The term \( h(\theta, \dot{\theta}) \) can be computed by calling the inverse dynamics algorithm with \( \dot{\theta} = 0 \) and \( \mathcal{F}_{\text{tip}} = 0 \). The inertia matrix \( M(\theta) \) can be computed using Equation (8.56). An alternative is to use \( n \) calls of the inverse dynamics algorithm to build \( M(\theta) \) column by column. In each of the \( n \) calls, set \( \mathcal{g} = 0 \), \( \dot{\theta} = 0 \), and \( \mathcal{F}_{\text{tip}} = 0 \). In the first call, the column vector \( \ddot{\theta} \) is all zeros except for a 1 in the first row. In the second call, \( \ddot{\theta} \) is all zeros except for a 1 in the second row, and so on. The \( \tau \) vector returned by the \( i \)th call is the \( i \)th column of \( M(\theta) \), and after \( n \) calls the \( n \times n \) matrix \( M(\theta) \) is constructed.

With \( M(\theta), h(\theta, \dot{\theta}), \) and \( \mathcal{F}_{\text{tip}} \), we can use any efficient algorithm for solving Equation (8.80), an equation of the form \( M \ddot{\theta} = b \), for \( \ddot{\theta} \).

The forward dynamics can be used to simulate the motion of the robot given its initial state, the joint forces/torques \( \tau(t) \), and an optional external wrench \( \mathcal{F}_{\text{tip}}(t) \), for \( t \in [0, t_f] \). First define the function \( \text{ForwardDynamics} \) returning the solution to Equation (8.80), i.e.,

\[
\ddot{\theta} = \text{ForwardDynamics}(\theta, \dot{\theta}, \tau, \mathcal{F}_{\text{tip}}).
\]
Defining the variables $q_1 = \theta$, $q_2 = \dot{\theta}$, the second-order dynamics (8.80) can be converted to two first-order differential equations

\[
\begin{align*}
\dot{q}_1 &= q_2 \\
\dot{q}_2 &= \text{ForwardDynamics}(q_1, q_2, \tau, \mathcal{F}_{\text{tip}}).
\end{align*}
\]

The simplest method for numerically integrating a system of first-order differential equations of the form $\dot{q} = f(q, t)$, $q \in \mathbb{R}^n$, is the first-order Euler iteration

\[
q(t + \delta t) = q(t) + \delta t f(q(t), t),
\]

where the positive scalar $\delta t$ denotes the timestep. The Euler integration of the robot dynamics is thus

\[
\begin{align*}
q_1(t + \delta t) &= q_1(t) + q_2(t)\delta t \\
q_2(t + \delta t) &= q_2(t) + \text{ForwardDynamics}(q_1, q_2, \tau, \mathcal{F}_{\text{tip}})\delta t.
\end{align*}
\]

Given a set of initial values for $q_1(0) = \theta(0)$ and $q_2(0) = \dot{\theta}(0)$, the above equations can be iterated forward in time to numerically obtain the motion $\theta(t) = q_1(t)$.

**Euler Integration Algorithm for Forward Dynamics**

- **Inputs**: Initial conditions $\theta(0)$ and $\dot{\theta}(0)$, input torques $\tau(t)$ and wrenches at the end-effector $\mathcal{F}_{\text{tip}}(t)$ for $t \in [0, t_f]$, and the number of integration steps $N$.
- **Initialization**: Set timestep $\delta t = t_f/N$, $\theta[0] = \theta(0)$, $\dot{\theta}[0] = \dot{\theta}(0)$.
- **Iteration**: For $k = 0$ to $N - 1$ do

  \[
  \begin{align*}
  \dot{\theta}[k] &= \text{ForwardDynamics}(\theta[k], \dot{\theta}[k], \tau(k\delta t), \mathcal{F}_{\text{tip}}(k\delta t)) \\
  \theta[k + 1] &= \theta[k] + \dot{\theta}[k]\delta t \\
  \dot{\theta}[k + 1] &= \dot{\theta}[k] + \dot{\theta}[k]\delta t
  \end{align*}
  \]

- **Output**: Joint trajectory $\theta(k\delta t) = \theta[k]$, $\dot{\theta}(k\delta t) = \dot{\theta}[k]$, $k = 0, \ldots, N$.

The result of the numerical integration converges to the theoretical result as the number of integration steps $N$ goes to infinity. Higher-order numerical integration schemes, such as fourth-order Runge-Kutta, can yield closer approximations with fewer computations than the simple first-order Euler method.

### 8.6 Dynamics in Task Space Coordinates

In this section we consider how the dynamic equations change under a transformation to coordinates of the end-effector frame (task space coordinates). To
8.6. Dynamics in Task Space Coordinates

To keep things simple, we consider a six-degree-of-freedom open chain with joint space dynamics,

\[ \tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}), \quad \theta \in \mathbb{R}^6, \quad \tau \in \mathbb{R}^6. \tag{8.81} \]

We also ignore for the time being any end-effector forces \( F_{\text{tip}} \). The twist \( \mathcal{V} = (\omega, v) \) of the end-effector is related to the joint velocity \( \dot{\theta} \) by

\[ \mathcal{V} = J(\theta)\dot{\theta}, \tag{8.82} \]

with the understanding that \( \mathcal{V} \) and \( J(\theta) \) are always expressed in terms of the same reference frame. The time derivative \( \dot{\mathcal{V}} \) is then

\[ \dot{\mathcal{V}} = J(\theta)\ddot{\theta} + J(\theta)\dot{J}(\theta)\dot{\theta}. \tag{8.83} \]

At configurations \( \theta \) where \( J(\theta) \) is invertible, we have

\[ \begin{align*}
\dot{\theta} &= J^{-1}\mathcal{V} \\
\ddot{\theta} &= J^{-1}\dot{\mathcal{V}} - J^{-1}\dot{J}(\theta)^{-1}J(\theta)^{-1}\mathcal{V}.
\end{align*} \tag{8.84, 8.85} \]

Substituting for \( \dot{\theta} \) and \( \ddot{\theta} \) in Equation (8.81) leads to

\[ \tau = M(\theta) \left( J^{-1}\dot{\mathcal{V}} - J^{-1}\dot{J}(\theta)^{-1}J(\theta)^{-1}\mathcal{V} \right) + h(\theta, J^{-1}\mathcal{V}), \tag{8.86} \]

where \( J^{-T} \) denotes \( (J^{-1})^T = (J^T)^{-1} \). Pre-multiply both sides by \( J^{-T} \) to get

\[ J^{-T}\tau = J^{-T}M(\theta)J^{-1}\dot{\mathcal{V}} - J^{-T}MJ^{-1}\dot{J}(\theta)^{-1}J(\theta)^{-1}\mathcal{V} + J^{-T}h(\theta, J^{-1}\mathcal{V}). \tag{8.87} \]

Expressing \( J^{-T}\tau \) as the wrench \( \mathcal{F} \), the above can be written

\[ \mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \eta(\theta, \mathcal{V}), \tag{8.88} \]

where

\[ \begin{align*}
\Lambda(\theta) &= J^{-T}M(\theta)J^{-1} \tag{8.89} \\
\eta(\theta, \mathcal{V}) &= J^{-T}h(\theta, J^{-1}\mathcal{V}) - \Lambda(\theta)\dot{J}(\theta)^{-1}J(\theta)^{-1}\mathcal{V}. \tag{8.90}
\end{align*} \]

These are the dynamic equations expressed in end-effector frame coordinates. If an external wrench \( \mathcal{F} \) is applied to the end-effector frame, then assuming zero actuator effort, the motion of the end-effector frame is governed by these equations. Note the dependence of \( \Lambda(\theta) \) and \( \eta(\theta, \mathcal{V}) \) on \( \theta \). If \( \theta \) were replaced by its inverse kinematics solution \( \theta = T^{-1}(X) \), then one would obtain a differential equation strictly in terms of the end-effector frame’s displacement \( X \in SE(3) \) and twist \( \mathcal{V} \). In practice, since \( X \) is usually obtained by measuring \( \theta \) and substituting into the forward kinematics, it is preferable to leave the dependence on \( \theta \) explicit.
8.7 Constrained Dynamics

Now consider the case that the \( n \)-joint robot is subject to a set of \( k \) holonomic or nonholonomic Pfaffian velocity constraints of the form

\[
A(\theta)\dot{\theta} = 0, \quad A(\theta) \in \mathbb{R}^{k \times n}.
\]  
(8.91)

(See Chapter 2.4 for an introduction to Pfaffian constraints.) Such constraints can come from loop-closure constraints. As an example, the motion of the end-effector of a robot arm opening a door is subject to \( k = 5 \) constraints due to the hinges of the door. As another example, a robot writing with a pen is subject to \( k = 1 \) constraint that keeps the height of the tip of the pen above the paper at zero. In any case, we assume that the constraints do no work on the robot, i.e., the generalized forces \( \tau_{\text{con}} \) due to the constraints satisfy

\[
\tau_{\text{con}}^T \dot{\theta} = 0.
\]

This assumption means that \( \tau_{\text{con}} \) must be a linear combination of the columns of \( A^T(\theta) \), i.e., \( \tau_{\text{con}} = A^T(\theta)\lambda \) for some \( \lambda \in \mathbb{R}^k \), since these are the generalized forces that do no work when \( \dot{\theta} \) is subject to the constraints (8.91):

\[
(A^T(\theta)\lambda)^T \dot{\theta} = \lambda^T A(\theta) \dot{\theta} = 0 \quad \text{for all } \lambda \in \mathbb{R}^k.
\]

For the writing robot example, the assumption that the constraint is workless means that there is no friction between the pen and the paper.

Adding the constraint forces \( A^T(\theta)\lambda \) to the equations of motion, we can write the \( n + k \) constrained equations of motion in the \( n + k \) unknowns \( \{\dot{\theta}, \lambda\} \) (forward dynamics) or the \( n + k \) unknowns \( \{\tau, \lambda\} \) (inverse dynamics):

\[
\begin{align*}
\tau &= M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}) + A^T(\theta)\lambda \\
A(\theta)\dot{\theta} &= 0,
\end{align*}
\]  
(8.92)  
(8.93)

where \( \lambda \) is a set of Lagrange multipliers and \( A^T(\theta)\lambda \) are the forces the robot creates against the constraints. From these equations, it should be clear that the robot has \( n - k \) velocity freedoms and \( k \) “force freedoms”—the robot is free to create any generalized force of the form \( A^T(\theta)\lambda \). (For the writing robot, this is not quite true; the robot can only apply pushing forces into the paper and table, not pulling forces.)

Often it is convenient to reduce the \( n + k \) equations in \( n + k \) unknowns to \( n \) equations in \( n \) unknowns, without explicitly calculating the Lagrange multipliers \( \lambda \). To do this, we can solve for \( \lambda \) in terms of the other quantities and plug our solution into Equation (8.92). Since the constraints are satisfied at all times, the time rate of change of the constraints satisfies

\[
\dot{A}(\theta)\dot{\theta} + A(\theta)\ddot{\theta} = 0.
\]  
(8.94)

Assuming that \( M(\theta) \) and \( A(\theta) \) are full rank, we solve Equation (8.92) for \( \ddot{\theta} \), plug in to Equation (8.94), and omit the dependencies on \( \theta \) and \( \dot{\theta} \) for conciseness to get

\[
\dot{\theta} + AM^{-1}(\tau - h - A^T\lambda) = 0.
\]  
(8.95)
Using $\dot{A}\ddot{\theta} = -\ddot{A}\dot{\theta}$, after some manipulation, we get

$$\lambda = (AM^{-1}A^T)^{-1}(AM^{-1}(\tau - h) - \dot{A}\ddot{\theta}). \quad (8.96)$$

Plugging Equation (8.96) into Equation (8.92) and manipulating further, we get

$$P\tau = P(M\ddot{\theta} + h) \quad (8.97)$$

where

$$P = I - A^T(AM^{-1}A^T)^{-1}AM^{-1}, \quad (8.98)$$

where $I$ is the $n \times n$ identity matrix. The $n \times n$ matrix $P(\theta)$ is rank $n - k$, and it maps the joint generalized forces $\tau$ to $P(\theta)\tau$, projecting away the generalized force components against the constraints while retaining the generalized forces that do work on the robot. The complementary projection $(I - P(\theta))$ maps $\tau$ to $(I - P(\theta))\tau$, the joint forces that act on the constraints and do no work on the robot.

Using $P(\theta)$, we can solve the inverse dynamics by evaluating the right-hand side of Equation (8.97). The solution $P(\theta)\tau$ is a set of generalized joint forces that achieves the feasible components of the desired joint acceleration $\ddot{\theta}$. To this solution we can add any constraint forces $A^T(\lambda)$ without changing the acceleration of the robot.

We can rearrange Equation (8.97) to the form

$$P\ddot{\theta} = P\ddot{\theta}M^{-1}(\tau - h), \quad (8.99)$$

where

$$P\ddot{\theta} = M^{-1}PM = I - M^{-1}A^T(AM^{-1}A^T)^{-1}A. \quad (8.100)$$

The rank $n - k$ projection matrix $P\ddot{\theta}(\theta) \in \mathbb{R}^{n \times n}$ projects away the components of an arbitrary joint acceleration $\ddot{\theta}$ that violate the constraints, leaving only the components $P\ddot{\theta}(\theta)\ddot{\theta}$ that satisfy the constraints. To solve the forward dynamics, evaluate the right-hand side of Equation (8.99), and the resulting $P\ddot{\theta}$ is the set of joint accelerations.

In Chapter 11.4 we discuss the related topic of hybrid motion-force control, in which the goal at each instant is to simultaneously achieve a desired acceleration satisfying $\dot{A}\ddot{\theta} = 0$ (consisting of $n - k$ motion freedoms) and a desired constraint force $A^T(\lambda)$ (consisting of $k$ force freedoms). In that chapter, we use the task-space dynamics to represent the task-space end-effector motions and forces more naturally.

### 8.8 Robot Dynamics in the URDF

As described in Chapter 4.2 and illustrated in the UR5 Universal Robot Description Format file, the inertial properties of link $i$ are described in the URDF by the link elements mass, origin (the position and orientation of the center of mass frame relative to a frame attached at joint $i$), and inertia, which specifies the six elements of the symmetric rotational inertia matrix on or above the
diagonal. To fully write the robot’s dynamics, for joint $i$ we need the element \textit{origin}, specifying the position and orientation of link $i$’s joint frame relative to link $i-1$’s joint frame when $\theta_i = 0$, and the element \textit{axis}, which specifies the axis of motion of joint $i$. It is left to the exercises to translate these elements to the quantities needed for the Newton-Euler inverse dynamics algorithm.

8.9 Actuation, Gearing, and Friction

Until now we have been assuming the existence of actuators that directly provide commanded forces and torques. In practice, there are many types of actuators (e.g., electric, hydraulic, and pneumatic) and mechanical power transformers (e.g., gearheads), and the actuators can be located at the joints themselves or remotely, with mechanical power transmitted by cables or timing belts. Each combination of these has its own characteristics that can play a significant role in the “extended dynamics” mapping the actual control inputs (e.g., the current requested of amplifiers connected to electric motors) to the motion of the robot.

In this section we provide an introduction to some of the issues associated with one particular, and common, configuration: geared DC electric motors located at each joint. This is the configuration used in the Universal Robots UR5, for example.

Figure 8.7 shows the electrical block diagram for a typical $n$-joint robot driven by DC electric motors. For concreteness, we assume each joint is revolute. A power supply converts the wall AC voltage to a DC voltage to power the amplifier associated with each motor. A control box takes user input, for example in the form of a desired trajectory, as well as position feedback from encoders located at each joint. Using the desired trajectory, a model of the robot’s dynamics, and any errors in the current robot state relative to the desired robot state, the controller calculates the torque required of each actuator. Since DC electric motors nominally provide a torque proportional to the current through the motor, this torque command is equivalent to a current command. Each motor amplifier then uses a current sensor (shown external to the amplifier in Figure 8.7, but actually internal to the amplifier) to continually adjust the voltage across the motor to try to achieve the requested current.\textsuperscript{3} The motion of the motor is sensed by the motor encoder, and the position information is sent back to the controller.

The commanded torque/current is typically updated in the neighborhood of 1000 times per second (1 kHz), and the amplifier’s voltage control loop may be updated at a rate ten times that or more.

Figure 8.8 is a conceptual representation of the motor and other components for a single axis. The motor has a single shaft extending from both ends of the motor: one end drives a rotary encoder, which measures the position of the joint, and the other end becomes the input to a gearhead. The gearhead increases the torque while reducing the speed, since most DC electric motors

\textsuperscript{3}The voltage is typically a time-averaged voltage achieved by the duty cycle of a voltage rapidly switching between a maximum positive voltage and a maximum negative voltage.
8.9. Actuation, Gearing, and Friction

Figure 8.7: A block diagram of a typical $n$-joint robot. Bold lines correspond to high-power signals, while thin lines correspond to communication signals.

Figure 8.8: The outer cases of the encoder, motor, gearhead, and bearing are fixed in link $i$, while the gearhead output shaft supported by the bearing is fixed in link $i + 1$.

with an appropriate power rating provide torques that are too low to be useful for robotics applications. The purpose of the bearing is to support the gearhead output, freely transmitting torques about the gearhead axis while isolating the gearhead (and motor) from wrench components due to link $i + 1$ in the other five directions. The outer cases of the encoder, motor, gearhead, and bearing are all fixed relative to each other and to link $i$. It is also common for the motor to have some kind of brake, not shown.
8.9.1 DC Motors and Gearing

A DC motor consists of a **stator** and a **rotor** that rotates relative to the stator. DC electric motors create torque by sending current through windings in a magnetic field created by permanent magnets, where the magnets are attached to the stator and the windings are attached to the rotor, or vice-versa. A DC motor has multiple windings, some of which are energized and some of which are inactive at any given time. The windings that are energized are chosen as a function of the angle of the rotor relative to the stator. This “commutation” of the windings occurs mechanically using brushes (brushed motors) or electrically using control circuitry (brushless motors). Brushless motors have the advantage of no brush wear and higher continuous torque, since the windings are typically attached to the motor housing where the heat due to the resistance of the windings can be more easily dissipated. In our basic introduction to DC motor modeling, we do not distinguish between brushed and brushless motors.

Figure 8.9 shows a brushed DC motor with an encoder and a gearhead.

The torque $\tau$, measured in newton-meters (Nm), created by a DC motor is governed by the equation

$$\tau = k_t I,$$

where $I$, measured in amps (A), is the current through the windings. The constant $k_t$, measured in newton-meters per amp (Nm/A), is called the **torque constant**. The power dissipated as heat by the windings, measured in watts (W), is governed by

$$P_{\text{heat}} = I^2 R,$$

where $R$ is the resistance of the windings in ohms ($\Omega$). To keep the motor windings from overheating, the continuous current flowing through the motor must be limited. Accordingly, in continuous operation, the motor torque must be kept below a continuous torque limit $\tau_{\text{cont}}$ determined by thermal properties of the motor.

A simplified model of a DC motor, where all units are in the SI system, can be derived by equating the electrical power consumed by the motor $P_{\text{elec}} = IV$ in watts (W) with the mechanical power $P_{\text{mech}} = \tau w$ (also in W) and other power produced by the motor,

$$IV = \tau w + I^2 R + L \frac{dI}{dt} + \text{friction and other power},$$

where $V$ is the voltage applied to the motor in volts (V), $w$ is the angular speed of the motor in radians per second (1/s), and $L$ is the inductance due to the windings in henries (H). The terms on the right-hand side are the mechanical power produced by the motor, the power lost to heating the windings due to the resistance of the wires, the power consumed or produced by energizing or de-energizing the inductance of the windings (since the energy stored in an inductor is $\frac{1}{2}LI^2$, and power is the time derivative of energy), and power lost to friction in bearings, etc. Dropping this last term, replacing $\tau w$ by $k_t I w$, and dividing
8.9. Actuation, Gearing, and Friction

Figure 8.9: (Top) A cutaway view of a Maxon brushed DC motor with an encoder and gearhead. (Cutaway image courtesy of Maxon Precision Motors, Inc., maxonmotorusa.com.) The motor’s rotor consists of the windings, commutator ring, and shaft. Each of the several windings connects different segments of the commutator, and as the motor rotates, the two brushes slide over the commutator ring and make contact with different segments, sending current through one or more windings. One end of the motor shaft turns the encoder, and the other end is input to the gearhead. (Bottom) A simplified cross-section of the motor only, showing the stator (brushes, housing, and magnets) in dark gray and the rotor (windings, commutator, and shaft) in light gray.

by $I$, we get the voltage equation

$$V = k_t w + IR + L \frac{dI}{dt}. \quad (8.101)$$

Often Equation (8.101) is written with the electrical constant $k_e$ (with units of Vs) instead of the torque constant $k_t$, but in SI units (Vs or Nm/A), the numerical value of the two is identical; they represent the same constant property of the motor. So we prefer to use $k_t$.

The voltage term $k_t w$ in Equation (8.101) is called the back electromotive
force or back-emf for short, and it is what differentiates a motor from simply being a resistor and inductor in series. It also allows a motor, which we usually think of as converting electrical power to mechanical power, to be a generator, converting mechanical power to electrical power. If the motor’s electrical inputs are disconnected (so no current can flow) and the shaft is forced to turn by some external torque, you can measure the back-emf voltage $k_t w$ across the motor’s inputs.

For simplicity in the rest of this section, we ignore the $L dI/dt$ term. This assumption is exactly satisfied when the motor is operating at a constant current. With this assumption, Equation (8.101) can be rearranged to

$$w = \frac{1}{k_t} (V - IR) = \frac{V}{k_t} - \frac{R}{k_t^2} \tau,$$

expressing the speed $w$ as a linear function of $\tau$ (with a slope of $-R/k_t^2$) for a constant $V$. Now assume that the voltage across the motor is limited to the range $[-V_{\text{max}}, +V_{\text{max}}]$ and the current through the motor is limited to $[-I_{\text{max}}, +I_{\text{max}}]$, perhaps by the amplifier or power supply. Then the operating region of the motor in the torque-speed plane is as shown in Figure 8.10. Note that the signs of $\tau$ and $w$ are opposite in the second and fourth quadrants of this plane, and therefore the product $\tau w$ is negative. When the motor operates in these quadrants, it is actually consuming mechanical power, not producing mechanical power. The motor is acting like a damper.

Figure 8.10: The operating region of a current- and voltage-limited DC electric motor, and its continuous operating region.
Focusing on the first quadrant ($\tau \geq 0, w \geq 0, \tau w \geq 0$), the bounding line segment of the operating region is called the speed-torque curve. The no-load speed $w_0 = V_{\text{max}}/k_t$ at one end of the speed-torque curve is the speed the motor spins at when it is powered by $V_{\text{max}}$ but providing no torque. In this operating condition, the back-emf $k_tw$ is equal to the applied voltage, so there is no voltage remaining to create current (or torque). The stall torque $\tau_{\text{stall}} = k_tV_{\text{max}}/R$ at the other end of the speed-torque curve is achieved when the shaft is blocked from spinning, so there is no back-emf.

Figure 8.10 also indicates the continuous operating region where $|\tau| \leq \tau_{\text{cont}}$. The motor may be operated intermittently outside the continuous operating region, but extended operation outside the continuous operating region raises the possibility that the motor will overheat.

The motor's rated mechanical power is $P_{\text{rated}} = \tau_{\text{cont}}w_{\text{cont}}$, where $w_{\text{cont}}$ is the speed on the speed-torque curve corresponding to $\tau_{\text{cont}}$. Even if the motor’s rated power is sufficient for a particular application, the torque generated by a DC motor is typically too low to be useful. Gearing is therefore used to increase the torque while also decreasing the speed. For a gear ratio $G$, the output speed of the gearhead is

$$w_{\text{gear}} = \frac{w_{\text{motor}}}{G}.$$ 

For an ideal gearhead, zero power is lost in the conversion, i.e., $\tau_{\text{motor}}w_{\text{motor}} = \tau_{\text{gear}}w_{\text{gear}}$, which implies

$$\tau_{\text{gear}} = G\tau_{\text{motor}}.$$ 

In practice, some mechanical power is lost due to friction and impact between gear teeth, bearings, etc., so

$$\tau_{\text{gear}} = \eta G\tau_{\text{motor}},$$

where $\eta \leq 1$ is the efficiency of the gearhead.

Figure 8.11 shows the operating region of the motor from Figure 8.10 when the motor is geared by $G = 2$ (with $\eta = 1$). The achievable torque doubles, while the achievable speed shrinks by a factor of two. Since many DC motors are capable of no-load speeds of 10,000 rpm or more, robot joints often have gear ratios of 100 or more to achieve an appropriate compromise between speed and torque.

### 8.9.2 Apparent Inertia

The motor’s stator is attached to one link and the rotor is attached to another link, possibly through a gearhead. Therefore, when calculating the contribution of a motor to the masses and inertias of the links, the mass and inertia of the stator must be assigned to one link and the mass and inertia of the rotor must be assigned to the other link.

Consider a stationary link 0 with the stator of the joint 1 gearmotor attached to it. The rotational speed of joint 1, the output of the gearhead, is $\dot{\theta}$. Therefore
the motor’s rotor rotates at $\dot{\theta}$. The kinetic energy of the rotor is therefore

$$K = \frac{1}{2} I_{\text{rotor}} (G\dot{\theta})^2 = \frac{1}{2} G^2 I_{\text{rotor}} \dot{\theta}^2,$$

where $I_{\text{rotor}}$ is the rotor’s scalar inertia about the rotation axis and $G^2 I_{\text{rotor}}$ is the **apparent inertia** (often called the **reflected inertia**) of the rotor about the axis. In other words, if you were to grab link 1 and rotate it manually, the inertia contributed by the rotor would feel as if it were a factor $G^2$ larger than its actual inertia, due to the gearhead.

While the inertia $I_{\text{rotor}}$ is typically much less than the inertia $I_{\text{link}}$ of the rest of the link about the rotation axis, the apparent inertia $G^2 I_{\text{rotor}}$ may be on the order of, or even larger than, $I_{\text{link}}$.

One consequence as the gear ratio becomes large is that the inertia seen by joint $i$ becomes increasingly dominated by the apparent inertia of the rotor. In other words, the torque required of joint $i$ becomes relatively more dependent on $\ddot{\theta}_i$ than on other joint accelerations, i.e., the robot’s mass matrix becomes more diagonal. In the limit when the mass matrix has negligible off-diagonal components (and in the absence of gravity), the dynamics of the robot are decoupled—the dynamics at one joint have no dependence on the configuration or motion of other joints.

As an example, consider the 2R arm of Figure 8.1 with $L_1 = L_2 = m_1 = m_2 = 1$. Now assume that each of joint 1 and joint 2 has a motor of mass 1, with a stator of inertia 0.005 and a rotor of inertia 0.00125, and a gear ratio $G$ (with $\eta = 1$). With a gear ratio $G = 10$, the mass matrix is

$$M(\theta) = \begin{bmatrix} 4.13 + 2 \cos \theta_2 & 1.01 + \cos \theta_2 \\ 1.01 + \cos \theta_2 & 1.13 \end{bmatrix}.$$
With a gear ratio $G = 100$, the mass matrix is

$$M(\theta) = \begin{bmatrix} 16.5 + 2 \cos \theta_2 & 1.13 + \cos \theta_2 \\ 1.13 + \cos \theta_2 & 13.5 \end{bmatrix}.$$ 

The off-diagonal components are relatively less important for the second robot. The available joint torques of the second robot are ten times that of the first robot, so despite the increases in the mass matrix, the second robot is capable of significantly higher accelerations and end-effector payloads. The top speed of each joint of the second robot is ten times less than that of the first robot, however.

If the apparent inertia of the rotor is non-negligible relative to the inertia of the rest of the link, the Newton-Euler inverse dynamics algorithm must be modified to account for it. One approach is to treat the link as consisting of two separate bodies, the geared rotor driving the link and the rest of the link, each with its own center of mass and inertial properties (where the link includes the mass and inertia of the stator of any motor mounted to it); calculate the twist and acceleration of each during the forward iterations, accounting for the gearhead in calculating the rotor’s motion; calculate the wrench on the link as in Equation (8.52) and the wrench on the rotor during the backward iterations; express the rotor wrench in the link’s frame and sum the wrenches to get the total wrench $\mathbf{F}_i$ on the link; and project the total wrench to the motor axis as in Equation (8.53) to calculate the required motor torque $\tau$. The current command to the DC motor is then $I_{\text{com}} = \tau / (\eta \kappa G)$.

Taking into account apparent inertias, the recursive Newton-Euler inverse dynamics algorithm can be modified as follows:

**Initialization:** Attach a frame $\{0\}_L$ to the base, frames $\{1\}_L$ to $\{n\}_L$ to the centers of mass of links $\{1\}$ to $\{n\}$, and frames $\{1\}_R$ to $\{n\}_R$ to the centers of mass of rotors $\{1\}$ to $\{n\}$ as in Figure 8.12. Frame $\{n + 1\}_L$ is attached to the
end-effector and assumed fixed with respect to frame \( \{ n \}_L \). \( M_{(i-1)_L,i_L} \) is the configuration of \( \{ i \}_L \) relative to \( \{ i-1 \}_L \) when \( \theta_i = 0 \). \( A_i \) is the screw axis of joint \( i \) expressed in \( \{ i \}_L \). Similarly, \( S_i \) is the screw axis of joint \( i \) expressed in \( \{ i \}_R \). \( G_{i_L} \) is the 6 \( \times \) 6 spatial inertia matrix of link \( i \) that includes the inertia of the attached stator, and \( G_{i_R} \) is the 6 \( \times \) 6 spatial inertia matrix of rotor \( i \). \( G_i \) is the gear ratio of motor \( i \). Lastly, \( \nu_{0_L} \), \( \dot{\nu}_{0_L} \) and \( F_{(n+1)_L} \) are defined in the same way as \( \nu_0 \), \( \dot{\nu}_0 \) and \( F_{n+1} \) in Section 8.3.2.

**Forward iterations:** Given \( \theta, \dot{\theta}, \ddot{\theta} \), for \( i = 1 \) to \( n \) do

\[
T_{(i-1)_L,i_L} = M_{(i-1)_L,i_L} e^{[A_i] \theta_i} \tag{8.102}
\]

\[
\nu_{i_L} = Ad_{T_{(i-1)_L,i_L}}(\nu_{(i-1)_L}) + A_i \dot{\theta}_i \tag{8.103}
\]

\[
\dot{\nu}_{i_L} = Ad_{T_{(i-1)_L,i_L}}(\dot{\nu}_{(i-1)_L}) + [\nu_{i_L}, A_i] \dot{\theta}_i + A_i \ddot{\theta}_i \tag{8.104}
\]

\[
\nu_{i_R} = S_i G_i \dot{\theta}_i \tag{8.105}
\]

\[
\dot{\nu}_{i_R} = S_i G_i \ddot{\theta}_i. \tag{8.106}
\]

**Backward iterations:** For \( i = n \) to 1 do

\[
F_{i_L} = Ad^T_{T_{(i+1)_L,i_L}}(F_{(i+1)_L}) + G_{i_L} \dot{\nu}_{i_L} - ad^T_{\nu_{i_L}}(G_{i_L} \nu_{i_L}) \tag{8.107}
\]

\[
F_{i_R} = G_{i_R} \dot{\nu}_{i_R} \tag{8.108}
\]

\[
\tau_i = F^T_{i_L} A_i + G_i F^T_{i_R} S_i. \tag{8.109}
\]

Note that all of the above discussion neglects the mass and inertia due to the gearhead itself. If there is no gearing, then no modification to the original Newton-Euler inverse dynamics algorithm is necessary; the stator is attached to one link and the rotor is attached to another link. Robots constructed with a motor at each axis and no gearheads are sometimes called direct-drive robots.

### 8.9.3 Friction

The Lagrangian and Newton-Euler dynamics do not account for friction at the joints, but friction forces and torques in gearheads and bearings may be significant. Friction is a complex phenomenon which is the subject of considerable current research, and any friction model is a gross attempt to capture average behavior of the micromechanics of contact.

Friction models often include a static friction term and a velocity-dependent viscous friction term. The static friction term means that nonzero torque is required to cause the joint to begin to move. The viscous friction term indicates that the amount of friction torque increases with increasing velocity of the joint. See Figure 8.13 for some examples of velocity-dependent friction.

Other factors may contribute to the friction at a joint, including the loading of the joint bearings, the time the joint has been at rest, temperature, etc. The friction in a gearhead often increases as the gear ratio \( G \) increases.
8.10. Summary

- Given a set of generalized coordinates $\theta$ and generalized forces $\tau$, the Euler-Lagrange equations can be written

$$\tau = \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}} \right) - \frac{\partial L}{\partial \theta},$$

where $L(\theta, \dot{\theta}) = K(\theta, \dot{\theta}) - P(\theta)$, $K$ is the kinetic energy of the robot, and $P$ is the potential energy of the robot.

- The equations of motion of a robot can be written in the following equiv-

Figure 8.13: Examples of velocity-dependent friction models. (a) Viscous friction, $\tau_{\text{fric}} = b_{\text{viscous}} \dot{\theta}$. (b) Coulomb friction, $\tau_{\text{fric}} = b_{\text{static}} \text{sgn}(\dot{\theta})$. $\tau_{\text{fric}}$ can take any value in $[-b_{\text{static}}, b_{\text{static}}]$ at zero velocity. (c) Static plus viscous friction, $\tau_{\text{fric}} = b_{\text{static}} \text{sgn}(\dot{\theta}) + b_{\text{viscous}} \dot{\theta}$. (d) Static and kinetic friction, requiring $\tau_{\text{fric}} \geq |b_{\text{static}}|$ to initiate motion, and then $\tau_{\text{fric}} = b_{\text{kinetic}} \text{sgn}(\dot{\theta})$ during motion, where $b_{\text{static}} > b_{\text{kinetic}}$. (e) Static, kinetic, and viscous friction. (f) A friction law exhibiting the Stribeck effect—at low velocities, friction decreases as velocity increases.

8.9.4 Joint and Link Flexibility

In practice, a robot’s joints and links are likely to exhibit some flexibility. For example, the flexspline element of a harmonic drive gearhead achieves essentially zero backlash by being somewhat flexible. A model of a joint with harmonic drive gearing, then, could include a relatively stiff torsional spring between the motor’s rotor and the link the gearhead attaches to.

Similarly, links themselves are not infinitely stiff. The finite stiffness is exhibited as vibrations along the link.

Flexible joints and links introduce extra states to the dynamics of the robot, significantly complicating the dynamics and control. While many robots are designed to be stiff to try to minimize these complexities, in some cases this is impractical due to the extra link mass required to create the stiffness.

8.10 Summary
alent forms:

\[
\tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta})
\]

\[
= M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta)
\]

\[
= M(\theta)\ddot{\theta} + \dot{\theta}^T\Gamma(\theta)\dot{\theta} + g(\theta)
\]

\[
= M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + g(\theta),
\]

where \( M(\theta) \) is the \( n \times n \) symmetric positive-definite mass matrix, \( h(\theta, \dot{\theta}) \) is the sum of generalized forces due to gravity and quadratic velocity terms, \( c(\theta, \dot{\theta}) \) are quadratic velocity forces, \( g(\theta) \) are gravitational forces, \( \Gamma(\theta) \) is an \( n \times n \times n \) matrix of Christoffel symbols of the first kind obtained from partial derivatives of \( M(\theta) \) with respect to \( \theta \), and \( C(\theta, \dot{\theta}) \) is the \( n \times n \) Coriolis matrix whose \((i,j)\) entry is given by

\[
c_{ij}(\theta, \dot{\theta}) = \sum_{k=1}^{n} \Gamma_{ijk}(\theta)\dot{\theta}_k.
\]

If the end-effector of the robot is applying a wrench \( F_{\text{tip}} \) to the environment, the term \( J^T(\theta)F_{\text{tip}} \) should be added to the right-hand side of the robot’s dynamic equations.

- The symmetric positive-definite rotational inertia matrix of a rigid body is

\[
I_b = \begin{bmatrix}
I_{xx} & I_{xy} & I_{xz} \\
I_{xy} & I_{yy} & I_{yz} \\
I_{xz} & I_{yz} & I_{zz}
\end{bmatrix},
\]

where

\[
I_{xx} = \int_B \rho(x, y, z) dx \quad I_{xy} = \int_B x \rho(x, y, z) dx \\
I_{yy} = \int_B \rho(x, y, z) dx \quad I_{yx} = -\int_B xy \rho(x, y, z) dx \\
I_{yz} = \int_B z \rho(x, y, z) dx \quad I_{xz} = -\int_B yz \rho(x, y, z) dx
\]

\( B \) is the body, \( dV \) is a differential volume element, and \( \rho(x, y, z) \) is the density function.

- If \( I_b \) is defined in a frame \( \{b\} \) at the center of mass, with axes aligned with the principal axes of inertia, then \( I_b \) is diagonal.

- If \( \{b\} \) is at the center of mass but its axes are not aligned with the principal axes of inertia, there always exists a rotated frame \( \{c\} \) defined by the rotation matrix \( R_{bc} \) such that \( I_c = R_{bc}^T I_b \) is diagonal.

- If \( I_b \) is defined in a frame \( \{b\} \) at the center of mass, then \( I_q \), the inertia in a frame \( \{q\} \) aligned with \( \{b\} \), but displaced from the origin of \( \{b\} \) by \( q \in \mathbb{R}^3 \) in \( \{b\} \) coordinates, is

\[
I_q = I_b + m(q^T q I_3 - qq^T).
\]
• The spatial inertia matrix $\mathcal{G}_b$ expressed in a frame \{b\} at the center of mass is defined as the $6 \times 6$ matrix
$$
\mathcal{G}_b = \begin{bmatrix} I_b & 0 \\ 0 & m I \end{bmatrix}.
$$

• The Lie bracket of two twists $V_1$ and $V_2$ is
$$
[V_1, V_2] = [\text{ad}_{V_1}] V_2 = [\text{ad}_{V_1}] V_2,
$$
where
$$
[\text{ad}_V] = \begin{bmatrix} [\omega] & 0 \\ [v] & [\omega] \end{bmatrix} \in \mathbb{R}^{6 \times 6}.
$$

• The twist-wrench formulation of the rigid-body dynamics of a single rigid body is
$$
\mathcal{F}_b = \mathcal{G}_b \dot{V}_b - [\text{ad}_{V_b}]^T \mathcal{G}_b V_b.
$$

• The kinetic energy of a rigid body is
$$
\frac{1}{2} V_b^T \mathcal{G}_b V_b,
$$
and the kinetic energy of an open-chain robot is
$$
\frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta}.
$$

• The forward-backward Newton-Euler inverse dynamics algorithm is the following:

**Forward iterations:** Given $\theta, \dot{\theta}, \ddot{\theta}$, for $i = 1$ to $n$ do

$$
\begin{align*}
T_{i-1,i} &= M_{i-1,i} e^{[A_i] \theta_i} \\
\mathcal{V}_i &= \text{Ad}_{T_{i-1,i}}(\mathcal{V}_{i-1}) + A_i \dot{\theta}_i \\
\dot{\mathcal{V}}_i &= \text{Ad}_{T_{i-1,i}}(\dot{\mathcal{V}}_{i-1}) + [\mathcal{V}_i, A_i] \dot{\theta}_i + A_i \ddot{\theta}_i.
\end{align*}
$$

**Backward iterations:** For $i = n$ to 1 do

$$
\begin{align*}
\mathcal{F}_i &= \text{Ad}_{T_{i+1,i}}^T(\mathcal{F}_{i+1}) + \mathcal{G}_i \dot{\mathcal{V}}_i - [\text{ad}_{V_i}]^T \mathcal{G}_i V_i \\
\tau_i &= \mathcal{F}_i^T A_i.
\end{align*}
$$

• Let $J_{ib}(\theta)$ to be the Jacobian relating $\dot{\theta}$ to the body twist $\mathcal{V}_i$ in link $i$’s center-of-mass frame \{i\}. Then the mass matrix $M(\theta)$ of the manipulator can be expressed as
$$
M(\theta) = \sum_{i=1}^{n} J_{ib}^T(\theta) \mathcal{G}_i J_{ib}(\theta).
$$

• The forward dynamics problem is to solve
$$
M(\theta) \ddot{\theta} = \tau(t) - h(\theta, \dot{\theta}) - J^T(\theta) \mathcal{F}_\text{tip}
$$
for $\ddot{\theta}$, using any efficient solver of equations of the form $Ax = b$. 

• The robot’s dynamics \( M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}) \) can be expressed in the task space as
\[
\mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \eta(\theta, \mathcal{V}),
\]
where \( \mathcal{F} \) is the wrench applied to the end-effector, \( \mathcal{V} \) is the twist of the end-effector, and \( \mathcal{F}, \mathcal{V} \), and the Jacobian \( J(\theta) \) are all defined in the same frame. The task-space mass matrix \( \Lambda(\theta) \) and gravity and quadratic velocity forces \( \eta(\theta, \mathcal{V}) \) are
\[
\Lambda(\theta) = J^{-T}M(\theta)J^{-1}, \\
\eta(\theta, \mathcal{V}) = J^{-T}h(\theta, J^{-1}\mathcal{V}) - \Lambda(\theta)\dot{J}^{-1}\mathcal{V}.
\]

• Define two \( n \times n \) projection matrices of rank \( n - k \)
\[
P(\theta) = I - A^T(AM^{-1}A^T)^{-1}AM^{-1} \\
P_\theta(\theta) = M^{-1}PM = I - M^{-1}A^T(AM^{-1}A^T)^{-1}A
\]
corresponding to the \( k \) Pfaffian constraints acting on the robot, \( A(\theta)\dot{\theta} = 0 \), \( A \in \mathbb{R}^{k \times n} \). Then the \( n + k \) constrained equations of motion
\[
\tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}) + A^T(\theta)\lambda \\
A(\theta)\dot{\theta} = 0
\]
can be reduced to these equivalent forms by eliminating the Lagrange multipliers \( \lambda \):
\[
P \tau = P(M\dot{\theta} + h) \\
P_\theta \dot{\theta} = P_\theta M^{-1}(\tau - h).
\]
The matrix \( P \) projects away joint force/torque components that act on the constraints without doing work on the robot, and the matrix \( P_\theta \) projects away acceleration components that do not satisfy the constraints.

• An ideal gearhead (100% efficient) with a gear ratio \( G \) multiplies the torque at the output of a motor by a factor \( G \) and divides the speed by a factor \( G \), leaving the mechanical power unchanged. The inertia of the motor’s rotor about its axis of rotation, as apparent at the output of the gearhead, is \( G^2I_{\text{rotor}} \).

\[8.11\] **Software**

\[\text{adV} = \text{ad}(V)\]
Computes \([\text{ad}_V]\).

\[\text{taulist} = \text{InverseDynamics(thetalist,dthetalist,ddthetalist,g,Ftip, Mlist,Glist,Slist)}\]
Uses Newton-Euler inverse dynamics to compute the \( n \)-vector \( \tau \) of required joint forces/torques given \( \theta, \dot{\theta}, \ddot{\theta}, g, F_{\text{tip}} \), a list of transforms \( M_{i-1,i} \) specifying the configuration of the center-of-mass frame of link \( \{i\} \) relative to \( \{i-1\} \) when the robot is at its home position, a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame.

\[
M = \text{MassMatrix}(\theta, M, G, S)
\]

Computes the mass matrix \( M(\theta) \) given the joint configuration \( \theta \), a list of transforms \( M_{i-1,i} \), a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame.

\[
c = \text{VelQuadraticForces}(\theta, \dot{\theta}, M, G, S)
\]

Computes \( c(\theta, \dot{\theta}) \) given the joint configuration \( \theta \), joint velocities \( \dot{\theta} \), a list of transforms \( M_{i-1,i} \), a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame.

\[
\text{grav} = \text{GravityForces}(\theta, g, M, G, S)
\]

Computes \( g(\theta) \) given the joint configuration \( \theta \), the gravity vector \( g \), a list of transforms \( M_{i-1,i} \), a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame.

\[
J_{T \text{F-tip}} = \text{EndEffectorForces}(\theta, F_{\text{tip}}, M, G, S)
\]

Computes \( J^T \tau_{\text{F-tip}} \) given the joint configuration \( \theta \), the wrench \( F_{\text{tip}} \) applied by the end-effector, a list of transforms \( M_{i-1,i} \), a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame.

\[
\text{ddtheta} = \text{ForwardDynamics}(\theta, \dot{\theta}, \ddot{\theta}, g, F_{\text{tip}}, M, G, S)
\]

Computes \( \ddot{\theta} \) given the joint configuration \( \theta \), joint velocities \( \dot{\theta} \), joint forces/torques \( \tau \), the gravity vector \( g \), the wrench \( F_{\text{tip}} \) applied by the end-effector, a list of transforms \( M_{i-1,i} \), a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame.

\[
[\theta_{\text{next}}, \dot{\theta}_{\text{next}}] = \text{EulerStep}(\theta, \dot{\theta}, \ddot{\theta}, \delta t)
\]

Computes a first-order Euler approximation to \( \{\theta(t+\delta t), \dot{\theta}(t+\delta t)\} \) given the joint configuration \( \theta(t) \), joint velocities \( \dot{\theta}(t) \), joint accelerations \( \ddot{\theta}(t) \), and a timestep \( \delta t \).

\[
\text{taumat} = \text{InverseDynamicsTrajectory}(\theta, \dot{\theta}, \ddot{\theta}, g, F_{\text{tip}}, M, G, S)
\]

The variable \( \theta \) is an \( N \times n \) matrix of robot joint variables \( \theta \), where the \( i \)th row corresponds to the \( n \)-vector of joint variables \( \theta(t) \) at time \( t = (i-1)\delta t \), where \( \delta t \) is the timestep. The variables \( \dot{\theta}, \ddot{\theta}, \text{Ftipmat} \) similarly represent \( \dot{\theta}, \ddot{\theta}, \text{Ftip} \) as a function of time. Other inputs include the gravity vector \( g \), a list of transforms \( M_{i-1,i} \), a list of link spatial inertia matrices \( G_i \), and a list of joint screw axes \( S_i \) expressed in the base frame. This function computes an \( N \times n \) matrix \( \text{taumat} \) representing the joint forces/torques \( \tau(t) \).
required to generate the trajectory specified by $\theta(t)$ and $F_{\text{tip}}(t)$. Note that it is
not necessary to specify $\delta t$. The velocities $\dot{\theta}(t)$ and accelerations $\ddot{\theta}(t)$ should be
consistent with $\theta(t)$.

\begin{verbatim}
[thetamat,dthetamat] = ForwardDynamicsTrajectory(thetalist,
dthetalist,taumat,g,Ftipmat,Mlist,Glist,Slist,dt,intRes)
\end{verbatim}
This function numerically integrates the robot’s equations of motion using Euler
integration. The outputs are $N \times n$ matrices $\text{thetamat}$ and $\text{dthetamat}$, where
the $i$th rows correspond to the $n$-vectors $\theta((i-1)\delta t)$ and $\dot{\theta}((i-1)\delta t)$. Inputs
are the initial state $\theta(0)$ and $\dot{\theta}(0)$, an $N \times n$ matrix of joint forces/torques $\tau(t)$,
the gravity vector $g$, an $N \times n$ matrix of end-effector wrenches $F_{\text{tip}}(t)$, a list
of transforms $M_{i-1,i}$, a list of link spatial inertia matrices $G_i$, a list of joint
screw axes $S_i$ expressed in the base frame, the timestep $\delta t$, and the number of
integration steps to take during each timestep (a positive integer).

8.12 Notes and References
8.13 Exercises

1. Center of mass and link inertias for some standard shapes.

2. Given a linear mapping \( L : U \rightarrow V \) between two vector spaces \( U \) and \( V \), a choice of basis for \( U \) and \( V \) then leads to a particular matrix representation for \( L \); different bases for \( U \) and \( V \) then lead to different matrix representations for \( L \). It is also possible to think of the spatial inertia matrix as a coordinate representation for a rigid body’s kinetic energy, in the following sense. Given a body-fixed reference frame \( \{b\} \) attached to the rigid body, let \( \mathbf{V}_b \) denote the rigid body’s body twist, and \( G_b \in \mathbb{R}^{6\times 6} \) be the spatial inertia matrix expressed relative to \( \{b\} \). The kinetic energy, recall, is then given by

\[
\text{Kinetic Energy} = \frac{1}{2} \mathbf{V}_b^T G_b \mathbf{V}_b.
\]

Now, given a different choice of body-fixed reference frame, say \( \{a\} \), denote by \( \mathbf{V}_a \) and \( G_a \in \mathbb{R}^{6\times 6} \) the twist and spatial inertia matrix expressed relative to frame \( \{b\} \), respectively. Since the kinetic energy is a physical invariant and should not depend on the choice of reference frame coordinates, it follows that

\[
\frac{1}{2} \mathbf{V}_b^T G_b \mathbf{V}_b = \frac{1}{2} \mathbf{V}_a^T G_a \mathbf{V}_a.
\]

It is therefore customary to express kinetic energy in the following reference frame-invariant way:

\[
\text{Kinetic energy} = \langle \mathbf{V}, \mathbf{V} \rangle = \frac{1}{2} \mathbf{V}^T G \mathbf{V},
\]

with the understanding that both \( \mathbf{V} \) and \( G \) are always expressed with respect to the same body-fixed reference frame.

(a) Determine the appropriate transformation rule between \( G_a \) and \( G_b \) such that the kinetic energy as defined above is reference frame-invariant.

(b) Given a body-fixed reference frame \( \{b\} \), Recall that the spatial momentum expressed in frame \( \{b\} \), denoted \( \mathcal{P}_b \), was defined as

\[
\mathcal{P}_b = G_b \mathbf{V}_b.
\]

Suppose one wishes to express the spatial momentum in terms of another body-fixed reference frame \( \{a\} \). Determine the appropriate transformation rule between \( \mathcal{P}_b \) and \( \mathcal{P}_a \).

(c) The dynamic equations for a single rigid body expressed in terms of a body-fixed reference frame \( \{b\} \) are of the form

\[
\mathcal{F}_b = G_b \dot{\mathbf{V}}_b - [\text{ad}_{\mathbf{V}_b}]^T \mathcal{P}_b.
\]

Given a difference choice of body-fixed reference frame, say \( \{a\} \), such that \( T_{ab} \) is constant, show that the corresponding dynamic equation expressed relative
Dynamics of Open Chains

to the \{a\} frame is given by

$$\mathbf{F}_a = G_a \dot{V}_a - [\text{ad}_{V_a}]^T \mathbf{P}_a.$$  

As a result the rigid body dynamic equations can be expressed in the general form

$$\mathbf{F} = G \dot{V} - [\text{ad}_V]^T \mathbf{P},$$

without explicit reference to a body-fixed frame.

**Solution:** (a) Recalling that 

$$V_a = \text{Ad}_{T_{ab}}(V_b),$$

and writing the kinetic energy in terms of \(V_a\) as 

$$\frac{1}{2} V_a^T G_a V_a,$$

it then follows that

$$V_b^T G_b V_b = V_a^T G_a V_a = V_b^T [\text{Ad}_{T_{ab}}]^T G_a [\text{Ad}_{T_{ab}}] V_b$$

where

$$[\text{Ad}_{T_{ab}}] = \begin{pmatrix} R_{ab} & 0 \\ [p_{ab}] & R_{ab} \end{pmatrix}.$$  

From the above it follows that

$$G_b = [\text{Ad}_{T_{ab}}]^T G_a [\text{Ad}_{T_{ab}}]$$

and

$$G_a = [\text{Ad}_{T_{ba}}]^T G_b [\text{Ad}_{T_{ba}}].$$

(b) Recalling also that the kinetic energy can also be expressed in terms of the spatial momentum of the moving rigid body, i.e., as 

$$\frac{1}{2} \mathbf{P}_b^T V_b,$$

The transformation rule for \(\mathbf{P}_b\) now follows from the invariance of the kinetic energy: if \(\{a\}\) denotes a second body-fixed frame as before, then clearly 

$$\mathbf{P}_b^T V_b = \mathbf{P}_a^T \text{Ad}_{T_{ba}}(V_a),$$

implying that \(\mathbf{P}_a\) and \(\mathbf{P}_b\) must be related by

$$\mathbf{P}_a = \text{Ad}_{T_{ba}}^T(\mathbf{P}_b).$$

(The above also follows from \(G_a V_a = [\text{Ad}_{T_{ba}}]^T G_b [\text{Ad}_{T_{ba}}] V_a = [\text{Ad}_{T_{ba}}]^T G_b V_b.\)  

The coordinate transformation rule for \(\mathbf{P}_b\) is thus given by the transpose of the adjoint map \(\text{Ad}^T\), which is the same as for the wrench \(\mathbf{F}_b.\)

3. Show that the time derivative of the mass matrix, \(\dot{M}(\theta)\), can be explicitly written

$$\dot{M} = \mathbf{A}^T \mathbf{L}^T \Gamma^T [\text{ad}_{\dot{\theta}}]\mathbf{L}^T \mathbf{G} \mathbf{L} \mathbf{A} + \mathbf{A}^T \mathbf{L}^T \mathbf{G} \mathbf{L} [\text{ad}_{\dot{\theta}}] \Gamma \mathbf{A}.$$  

4. In the dynamic equations for an open chain, i.e.,

$$\tau = M(\theta) \ddot{\theta} + c(\theta, \dot{\theta}) + n(\theta),$$
c(θ, ˙θ) is often referred to as the Coriolis terms. Earlier in Equation (8.78) we showed that c(θ, ˙θ) can expressed as c(θ, ˙θ) = A(θ, ˙θ) ˙θ, where
\[ A(θ, ˙θ) = AT LT (GL [ad_A ˙θ]I - [ad_V]T G) LA ∈ R^{n×n}. \] (8.110)
In the Lagrangian formulation we further showed that the Coriolis terms can be expressed as c(θ, ˙θ) = C(θ, ˙θ) ˙θ, where the \((k, j)\) entry of \(C(θ, ˙θ) ∈ R^{n×n}\) is given by
\[ C_{kj} = \sum_{i=1}^{n} \Gamma_{ijk} ˙θ_i, \]
where the \(\Gamma_{ijk}\) are the Christoffel symbols of the first kind associated with the mass matrix \(M(θ) = (m_{ij}) ∈ R^{n×n}\). It turns out that the matrix \(\dot{M} - 2C\) with \(C\) defined as above is skew-symmetric; this property turns out to be important in proving the stability of certain dynamic model-based control laws (our later chapter on robot control also relies upon this result).
(a) Show that \(\dot{M} - 2C\) is skew-symmetric.
(b) Show that \(\dot{M} - 2A\) with \(A\) defined as above is not skew-symmetric.
(c) *(See Ploen thesis)*

5. Efficient recursive formulas for evaluating the mass matrix.

6.

7. Assuming the joint space dynamics are of the form
\[ \tau = M(θ) ˙θ + C(θ, ˙θ) + N(θ), \] (8.111)
and the differential forward kinematics is of the form \(V = J(θ) ˙θ\), where \(V ∈ se(3)\) denotes the velocity of the tip frame, the corresponding dynamics in \(SE(3)\) assume the form
\[ F = \dot{M}V + \dot{C} + \dot{N}, \] (8.112)
where \(F = J^{-T}τ ∈ se(3)^*\) corresponds to the moment-force pair at the tip, and
\[ \dot{M} = (JM^{-1}J^T)^{-1} \] (8.113)
\[ \dot{C} = J^{-T}M \left( \frac{d}{dt}J^{-1} \right) V + J^{-T}C \] (8.114)
\[ \dot{N} = J^{-T}N. \] (8.115)
The above can be verified by noting that \(\dot{V} = J\ddot{θ} + J\dot{θ}\), and replacing \(\ddot{θ}\) in the original dynamics equations by \(\ddot{θ} = J^{-1}\dot{V} + (\frac{d}{dt}J^{-1})V\).
The physical meaning that can be associated with these equations is that, assuming zero joint torques, and given an external input force $F(t) \in se(3)^*$ applied at the tip frame, the motion of the tip frame is then governed by (8.112). Note the dependence of $\hat{M}(\theta)$, $C(\theta, \dot{\theta})$, and $\hat{N}$ on $\theta$ and $\dot{\theta}$; if $\theta$ and $\dot{\theta}$ in (8.112) are replaced by their inverse kinematics solutions $\theta = \theta^{-1}(X)$ and $\dot{\theta} = J^{-1}(\theta)\dot{V}$, then a differential equation strictly in terms of the tip frame parameters $X \in SE(3)$ and $V \in se(3)$ can be obtained. In practice, however, integrating these equations is cumbersome; to obtain the resulting tip frame trajectory $X(t)$, it is far more preferable to integrate $J^T F = \hat{M}(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + \hat{N}(\theta)$ to obtain $\theta(t)$, and to substitute this into the forward kinematics relation to obtain $X(t) = f(\theta(t))$.

8. The 2R open chain robot of Figure 8.15(a) is shown in its zero configuration. Assuming the mass of each link is concentrated at the tip and neglecting its thickness, the robot can be modeled as shown in Figure 8.15(b). Assume $m_1 = m_2 = 2$, $L_1 = L_2 = 1$, $g = 10 \text{ m/sec}$, and the link inertias $\mathcal{I}_1$ and $\mathcal{I}_2$ (expressed in their respective link body frames $\{b\}_1$ and $\{b\}_2$) are

$$
\mathcal{I}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix}, \quad \mathcal{I}_2 = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix},
$$

Assume the total potential energy of the robot at its zero configuration is zero.

(a) For arbitrary $\theta_1$, $\theta_2$, $\dot{\theta}_1$, $\dot{\theta}_2$, determine the body twists $\mathcal{V}_{b_1}$ and $\mathcal{V}_{b_2}$ of the link frames $\{b\}_1$ and $\{b\}_2$.

(b) For arbitrary $\theta_1$, $\theta_2$, $\dot{\theta}_1$, $\dot{\theta}_2$, determine the kinetic energies $E_{k_1}$ and $E_{k_2}$ of the two links, and the total potential energy $E_p$ of the robot.

(c) Use the Lagrangian method to derive the dynamic equations, and determine the input torques $\tau_1$ and $\tau_2$ when $\theta_1 = \theta_2 = \pi/4$ and $\dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_1 = \dot{\theta}_2 = 0$.

(d) Determine the mass matrix $M(\theta)$ when $\theta_1 = \theta_2 = \pi/4$.

Solution:
See figure.

9. Intuitively explain the shapes of the end-effector force ellipsoids in Figure 8.4 based on the the point masses and the Jacobians.

10. Prove the parallel axis theorem, and then prove Steiner’s theorem. If it is helpful, you may assume the rigid body consists of point masses and that the axes of $\{b\}$ are aligned with the principal axes of inertia of the body, but then you should show that these assumptions are not needed.

11. Provide a general expression for the rotational inertia matrix $\mathcal{I}_c$ in a frame
\[ T_{bc} = (R_{bc}, p) \] and \( I_b \) expressed in a center-of-mass frame \( \{b\} \).

12. Consider a cast iron dumbbell consisting of a cylinder connecting two solid spheres at either end of the cylinder. The density of the dumbbell is 7500 kg/m\(^3\). The cylinder has a diameter of 4 cm and a length of 20 cm. Each sphere has a diameter of 20 cm. (a) Find the approximate rotational inertia matrix \( I_b \) in a frame \( \{b\} \) at the center of mass with axes aligned with the principal axes of inertia of the dumbbell. (b) Write the spatial inertia matrix \( G_b \).

13. Program a function to efficiently calculate \( h(\theta, \dot{\theta}) = c(\theta, \dot{\theta}) + g(\theta) \) using Newton-Euler inverse dynamics.

14. (a) Conceptually develop a computationally efficient algorithm for determining the mass matrix \( M(\theta) \) using Equation (8.56). (b) Implement this algorithm.

15. The function \texttt{InverseDynamicsTrajectory} requires the user to enter not only a time sequence of joint variables \texttt{thetamat}, but also a time sequence of joint velocities \texttt{dthetamat} and accelerations \texttt{ddthetamat}. Instead, the function could use numerical differencing to approximately find the joint velocities and accelerations at each timestep, using only \texttt{thetamat}. Write an alternative \texttt{InverseDynamicsTrajectory} function that does not require the user to enter \texttt{dthetamat} and \texttt{ddthetamat}. Verify that it yields similar results.

16. Give the steps that rearrange Equation (8.97) to get Equation (8.99). Remember that \( P(\theta) \) is not full rank and cannot be inverted.

17. Give the equations that would convert the joint and link descriptions in a robot’s URDF file to the data \texttt{Mlist}, \texttt{Glist}, and \texttt{Slist}, suitable for using with...
18. Consider a motor with rotor inertia $I_{\text{rotor}}$ connected through a gearhead of gear ratio $G$ to a load with a scalar inertia $I_{\text{link}}$ about the rotation axis. The load and motor are said to be \textbf{inertia matched} if, for any given torque $\tau_m$ at the motor, the acceleration of the load is maximized. The acceleration of the load can be written

$$\ddot{\theta} = \frac{G \tau_m}{I_{\text{link}} + G^2 I_{\text{rotor}}}.$$ 

Solve for the inertia matching gear ratio $\sqrt{\frac{I_{\text{link}}}{I_{\text{rotor}}}}$ by solving $d\dot{\theta}/dG = 0$. Show your work.
Chapter 9

Trajectory Generation

During robot motion, the robot controller is provided with a steady stream of goal positions and velocities to track. This specification of the robot position as a function of time is called a trajectory. In some cases, the trajectory is completely specified by the task—for example, the end-effector may be required to track a known moving object. In other cases, as when the task is simply to move from one position to another in a given time, we have freedom to design the trajectory to meet these constraints. This is the domain of trajectory planning. The trajectory should be a sufficiently smooth function of time, and it should respect any given limits on joint velocities, accelerations, or torques.

In this chapter we consider a trajectory as the combination of a path, a purely geometric description of the sequence of configurations achieved by the robot, and a time scaling, which specifies the times when those configurations are reached. We consider three cases: point-to-point straight-line trajectories in both joint space and task space; trajectories passing through a sequence of timed via points; and minimum-time trajectories along specified paths. Finding paths that avoid obstacles is left to Chapter 10.

9.1 Definitions

A path \( \theta(s) \) maps a scalar path parameter \( s \), assumed to be zero at the start of the path and one at the end, to a point in the robot’s configuration space \( \Theta \), \( \theta : [0, 1] \to \Theta \). As \( s \) increases from 0 to 1, the robot moves along the path. Sometimes \( s \) is taken to be time, and is allowed to vary from time \( s = 0 \) to the total motion time \( s = T \), but it is often useful to separate the role of the geometric path parameter \( s \) from the time parameter \( t \). A time scaling \( s(t) \) assigns a value \( s \) to each time \( t \in [0, T] \), \( s : [0, T] \to [0, 1] \).

Together a path and time scaling define a trajectory \( \theta(s(t)) \), or \( \theta(t) \) for short. Using the chain rule, the velocity and acceleration along the trajectory can be
Trajectory Generation

written as

\[ \dot{\theta} = \frac{d\theta}{ds} \dot{s} \]  
(9.1)

\[ \ddot{\theta} = \frac{d\theta}{ds} \ddot{s} + \frac{d^2\theta}{ds^2} s^2. \]  
(9.2)

To ensure that the robot’s acceleration (and therefore dynamics) are well defined, each of \( \theta(s) \) and \( s(t) \) must be twice differentiable.

### 9.2 Point-to-Point Trajectories

The simplest type of motion is from rest at one configuration to rest at another. We call this a point-to-point motion. The simplest type of path for point-to-point motion is a straight line. Straight-line paths and their time scalings are discussed below.

#### 9.2.1 Straight-Line Paths

A “straight line” from a start configuration \( \theta_{\text{start}} \) to an end configuration \( \theta_{\text{end}} \) could be defined in joint space or in task space. The advantage of a straight-line path from \( \theta_{\text{start}} \) to \( \theta_{\text{end}} \) in joint space is simplicity: since joint limits typically take the form \( \theta_{i,\text{min}} \leq \theta_i \leq \theta_{i,\text{max}} \) for each joint \( i \), the allowable joint configurations form a convex set \( \Theta_{\text{free}} \) in joint space, so the straight line between any two endpoints in \( \Theta_{\text{free}} \) also lies in \( \Theta_{\text{free}} \). The straight line can be written

\[ \theta(s) = \theta_{\text{start}} + s(\theta_{\text{end}} - \theta_{\text{start}}), \quad s \in [0, 1] \]  
(9.3)

with derivatives

\[ \frac{d\theta}{ds} = \theta_{\text{end}} - \theta_{\text{start}} \]  
(9.4)

\[ \frac{d^2\theta}{ds^2} = 0. \]  
(9.5)

Straight lines in joint space generally do not yield straight-line motion of the end-effector in task space. If task space straight-line motions are desired, the start and end configurations can be specified by \( X_{\text{start}} \) and \( X_{\text{end}} \) in task space. If \( X_{\text{start}} \) and \( X_{\text{end}} \) are represented by a minimum set of coordinates, then a straight line is defined as \( X(s) = X_{\text{start}} + s(X_{\text{end}} - X_{\text{start}}), \ s \in [0, 1] \). Compared to joint coordinates, however, the following are issues that must be addressed:

- If the path passes near a kinematic singularity, joint velocities may become unreasonably large for almost all time scalings of the path.

- Since the robot’s reachable task space may not be convex in \( X \) coordinates, some points on a straight line between two reachable endpoints may not be reachable (Figure 9.1).
In addition to the issues above, if $X_{\text{start}}$ and $X_{\text{end}}$ are represented as elements of $\mathbb{SE}(3)$ instead of as a minimum set of coordinates, then there is the question of how to define a “straight” line in $\mathbb{SE}(3)$. A configuration of the form $X_{\text{start}} + s(X_{\text{end}} - X_{\text{start}})$ does not generally lie in $\mathbb{SE}(3)$.

One option is to use the screw motion (simultaneous rotation about and translation along a fixed screw axis) that moves the robot’s end-effector from $X_{\text{start}} = X(0)$ to $X_{\text{end}} = X(1)$. To derive this $X(s)$, we can write the start and end configurations explicitly in the $\{s\}$ frame as $X_{s,\text{start}}$ and $X_{s,\text{end}}$ and use our subscript cancellation rule to express the end configuration in the start frame:

$$X_{\text{start,end}} = X_{\text{start},s}X_{s,\text{end}} = X_{s,\text{start}}^{-1}X_{s,\text{end}}.$$  

Then $\log(X_{s,\text{start}}^{-1}X_{s,\text{end}})$ is the matrix representation of the twist, expressed in the $\{\text{start}\}$ frame, that takes $X_{\text{start}}$ to $X_{\text{end}}$ in unit time. The path can therefore be written as

$$X(s) = X_{\text{start}} \exp(\log(X_{s,\text{start}}^{-1}X_{s,\text{end}})s)$$  

where $X_{\text{start}}$ is post-multiplied by the matrix exponential since the twist is
Figure 9.2: A path following a constant screw motion vs. a decoupled path where the frame origin follows a straight line and the angular velocity is constant.

represented in the \{\text{start}\} frame, not the fixed world frame \{\text{s}\}.

This screw motion provides a “straight-line” motion in the sense that the screw axis is constant. The origin of the end-effector does not generally follow a straight line in Cartesian space, however, since it follows a screw motion. It may be preferable to decouple the rotational motion from the translational motion. Writing $X = (R, p)$, we can define the path

\[ p(s) = p_{\text{start}} + s(p_{\text{end}} - p_{\text{start}}) \quad (9.7) \]

\[ R(s) = R_{\text{start}} \exp(\log(R_{\text{start}} R_{\text{end}}) s) \quad (9.8) \]

where the frame origin follows a straight line while the axis of rotation is constant in the body frame. Figure 9.2 illustrates a screw path and a decoupled path for the same $X_{\text{start}}$ and $X_{\text{end}}$.

9.2.2 Time Scaling a Straight-Line Path

A time scaling $s(t)$ of a path should ensure that the motion is appropriately smooth and that any constraints on robot velocity and acceleration are satisfied. For a straight-line path in joint space of the form (9.3), the time-scaled joint velocities and accelerations are $\dot{\theta} = \dot{s}(\theta_{\text{end}} - \theta_{\text{start}})$ and $\ddot{\theta} = \ddot{s}(\theta_{\text{end}} - \theta_{\text{start}})$, respectively. For a straight-line path in task space, parametrized by minimum coordinates $X \in \mathbb{R}^m$, simply replace $\theta$, $\dot{\theta}$, and $\ddot{\theta}$ by $X$, $\dot{X}$, and $\ddot{X}$.

9.2.2.1 Polynomial Time Scaling

Third-order Polynomials A convenient form for the time scaling $s(t)$ is a cubic polynomial of time,

\[ s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (9.9) \]

A point-to-point motion in time $T$ imposes the initial constraints $s(0) = \dot{s}(0) = 0$ and the terminal constraints $s(T) = 1$ and $\dot{s}(T) = 0$. Evaluating Equation (9.9) and its derivative

\[ \dot{s}(t) = a_1 + 2a_2 t + 3a_3 t^2 \quad (9.10) \]
9.2. Point-to-Point Trajectories

\[ s(t), \frac{ds}{dt}, \frac{d^2s}{dt^2} \]

Figure 9.3: Plots of \( s(t) \), \( \dot{s}(t) \), and \( \ddot{s}(t) \) for a third-order polynomial time scaling.

at \( t = 0 \) and \( t = T \) and solving the four constraints for \( a_0, \ldots, a_3 \), we find

\[
\begin{align*}
    a_0 &= 0, \quad a_1 = 0, \quad a_2 = \frac{3}{T^2}, \quad a_3 = -\frac{2}{T^3}.
\end{align*}
\]

Plots of \( s(t) \), \( \dot{s}(t) \), and \( \ddot{s}(t) \) are shown in Figure 9.3.

Plugging \( s = a_2t^2 + a_3t^3 \) into Equation (9.3) yields

\[
\begin{align*}
    \theta(t) &= \theta_{\text{start}} + \left( \frac{3t^2}{T^2} - \frac{2t^3}{T^3} \right) (\theta_{\text{end}} - \theta_{\text{start}}) \quad (9.11) \\
    \dot{\theta}(t) &= \frac{6t}{T^2} - \frac{6t^2}{T^3} (\theta_{\text{end}} - \theta_{\text{start}}) \quad (9.12) \\
    \ddot{\theta}(t) &= \frac{6}{T^2} - \frac{12t}{T^3} (\theta_{\text{end}} - \theta_{\text{start}}). \quad (9.13)
\end{align*}
\]

The maximum joint velocities are achieved at the halfway point of the motion \( t = T/2 \):

\[
\dot{\theta}_{\text{max}} = \frac{3}{2T} (\theta_{\text{end}} - \theta_{\text{start}}).
\]

The maximum joint accelerations and decelerations are achieved at \( t = 0 \) and \( t = T \):

\[
\begin{align*}
    \ddot{\theta}_{\text{max}} &= \left| \frac{6}{T^2} (\theta_{\text{end}} - \theta_{\text{start}}) \right|, \quad \ddot{\theta}_{\text{min}} = -\left| \frac{6}{T^2} (\theta_{\text{end}} - \theta_{\text{start}}) \right|.
\end{align*}
\]

If there are known limits on the maximum joint velocities \( |\dot{\theta}| \leq \dot{\theta}_{\text{limit}} \) and maximum joint accelerations \( |\ddot{\theta}| \leq \ddot{\theta}_{\text{limit}} \), these bounds can be checked to see if the requested motion time \( T \) is feasible. Alternatively, \( T \) could be solved for to find the minimum possible motion time that satisfies the most restrictive velocity or acceleration constraint.

**Fifth-order Polynomials**  Because the third-order time scaling does not constrain the endpoint path accelerations \( \ddot{s}(0) \) and \( \ddot{s}(T) \) to be zero, the robot is asked to achieve a discontinuous jump in acceleration at both \( t = 0 \) and \( t = T \). This implies infinite jerk, the derivative of acceleration, which may cause vibration of the robot.
Figure 9.4: Plots of $s(t)$, $\dot{s}(t)$, and $\ddot{s}(t)$ for a fifth-order polynomial time scaling.

Figure 9.5: Plots of $s(t)$ and $\dot{s}(t)$ for a trapezoidal motion profile.

One solution is to constrain the endpoint accelerations to $\ddot{s}(0) = \ddot{s}(T) = 0$. The addition of these two constraints requires the addition of two more design freedoms in the polynomial, yielding a quintic polynomial of time, $s(t) = a_0 + \ldots + a_5 t^5$. We can use the six terminal position, velocity, and acceleration constraints to solve uniquely for $a_0 \ldots a_5$ (Exercise 5), which yields a smoother motion with a higher maximum velocity than a cubic time scaling. A plot of the time scaling is shown in Figure 9.4.

**9.2.2.2 Trapezoidal Motion Profiles**

Trapezoidal time scalings are quite common in motor control, particularly for the motion of a single joint, and they get their name from their velocity profiles. The point-to-point motion consists of a constant acceleration phase $\ddot{s} = a$ of time $t_a$, followed by a constant velocity phase $\dot{s} = v$ of time $t_v = T - 2t_a$, followed by a constant deceleration phase $\ddot{s} = -a$ of time $t_a$. The resulting $\dot{s}$ profile is a trapezoid and the $s$ profile is the concatenation of a parabola, linear segment, and parabola as a function of time (Figure 9.5).

The trapezoidal time scaling is not as smooth as the cubic time scaling, but it has the advantage that if there are known limits on joint velocities $\dot{\theta}_{\text{limit}} \in \mathbb{R}^n$ and joint accelerations $\ddot{\theta}_{\text{limit}} \in \mathbb{R}^n$, the trapezoidal motion using the largest $v$
and \( a \) satisfying

\[
|\theta_{\text{end}} - \theta_{\text{start}}| \leq \dot{\theta}_{\text{limit}}
\]

\[
|\theta_{\text{end}} - \theta_{\text{start}}| \leq \ddot{\theta}_{\text{limit}}
\]

is the fastest straight-line motion possible. (See Exercise 8.)

If \( v^2/a > 1 \), the robot never reaches the velocity \( v \) during the motion (Exercise 10). The three-phase accelerate-coast-decelerate motion becomes a two-phase accelerate-decelerate “bang-bang” motion, and the trapezoidal profile \( \dot{s}(t) \) in Figure 9.5 becomes a triangle.

Assuming that \( v^2/a \leq 1 \), the trapezoidal motion is fully specified by \( v, a, t_a, \) and \( T \), but only two of these can be specified independently, since they must satisfy \( s(T) = 1 \) and \( v = at_a \). It is unlikely that we would specify \( t_a \) independently, so we can eliminate it from the equations of motion by the substitution \( t_a = v/a \). The motion profile during the three stages (acceleration, coast, deceleration) can then be written in terms of \( v, a, \) and \( T \) as

\[
0 \leq t \leq \frac{v}{a} : \quad \ddot{s}(t) = a \tag{9.16}
\]

\[
\dot{s}(t) = at \tag{9.17}
\]

\[
s(t) = \frac{1}{2}at^2 \tag{9.18}
\]

\[
\frac{v}{a} < t \leq T - \frac{v}{a} : \quad \ddot{s}(t) = 0 \tag{9.19}
\]

\[
\dot{s}(t) = v \tag{9.20}
\]

\[
s(t) = vt - \frac{v^2}{2a} \tag{9.21}
\]

\[
T - \frac{v}{a} < t \leq T : \quad \ddot{s}(t) = -a \tag{9.22}
\]

\[
\dot{s}(t) = a(T - t) \tag{9.23}
\]

\[
s(t) = \frac{2avT - 2v^2 - a^2(t - T)^2}{2a} \tag{9.24}
\]

Since only two of \( v, a, \) and \( T \) can be chosen independently, we have three options:

- Choose \( v \) and \( a \) such that \( v^2/a \leq 1 \), assuring a three-stage trapezoidal profile, and solve \( s(T) = 1 \) (Equation (9.24)) for \( T \):

\[
T = \frac{a + v^2}{va}.
\]

If \( v \) and \( a \) correspond to the highest possible joint velocities and accelerations, this is the minimum possible time for the motion.
Figure 9.6: Plot of $\dot{s}(t)$ for an S-curve motion profile consisting of seven stages: (1) constant positive jerk, (2) constant acceleration, (3) constant negative jerk, (4) constant velocity, (5) constant negative jerk, (6) constant deceleration, and (7) constant positive jerk.

- Choose $v$ and $T$ such that $2 \geq vT > 1$, assuring a three-stage trapezoidal profile and that the top speed $v$ is sufficient to reach $s = 1$ in time $T$, and solve $s(T) = 1$ for $a$:

$$a = \frac{v^2}{vT - 1}.$$ 

- Choose $a$ and $T$ such that $aT^2 \geq 4$, assuring that the motion is completed in time, and solve $s(T) = 1$ for $v$:

$$v = \frac{1}{2} \left( aT - \sqrt{a} \sqrt{aT^2 - 4} \right).$$

### 9.2.2.3 S-Curve Time Scalings

Just as cubic polynomial time scalings lead to infinite jerk at the beginning and end of the motion, trapezoidal motions cause discontinuous jumps in acceleration at $t \in \{0, t_a, T - t_a, T\}$. A solution is a smoother S-curve time scaling, a popular motion profile in motor control because it avoids vibrations or oscillations induced by step changes in acceleration. An S-curve time scaling consists of seven stages: (1) constant jerk $s^{(3)} = J$ until a desired acceleration $\ddot{s} = a$ is achieved; (2) constant acceleration until the desired $\ddot{s} = v$ is being approached; (3) constant negative jerk $-J$ until $\ddot{s}$ equals zero exactly at the time $\dot{s}$ reaches $v$; (4) coasting at constant $v$; (5) constant negative jerk $-J$; (6) constant deceleration $-a$; and (7) constant positive jerk $J$ until $\ddot{s}$ and $\dot{s}$ reach zero exactly at the time $s$ reaches 1.

The $\dot{s}(t)$ profile for an S-curve is shown in Figure 9.6.

Given some subset of $v$, $a$, $J$, and the total motion time $T$, algebraic manipulation reveals the switching time between stages and conditions that ensure that all seven stages are actually achieved, similar to the case of the trapezoidal motion profile.
9.3 Polynomial Via Point Trajectories

If the goal is to have the robot joints pass through a series of via points at specified times, without a strict specification on the shape of path between consecutive points, a simple solution is to use polynomial interpolation to directly find joint histories \( \theta(t) \) without first specifying a path \( \theta(s) \) and then a time scaling \( s(t) \) (Figure 9.7).

Let the trajectory be specified by \( k \) via points, with the start point occurring at \( T_1 = 0 \) and the final point at \( T_k = T \). Since each joint history is interpolated individually, we focus on a single joint variable and call it \( \beta \) to avoid proliferation of subscripts. At each via point \( i \in \{1,\ldots,k\} \), the user specifies the desired position \( \beta(T_i) = \beta_i \) and velocity \( \dot{\beta}(T_i) = \dot{\beta}_i \). The trajectory has \( k-1 \) segments, and the duration of segment \( j \in \{1,\ldots,k-1\} \) is \( \Delta T_j = T_{j+1} - T_j \). The joint trajectory during segment \( j \) is expressed as the third-order polynomial

\[
\beta(T_j + \Delta t) = a_{j0} + a_{j1} \Delta t + a_{j2} \Delta t^2 + a_{j3} \Delta t^3 \quad (9.25)
\]

in terms of the time \( \Delta t \) elapsed in segment \( j \), where \( 0 \leq \Delta t \leq \Delta T_j \). Segment \( j \) is subject to the four constraints

\[
\begin{align*}
\beta(T_j) &= \beta_j \\
\dot{\beta}(T_j) &= \dot{\beta}_j \\
\beta(T_j + \Delta T_j) &= \beta_{j+1} \\
\dot{\beta}(T_j + \Delta T_j) &= \dot{\beta}_{j+1}.
\end{align*}
\]
Solving these constraints for \( a_{j0}, \ldots, a_{j3} \) yields

\[
\begin{align*}
    a_{j0} &= \beta_j \\
    a_{j1} &= \dot{\beta}_j \\
    a_{j2} &= \frac{3\beta_{j+1} - 3\beta_j - 2\dot{\beta}_j \Delta T_j - \dot{\beta}_{j+1} \Delta T_j}{\Delta T_j^2} \\
    a_{j3} &= \frac{2\beta_j + (\dot{\beta}_j + \beta_{j+1}) \Delta T_j - 2\dot{\beta}_{j+1}}{\Delta T_j^3}.
\end{align*}
\]

Figure 9.8 shows the time histories for the interpolation of Figure 9.7(a). In this 2D \((x,y)\) coordinate space, the via points \(1 \ldots 4\) occur at times \(T_1 = 0\), \(T_2 = 1\), \(T_3 = 2\), and \(T_4 = 3\). The via points are at \((0,0)\), \((0,1)\), \((1,1)\), and \((1,0)\) with velocities \((0,0)\), \((1,0)\), \((0,-1)\), and \((0,0)\).

Two issues are worth mentioning:

- The quality of the interpolated trajectories is improved by “reasonable” combinations of via point times and via point velocities. For example, if the user wants to specify a via point location and time, but not the velocity, a heuristic could be used to choose a via velocity based on the times and coordinate vectors to the via points before and after the via in question. As an example, the trajectory of Figure 9.7(b) is smoother than the trajectory of Figure 9.7(a).

- Cubic via point interpolation ensures that velocities are continuous at via points, but not accelerations. The approach is easily generalized to use fifth-order polynomials and specifications of the accelerations at the via points, at the cost of increased complexity of the solution.

If only two points are used (the start and end point), and the velocities at each are zero, the resulting trajectory is identical to the straight-line cubic polynomial time-scaled trajectory discussed in Section 9.2.2.1.
9.4 Time-Optimal Time Scaling

In the case that the path $\theta(s)$ is fully specified by the task or an obstacle-avoiding path planner (e.g., Figure 9.9), the trajectory planning problem reduces to finding a time scaling $s(t)$. One could choose the time scaling to minimize energy consumed while meeting a time constraint, or to prevent spilling a glass of water the robot is carrying. One of the most useful time scalings, however, is the one that minimizes the time of motion along the path, subject to the robot’s actuator limits. Such time-optimal trajectories maximize the robot’s productivity.

While the trapezoidal time scalings of Section 9.2.2.2 can yield time-optimal trajectories, this is only under the assumption of straight-line motions and constant maximum acceleration $a$ and maximum coasting velocity $v$. For most robots, because of state-dependent joint actuator limits and the state-dependent dynamics
\[ M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = \tau, \] (9.30)
the maximum available velocities and accelerations change along the path.

Figure 9.9: A path planner has returned a semicircular path of radius $R$ around an obstacle in $(x, y)$ space for a robot with two prismatic joints. The path can be represented as a function of a path parameter $s$ as $x(s) = x_c + R \cos s\pi$ and $y(s) = y_c - R \sin s\pi$ for $s \in [0, 1]$. For a 2R robot, inverse kinematics would be used to express the path as a function of $s$ in joint coordinates.

There are many other methods for interpolating a set of via points. For example, B-spline interpolation is popular. In B-spline interpolation, the path may not pass exactly through the via points, but the path is guaranteed to be confined to the convex hull of the via points, unlike the paths in Figure 9.7. This can be important to ensure that joint limits or workspace obstacles are respected.
In this section we consider the problem of finding the fastest possible time scaling $s(t)$ that respects the robot’s actuator limits. We write the limits on the $i$th actuator as
\[ \tau_i \min (\theta, \dot{\theta}) \leq \tau_i \leq \tau_i \max (\theta, \dot{\theta}). \] (9.31)

The available actuator torque is typically a function of the current joint speed (see Chapter 8.9.1). For example, for a given maximum voltage of a DC motor, the maximum torque available from the motor drops linearly with the motor’s speed.

Before proceeding, we note that the quadratic velocity terms in Equation (9.30) can be written equivalently as
\[ c(\theta, \dot{\theta}) = \dot{\theta}^T \Gamma(\theta) \dot{\theta}, \]
where $\Gamma(\theta)$ is the three-dimensional tensor of Christoffel symbols constructed of partial derivatives of components of the mass matrix $M(\theta)$ with respect to $\theta$. This form more clearly shows the quadratic dependence on velocities. Now beginning with Equation (9.30), replacing $\dot{\theta}$ by $(d\theta/ds)\dot{s}$ and $\ddot{\theta}$ by $(d\theta/ds)\ddot{s} + (d^2\theta/ds^2)\dot{s}^2$, and rearranging, we get
\[
\begin{align*}
\left( m(s) \left( \frac{d\theta}{ds} \right) \right) \ddot{s} + \left( c(s) \dot{s} + g(\theta(s)) \right) s^2 = \tau,
\end{align*}
\]
expressed more compactly as the vector equation
\[ m(s) \ddot{s} + c(s) \dot{s}^2 + g(s) = \tau, \] (9.33)
where $m(s)$ is the effective inertia of the robot confined to the path $\theta(s)$, $c(s)\dot{s}^2$ are the quadratic velocity terms, and $g(s)$ is the gravitational torque.

Similarly, the actuation constraints (9.31) can be expressed as a function of $s$:
\[ \tau_i \min (s, \dot{s}) \leq \tau_i \leq \tau_i \max (s, \dot{s}). \] (9.34)

Plugging in the components of Equation (9.33), we get
\[ \tau_i \min (s, \dot{s}) \leq m_i(s) \ddot{s} + c_i(s) \dot{s}^2 + g_i(s) \leq \tau_i \max (s, \dot{s}). \] (9.35)

Let $L_i(s, \dot{s})$ and $U_i(s, \dot{s})$ be the minimum and maximum accelerations $\ddot{s}$ satisfying the $i$th component of Equation (9.35). Depending on the sign of $m_i(s)$,
we have three possibilities:

\[
\begin{align*}
\text{if } m_i(s) > 0: \quad L_i(s) &= \tau_i^{\text{min}}(s, \dot{s}) - c(s)\dot{s}^2 - g(s) \quad \frac{m_i(s)}{m_i(s)} \\
U_i(s) &= \tau_i^{\text{max}}(s, \dot{s}) - c(s)\dot{s}^2 - g(s) \quad \frac{m_i(s)}{m_i(s)} \\
\text{if } m_i(s) < 0: \quad L_i(s) &= \tau_i^{\text{max}}(s, \dot{s}) - c(s)\dot{s}^2 - g(s) \quad \frac{m_i(s)}{m_i(s)} \\
U_i(s) &= \tau_i^{\text{min}}(s, \dot{s}) - c(s)\dot{s}^2 - g(s) \quad \frac{m_i(s)}{m_i(s)} \\
\text{if } m_i(s) = 0: \quad \text{zero-inertia point, discussed in Section 9.4.3}
\end{align*}
\]

Defining

\[
L(s, \dot{s}) = \max_i L_i(s, \dot{s}) \quad \text{and} \quad U(s, \dot{s}) = \min_i U_i(s, \dot{s}),
\]

the actuator limits (9.35) can be written as the state-dependent time-scaling constraints

\[
L(s, \dot{s}) \leq \ddot{s} \leq U(s, \dot{s}). \tag{9.37}
\]

The time-optimal time-scaling problem can now be stated:

Given a path \( \theta(s), s \in [0, 1] \), an initial state \((s_0, \dot{s}_0) = (0, 0)\), and a final state \((s_f, \dot{s}_f) = (1, 0)\), find a monotonically increasing twice-differentiable time scaling \( s : [0, T] \to [0, 1] \) that

(i) satisfies \( s(0) = \dot{s}(0) = \dot{s}(T) = 0 \) and \( s(T) = 1 \), and

(ii) minimizes the total travel time \( T \) along the path while respecting the actuator constraints (9.37).

The problem formulation is easily generalized to the case of nonzero initial and final velocity along the path, \( \dot{s}(0) > 0 \) and \( \dot{s}(T) > 0 \).

### 9.4.1 The \((s, \dot{s})\) Phase Plane

The problem is easily visualized in the \((s, \dot{s})\) phase plane of the path-constrained robot, with \( s \) running from 0 to 1 on a horizontal axis and \( \dot{s} \) on a vertical axis. Since \( s(t) \) is monotonically increasing, \( \dot{s}(t) \geq 0 \) for all times \( t \) and for all \( s \in [0, 1] \). A time scaling of the path is any curve in the phase plane that moves monotonically to the right from \((0, 0)\) to \((1, 0)\) (Figure 9.10). Not all such curves satisfy the acceleration constraints (9.37), however.

To see the effect of the acceleration constraints, at each \((s, \dot{s})\) in the phase plane, we can plot the limits \( L(s, \dot{s}) \leq \ddot{s} \leq U(s, \dot{s}) \) as a cone, as illustrated in Figure 9.11(a). If \( L(s, \dot{s}) \geq U(s, \dot{s}) \), the cone disappears—there are no actuator commands that can keep the robot on the path at this state. These inadmissible states are indicated in gray in Figure 9.11(a). For any \( s \), typically there is a
302

Trajectory Generation

Figure 9.10: A time scaling in the \((s, \dot{s})\) phase plane is a curve with \(\dot{s} \geq 0\) at all times connecting the initial path position and velocity \((0, 0)\) to the final position and velocity \((1, 0)\).

Figure 9.11: (a) Acceleration-limited motion cones at four different states. The upper ray of the cone is the sum of \(U(s, \dot{s})\) plotted in the vertical direction (change in velocity) and \(\dot{s}\) plotted in the horizontal direction (change in position). The lower ray of the cone is constructed from \(L(s, \dot{s})\) and \(\dot{s}\). Points in gray, bounded by the velocity limit curve, have \(L(s, \dot{s}) \geq U(s, \dot{s})\) — the state is inadmissible. On the velocity limit curve, the cone is reduced to a single vector. (b) The proposed time scaling is infeasible because the tangent to the curve is outside the motion cone at the state indicated.

For a minimum-time motion, the “speed” \(\dot{s}\) must be as high as possible...
9.4. Time-Optimal Time Scaling

303

A time-optimal bang-bang time scaling integrates $U(s, \dot{s})$ from $(0, 0)$ and switches to $L(s, \dot{s})$ at a switching point $s^*$. Also shown is a non-optimal time scaling with a tangent inside a motion cone. (b) Sometimes the velocity limit curve prevents a single-switch solution.

At every $s$ while still satisfying the acceleration constraints and the endpoint constraints. To see this, consider that the total time of motion $T$ is given by

$$T = \int_0^T 1 \, dt.$$  (9.38)

Making the substitution $ds/ds = 1$, and changing the limits of integration from $0$ to $T$ (time) to $0$ to $1$ ($s$), we get

$$T = \int_0^T 1 \, dt = \int_0^T \frac{ds}{ds} \, dt = \int_0^T \frac{dt}{ds} \, ds = \int_0^1 \dot{s}^{-1}(s) \, ds.$$  (9.39)

Thus for time to be minimized, $\dot{s}^{-1}(s)$ should be as small as possible, and therefore $\dot{s}(s)$ must be as large as possible, at all $s$, while still satisfying the acceleration constraints (9.37) and the boundary constraints.

This implies that the time scaling must always operate at the limit $U(s, \dot{s})$ or $L(s, \dot{s})$, and our only choice is when to switch between these limits. A common solution is a bang-bang trajectory: maximum acceleration $U(s, \dot{s})$ followed by a switch to maximum deceleration $L(s, \dot{s})$. (This is similar to the trapezoidal motion profile that never reaches the coasting velocity $v$ in Section 9.2.2.2.)

In this case, the time scaling is calculated by numerically integrating $U(s, \dot{s})$ forward in $s$ from $(0, 0)$, integrating $L(s, \dot{s})$ backward in $s$ from $(1, 0)$, and finding the intersection of these curves (Figure 9.12(a)). The switch between maximum acceleration and maximum deceleration occurs at the intersection.

In some cases, however, the velocity limit curve prevents a single-switch solution (Figure 9.12(b)). These cases require an algorithm to find multiple switching points.
9.4.2 The Time-Scaling Algorithm

Finding the optimal time scaling is reduced to finding the switches between maximum acceleration \( U(s, \dot{s}) \) and maximum deceleration \( L(s, \dot{s}) \), maximizing the “height” of the curve in the \((s, \dot{s})\) phase plane.

**Time-scaling algorithm.**

1. Initialize an empty list of switches \( S = \{ \} \) and a switch counter \( i = 0 \). Set \((s_i, \dot{s}_i) = (0, 0)\).

2. Integrate the equation \( \ddot{s} = L(s, \dot{s}) \) backward in time from \((1, 0)\) until \( L(s, \dot{s}) > U(s, \dot{s}) \) (the velocity limit curve is penetrated) or \( s = 0 \). Call this phase plane curve \( F \).

3. Integrate the equation \( \ddot{s} = U(s, \dot{s}) \) forward in time from \((s_i, \dot{s}_i)\) until it crosses \( F \) or until \( U(s, \dot{s}) < L(s, \dot{s}) \) (the velocity limit curve is penetrated). Call this curve \( A_i \). If \( A_i \) crosses \( F \), then increment \( i \), set \((s_i, \dot{s}_i)\) to the \((s, \dot{s})\) value at which the crossing occurs, and append \( s_i \) to the list of switches \( S \). This is a switch from maximum acceleration to maximum deceleration. The problem is solved and \( S \) is the set of switches expressed in the path parameter. If instead the velocity limit curve is penetrated, let \((s_{lim}, \dot{s}_{lim})\) be the point of penetration and proceed to the next step.

4. Perform a binary search on the velocity in the range \([0, \dot{s}_{lim}]\) to find the velocity \( \dot{s}' \) such that the curve integrating \( \ddot{s} = L(s, \dot{s}) \) forward from \((s_{lim}, \dot{s}')\) touches the velocity limit curve without penetrating it. The binary search is initiated with \( \dot{s}_{high} = \dot{s}_{lim} \) and \( \dot{s}_{low} = 0 \).
   
   (a) Set the test velocity halfway between \( \dot{s}_{low} \) and \( \dot{s}_{high} \), \( \dot{s}_{test} = (\dot{s}_{high} + \dot{s}_{low})/2 \). The test point is \((s_{lim}, \dot{s}_{test})\).
   
   (b) If the curve from the test point penetrates the velocity limit curve, set \( \dot{s}_{high} \) equal to \( \dot{s}_{test} \). If instead the curve from the test point hits \( \dot{s} = 0 \), set \( \dot{s}_{low} \) equal to \( \dot{s}_{test} \). Return to step 4a.

Continue the binary search until a specified tolerance. Let \((s_{tan}, \dot{s}_{tan})\) equal the point where the resulting curve just touches the velocity limit curve tangentially (or comes closest to the curve without hitting). The motion cone at this point is reduced to a single tangent vector \((L(s, \dot{s}) = U(s, \dot{s}))\), tangent to the velocity limit curve.

5. Integrate \( \ddot{s} = L(s, \dot{s}) \) backwards from \((s_{tan}, \dot{s}_{tan})\) until it intersects \( A_i \). Increment \( i \), set \((s_i, \dot{s}_i)\) to the \((s, \dot{s})\) value at the intersection, and label as \( A_i \) the curve segment from \((s_i, \dot{s}_i)\) to \((s_{tan}, \dot{s}_{tan})\). Append \( s_i \) to the list of switches \( S \). This is a switch from maximum acceleration to maximum deceleration.

6. Increment \( i \) and set \((s_i, \dot{s}_i)\) to \((s_{tan}, \dot{s}_{tan})\). Append \( s_i \) to the list of switches \( S \). This is a switch from maximum deceleration to maximum acceleration. Go to step 3.

The algorithm is illustrated in Figure 9.13.
9.4. Time-Optimal Time Scaling

Step 2: \( i = 0, S = \{ \} \)

Step 3: \( i = 0, S = \{ \} \)

Step 4: \( i = 0, S = \{ \} \)

Step 5: \( i = 1, S = \{ s_1 \} \)

Step 6: \( i = 2, S = \{ s_1, s_2 \} \)

Step 3: \( i = 3, S = \{ s_1, s_2, s_3 \} \)

Figure 9.13: The time-scaling algorithm. (Step 2) Integrating \( \ddot{s} = L(s, \dot{s}) \) backward from \((1, 0)\) until the velocity limit curve is reached. (Step 3) Integrating \( \ddot{s} = U(s, \dot{s}) \) forward from \((0, 0)\) to the intersection \((s_{lim}, \dot{s}_{lim})\) with the velocity limit curve. (Step 4) Binary search to find \((s_{lim}, \dot{s}’)\) from which \( \ddot{s} = L(s, \dot{s}) \), integrated forward from \((s_{lim}, \dot{s}’), \) touches the velocity limit curve tangentially. (Step 5) Integrating backward along \( L(s, \dot{s}) \) from \((s_{tan}, \dot{s}_{tan})\) to find the first switch from acceleration to deceleration. (Step 6) The second switch, from deceleration to acceleration, is at \((s_2, \dot{s}_2) = (s_{tan}, \dot{s}_{tan})\). (Step 3) Integrating forward along \( U(s, \dot{s}) \) from \((s_2, \dot{s}_2)\) results in intersection with \( F \) at \((s_3, \dot{s}_3)\), where a switch occurs from acceleration to deceleration. The optimal time scaling consists of switches at \( S = \{ s_1, s_2, s_3 \} \).
9.4.3 Assumptions and Caveats

The description above covers the major points of the optimal time-scaling algorithm. A few assumptions were glossed over, however; they are made explicit below.

- **Static posture maintenance.** The algorithm, as described, assumes that the robot can maintain its configuration against gravity at any state \((s, \dot{s} = 0)\). This ensures the existence of valid time scalings, namely, time scalings that move the robot along the path arbitrarily slowly. For some robots and paths, this assumption may be violated due to weak actuators. For example, some paths may require momentum to carry motion through configurations the robot cannot maintain statically. The algorithm can be modified to handle such cases.

- **Inadmissible states.** The algorithm assumes that at every \(s\) there is a unique velocity limit \(\dot{s}_{\text{lim}}(s) > 0\) such that all velocities \(\dot{s} \leq \dot{s}_{\text{lim}}(s)\) are admissible and all velocities \(\dot{s} > \dot{s}_{\text{lim}}(s)\) are inadmissible. For some models of actuator dynamics or friction, this assumption may be violated—there may be isolated “islands” of inadmissible states. The algorithm can be modified to handle this case.

- **Zero-inertia points.** The algorithm assumes no zero-inertia points (Equation (9.36)). If \(m_i(s) = 0\) in (9.36), then the torque provided by actuator \(i\) has no dependence on the acceleration \(\ddot{s}\), and the \(i\)th actuator constraint in (9.35) directly defines a velocity constraint on \(\dot{s}\). At a point \(s\) with one or more zero components in \(m(s)\), the velocity limit curve is defined by the minimum of (a) the velocity constraints defined by the zero-inertia components and (b) the \(\dot{s}\) values satisfying \(L_i(s, \dot{s}) = U_i(s, \dot{s})\) for the other components. For the algorithm as described, singular arcs of zero-inertia points on the velocity limit curve may lead to rapid switching between \(\ddot{s} = U(s, \dot{s})\) and \(\ddot{s} = L(s, \dot{s})\). In such cases, choosing an acceleration tangent to the velocity limit curve, and between \(U(s, \dot{s})\) and \(L(s, \dot{s})\), preserves time optimality without chattering controls.

It is worth noting that the time-scaling algorithm generates trajectories with discontinuous acceleration, which could lead to vibrations. Beyond this, inaccuracies in models of robot inertial properties and friction make direct application of the time-scaling algorithm impractical. Finally, since a minimum-time time scaling always saturates at least one actuator, if the robot gets off the planned trajectory, there may be no torque left for corrective action by a feedback controller.

Despite these drawbacks, the time-scaling algorithm provides a deep understanding of the true maximum capabilities of a robot following a path.
9.5 Summary

- A trajectory \( \theta(t), \theta : [0, T] \rightarrow \Theta \), can be written as \( \theta(s(t)) \), the composition of a path \( \theta(s), \theta : [0, 1] \rightarrow \Theta \), and a time scaling \( s(t), s : [0, T] \rightarrow [0, 1] \).

- A straight-line path in joint space can be written \( \theta(s) = \theta_{\text{start}} + s(\theta_{\text{end}} - \theta_{\text{start}}), s \in [0, 1] \). A similar form holds for straight-line paths in a minimum set of task space coordinates. A “straight-line” path in \( SE(3) \), where \( X = (R, p) \), can be decoupled to a Cartesian path and a rotation path:

\[
\begin{align*}
p(s) &= p_{\text{start}} + s(p_{\text{end}} - p_{\text{start}}) \quad (9.40) \\
R(s) &= R_{\text{start}} \exp(\log(R_{\text{start}}^T R_{\text{end}})s). \quad (9.41)
\end{align*}
\]

- A cubic polynomial \( s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \) can be used to time scale a point-to-point motion with zero initial and final velocity. The acceleration undergoes a step change (infinite jerk) at \( t = 0 \) and \( t = T \), however. An impulse in jerk can cause vibration of the robot.

- A quintic polynomial \( s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \) can be used to time-scale a point-to-point motion with zero initial and final velocities and accelerations. Jerk is finite at \( t = 0 \) and \( t = T \).

- The trapezoidal motion profile is a popular time scaling in point-to-point control, particularly control of a single motor. The motion consists of three phases: constant acceleration, constant velocity, and constant deceleration, resulting in a trapezoid in \( \dot{s}(t) \). Trapezoidal motion involves step changes in acceleration.

- The S-curve motion profile is also popular in point-to-point control of a single motor. It consists of seven phases: (1) constant positive jerk, (2) constant acceleration, (3) constant negative jerk, (4) constant velocity, (5) constant negative jerk, (6) constant deceleration, and (7) constant positive jerk.

- Given a set of via points including a start state, a goal state, and other via states through which the robot’s motion must pass, as well as the times \( T_i \) these states should be reached, a series of cubic polynomial time scalings can be used to generate a trajectory \( \theta(t) \) interpolating the via points. To prevent step changes in the acceleration at the via points, a series of quintic polynomials can be used instead.

- Given a robot path \( \theta(s) \), the dynamics of the robot, and limits on the actuator torques, the actuator constraints can be expressed in terms of \( (s, \dot{s}) \) as the vector inequalities

\[
L(s, \dot{s}) \leq \dot{s} \leq U(s, \dot{s}).
\]

The time-optimal time scaling is the \( s(t) \) such that the “height” of the curve in the \( (s, \dot{s}) \) phase plane is maximized while satisfying \( s(0) = \dot{s}(0) = \)}
Trajectory Generation

\[ s(T) = 0, \quad s(T) = 1, \quad \text{and the actuator constraints. The optimal solution always operates at maximum acceleration } U(s, \dot{s}) \text{ or maximum deceleration } L(s, \dot{s}). \]

9.6 Software

\[ s = \text{CubicTimeScaling}(T_f, t) \]
Computes \( s(t) \) for a cubic time-scaling, given \( t \) and the total time of motion \( T_f \).

\[ s = \text{QuinticTimeScaling}(T_f, t) \]
Computes \( s(t) \) for a quintic time-scaling, given \( t \) and the total time of motion \( T_f \).

\[ \text{traj} = \text{JointTrajectory}(\theta_{\text{start}}, \theta_{\text{end}}, T_f, N, \text{method}) \]
Computes a straight-line trajectory in joint space as an \( N \times n \) matrix, where each of the \( N \) rows is an \( n \)-vector of joint variables at an instant in time. The first row is \( \theta_{\text{start}} \) and the \( N \)th row is \( \theta_{\text{end}} \). The elapsed time between each row is \( T_f/(N - 1) \). The parameter \( \text{method} \) is either 3 for a cubic time scaling or 5 for a quintic time scaling.

\[ \text{traj} = \text{ScrewTrajectory}(X_{\text{start}}, X_{\text{end}}, T_f, N, \text{method}) \]
Computes a trajectory as a list of \( N SE(3) \) matrices, where each matrix represents the configuration of the end-effector at an instant in time. The first matrix is \( X_{\text{start}} \), the \( N \)th matrix is \( X_{\text{end}} \), and the motion is along a constant screw axis. The elapsed time between each matrix is \( T_f/(N - 1) \). The parameter \( \text{method} \) is either 3 for a cubic time scaling or 5 for a quintic time scaling.

\[ \text{traj} = \text{CartesianTrajectory}(\theta_{\text{start}}, \theta_{\text{end}}, T_f, N, \text{method}) \]
Computes a trajectory as a list of \( N SE(3) \) matrices, where each matrix represents the configuration of the end-effector at an instant in time. The first matrix is \( X_{\text{start}} \), the \( N \)th matrix is \( X_{\text{end}} \), and the origin of the end-effector frame follows a straight line, decoupled from the rotation. The elapsed time between each matrix is \( T_f/(N - 1) \). The parameter \( \text{method} \) is either 3 for a cubic time scaling or 5 for a quintic time scaling.

9.7 Notes and References

In 1985, Bobrow et al. [12] and Shin and McKay [122] published papers nearly simultaneously that independently derived the essence of the time-optimal time-scaling algorithm outlined in Section 9.4. A year earlier, Hollerbach addressed the restricted problem of finding dynamically feasible time-scaled trajectories for uniform time scalings where the time variable \( t \) is replaced by \( ct \) for \( c > 0 \) [41].

The original papers of Bobrow et al. and Shin and McKay were followed by a number of papers refining the methods by addressing singularities, algorithm
efficiency, and even the presence of constraints and obstacles [99, 126, 117, 118, 119, 120, 121]. Other research has focused on numerical methods such as dynamic programming or nonlinear optimization to minimize cost functions such as actuator energy. One early example of work in this area is by Vukobratović and Kirćanski [138].
9.8 Exercises

1. Consider an elliptical path in the \((x, y)\) plane. The path starts at \((0, 0)\) and proceeds clockwise to \((2, 1)\), \((4, 0)\), \((2, -1)\), and back to \((0, 0)\) (Figure 9.14). Write the path as a function of \(s \in [0, 1]\).

2. A cylindrical path in \(X = (x, y, z)\) is given by \(x = \cos 2\pi s\), \(y = \sin 2\pi s\), \(z = 2s\), \(s \in [0, 1]\), and its time scaling is \(s(t) = \frac{1}{4} t + \frac{1}{8} t^2\), \(t \in [0, 2]\). Write \(X\) and \(\dot{X}\).

3. Consider a path from \(X(0) = X_{\text{start}} \in SE(3)\) to \(X(1) = X_{\text{end}} \in SE(3)\) consisting of motion along a constant screw axis. The path is time scaled by some \(s(t)\). Write the twist \(V\) and acceleration \(\ddot{V}\) at any point on the path given \(\dot{s}\) and \(\ddot{s}\).

4. Consider a straight-line path \(\theta(s) = \theta_{\text{start}} + s(\theta_{\text{end}} - \theta_{\text{start}}), s \in [0, 1]\) from \(\theta_{\text{start}} = (0, 0)\) to \(\theta_{\text{end}} = (\pi, \pi/3)\). The feasible joint velocities are \(|\dot{\theta}_1|, |\dot{\theta}_2| \leq 2\text{ rad/s}\) and the feasible joint accelerations are \(|\ddot{\theta}_1|, |\ddot{\theta}_2| \leq 0.5\text{ rad/s}^2\). Find the fastest motion time \(T\) using a cubic time scaling that satisfies the joint velocity and acceleration limits.

5. Find the fifth-order polynomial time scaling that satisfies \(s(T) = 1\) and \(s(0) = \dot{s}(0) = \ddot{s}(0) = \dddot{s}(T) = 0\).

6. As a function of the total time of motion \(T\), find the times at which the acceleration \(\ddot{s}\) of the fifth-order polynomial point-to-point time scaling is maximum and minimum.

7. If you want to use a polynomial time scaling for point-to-point motion with zero initial and final velocity, acceleration, and jerk, what would be the
9.8. Exercises

minimum order of the polynomial?

8. Prove that the trapezoidal time scaling, using the maximum allowable acceleration $a$ and velocity $v$, minimizes the time of motion $T$.

9. Plot by hand the acceleration profile $\ddot{s}(t)$ for a trapezoidal time scaling.

10. If $v$ and $a$ are specified for a trapezoidal time scaling, prove that $v^2/a \leq 1$ is a necessary condition for the robot to reach the maximum velocity $v$ during the path.

11. If $v$ and $T$ are specified for a trapezoidal time scaling, prove that $vT > 1$ is a necessary condition for the motion to be able to complete in time $T$. Prove that $vT \leq 2$ is a necessary condition for a three-stage trapezoidal motion.

12. If $a$ and $T$ are specified for a trapezoidal time scaling, prove that $aT^2 \geq 4$ is a necessary condition to ensure that the motion completes in time.

13. Consider the case where the maximum velocity $v$ is never reached in a trapezoidal time scaling. The motion becomes a bang-bang motion: constant acceleration $a$ for time $T/2$ followed by constant deceleration $-a$ for time $T/2$. Write the position $s(t)$, velocity $\dot{s}(t)$, and acceleration $\ddot{s}(t)$ for both phases, analogous to Equations (9.16)-(9.24).

14. Plot by hand the acceleration profile $\ddot{s}(t)$ for an S-curve time scaling.

15. A seven-stage S-curve is fully specified by the time $t_J$ (duration of constant positive or negative jerk), the time $t_a$ (duration of constant positive or negative acceleration), the time $t_v$ (duration of constant velocity), the total time $T$, the jerk $J$, the acceleration $a$, and the velocity $v$. Of these seven quantities, how many can be specified independently? You may assume that any inequality constraints needed for a seven-stage profile are satisfied.

16. A nominal S-curve has seven stages, but it can have fewer if certain inequality constraints are not satisfied. Indicate which cases are possible with fewer than seven stages. By hand, approximately draw the $\dot{s}(t)$ velocity profiles for these cases.

17. If the S-curve achieves all seven stages and uses jerk $J$, acceleration $a$, and velocity $v$, what is the constant velocity coasting time $t_v$ in terms of $v$, $a$, $J$, and the total motion time $T$?

18. Write your own via-point cubic polynomial interpolation trajectory gener-
A new position and velocity specification is required for each joint at 1000 Hz. The user specifies a sequence of via point positions, velocities, and times, and the program generates an array consisting of the joint angles and velocities at every 1 ms from time $t = 0$ to time $t = T$, the total duration of the movement. For a test case with at least three via points (start and end at rest, and at least one more via), plot (a) the path in the joint angle space (similar to Figure 9.7) and (b) the position and velocity of each joint as a function of time (similar to Figure 9.8).

19. Via points with specified positions, velocities, and accelerations can be interpolated using fifth-order polynomials of time. For a fifth-order polynomial segment between via points $j$ and $j+1$, of duration $\Delta T_j$, with $\beta_j, \dot{\beta}_j, \ddot{\beta}_j, \beta_{j+1}$, $\dot{\beta}_{j+1}$ specified, solve for the coefficients of the fifth-order polynomial (similar to Equations (9.26)–(9.29)). A symbolic math solver will simplify the problem.

20. By hand or by computer, plot a trapezoidal motion profile in the $(s, \dot{s})$ plane.

21. Figure 9.15 shows three candidate motion curves in the $(s, \dot{s})$ plane (A, B, and C) and three candidate motion cones at $\dot{s} = 0$ (a, b, and c). Two of the three curves and two of the three motion cones cannot be correct for any robot dynamics. Indicate which are incorrect and explain your reasoning. Explain why the others are possibilities.

22. Under the assumptions of Section 9.4.3, explain why the time-scaling algorithm of Section 9.4.2 is correct. In particular, (a) explain why in the binary search of Step 4, the curve integrated forward from $(s_{\text{lim}}, \dot{s}_{\text{test}})$ must either hit
(or run tangent to) the velocity limit curve or hit the $\dot{s} = 0$ axis (and does not hit the curve $F$, for example); (b) explain why the final time scaling can only touch the velocity limit curve tangentially; and (c) explain why the acceleration switches from minimum to maximum at points where the time scaling touches the velocity limit curve.

23. Explain how the time-scaling algorithm should be modified, if at all, to handle the case where the initial and final velocity, at $s = 0$ and $s = 1$, are nonzero.

24. Explain how the time-scaling algorithm should be modified if the robot’s actuators are too weak to hold it statically at some configurations of the path (static posture maintenance assumption is violated), but the assumptions on inadmissible states and zero-inertia points are satisfied. Valid time scalings may no longer exist. Under what condition(s) should the algorithm terminate and indicate that no valid time scaling exists? (Under the assumptions of Section 9.4.3, the original algorithm always finds a solution, and therefore does not check for failure cases.) What do the motion cones look like at states $(s, \dot{s} = 0)$ where the robot cannot hold itself statically?

25. Create a computer program that plots the motion cones in the $(s, \dot{s})$ plane for a 2R robot in a horizontal plane. The path is a straight line in joint space from $(\theta_1, \theta_2) = (0, 0)$ to $(\pi/2, \pi/2)$. First derive the dynamics of the arm, then rewrite the dynamics in terms of $s, \dot{s}, \ddot{s}$ instead of $\theta, \dot{\theta}, \ddot{\theta}$. The actuators can provide torques in the range $-\tau_{i,\text{limit}} - b\dot{\theta}_i \leq \tau_i \leq \tau_{i,\text{limit}} - b\dot{\theta}_i$, where $b > 0$ indicates the velocity dependence of the torque. The cones should be drawn at a grid of points in $(s, \dot{s})$. To keep the figure manageable, normalize each cone ray to the same length.
Chapter 10

Motion Planning

Motion planning is the problem of finding a robot motion from a start state to a goal state while avoiding obstacles in the environment and satisfying other constraints, such as joint limits or torque limits. Motion planning is one of the most active subfields of robotics, and it is the subject of entire books. The purpose of this chapter is to provide a practical overview of a few common techniques, using robot arms and mobile robots as the primary example systems (Figure 10.1).

The chapter begins with a brief overview of motion planning, followed by foundational material including configuration space obstacles, and concludes with summaries of several different planning methods.

10.1 Overview of Motion Planning

A key concept in motion planning is configuration space, or C-space for short. Every point in the C-space $C$ corresponds to a unique configuration $q$ of the robot, and every configuration of the robot can be represented as a point in

Figure 10.1: (Left) A robot arm executing an obstacle-avoiding motion plan. The motion plan was generated using MoveIt! [132] and visualized using rviz in ROS (the Robot Operating System). (Right) A car-like mobile robot parallel parking.
C-space. For example, the configuration of a robot arm with \( n \) joints can be represented as a list of \( n \) joint positions, \( q = (\theta_1, \ldots, \theta_n) \). The free C-space \( C_{\text{free}} \) consists of the configurations where the robot does not penetrate any obstacle nor violate a joint limit.

In this chapter, unless otherwise stated, we assume that \( q \) is an \( n \)-vector and that \( C \subset \mathbb{R}^n \). With some generalization, however, the concepts of this chapter apply to non-Euclidean C-spaces like \( C = SE(3) \ (n = 6) \).

The control inputs available to drive the robot are written as an \( m \)-vector \( u \in U \subset \mathbb{R}^m \), where \( m = n \) for a typical robot arm. If the robot has second-order dynamics, like a robot arm, and the control inputs are forces (equivalently, accelerations), the state of the robot is its configuration and velocity, \( x = (q, v) \in \mathcal{X} \). For \( q \in \mathbb{R}^n \), typically we write \( v = \dot{q} \). If we can treat the control inputs as velocities, the state \( x \) is simply the configuration \( q \). The notation \( q(x) \) indicates the configuration \( q \) corresponding to the state \( x \), and \( \mathcal{X}_{\text{free}} = \{ x \mid q(x) \in C_{\text{free}} \} \).

The equations of motion of the robot are written
\[
\dot{x} = f(x, u),
\]

or, in integral form,
\[
x(T) = x(0) + \int_0^T f(x(t), u(t)) dt.
\]

### 10.1.1 Types of Motion Planning Problems

Given the definitions above, a fairly broad specification of the motion planning problem is the following:

*Given an initial state \( x(0) = x_{\text{start}} \) and a desired final state \( x_{\text{goal}} \), find a time \( T \) and a set of controls \( u : [0, T] \to U \) such that the motion (10.2) satisfies \( x(T) = x_{\text{goal}} \) and \( q(x(t)) \in C_{\text{free}} \) for all \( t \in [0, T] \).*

It is assumed that a feedback controller (Chapter 11) is available to ensure that the planned motion \( x(t) \), \( t \in [0, T] \), is followed closely. It is also assumed that an accurate geometric model of the robot and environment is available to evaluate \( C_{\text{free}} \) during motion planning.

There are many variations of the basic problem; some are discussed below.

**Path planning vs. motion planning.** The path planning problem is a subproblem of the general motion planning problem. Path planning is the purely geometric problem of finding a collision-free path \( q(s) \), \( s \in [0, 1] \), from a start configuration \( q(0) = q_{\text{start}} \) to a goal configuration \( q(1) = q_{\text{goal}} \), without concern for dynamics, the duration of motion, or constraints on the motion or control inputs. It is assumed that the path returned by the path planner can be time scaled to create a feasible trajectory (Chapter 9). This problem is sometimes called the piano mover’s problem, emphasizing the focus on the geometry of cluttered spaces.
10.1. Overview of Motion Planning

**Control inputs:** $m = n$ vs. $m < n$. If there are fewer control inputs $m$ than degrees of freedom $n$, the robot is incapable of following many paths, even if they are collision-free. For example, a car has $n = 3$ (position and orientation of the chassis in the plane) but $m = 2$ (forward/backward motion and steering)—it cannot slide directly sideways into a parking space.

**Online vs. offline.** A motion planning problem requiring an immediate result, perhaps because obstacles appear, disappear, or move unpredictably, calls for a fast, online planner. If the environment is static, a slower offline planner may suffice.

**Optimal vs. satisficing.** In addition to reaching the goal state, we might want the motion plan to minimize (or approximately minimize) a cost $J$, e.g.,

$$J = \int_0^T L(x(t), u(t)) dt.$$  

For example, minimizing with $L = 1$ yields a time-optimal motion while minimizing with $L = u^T(r)u(r)$ yields a “minimum-effort” motion.

**Exact vs. approximate.** We may be satisfied with a final state $x(T)$ that is sufficiently close to $x_{\text{goal}}$, e.g., $\|x(T) - x_{\text{goal}}\| < \epsilon$.

**With or without obstacles.** The motion planning problem can be challenging even in the absence of obstacles, particularly if $m < n$ or optimality is desired.

10.1.2 Properties of Motion Planners

Planners must conform to the properties of the motion planning problem as outlined above. In addition, planners can be distinguished by the following properties:

**Multiple-query vs. single-query planning.** If the environment is unchanging and the robot will be asked to solve a number of motion planning problems in the environment, it may be worth spending the time to build a data structure that accurately represents $C_{\text{free}}$. This data structure can then be searched to efficiently solve multiple planning queries. Single-query planners solve each new problem from scratch.

**“Anytime” planning.** An anytime planner is one that continues to look for better solutions after a first solution is found. The planner can be stopped at any time, for example when a specified time limit has passed, and the best solution is returned.

**Completeness.** A motion planner is said to be complete if it is guaranteed to find a solution in finite time if one exists, and to report failure if there is no feasible motion plan. A weaker notion is resolution completeness.
A planner is resolution complete if it is guaranteed to find a solution if one exists at the resolution of a discretized representation of the problem, such as the resolution of a grid representation of $C_{\text{free}}$. Finally, a planner is **probabilistically complete** if the probability of finding a solution, if one exists, tends to 1 as planning time goes to infinity.

**Computational complexity.** The computational complexity of a planner refers to characterizations of the amount of time the planner takes to run or the amount of memory it requires. These are measured in terms of the description of the planning problem, such as the dimension of the C-space or the number of vertices in the representation of the robot and obstacles. For example, the time for a planner to run may be exponential in $n$, the dimension of the C-space. The computational complexity may be expressed in terms of the average case or the worst case. Some planning algorithms lend themselves easily to computational complexity analysis, while others do not.

### 10.1.3 Motion Planning Methods

There is no single planner applicable to all motion planning problems. Below is a broad overview of some of the many motion planners available. Details are left to the sections indicated.

**Complete methods (Section 10.3).** These methods focus on exact representations of the geometry or topology of $C_{\text{free}}$, ensuring completeness. For all but simple or low-degree-of-freedom problems, these representations are mathematically or computationally prohibitive to derive.

**Grid methods (Section 10.4).** These methods discretize $C_{\text{free}}$ into a grid and search the grid for a motion from $q_{\text{start}}$ to a grid point in the goal region. Modifications of the approach may discretize the state space or control space, or use multi-scale grids to refine the representation of $C_{\text{free}}$ near obstacles. These methods are relatively easy to implement and can return optimal solutions, but for a fixed resolution, the memory required to store the grid, and the time to search it, grows exponentially with the number of dimensions of the space. This limits the approach to low-dimensional problems.

**Sampling methods (Section 10.5).** A generic sampling method relies on a random or deterministic function to choose a sample from the C-space or state space; a function to evaluate whether the sample is in $X_{\text{free}}$; a function to determine the “closest” previous free-space sample; and a local planner to try to connect to, or move toward, the new sample from the previous sample. This process builds up a graph or tree representing feasible motions of the robot. Sampling methods are easy to implement, tend to be probabilistically complete, and can even solve high-degree-of-freedom motion planning problems. The solutions tend to be satisfying,
not optimal, and it can be difficult to characterize the computational complexity.

**Virtual potential fields (Section 10.6).** Virtual potential fields create forces on the robot that pull it toward the goal and push it away from obstacles. The approach is relatively easy to implement, even for high-degree-of-freedom systems, and fast to evaluate, often allowing online implementation. The drawback is local minima in the potential function: the robot may get stuck in configurations where the attractive and repulsive forces cancel, but the robot is not at the goal state.

**Nonlinear optimization (Section 10.7).** The motion planning problem can be converted to a nonlinear optimization problem by representing the path or controls by a finite number of design parameters, such as the coefficients of a polynomial or a Fourier series. The problem is to solve for the design parameters that minimize a cost function while satisfying constraints on the controls, obstacles, and goal. While these methods can produce near-optimal solutions, they require an initial guess at the solution. Because the objective function and feasible solution space are generally not convex, the optimization process can get stuck far away from a solution, let alone an optimal solution.

**Smoothing (Section 10.8).** Often the motions found by a planner are jerky. A smoothing algorithm can be run on the result of the motion planner to improve the smoothness.

The major trend in recent years has been toward sampling methods, which are easy to implement and can handle high-dimensional problems.

## 10.2 Foundations

Before discussing motion planning algorithms, we establish concepts used in many of them: configuration space obstacles, collision detection, graphs, and graph search.

### 10.2.1 Configuration Space Obstacles

Determining whether a robot at a configuration $q$ is in collision with a known environment generally requires a complex operation involving a CAD model of the environment and robot. There are a number of free and commercial software packages that can perform this operation, and we will not delve into them here. For our purposes, it is enough to know that the workspace obstacles partition the configuration space $\mathcal{C}$ into two sets, the free space $\mathcal{C}_{\text{free}}$ and the obstacle space $\mathcal{C}_{\text{obs}}$, where $\mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{obs}}$. Joint limits are treated as obstacles in the configuration space.

With the concepts of $\mathcal{C}_{\text{free}}$ and $\mathcal{C}_{\text{obs}}$, the path planning problem becomes the problem of finding a path for a point robot among the obstacles $\mathcal{C}_{\text{obs}}$. If the
obstacles break $\mathcal{C}_{\text{free}}$ into disconnected connected components, and $q_{\text{start}}$ and $q_{\text{goal}}$ do not lie in the same connected component, then there is no collision-free path.

The explicit mathematical representation of a C-obstacle can be exceedingly complex, and for that reason C-obstacles are rarely represented exactly. Despite this, the concept of C-obstacles is very important for understanding motion planning algorithms. The ideas are best illustrated by examples.

10.2.1.1 A 2R Planar Arm

Figure 10.2 shows a 2R planar robot arm, with configuration $q = (\theta_1, \theta_2)$, among obstacles A, B, and C in the workspace. The C-space of the robot is represented by a portion of the plane with $0 \leq \theta_1 < 2\pi$, $0 \leq \theta_2 < 2\pi$. In fact, however, the topology of the C-space is a torus (or doughnut), since the edge of the square at $\theta_1 = 2\pi$ is connected to the edge $\theta_1 = 0$; similarly, $\theta_2 = 2\pi$ is connected to $\theta_2 = 0$. The square region of $\mathbb{R}^2$ is obtained by slicing the surface of the doughnut twice, at $\theta_1 = 0$ and $\theta_2 = 0$, and laying it flat on the plane.

The C-space in Figure 10.2 shows the workspace obstacles A, B, and C represented as C-obstacles. Any configuration inside a C-obstacle corresponds to penetration of the obstacle by the robot arm in the workspace. A free path for the robot arm from one configuration to another is shown in both the workspace and C-space. The path and obstacles illustrate the topology of the C-space. Note that the obstacles break $\mathcal{C}_{\text{free}}$ into three connected components.

10.2.1.2 A Circular Planar Mobile Robot

Figure 10.3 shows a top view of a circular mobile robot whose configuration is given by the location of its center, $(x, y) \in \mathbb{R}^2$. The robot translates in a plane with a single obstacle. The corresponding C-obstacle is obtained by “growing” the workspace obstacle by the radius of the mobile robot. Any point outside
10.2. Foundations

Figure 10.3: (a) A circular mobile robot (white) and a workspace obstacle (gray). The configuration of the robot is represented by \((x, y)\), the center of the robot. (b) In the C-space, the obstacle is “grown” by the radius of the robot and the robot is treated as a point. Any \((x, y)\) configuration outside the dark boundary is collision-free.

Figure 10.4: The grown C-space obstacles corresponding to two workspace obstacles and a circular mobile robot. The overlapping boundaries mean that the robot cannot move between the two obstacles.

This C-obstacle represents a free configuration of the robot. Figure 10.4 shows the workspace and C-space for two obstacles, indicating that the mobile robot cannot pass between the two obstacles.

10.2.1.3 A Polygonal Planar Mobile Robot That Translates

Figure 10.5 shows the C-obstacle for a polygonal mobile robot translating in the presence of a polygonal obstacle. The C-obstacle is obtained by sliding the robot along the boundary of the of the obstacle and tracing the position of the robot’s reference point.
Figure 10.5: (a) The configuration of a triangular mobile robot, which can translate but not rotate, is represented by the \((x, y)\) location of a reference point. Also shown is a workspace obstacle in gray. (b) The corresponding C-space obstacle is obtained by sliding the robot around the boundary of the obstacle and tracing the position of the reference point.

10.2.1.4 A Polygonal Planar Mobile Robot That Translates and Rotates

Figure 10.6 illustrates the C-obstacle for the workspace obstacle and triangular mobile robot of Figure 10.5 if the robot is now allowed to rotate. The C-space is now three-dimensional, given by \((x, y, \theta) \in \mathbb{R}^2 \times S^1\). The three-dimensional C-obstacle is the union of two-dimensional C-obstacle slices at angles \(\theta \in [0, 2\pi)\). Even for this relatively low-dimensional C-space, an exact representation of the C-obstacle is quite complex. For this reason, C-obstacles are rarely described exactly.
10.2. Foundations

Figure 10.6: (Top) A triangular mobile robot that can rotate and translate, represented by the configuration \((x, y, \theta)\). (Left) The C-space obstacle from Figure 10.5(b) when the robot is restricted to \(\theta = 0\). (Right) The full 3-D C-space obstacle shown in slices at 10° increments.

10.2.2 Distance to Obstacles and Collision Detection

Given a C-obstacle \(B\) and a configuration \(q\), let \(d(q, B)\) be the distance between the robot and the obstacle, where

\[
\begin{align*}
    d(q, B) > 0 & \quad \text{no contact with the obstacle} \\
    d(q, B) = 0 & \quad \text{contact} \\
    d(q, B) < 0 & \quad \text{penetration}.
\end{align*}
\]

The distance could be defined as the Euclidean distance between the two closest points of the robot and the obstacle.

A **distance-measurement algorithm** is one that determines \(d(q, B)\). A **collision-detection routine** determines whether \(d(q, B_i) \leq 0\) for any C-obstacle \(B_i\). A collision-detection routine returns a binary result, and may or may not utilize a distance-measurement algorithm at its core.

One popular distance-measurement algorithm is called the GJK (Gilbert-Johnson-Keerthi) algorithm, which efficiently computes the distance between two convex bodies, possibly represented by triangular meshes. Any robot or obstacle can be treated as the union of multiple convex bodies. Extensions of this algorithm are used in many distance-measurement algorithms and collision-detection routines for robotics, graphics, and game physics engines.

A simpler approach is to approximate the robot and obstacles as unions of overlapping spheres. Approximations must always be **conservative**—the approximation must cover all points of the object—so that if a collision-detection
routine indicates a free configuration $q$, we are guaranteed that the actual geometry is collision-free. As the number of spheres in the representation of the robot and obstacles increases, the closer the approximations come to the actual geometry. An example is shown in Figure 10.7.

Given a robot at $q$ represented by $k$ spheres of radius $R_i$ centered at $r_i(q)$, $i = 1 \ldots k$, and an obstacle $B$ represented by $\ell$ spheres of radius $B_j$ centered at $b_j$, $j = 1 \ldots \ell$, the distance between the robot and the obstacle can be calculated as

$$d(q, B) = \min_{i,j} \|r_i(q) - b_j\| - R_i - B_j.$$ 

Apart from determining whether a particular configuration of the robot is in collision, another useful operation is determining whether the robot collides during a particular motion segment. While exact solutions have been developed for particular object geometries and motion types, the general approach is to sample the path at finely spaced points and to “grow” the robot to ensure that if two consecutive configurations are collision-free for the grown robot, then the swept volume of the actual robot between the two configurations is also collision-free.

**10.2.3 Graphs and Trees**

Many motion planners explicitly or implicitly represent the C-space or state space as a graph. A graph consists of a collection of nodes $\mathcal{N}$ and a collection of edges $\mathcal{E}$, where each edge $e$ connects two nodes. In motion planning, a node typically represents a configuration or state, while an edge between nodes $n_1$ and $n_2$ indicates the ability to move from $n_1$ to $n_2$ without penetrating an obstacle or violating other constraints.
10.2. Foundations

A graph can be either directed or undirected. In an undirected graph, each edge is bidirectional: if the robot can travel from \( n_1 \) to \( n_2 \), then it can also travel from \( n_2 \) to \( n_1 \). In a directed graph, or digraph for short, each edge allows travel in only one direction. The same two nodes can have two edges between them, in opposite directions.

Graphs can also be weighted or unweighted. In a weighted graph, each edge has its own positive cost associated with traversing it. In an unweighted graph, each edge has the same cost (e.g., 1). Thus the most general type of graph we consider is a weighted digraph.

A tree is a digraph in which (1) there are no cycles and (2) each node has at most one parent node (i.e., at most one edge leading to the node). A tree has one root node with no parents and a number of leaf nodes with no children.

A digraph, undirected graph, and tree are illustrated in Figure 10.8.

A graph can be represented by a matrix \( A \in \mathbb{R}^{N \times N} \), where element \( a_{ij} \) of the matrix represents the cost of the edge from node \( i \) to node \( j \) (a zero indicates no edge between the nodes). A tree can be represented more compactly as a list of nodes, each with a link (and perhaps a cost) to at most one parent and a list of links (and costs) to its children.

10.2.4 Graph Search

Once the free space is represented as a graph, a motion plan can be found by searching the graph for a path from the start to the goal. One of the most powerful and popular graph search algorithms is A* (pronounced “A star”) search.

10.2.4.1 A* Search

A* search efficiently finds a minimum-cost path on a graph when the cost of the path is simply the sum of the positive edge costs along the path.

Given a graph described by a set of nodes \( \mathcal{N} = \{1, \ldots, N\} \), where node 1 is the start node, and a set of edges \( \mathcal{E} \), A* makes use of the following data structures:
• a sorted list \( \text{OPEN} \) of the nodes to be explored from, and a list \( \text{CLOSED} \) of nodes that have already been explored from;

• a matrix \( \text{cost[node1,node2]} \) encoding the set of edges, where the positive value corresponds to the cost of moving from \( \text{node1} \) to \( \text{node2} \) (a negative value indicates that no edge exists);

• an array \( \text{past_cost[node]} \) of the minimum cost found so far to reach node \( \text{node} \) from the start node; and

• a search tree defined by an array \( \text{parent[node]} \), which contains a link for each \( \text{node} \) to the node preceding it in the shortest path found so far from the start node to \( \text{node} \).

To initialize the search, the matrix \( \text{cost} \) is constructed to encode the edges, the list \( \text{OPEN} \) is initialized to the start node 1, the cost to reach the start node (\( \text{past_cost[1]} \)) is initialized as 0, and \( \text{past_cost[node]} \) for \( \text{node} \in \{2,\ldots,N\} \) is initialized as infinity (or a large number), indicating that we currently have no idea of the cost to reach those nodes.

At each step of the algorithm, the first node in \( \text{OPEN} \) is removed from \( \text{OPEN} \) and called \( \text{current} \). The node \( \text{current} \) is also added to \( \text{CLOSED} \). The first node in \( \text{OPEN} \) is one that minimizes the total estimated cost of the best path to the goal that passes through that node, and it is calculated as

\[
\text{est_total_cost[node]} = \text{past_cost[node]} + \text{heuristic_cost_to_go(node)}
\]

where \( \text{heuristic_cost_to_go(node)} \geq 0 \) is an optimistic (underestimating) estimate of the actual cost-to-go to the goal from \( \text{node} \). For the visibility graph example, an appropriate choice for the heuristic is the straight-line distance to the goal, ignoring any obstacles.

Because \( \text{OPEN} \) is a sorted list according to the estimated total cost, inserting a new node at the correct location in \( \text{OPEN} \) entails a small computational price.

If the node \( \text{current} \) is in the goal set, then the search is finished, and the path is reconstructed from the \( \text{parent} \) links. If not, for each neighbor \( \text{nbr} \) of \( \text{current} \) in the graph, which is not also in \( \text{CLOSED} \), the tentative_past_cost for \( \text{nbr} \) is calculated as \( \text{past_cost[current]} + \text{cost[current,nbr]} \). If \( \text{tentative_past_cost} < \text{past_cost[nbr]} \), then \( \text{nbr} \) can be reached less expensively than previously thought, so \( \text{past_cost[nbr]} \) is set to \( \text{tentative_past_cost} \) and \( \text{parent[nbr]} \) is set to \( \text{current} \). The node \( \text{nbr} \) is then added (or moved) in \( \text{OPEN} \) according to its estimated total cost.

The algorithm proceeds by returning to pop off of \( \text{OPEN} \) the node with the lowest estimated total cost. If \( \text{OPEN} \) is empty, then there is no solution.

The \( A^* \) algorithm is guaranteed to return a minimum-cost path, as nodes are only checked for inclusion in the goal set when they have the minimum total estimated cost of all nodes. If the node \( \text{current} \) is in the goal set, then \( \text{heuristic_cost_to_go(current)} \) is zero, and since all edge costs are positive, we know that any path found in the future must have a cost greater than
or equal to \( \text{past\_cost}[\text{current}] \). Therefore the path to \text{current} must be a shortest path. (There may be other paths of the same cost.)

If the heuristic cost-to-go is calculated exactly, considering obstacles, then \( A^* \) will expand the minimum number of nodes necessary to solve the problem. Of course, calculating the cost-to-go exactly is equivalent to solving the path planning problem, so this is impractical. Instead, the heuristic cost-to-go should be calculated quickly and should be as close as possible to the actual cost-to-go to ensure that the algorithm runs efficiently. Using an optimistic cost-to-go ensures an optimal solution.

\( A^* \) is an example of the general class of best-first searches, which always explore from the node currently deemed "best" by some measure. Not all types of best-first searches are guaranteed to return a minimum-cost path, however.

The \( A^* \) search algorithm is described in pseudocode in Algorithm 1.

\[\text{Algorithm 1 } A^* \text{ search.}\]

1: \( \text{OPEN} \leftarrow \{1\} \)
2: \( \text{past\_cost}[1] \leftarrow 0, \text{past\_cost}[\text{node}] \leftarrow \infty \) for node \( \in \{2, \ldots, N\} \)
3: while \( \text{OPEN} \) is not empty do
4: \( \text{current} \leftarrow \text{first node in } \text{OPEN}, \text{remove from } \text{OPEN} \)
5: add \( \text{current} \) to \( \text{CLOSED} \)
6: if \( \text{current} \) is in the goal set then
7: return SUCCESS and the path to \( \text{current} \)
8: end if
9: for each \( \text{nbr} \) of \( \text{current} \) not in \( \text{CLOSED} \) do
10: \( \text{tentative\_past\_cost} \leftarrow \text{past\_cost}[\text{current}] + \text{cost}[\text{current}, \text{nbr}] \)
11: if \( \text{tentative\_past\_cost} < \text{past\_cost}[\text{nbr}] \) then
12: \( \text{past\_cost}[\text{nbr}] \leftarrow \text{tentative\_past\_cost} \)
13: \( \text{parent}[\text{nbr}] \leftarrow \text{current} \)
14: put (or move) \( \text{nbr} \) in sorted list \( \text{OPEN} \) according to
15: \( \text{est\_total\_cost}[\text{nbr}] \leftarrow \text{past\_cost}[\text{nbr}] + \text{heuristic\_cost\_to\_go}(\text{nbr}) \)
16: end if
17: end for
18: end while
19: return FAILURE

10.2.4.2 Other Search Methods

- \textit{Dijkstra's method}. If the heuristic cost-to-go is always estimated as zero, then \( A^* \) always explores from the \( \text{OPEN} \) node that has been reached with minimum past cost. This variant is called \textit{Dijkstra's algorithm}, which preceded \( A^* \) historically. Dijkstra's algorithm is also guaranteed to find a minimum-cost path, but on many problems it runs slower than \( A^* \) due to the lack of a heuristic look-ahead function to help guide the search.
• **Breadth-first search.** If each edge in $\mathcal{E}$ has the same cost, Dijkstra’s algorithm reduces to bread-first search. All nodes one edge away from the start node are considered first, then all nodes two edges away, etc. The first solution found is therefore a minimum-cost path.

• **Suboptimal $A^*$ search.** If the heuristic cost-to-go is overestimated by multiplying the optimistic heuristic by a constant factor $\eta > 1$, $A^*$ search will be biased to explore from nodes closer to the goal rather than nodes with a low past cost. This may cause a solution to be found more quickly, but unlike the case of an optimistic cost-to-go heuristic, the solution will not be guaranteed to be optimal. One possibility is to run $A^*$ with an inflated cost-to-go to find an initial solution, then re-run the search with progressively smaller values of $\eta$ until the time allotted for the search has expired or a solution is found with $\eta = 1$.

### 10.3 Complete Path Planners

Complete path planners rely on an exact representation of $\mathcal{C}_{\text{free}}$. These techniques tend to be mathematically and algorithmically sophisticated, and impractical for many real systems, so we do not delve into them in detail.

One approach to complete path planning, which we will see in modified form in Section 10.5, is based on representing the complex, high-dimensional space $\mathcal{C}_{\text{free}}$ by a one-dimensional roadmap $R$ with the following properties:

(i) **Reachability.** From every point $q \in \mathcal{C}_{\text{free}}$, a free path to a point $q' \in R$ can be found trivially (e.g., a straight-line path).

(ii) **Connectivity.** For each connected component of $\mathcal{C}_{\text{free}}$, there is one connected component of $R$.

With such a roadmap, the planner can find a path between any two points in the same connected component of $\mathcal{C}_{\text{free}}$ by simply finding paths from $q_{\text{start}}$ to a point $q'_{\text{start}} \in R$, from a point $q'_{\text{goal}} \in R$ to $q_{\text{goal}}$, and from $q'_{\text{start}}$ to $q'_{\text{goal}}$ on the roadmap $R$. If a path can be found trivially between $q_{\text{start}}$ and $q_{\text{goal}}$, the roadmap may not even be used.

While constructing a roadmap of $\mathcal{C}_{\text{free}}$ is complex in general, some problems admit simple roadmaps. For example, consider a polygon translating among polygonal obstacles in the plane. As we have seen in Figure 10.5, the C-obstacles in this case are also polygons. A suitable roadmap is the weighted undirected visibility graph, with nodes at the vertices of the C-obstacles and edges between nodes that can “see” each other (i.e., the line segment between the vertices does not intersect an obstacle). The weight associated with each edge is the Euclidean distance between the nodes.

Not only is this a suitable roadmap $R$, but it allows us to use $A^*$ search to find a shortest path between any two configurations in the same connected component of $\mathcal{C}_{\text{free}}$, as the shortest path is guaranteed to either be a straight line from $q_{\text{start}}$ to $q_{\text{goal}}$, or consist of a straight line from $q_{\text{start}}$ to a node $q'_{\text{start}} \in R$, a
10.4 Grid Methods

A search like A* requires a discretization of the search space. The simplest discretization of C-space is a grid. For example, if the configuration space is n-dimensional and we desire k grid points along each dimension, the C-space is represented by $k^n$ grid points.

The $A^*$ algorithm can be used as a path planner for a C-space grid, with the following minor modifications:

- The definition of a “neighbor” of a grid point must be chosen: is the robot constrained to move in axis-aligned directions in configuration space, or can it move in multiple dimensions simultaneously? For example, for a two-dimensional C-space, neighbors could be 4-connected (on the cardinal points of a compass: north, south, east, and west) or 8-connected (diagonals allowed), as shown in Figure 10.10(a). If diagonal motions are allowed, the cost to diagonal neighbors should be penalized appropriately.

Figure 10.9: (a) The start and goal configurations for a square mobile robot (reference point shown) in an environment with a triangular and a rectangular obstacle. (b) The grown C-obstacles. (c) The visibility graph roadmap $R$ of $C_{\text{free}}$. (d) The full graph consists of $R$ plus nodes at $q_{\text{start}}$ and $q_{\text{goal}}$, along with the links connecting these nodes to visible nodes of $R$. (e) Searching the graph results in the shortest path in bold. (f) The robot traversing the path.

straight line from a node $q'_{\text{goal}} \in R$ to $q_{\text{goal}}$, and a path along the straight edges of $R$ from $q'_{\text{start}}$ to $q'_{\text{goal}}$ (Figure 10.9). Note that the shortest path requires the robot to graze the obstacles, so we implicitly treat $C_{\text{free}}$ as including its boundary.
Figure 10.10: (a) A 4-connected grid point and an 8-connected grid point for a space \( n = 2 \). (b) Grid points spaced at unit intervals. The Euclidean distance between the two points indicated is \( \sqrt{5} \) while the Manhattan distance is 3. (c) A grid representation of the C-space and a minimum-length Manhattan-distance path for the problem of Figure 10.2.

For example, the cost to a N/S/E/W neighbor could be 1, while the cost to a diagonal neighbor could be \( \sqrt{2} \). If integers are desired, for efficiency of the implementation, the approximate costs 5 and 7 could be used.

- If only axis-aligned motions are used, the heuristic cost-to-go should be based on the Manhattan distance, not the Euclidean distance. The Manhattan distance counts the number of “city blocks” that must be traveled, considering that diagonals through a block are not possible (Figure 10.10(b)).

- A node \( \text{nbr} \) is only added to OPEN if the step from \( \text{current} \) to \( \text{nbr} \) is collision-free. (The step may be considered collision-free if a grown version of the robot at \( \text{nbr} \) does not intersect any obstacles.)

- Other optimizations are possible due to the known regular structure of the grid.

An A* grid-based path planner is resolution-complete: it will find a solution if one exists at the level of discretization of the C-space. The path will be a shortest path subject to the allowed motions.

Figure 10.10(c) illustrates grid-based path planning for the 2R robot example of Figure 10.2. The C-space is represented as a grid with \( k = 32 \), i.e., a resolution of \( 360^\circ / 32 = 11.25^\circ \) for each joint. This yields a total of \( 32^2 = 1024 \) grid points.

The grid-based planner, as described, is a single-query planner: it solves each path planning query from scratch. On the other hand, if the same \( q_{\text{goal}} \)
10.4. Grid Methods

Figure 10.11: A wavefront planner on a 2D grid. The goal configuration is
given a score of 0. Then all collision-free 4-neighbors are given a score of 1. The
process continues, breadth-first, with each free neighbor (that does not have a
score already) assigned the score of its parent plus 1. Once every grid cell in
the connected component of the goal configuration is assigned a score, planning
from any location in the connected component is trivial: at every step, the
robot simply moves “downhill” to a neighbor with a lower score. Grid points in
collision receive a high score.

will be used in the same environment for multiple path planning queries, it may
be worth preprocessing the entire grid to enable fast path planning. This is the
wavefront planner, illustrated in Figure 10.11.

Although grid-based path planning is easy to implement, it is only appro-
priate for low-dimensional C-spaces. This is because the number of grid points,
and hence the computational complexity of the path planner, increases expo-
nentially with the number of dimensions $n$. For instance, a resolution $k = 100$ in
a C-space with $n = 3$ dimensions leads to 1 million grid nodes, while $n = 5$ leads
to 10 billion grid nodes and $n = 7$ leads to 100 trillion nodes. An alternative
is to reduce the resolution $k$ along each dimension, but this leads to a coarse
representation of C-space that may miss free paths.

10.4.1 Multi-Resolution Grid Representation

One way to reduce the computational complexity of a grid-based planner is to
use a multi-resolution grid representation of $C_{\text{free}}$. Conceptually, a grid point is
considered an obstacle if any part of the rectilinear cell centered on the grid point
touches a C-obstacle. To refine the representation of the obstacle, an obstacle
cell can be subdivided into smaller cells. Each dimension of the original cell is
split in half, resulting in $2^n$ sub-cells for an $n$-dimensional space. Any of the
cells that are still in contact with a C-obstacle are then subdivided further, up
to a specified maximum resolution.

The advantage of this representation is that only portions of C-space near
obstacles are refined to high resolution, while portions away from obstacles
are represented by a coarse resolution. This allows the planner to find paths
using short steps through cluttered spaces while taking large steps through wide
open space. The idea is illustrated in Figure 10.12, which uses only 10 cells to
represent an obstacle at the same resolution as a fixed grid that uses 64 cells.
332

Motion Planning

Figure 10.12: At the original C-space cell resolution, a small obstacle (indicated by a dark square) causes the whole cell to be labeled an obstacle. Subdividing the cell once shows that at least 3/4 of the cell is actually free. Three levels of subdivision results in a representation using ten total cells: four at subdivision level 3, three at subdivision level 2, and three at subdivision level 1. The cells shaded gray are the obstacle cells in the final representation. The subdivision of the original cell is also shown as a tree, specifically a quadtree, where the leaves of the tree are the final cells in the representation.

For \( n = 2 \), this multiresolution representation is called a quadtree, as each obstacle cell subdivides into \( 2^n = 4 \) cells. For \( n = 3 \), each obstacle cell subdivides into \( 2^n = 8 \) cells, and the representation is called an octree.

The multi-resolution representation of \( C_{\text{free}} \) can be built in advance of the search or incrementally as the search is being performed. In the latter case, if the step from current to nbr is found to be in collision, the step size can be halved until the step is free or the minimum step size is reached.

10.4.2 Grid Methods with Motion Constraints

The previous grid-based planners operate under the assumption that the robot can go from a cell to any neighbor cell in a regular C-space grid. This may not be possible for some robots. For example, a car cannot reach, in one step, a “neighbor” cell that is to the side of it. Also, motions for a fast-moving robot arm should be planned in the state space, not just C-space, to take the arm dynamics into account. In the state space, the robot arm cannot move in certain directions (Figure 10.13).

Grid-based planners must be adapted to account for the motion constraints of the particular robot. In particular, the constraints may result in a directed grid graph. One approach is to discretize the robot controls while still making
10.4. Grid Methods

Figure 10.13: Sample trajectories emanating from three initial states in the phase space of a dynamic system with $q \in \mathbb{R}$. If the initial state has $\dot{q} > 0$, the trajectory cannot move to the left (negative motion in $q$) instantaneously. Similarly, if the initial state has $q < 0$, the trajectory cannot move to the right instantaneously.

Figure 10.14: Discretizations of the control sets for unicycle, diff-drive, and car-like robots.

The use of a grid on the C-space or state space, as appropriate. Details for a wheeled mobile robot and a dynamic robot arm are described next.

10.4.2.1 Grid-Based Path Planning for a Wheeled Mobile Robot

As described in Chapter 13.3, the controls for simplified models of unicycle, diff-drive, and car-like robots are $(v, \omega)$, the forward-backward linear velocity and the angular velocity. The control sets for these mobile robots are shown in Figure 10.14. Also shown are proposed discretizations of the controls, as dots. Other discretizations could be chosen.

Using the control discretization, we can use a variant of Dijkstra’s algorithm to find approximately shortest paths (Algorithm 2).

The search expands from $q_{\text{start}}$ by integrating forward each of the controls for a time $\Delta t$, creating new nodes for the paths that are collision-free. Each node keeps track of the control used to reach the node as well as the cost of the path to the node. The cost of the path to a new node is the sum of the cost of the previous node current plus the cost of the action.
Algorithm 2 Grid-based Dijkstra planner for a wheeled mobile robot.

1: OPEN ← \{q\text{start}\}
2: past\_cost[q\text{start}] ← 0
3: counter ← 1
4: while OPEN is not empty and counter < MAXCOUNT do
5: current ← first node in OPEN, remove from OPEN
6: if current is in the goal set then
7: return SUCCESS and the path to current
8: end if
9: if current is not in a previously occupied C-space grid cell then
10: mark grid cell occupied
11: counter ← counter + 1
12: for each control in the discrete control set do
13: integrate control forward a short time \(\Delta t\) from current to \(q_{\text{new}}\)
14: if the path to \(q_{\text{new}}\) is collision-free then
15: compute cost of the path to \(q_{\text{new}}\)
16: place \(q_{\text{new}}\) in OPEN, sorted by cost
17: parent[\(q_{\text{new}}\)] ← current
18: end if
19: end for
20: end if
21: end while
22: return FAILURE

Integration of the controls does not move the mobile robot to exact grid points. Instead, the C-space grid comes into play in lines 9 and 10. When a node is expanded, the grid cell it sits in is marked “occupied.” In future, any node in this occupied cell will be pruned from the search. This prevents the search from expanding nodes that are close by nodes reached with a lower cost.

No more than MAXCOUNT nodes, where MAXCOUNT is a value chosen by the user, are considered during the search.

The time \(\Delta t\) should be chosen small enough that each motion step is “small.” The size of the grid cells should be chosen as large as possible while ensuring that integration of any control for time \(\Delta t\) will move the mobile robot outside of its current grid cell.

The planner terminates when current lies inside the goal region, when there are no more nodes left to expand (perhaps due to obstacles), or when MAXCOUNT nodes have been considered. Any path found is optimal for the choice of cost function and other parameters to the problem. The planner actually runs faster in somewhat cluttered spaces, as the obstacles help guide the exploration.

Some examples of motion plans for a car are shown in Figure 10.15.
Figure 10.15: (Left) A minimum-cost path for a car-like robot where each action has identical cost, favoring a short path. (Right) A minimum-cost path where reversals are penalized. Penalizing reversals requires a modification to Algorithm 2.

10.4.2.2 Grid-Based Motion Planning for a Robot Arm

One method for planning the motion for a robot arm is to decouple the problem into a path planning problem followed by a time-scaling of the path:

(i) Apply a grid-based or other path planner to find an obstacle-free path in C-space.

(ii) Time scale the path to find the fastest trajectory along the path that respects the robot’s dynamics, as described in Chapter 9.4. Or use any less aggressive time scaling.

Since the motion planning problem is broken into two steps (path planning plus time scaling), the resultant motion will not be time-optimal in general.

Another approach is to plan directly in the state space. Given a state \((q, \dot{q})\) of the robot arm, let \(A(q, \dot{q})\) represent the set of feasible accelerations based on the limited joint torques. To discretize the controls, the set \(A(q, \dot{q})\) is intersected with a grid of points of the form

\[
\sum_{i=1}^{n} c_i a_i \hat{e}_i,
\]

where \(c_i\) is an integer, \(a_i > 0\) is the acceleration step size in the \(\ddot{q}_i\) direction, and \(\hat{e}_i\) is a unit vector in the \(i\)th direction (Figure 10.16).

As the robot moves, the acceleration set \(A(q, \dot{q})\) changes, but the grid remains fixed. Because of this, and assuming a fixed integration time \(\Delta t\) at each “step” in a motion plan, the reachable states of the robot (after any integral number of steps) are confined to a grid in state space. To see this, consider a single joint angle of the robot, \(q_1\), and assume for simplicity zero initial velocity \(\dot{q}_1(0) = 0\). The velocity at timestep \(k\) takes the form

\[
\dot{q}_1(k) = \dot{q}_1(k-1) + c_1(k)a_1 \Delta t,
\]
where $c_1(k)$ is chosen from a finite set of integers. By induction, the velocity at any timestep must be of the form $a_1k_v\Delta t$, where $k_v$ is an integer. The position at timestep $k$ takes the form

$$q_1(k) = q_1(k - 1) + \dot{q}_1(k - 1)\Delta t + \frac{1}{2}c_1(k)a_1(\Delta t)^2.$$  

Plugging in the form of the velocity, we find that the position at any timestep must be of the form $a_1k_p(\Delta t)^2/2 + q_1(0)$, where $k_p$ is an integer.

To find a trajectory from a start node to a goal set, a breadth-first search can be employed to create a search tree on the state space nodes. When a node $(q, \dot{q})$ in the state space is explored from, the set $\mathcal{A}(q, \dot{q})$ is evaluated to find the discrete set of control actions. New nodes are created by integrating the control actions for time $\Delta t$. A node is discarded if the path to it is in collision or if it has been reached previously (i.e., by a trajectory taking the same or less time).

Because the joint angles and angular velocities are bounded, the state space grid is finite, and therefore it can be searched in finite time. The planner is resolution complete and returns a time-optimal trajectory, subject to the resolution specified in the control grid and timestep $\Delta t$.

The control grid step sizes $a_i$ must be chosen small enough that $\mathcal{A}(q, \dot{q})$, for any feasible state $(q, \dot{q})$, contains a representative set of points of the control grid. Choosing a finer grid for the controls, or a smaller timestep $\Delta t$, creates a finer grid in the state space and a higher likelihood of finding a solution amidst obstacles. It also allows choosing a smaller goal set while keeping points of the state space grid inside the set.

Finer discretization comes at a computational cost, however. If the resolution of the control discretization is increased by a factor of $r$ in each dimension (i.e., each $a_i$ is reduced to $a_i/r$), and the timestep size is divided by a factor of $\tau$, the computation time spent growing the search tree for a given robot motion
duration increases by a factor of \( r^n \), where \( n \) is the number of joints. For example, increasing the control grid resolution by a factor of \( r = 2 \) and decreasing the timestep by a factor of \( \tau = 4 \) for a three-joint robot results in a search that is likely to take \( 2^{3+4} = 4096 \) times longer to complete. The high computational complexity of the planner makes it impractical beyond a few degrees of freedom.

The description above ignores one important issue: the feasible control set \( \mathcal{A}(q, \dot{q}) \) changes during a timestep, so the control chosen at the beginning of the timestep may no longer be feasible by the end of the timestep. For that reason, a conservative approximation \( \tilde{\mathcal{A}}(q, \dot{q}) \subset \mathcal{A}(q, \dot{q}) \) should be used instead. This set should remain feasible over the duration of a timestep regardless of which control action is chosen. How to determine a conservative approximation \( \tilde{\mathcal{A}}(q, \dot{q}) \) is beyond the scope of this chapter, but it has to do with bounds on how rapidly the arm’s mass matrix \( M(q) \) changes with \( q \) and how fast the robot is moving. At low speeds \( \dot{q} \) and short durations \( \Delta t \), the conservative set \( \tilde{\mathcal{A}}(q, \dot{q}) \) is very close to \( \mathcal{A}(q, \dot{q}) \).

10.5 Sampling Methods

Each of the grid-based methods discussed above delivers optimal solutions subject to the chosen discretization. A drawback of the approaches is their high computational complexity, making them unsuitable for systems of more than a few degrees of freedom.

A different class of planners, known as sampling methods, relies on a random or deterministic function to choose a sample from the C-space or state space; a function to evaluate whether a sample or motion is in \( \mathcal{X}_{\text{free}} \); a function to determine nearby previous free-space samples; and a simple local planner to try to connect to, or move toward, the new sample. These functions are used to build up a graph or tree representing feasible motions of the robot.

Sampling methods generally give up on the resolution-optimal solutions of a grid search in exchange for the ability to find satisficing solutions quickly in high-dimensional state spaces. The samples are chosen to form a roadmap or search tree that quickly approximates the free space \( \mathcal{X}_{\text{free}} \) using fewer samples than would typically be required by a fixed high-resolution grid, where the number of grid points increases exponentially with the dimension of the search space. Most sampling methods are probabilistically complete: the probability of finding a solution, when one exists, approaches 100% as the number of samples goes to infinity.

Two major classes of sampling methods are rapidly-exploring random trees (RRTs) and probabilistic roadmaps (PRMs). RRTs use a tree representation for single-query planning in either C-space or state space, while PRMs are primarily C-space planners that create a roadmap graph for multiple-query planning.
10.5.1 The RRT

The RRT algorithm searches for a collision-free motion from an initial state \( x_{\text{start}} \) to a goal set \( X_{\text{goal}} \). It applies to kinematic problems, where the state \( x \) is simply the configuration \( q \), as well as dynamic problems, where the state includes the velocity. The basic RRT grows a single tree from \( x_{\text{start}} \) as outlined in Algorithm 3.

**Algorithm 3** RRT algorithm.

```
1: initialize search tree \( T \) with \( x_{\text{start}} \)
2: while \( T \) is less than the maximum tree size do
3: \( x_{\text{samp}} \leftarrow \) sample from \( X \)
4: \( x_{\text{nearest}} \leftarrow \) nearest node in \( T \) to \( x_{\text{samp}} \)
5: employ a local planner to find a motion from \( x_{\text{nearest}} \) to \( x_{\text{new}} \) in
   the direction of \( x_{\text{samp}} \)
6: if the motion is collision-free then
7:   add \( x_{\text{new}} \) to \( T \) with an edge from \( x_{\text{nearest}} \) to \( x_{\text{new}} \)
8:   if \( x_{\text{new}} \) is in \( X_{\text{goal}} \) then
9:     return SUCCESS and the motion to \( x_{\text{new}} \)
10: end if
11: end if
12: end while
13: return FAILURE
```

In a typical implementation for a kinematic problem (where \( x \) is simply \( q \)), the sampler in line 3 chooses \( x_{\text{samp}} \) randomly from an almost-uniform distribution over \( X \), with a slight bias toward states in \( X_{\text{goal}} \). The closest node \( x_{\text{nearest}} \) in \( T \) (line 4) is the one minimizing the Euclidean distance to \( x_{\text{samp}} \). The state \( x_{\text{new}} \) (line 5) is chosen as the state a small distance \( d \) from \( x_{\text{nearest}} \) on the straight line to \( x_{\text{samp}} \). Because \( d \) is small, a very simple local planner, e.g., one that returns a straight line motion, will often find a motion connecting \( x_{\text{nearest}} \) to \( x_{\text{new}} \). If the motion is collision-free, the new state \( x_{\text{new}} \) is added to \( T \).

The net effect is that the nearly uniformly distributed samples “pull” the tree toward them, causing the tree to rapidly explore \( X_{\text{free}} \). An example of the effect of this pulling action on exploration is shown in Figure 10.17.

The basic algorithm leaves many choices: how to sample from \( X \) (line 3), how to define the “nearest” node in \( T \) (line 4), and how to plan the motion to make progress toward \( x_{\text{samp}} \) (line 5). Even a small change to the sampling method, for example, can yield a dramatic change in the running time of the planner. A wide variety of planners have been proposed in the literature based on these choices and other variations. Some of these variations are described below.
10.5. Sampling Methods

The most obvious sampler is one that samples randomly from a uniform distribution over $X$. This is straightforward for Euclidean C-spaces $\mathbb{R}^n$; for $n$-joint robot C-spaces $T^n = S^1 \times \ldots \times S^1$ ($n$ times), where we can choose a uniform distribution over each joint angle; and for the C-space $\mathbb{R}^2 \times S^1$ for a mobile robot in the plane, where we can choose a uniform distribution over $\mathbb{R}^2$ and $S^1$ individually. The notion of a uniform distribution on some other curved C-spaces, for example $SO(3)$, is less straightforward.

For dynamic systems, a uniform distribution over the state space can be defined as the cross-product of a uniform distribution over C-space and a uniform distribution over a bounded velocity set.

Although the name “rapidly-exploring random trees” gets its name from the idea of a random sampling strategy, the samples need not be generated randomly. For example, a deterministic sampling scheme that generates a progressively finer (multi-resolution) grid on $X$ could be employed instead. To reflect this more general view, the approach has been called rapidly-exploring dense trees (RDTs), emphasizing the key point that the samples should eventually become dense in the state space (i.e., as the number of samples goes to infinity, the samples become arbitrarily close to every point in $X$).

10.5.1.2 Line 4: Defining the Nearest Node

Finding the “nearest” node depends on a definition of distance on $X$. For an unconstrained kinematic robot on $C = \mathbb{R}^n$, a natural choice for the distance between two points is simply the Euclidean distance. For other spaces, the choice is less obvious.

As an example, for a car-like robot with a C-space $\mathbb{R}^2 \times S^1$, which configuration is closest to the configuration $x_{\text{amp}}$: one that is rotated twenty degrees relative to $x_{\text{amp}}$, one that is 2 meters straight behind it, or one that is 1 meter
Motion Planning

Figure 10.18: Which of the three dashed configurations of the car is “closest” to the configuration in gray?

straight to the side of it (Figure 10.18)? Since the motion constraints prevent spinning in place or moving directly sideways, the configuration that is 2 meters straight behind is best positioned to make progress toward $x_{s_{amp}}$. Thus defining a notion of distance requires

• combining components of different units (e.g., degrees, meters, degrees/second, meters/second) into a single distance measure; and

• taking into account the motion constraints of the robot.

The closest node $x_{n_{earest}}$ should perhaps be defined as the one that can reach $x_{s_{amp}}$ the fastest, but computing this is as hard as solving the motion planning problem.

A simple choice of a distance measure from $x$ to $x_{s_{amp}}$ is the weighted sum of the distances along the different components of $x_{s_{amp}} - x$. The weights choose the relative importance of the different components. If more is known about the time-limited reachable sets of a motion-constrained robot from a state $x$, this information can be used in determining the nearest node. In any case, the nearest node should be computed quickly. Finding a nearest neighbor is a common problem in computational geometry, and various algorithms, such as $kd$ trees and hashing, can be used to solve it efficiently.

10.5.1.3 Line 5: The Local Planner

The job of the local planner is to find a motion from $x_{n_{earest}}$ to some point $x_{n_{ew}}$ which is closer to $x_{s_{amp}}$. The planner should be simple and it should run quickly. Three examples are:

**Straight-line planner.** The plan is a straight line to $x_{n_{ew}}$, which may be chosen at $x_{s_{amp}}$ or at a fixed distance $d$ from $x_{n_{earest}}$ on the straight line to $x_{s_{amp}}$. This is for kinematic systems with no motion constraints.

**Discretized controls.** For systems with motion constraints, such as wheeled mobile robots or dynamic systems, the controls can be discretized into a discrete set $\{u_1, u_2, \ldots\}$, as in the grid methods with motion constraints (Section 10.4.2 and Figures 10.14 and 10.16). Each control is integrated
from \( x_{\text{nearest}} \) for a fixed time \( \Delta t \) using \( \dot{x} = f(x,u) \). The resulting state that is closest to \( x_{\text{samp}} \) is chosen as \( x_{\text{new}} \).

**Wheeled robot planners.** For a wheeled mobile robot, local plans can be found using Reeds-Shepp curves or polynomial functions of time of the differentially flat output, as described in Chapter 13.3.3.

Other robot-specific local planners can be designed.

### 10.5.1.4 Other RRT Variants

The performance of the basic RRT algorithm depends heavily on the choice of the sampling method, the distance measure, and the local planner. Beyond these choices, two other variants of the basic RRT are outlined below.

#### Bidirectional RRT.

The bidirectional RRT grows two trees: one “forward” from \( x_{\text{start}} \) and one “backward” from \( x_{\text{goal}} \). The algorithm alternates between growing the forward tree and the backward tree, and every so often attempts to connect the two trees by choosing \( x_{\text{samp}} \) from the other tree. The advantage of this approach is that a single goal state \( x_{\text{goal}} \) can be reached exactly, rather than just a goal set \( X_{\text{goal}} \). Another advantage is that in many environments, the two trees are likely to find each other much faster than a single “forward” tree will find a goal set.

The major problem is that the local planner might not be able to connect the two trees exactly. For example, the discretized controls planner of Section 10.5.1.3 is highly unlikely to create a motion exactly to a node in the other tree. In this case, the two trees may be considered more-or-less connected when points on each tree are sufficiently close. The “broken” discontinuous trajectory can be returned and patched by a smoothing method (Section 10.8).

#### RRT*. The basic RRT algorithm returns SUCCESS once a motion to \( X_{\text{goal}} \) is found. An alternative is to continue running the algorithm and to terminate the search only when another termination condition is reached (e.g., maximum running time or maximum tree size). Then the motion with the minimum cost can be returned. In this way, the RRT solution may continue to improve as time goes by. Because edges in the tree are never deleted or changed, however, the RRT generally does not converge to an optimal solution.

RRT* is a variation of the single-tree RRT that continually rewire the search tree to ensure that it always encodes the shortest path from \( x_{\text{start}} \) to each node in the tree. The basic approach works for C-space path planning with no motion constraints, allowing exact paths from any node to any other node.

To modify the RRT to the RRT*, line 7 of the RRT algorithm, which inserts \( x_{\text{new}} \) in \( T \) with an edge from \( x_{\text{nearest}} \) to \( x_{\text{new}} \), is replaced by a test of all nodes \( x \in X_{\text{near}} \) in \( T \) that are sufficiently near to \( x_{\text{new}} \). An edge to \( x_{\text{new}} \) is created from the \( x \in X_{\text{near}} \) that (1) has a collision-free motion by the local planner and (2) minimizes the total cost of the path from \( x_{\text{start}} \) to \( x_{\text{new}} \), not just the cost of
the added edge. The total cost is the cost to reach the candidate \( x \in X_{\text{near}} \) plus the cost of the new edge.

The next step is to consider each \( x \in X_{\text{near}} \) to see if it could be reached by lower cost by a motion through \( x_{\text{new}} \). If so, the parent of \( x \) is changed to \( x_{\text{new}} \). In this way, the tree is incrementally rewired to eliminate high-cost motions in favor of the minimum-cost motions available so far.

The definition of \( X_{\text{near}} \) depends on the number of samples in the tree, details of the sampling method, the dimension of the search space, and other factors.

Unlike the RRT, the solution provided by RRT* approaches the optimal solution as the number of sample nodes increases. Like the RRT, RRT* is probabilistically complete. Figure 10.19 demonstrates the rewiring behavior of RRT* compared to RRT for a simple example in \( \mathcal{C} = \mathbb{R}^2 \).

### 10.5.2 The PRM

The PRM uses sampling to build a roadmap representation of \( C_{\text{free}} \) (Section 10.3) before answering any specific queries. The roadmap is an undirected graph: the robot can move in either direction along any edge exactly from one node to the next. For this reason, PRMs primarily apply to kinematic problems for which an exact local planner exists that can find a path (ignoring obstacles) from any \( q_1 \) to any \( q_2 \). The simplest example is a straight-line planner for a robot with no kinematic constraints.

Once the roadmap is built, a particular start node \( q_{\text{start}} \) can be added to the graph by attempting to connect it to the roadmap, starting with the closest node. The same is done for the goal node \( q_{\text{goal}} \). The graph is then searched for
10.5. Sampling Methods

Figure 10.20: An example PRM roadmap for a point robot in \( \mathcal{C} = \mathbb{R}^2 \). The \( k = 3 \) closest neighbors are considered for connection to a sample node \( q \). The degree of a node can be greater than three since it may be a close neighbor of many nodes. Figure from [19] used with permission.

a path, typically using \( A^* \). Thus the query can be answered efficiently once the roadmap has been built.

PRMs allow the possibility of building a roadmap quickly and efficiently relative to constructing a roadmap using a high-resolution grid representation. This is because the volume fraction of the C-space that is “visible” by the local planner from a given configuration does not typically decrease exponentially with increasing dimension of the C-space.

The algorithm to construct a roadmap \( R \) with \( N \) nodes is outlined in Algorithm 4 and illustrated in Figure 10.20.

**Algorithm 4** PRM roadmap construction algorithm (undirected graph).

```plaintext
1: for \( i = 1 \ldots N \) do
2: \( q_i \leftarrow \) sample from \( \mathcal{C}_{\text{free}} \)
3: add \( q_i \) to \( R \)
4: end for
5: for \( i = 1 \ldots N \) do
6: \( \mathcal{N}(q_i) \leftarrow k \) closest neighbors of \( q_i \)
7: for each \( q \in \mathcal{N}(q_i) \) do
8: if there is a collision-free local path from \( q \) to \( q_i \) and there is not already an edge from \( q \) to \( q_i \) then
9: add an edge from \( q \) to \( q_i \) to the roadmap \( R \)
10: end if
11: end for
12: end for
13: return \( R \)
```
A key choice in the PRM roadmap construction algorithm is how to sample from $C_{\text{free}}$. While the default might be sampling randomly from a uniform distribution on $C$ and eliminating configurations in collision, it has been shown that sampling more densely near obstacles can improve the likelihood of finding narrow passages, thus significantly reducing the number of samples needed to properly represent the connectivity of $C_{\text{free}}$. Another option is deterministic multi-resolution sampling.

10.6 Virtual Potential Fields

Virtual potential field methods are inspired by potential energy fields in nature, such as gravitational and magnetic fields. From physics we know that a potential field $P(q)$ defined over $C$ induces a force $F = -\partial P/\partial q$ that drives an object from high to low potential. For example, if $h$ is the height above the Earth’s surface in a uniform gravitational potential field ($g = 9.81 \text{ m/s}^2$), then the potential energy of a mass $m$ is $P(h) = mgh$ and the force acting on it is $F = -\partial P/\partial h = -mg$. The force will cause the mass to fall to the Earth’s surface.

In robot motion control, the goal configuration $q_{\text{goal}}$ is assigned a low virtual potential energy and obstacles are assigned a high virtual potential. Applying a force to the robot proportional to the negative gradient of the virtual potential naturally pushes the robot toward the goal and away from the obstacles.

A virtual potential field is very different from the planners we have seen so far. Typically the gradient of the field can be calculated quickly, so the motion can be calculated in real time (reactive control) instead of planned in advance. With appropriate sensors, the method can even handle obstacles that move or appear unexpectedly. The drawback of the basic method is that the robot can get stuck in local minima of the potential field, away from the goal, even when a feasible motion to the goal exists. In certain cases it is possible to design the potential to guarantee that the only local minimum is at the goal, eliminating this problem.

10.6.1 A Point in C-space

Let’s begin by assuming a point robot in its C-space. A goal configuration $q_{\text{goal}}$ is typically encoded by a quadratic potential energy “bowl” with zero energy at the goal,

$$P_{\text{goal}}(q) = \frac{1}{2}(q - q_{\text{goal}})^T K (q - q_{\text{goal}}),$$

where $K$ is a symmetric positive-definite weighting matrix (for example, the identity matrix). The force induced by this potential is

$$F_{\text{goal}}(q) = -\frac{\partial P_{\text{goal}}}{\partial q} = K(q_{\text{goal}} - q),$$

an attractive force proportional to the distance from the goal.
The repulsive force induced by a C-obstacle $B$ can be calculated from the distance $d(q,B)$ to the obstacle (Section 10.2.2):

$$P_B(q) = \frac{k}{2d^2(q,B)},$$

(10.3)

where $k > 0$ is a scaling factor. The potential is only properly defined for points outside the obstacle, $d(q,B) > 0$. The force induced by the obstacle potential is

$$F_B(q) = -\frac{\partial P_B}{\partial q} = \frac{k}{d^3(q,B)} \frac{\partial d}{\partial q}.$$

The total potential is obtained by summing the attractive goal potential and the repulsive obstacle potentials,

$$P(q) = P_{\text{goal}}(q) + \sum_i P_{B_i}(q),$$

yielding a total force

$$F(q) = F_{\text{goal}}(q) + \sum_i F_{B_i}(q).$$

Note that the sum of the attractive and repulsive potentials may not give a minimum (zero force) exactly at $q_{\text{goal}}$. Also, it is common to put a bound on the maximum potential and force, as the simple obstacle potential (10.3) would otherwise yield unbounded potentials and forces near the boundaries of obstacles.

Figure 10.21 shows a potential field for a point in $\mathbb{R}^2$ with three circular obstacles. The contour plot of the potential field clearly shows the global minimum near the center of the space (near the goal marked with a +), a local minimum near the two obstacles on the left, as well as saddles (critical points that are a maximum in one direction and a minimum in the other direction) near the obstacles. Saddles are generally not a problem, as a small perturbation allows continued progress toward the goal. Local minima away from the goal are a problem, however, as they attract nearby starting points.

To actually control the robot using the calculated $F(q)$, we have several options, two of which are:

- Apply the calculated force plus damping,

$$u = F(q) + B\ddot{q}.$$  

(10.4)

If $B$ is positive definite, then it dissipates energy for all $\dot{q} \neq 0$, reducing oscillation and guaranteeing that the robot will come to rest. If $B = 0$, the robot continues to move while maintaining constant total energy, which is the sum of the initial kinetic energy $\frac{1}{2}\dot{q}^T(q(0))M(q(0))\dot{q}(0)$ plus the initial virtual potential energy $P(q(0))$.

The motion of the robot under the control law (10.4) can be visualized as a ball rolling in gravity on the potential surface of Figure 10.21, where the dissipative force is rolling friction.
• Treat the calculated force as a commanded velocity instead:

\[ \dot{q} = F(q). \]  \hspace{1cm} (10.5)

This automatically eliminates oscillations.

Using the simple obstacle potential (10.3), even distant obstacles have a nonzero effect on the motion of the robot. To speed up evaluation of the repulsive terms, distant obstacles could be ignored. We can define a range of influence
of the obstacles \( d_{\text{range}} > 0 \) so that the potential is zero for all \( d(q, \mathcal{B}) \geq d_{\text{range}} \):

\[
U_B(q) = \begin{cases} 
  k \left( \frac{d_{\text{range}} - d(q, \mathcal{B})}{d_{\text{range}} - d_{\text{range}}} \right)^2 & \text{if } d(q, \mathcal{B}) < d_{\text{range}} \\
  0 & \text{otherwise.}
\end{cases}
\]

Another issue is that \( d(q, \mathcal{B}) \) and its gradient are generally difficult to calculate. An approach to dealing with this is described in Section 10.6.3.

### 10.6.2 Navigation Functions

A significant problem with the potential field method is local minima. While potential fields may be appropriate for relatively uncluttered spaces, or for rapid response to unexpected obstacles, they are likely to get the robot stuck in local minima for many practical applications.

One method that avoids this issue is the wavefront planner of Figure 10.11. The wavefront algorithm creates a local-minimum-free potential function by a breadth-first traversal of every cell reachable from the goal cell in a grid representation of the free space. Therefore, if a solution exists to the motion planning problem, then simply moving “downhill” at every step is guaranteed to bring the robot to the goal.

Another approach to local-minimum-free gradient following is based on replacing the virtual potential function with a navigation function. A navigation function \( \varphi(q) \) is a type of virtual potential function that

(i) is smooth (or at least twice differentiable) on \( q \);

(ii) has a bounded maximum value (e.g., 1) on the boundaries of all obstacles;

(iii) has a single minimum at \( q_{\text{goal}} \); and

(iv) has a full-rank Hessian \( \partial^2 \varphi / \partial q^2 \) at all critical points \( q \) where \( \partial \varphi / \partial q = 0 \) (i.e., \( \varphi(q) \) is a Morse function).

Condition 1 ensures that the Hessian \( \partial^2 \varphi / \partial q^2 \) exists. Condition 2 puts an upper bound on the virtual potential energy of the robot. The key conditions are 3 and 4. Condition 3 ensures that of the critical points of \( \varphi(q) \) (including minima, maxima, and saddles), there is only one minimum, at \( q_{\text{goal}} \). This ensures that \( q_{\text{goal}} \) is at least locally attractive. However, there may be saddle points which are minima along a subset of directions. Condition 4 ensures that the set of initial states that are attracted to any saddle point has empty interior (zero measure), and therefore almost every initial state converges to the unique minimum \( q_{\text{goal}} \).

While constructing navigation potential functions with only a single minimum is non-trivial, Rimon and Koditschek show how to construct them for the particular case of an \( n \)-dimensional \( \mathcal{C}_{\text{free}} \) consisting of all points inside an \( n \)-sphere of radius \( R \) and outside of smaller spherical obstacles \( \mathcal{B}_i \) of radius \( r_i \) centered at \( q_i \), i.e., \( \{ q \in \mathbb{R}^n \mid \|q\| \leq R \text{ and } \|q - q_i\| > r_i \text{ for all } i \} \). This is called
a sphere world. While a real C-space is unlikely to be a sphere world, Rimon
and Koditschek show that the boundaries of the obstacles, and the associated
navigation function, can be deformed to a much broader class of star-shaped ob-
stances. A star-shaped obstacle is one that has a center point from which the line
segment to any point on the obstacle boundary is contained completely within
the obstacle. A star world is a star-shaped C-space which has star-shaped ob-
stances. Thus finding a navigation function for an arbitrary star world reduces
to finding a navigation function for a “model” sphere world that has centers
at the centers of the star-shaped obstacles, then stretching and deforming that
navigation function to one that fits the star world. Rimon and Koditschek give
a systematic procedure to accomplish this.

Figure 10.22 shows a deformation of a navigation function on a model sphere
world to a star world for the case $C \subset \mathbb{R}^2$.

10.6.3 Workspace Potential

A difficulty in calculating the repulsive force from an obstacle is calculating the
distance to the obstacle, $d(q, B)$. One approach that avoids the exact calculation
is to represent the boundary of an obstacle as a set of point obstacles, and to
represent the robot by a small set of control points. Let the Cartesian location
of control point $i$ on the robot be written $f_i(q) \in \mathbb{R}^3$ and boundary point $j$ of
the obstacle be $c_j \in \mathbb{R}^3$. Then the distance between the two points is $\|f_i(q) - c_j\|$, and the potential at the control point $i$ due to the obstacle point $j$ is

$$P'_{ij}(q) = \frac{k}{2\|f_i(q) - c_j\|^2},$$

Figure 10.22: (Left) A model “sphere world” with five circular obstacles. The
contour plot of a navigation function is shown. The goal is at $(0, 0)$. Note
that the obstacles induce saddle points near the obstacles, but no local minima.
(Right) A “star world” obtained by deforming the obstacles and the potential
while retaining a navigation function. Figure from [109] used with permission.
yielding the repulsive force at the control point

\[ F'_{ij}(q) = -\frac{\partial F'_{ij}}{\partial q} = k \frac{\partial f_i}{\partial q}^T (f_i(q) - c_j) \in \mathbb{R}^3. \]

To turn the linear force \( F'_{ij}(q) \in \mathbb{R}^3 \) into a generalized force \( F_{ij}(q) \in \mathbb{R}^n \) acting on the robot arm or mobile robot, we first find the Jacobian \( J_i(q) \in \mathbb{R}^{3 \times n} \) relating \( \dot{q} \) to the linear velocity of the control point \( \dot{f}_i \):

\[ \dot{f}_i = \frac{\partial f_i}{\partial q} \dot{q} = J_i(q)\dot{q}. \]

By the principle of virtual work, the generalized force \( F_{ij}(q) \in \mathbb{R}^n \) due to the repulsive linear force \( F'_{ij}(q) \in \mathbb{R}^3 \) is simply

\[ F_{ij}(q) = J_i^T(q) F'_{ij}(q). \]

Now the total force \( F(q) \) acting on the robot is the sum of the easily calculated attractive force \( F_{\text{goal}}(q) \) and the repulsive forces \( F_{ij}(q) \) for all \( i \) and \( j \).

### 10.6.4 Wheeled Mobile Robots

The preceding analysis assumes that a control force \( u = F(q) + B\dot{q} \) (for control law (10.4)) or a velocity \( \dot{q} = F(q) \) (for control law (10.5)) can be applied in any direction. If the robot is a wheeled mobile robot subject to rolling constraints \( A(q)\dot{q} = 0 \), however, the calculated \( F(q) \) must be projected to controls \( F_{\text{proj}}(q) \) that move the robot tangent to the constraints. For a kinematic robot employing the control law \( \dot{q} = F_{\text{proj}}(q) \), a suitable projection is

\[ F_{\text{proj}}(q) = \left( I - A^T(q)(A(q)A^T(q))^{-1}A(q) \right) F(q). \]

For a dynamic robot employing the control law \( u = F_{\text{proj}}(q) + B\dot{q} \), the projection is discussed in Chapter 8.7.

### 10.6.5 Use of Potential Fields in Planners

A potential field can be used in conjunction with a path planner. For example, a best-first search like \( A^* \) can use the potential as an estimate of the cost-to-go. Incorporating search prevents the planner from getting permanently stuck in local minima.

### 10.7 Nonlinear Optimization

The motion planning problem can be expressed as a general nonlinear optimization with equality and inequality constraints, taking advantage of a number of software packages to solve such problems. Nonlinear optimization problems can
be solved by gradient-based methods, like sequential quadratic programming (SQP), and non-gradient methods, like simulated annealing, Nelder-Mead optimization, and genetic programming. Like many nonlinear optimization problems, however, these methods are not generally guaranteed to find a feasible solution when one exists, let alone an optimal one. For methods that use gradients of the objective function and constraints, however, we can expect a locally optimal solution if we start the process with a guess that is “close” to a solution.

The general problem can be written

\[
\begin{align*}
\text{find} & \quad u(t), q(t), T \\
\text{minimizing} & \quad J(u(t), q(t), T) \\
\text{subject to} & \quad \dot{x}(t) = f(x(t), u(t)) \quad \forall t \in [0, T] \\
& \quad u(t) \in \mathcal{U} \quad \forall t \in [0, T] \\
& \quad q(t) \in \mathcal{C}_{\text{free}} \quad \forall t \in [0, T] \\
& \quad x(0) = x_{\text{start}} \\
& \quad x(T) = x_{\text{goal}}.
\end{align*}
\]

To approximately solve this problem by nonlinear optimization, the control \(u(t)\), trajectory \(q(t)\), and equality and inequality constraints (10.8)-(10.12) must be discretized. This is typically done by ensuring that the constraints are satisfied at a fixed number of points distributed evenly over the interval \([0, T]\), and by choosing a finite-parameter representation of the position and/or control histories. We have at least three choices of how to parametrize the position and controls:

(i) \textit{Parametrize the trajectory} \(q(t)\). In this case, we solve for the parametrized trajectory directly. The control forces \(u(t)\) at any time are calculated using the equations of motion. This approach does not apply to systems with fewer controls than configuration variables, \(m < n\).

(ii) \textit{Parametrize the control} \(u(t)\). We solve for \(u(t)\) directly. Calculating the state \(x(t)\) requires integrating the equations of motion.

(iii) \textit{Parametrize both} \(q(t)\) \textit{and} \(u(t)\). We have a larger number of variables, since we are parametrizing both \(q(t)\) and \(u(t)\). Also, we have a larger number of constraints, as \(q(t)\) and \(u\) must satisfy the dynamic equations \(\dot{x} = f(x, u)\) explicitly, typically at a fixed number of points distributed evenly over the interval \([0, T]\). We must be careful to choose the parametrizations of \(q(t)\) and \(u(t)\) to be consistent with each other, so that the dynamic equations can be satisfied at these points.

A trajectory or control history can be parametrized in any number of ways. The parameters can be the coefficients of a polynomial in time, the coefficients of a truncated Fourier series, spline coefficients, wavelet coefficients, piecewise constant acceleration or force segments, etc. For example, the control \(u_i(t)\)
could be represented by \( p + 1 \) coefficients \( a_j \) of a polynomial in time:

\[
    u_i(t) = \sum_{j=0}^{p} a_j t^j.
\]

In addition to the parameters for the state or control history, the total time \( T \) may be another control parameter. The choice of parametrization has implications for the efficiency of the calculation of \( q(t) \) and \( u(t) \) at a given time \( t \). The choice of parametrization also determines the sensitivity of the state and control to the parameters, and whether each parameter affects the profiles at all times \([0, T]\) or just on a finite-time support base. These are important factors in the stability and efficiency of the numerical optimization.

\section*{10.8 Smoothing}

The axis-aligned motions of a grid planner and the randomized motions of sampling planners may lead to jerky motion of the robot. One approach to dealing with this issue is to let the planner handle the work of searching globally for a solution, then post-process the resulting motion to make it smoother.

There are many ways to do this; two possibilities are outlined below.

\textbf{Nonlinear optimization.} While gradient-based nonlinear optimization may fail to find a solution if initialized with a random initial trajectory, it can make an effective post-processing step, since the plan initializes the optimization with a “reasonable” solution. The initial motion must be converted to a parametrized representation of the controls, and the cost \( J(u(t), q(t), T) \) can be expressed as a function of \( u(t) \) or \( q(t) \). For example,

\[
    J = \frac{1}{2} \int_0^T \dot{u}^T(t) \dot{u}(t) dt
\]

penalizes the rate of control change. This has an analogy in human motor control, where the smoothness of human arm motions has been attributed to minimizing the rate of change of acceleration of the joints.

\textbf{Subdivide and reconnect.} A local planner can be used to attempt a connection between two distant points on a path. If this new connection is collision-free, it replaces the original path segment. Since the local planner is designed to produce short, smooth paths, the new path is likely shorter and smoother than the original. This test-and-replace procedure can be applied iteratively to randomly chosen points on the path. Another possibility is to use a recursive procedure that subdivides the path first into two pieces and attempts to replace each piece with a shorter path; then, if either portion could not be replaced by a shorter path, subdivide again; etc.
10.9 Summary

- A fairly general statement of the motion planning problem is: Given an initial state \( x(0) = x_{\text{start}} \) and a desired final state \( x_{\text{goal}} \), find a time \( T \) and a set of controls \( u : [0, T] \to \mathcal{U} \) such that the motion satisfies \( x(T) \in \mathcal{X}_{\text{goal}} \) and \( q(x(t)) \in \mathcal{C}_{\text{free}} \) for all \( t \in [0, T] \).

- Motion planning problems can be classified in the following categories: path planning vs. motion planning; fully actuated vs. constrained or underactuated; online vs. offline; optimal vs. satisficing; exact vs. approximate; with or without obstacles.

- Motion planners can be characterized by the following properties: multiple-query vs. single-query; anytime planning or not; complete, resolution complete, probabilistically complete, or none of the above; and computational complexity.

- Obstacles partition the C-space into free C-space \( \mathcal{C}_{\text{free}} \) and obstacle space \( \mathcal{C}_{\text{obs}} \), \( \mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{obs}} \). Obstacles may split \( \mathcal{C}_{\text{free}} \) into multiple connected components. There is no feasible path between configurations in different connected components.

- A conservative check of whether a configuration \( q \) is in collision uses a simplified “grown” representation of the robot and obstacles. If there is no collision between the grown bodies, then the configuration is guaranteed collision-free. Checking if a path is collision-free usually involves sampling the path at finely spaced points, and ensuring that if the individual configurations are collision-free, then the swept volume of the robot path is collision-free.

- The C-space geometry is often represented by a graph consisting of nodes and edges between the nodes, where edges represent feasible paths. The graph can be undirected (edges flow both directions) or directed (edges only flow one direction). Edges can be unweighted or weighted according to their cost of traversal. A tree is a directed graph with no cycles in which each node has at most one parent.

- A roadmap path planner uses a graph representation of \( \mathcal{C}_{\text{free}} \), and path planning problems can be solved using a simple path from \( q_{\text{start}} \) onto the roadmap, a path along the roadmap, and a simple path from the roadmap to \( q_{\text{goal}} \).

- A* is a popular search method that finds minimum-cost paths on a graph. It operates by always exploring from the node that is (1) unexplored and (2) expected to be on a path with minimum estimated total cost. The estimated total cost is the sum of edge weights to reach the node from the start node plus an estimate of the cost-to-go to the goal. The cost-to-go estimate should be optimistic to ensure that the search returns an optimal solution.
10.9. Summary

- A grid-based path planner discretizes the C-space into a graph consisting of neighboring points on a regular grid. A multi-resolution grid can be used to allow large steps in wide-open spaces and smaller steps near obstacle boundaries.

- Discretizing the control set allows robots with motion constraints to take advantage of grid-based methods. If integrating a control does not land the robot exactly on a grid point, the new state may still be pruned if a state in the same grid cell was already achieved with a lower cost.

- The basic RRT algorithm grows a single search tree from \( x_{\text{start}} \) to find a motion to \( x_{\text{goal}} \). It relies on a sampler to find a sample \( x_{\text{samp}} \) in \( \mathcal{X} \); an algorithm to find the closest node \( x_{\text{nearest}} \) in the search tree; and a local planner to find a motion from \( x_{\text{nearest}} \) to a point closer to \( x_{\text{samp}} \). The sampling is chosen to cause the tree to explore \( \mathcal{X}_{\text{free}} \) quickly.

- The bidirectional RRT grows a search tree from both \( x_{\text{start}} \) and \( x_{\text{goal}} \) and attempts to join them up. RRT* returns solutions that tend toward the optimal as planning time goes to infinity.

- The PRM builds a roadmap of \( \mathcal{C}_{\text{free}} \) for multiple-query planning. The roadmap is built by sampling \( \mathcal{C}_{\text{free}} \) \( N \) times, then using a local planner to attempt to connect each sample with several of its nearest neighbors. The roadmap is searched for plans using \( A^* \).

- Virtual potential fields are inspired by potential energy fields such as gravitational and electromagnetic fields. The goal point creates an attractive potential while obstacles create a repulsive potential. The total potential \( P(q) \) is the sum of these, and the virtual force applied to the robot is \( F(q) = -\partial P/\partial q \). The robot is controlled by applying this force plus damping, or by simulating first-order dynamics and driving the robot with \( F(q) \) as a velocity. Potential field methods are conceptually simple but may get the robot stuck in local minima away from the goal.

- A navigation function is a potential function with no local minima. Navigation functions result in near-global convergence to \( q_{\text{goal}} \). While navigation functions are difficult to design in general, they can be designed systematically for certain environments.

- Motion planning problems can be converted to general nonlinear optimization problems with equality and inequality constraints. While optimization methods can be used to find smooth, near-optimal motions, they tend to get stuck in local minima in cluttered C-spaces. Optimization methods typically require a good initial guess at a solution.

- Motions returned by grid-based and sampling-based planners tend to be jerky. Smoothing the plan using nonlinear optimization or subdivide-and-reconnect can improve the quality of the motion.
10.10 Notes and References

Excellent textbooks covering motion planning broadly include the original text by Latombe [57] in 1991 and the more recent texts by Choset et al. [19] and LaValle [59]. Other summaries of the state-of-the-art in motion planning can be found in the Handbook of Robotics [49], and, particularly for robots subject to nonholonomic and actuation constraints, in the Control Handbook [74], the Encyclopedia of Systems and Control [73], and the textbook by Murray, Li, and Sastry [90]. Search algorithms and other algorithms for artificial intelligence are covered in detail by Russell and Norvig [111].

Landmark early work on motion planning for Shakey the Robot at SRI led to the development of $A^*$ search in 1968 by Hart, Nilsson, and Raphael [35]. This work built on the newly established approach to dynamic programming for optimal decision-making, as described by Bellman and Dreyfus [8], and improved on the performance of Dijkstra’s algorithm [25]. A suboptimal anytime variant of $A^*$ was proposed in [66]. Early work on multiresolution path planning is described in [45, 70, 31, 36] based on hierarchical decompositions of C-space [112].

One early line of work focused on exact characterization of the free C-space in the presence of obstacles. The visibility graph approach for polygons moving among polygons was developed by Lozano-Pérez and Wesley in 1979 [71]. In more general settings, researchers used sophisticated algorithms and mathematical methods to develop cellular decompositions and exact roadmaps of the free C-space. Important examples of this work are a series of papers by Schwartz and Sharir on the piano movers’ problem [113, 114, 115] and Canny’s PhD thesis [16].

As a result of the mathematical sophistication and high computational complexity needed to exactly represent the topology of C-spaces, a movement formed in the 1990s to approximately represent C-space using samples, and that movement carries on strong today. That line of work has followed two main branches, probabilistic roadmaps (PRMs) [48] and rapidly exploring random trees (RRTs) [60, 62, 61]. Due to their ability to handle complex high-dimensional C-spaces relatively efficiently, research in sampling-based planners has exploded, and some of the subsequent work is summarized in [19, 59]. The bidirectional RRT and RRT*, highlighted in this chapter, are described in [59] and [47], respectively.

The grid-based approach to motion planning for a wheeled mobile robot was introduced by Barraquand and Latombe [6], and the grid-based approach to time-optimal motion planning for a robot arm with dynamic constraints was introduced in [17, 27, 26].

The GJK algorithm for collision detection was derived in [34]. Open-source collision-detection packages are implemented in the Open Motion Planning Library (OMPL) [133] and the Robot Operating System (ROS). An approach to approximating polyhedra with spheres for fast collision detection is described in [43].

The potential field approach to motion planning and real-time obstacle avoidance was first introduced by Khatib and is summarized in [51]. A search-
based planner using a potential field to guide the search is described by Barraquand et al. [5]. The construction of navigation functions, potential functions with a unique local minimum, is described in a series of papers by Koditschek and Rimon [55, 53, 54, 109, 110].

Nonlinear optimization-based motion planning has been formulated in a number of publications, including the classic computer graphics paper by Witkin and Kass [140] using optimization to generate the motions of an animated jumping lamp; work on generating motion plans for dynamic nonprehensile manipulation [76]; Newton algorithms for optimal motions of mechanisms [64]; and more recent developments in short-burst sequential action control, which solves both the motion planning and feedback control problems [2, 136]. Path smoothing for mobile robot paths by subdivide and reconnect is described by Laumond et al. [58].
10.11 Exercises

1. A path is *homotopic* to another if it can be continuously deformed into the other without moving the endpoints. In other words, it can be stretched and pulled like a rubber band, but it cannot be cut and pasted back together. For the C-space of Figure 10.2, draw a path from the start to the goal that is not homotopic to the one shown.

2. Label the connected components in Figure 10.2. For each connected component, draw a picture of the robot for one configuration in the connected component.

3. Assume that \( \theta_2 \) joint angles in the range \([175^\circ, 185^\circ]\) result in self-collision for the robot of Figure 10.2. Draw the new joint limit C-obstacle on top of the existing C-obstacles and label the resulting connected components of \( C_{free} \). For each connected component, draw a picture of the robot for one configuration in the connected component.

4. Draw the C-obstacle corresponding to the obstacle and translating planar robot in Figure 10.23.

5. Write a program that accepts as input the coordinates of a polygonal robot (relative to a reference point on the robot) and the coordinates of a polygonal obstacle and produces as output a drawing of the corresponding C-space obstacle. In Mathematica, you may find the function `ConvexHull` useful. In MATLAB, try `convhull`.

6. Calculating a square root is typically computationally expensive. For a robot and an obstacle represented as collections of spheres (Section 10.2.2), provide a method for calculating the distance between the robot and obstacle.
7. For Figure 10.8(c), give the order that the nodes in the tree are visited in a complete breadth-first search and a complete depth-first search. Assume that leftmost branches are always followed first.

8. Draw the visibility roadmap for the C-obstacles and $q_{\text{start}}$ and $q_{\text{goal}}$ in Figure 10.24. Indicate the shortest path.

9. Not all edges of the visibility roadmap described in Section 10.3 are needed. Prove that an edge between two vertices of C-obstacles need not be included in the roadmap if either end of the edge does not hit the obstacle tangentially. In other words, if the edge ends by “colliding” with an obstacle, it will never be used in a shortest path.

10. You will implement an $A^*$ path planner for a point robot in a plane with obstacles. The planar region is a $100 \times 100$ area. The program will generate a graph consisting of $N$ nodes and $E$ edges, where $N$ and $E$ are chosen by the user. After generating $N$ randomly chosen nodes, the program should connect randomly chosen nodes with edges until $E$ unique edges have been generated. The cost associated with each edge is the Euclidean distance between the nodes. Finally, the program should display the graph, search the graph using $A^*$ for the shortest path between nodes 1 and $N$, and display the shortest path or indicate FAILURE if no path exists. The heuristic cost-to-go is the Euclidean distance to the goal.

11. Modify the $A^*$ planner in Exercise 10 to use a heuristic cost-to-go equal to ten times the distance to the goal node. Compare the running time to the original $A^*$ when they are run on the same graphs. (You may need to use large graphs to notice any effect.) Are the solutions found with the new heuristic that minimizes the use of square roots.
optimal?

12. Modify the $A^*$ algorithm from Exercise 10 to use Dijkstra’s algorithm instead. Comment on the relative running time between $A^*$ and Dijkstra’s algorithm when each is run on the same graphs.

13. Write a program that accepts the vertices of polygonal obstacles from a user, as well as the specification of a 2R robot arm, rooted at $(x, y) = (0, 0)$, with link lengths $L_1$ and $L_2$. Each link is simply a line segment. Generate the C-space obstacles for the robot by sampling the two joint angles at $k$-degree intervals (e.g., $k = 5$) and checking for intersection between the line segments and the polygon. Plot the obstacles in the workspace, and in the C-space grid use a black square or dot for C-obstacles. (Hint: At the core of this program is a subroutine to see if two line segments intersect. If the segments’ corresponding infinite lines intersect, you can check if this intersection is within the line segments.)

14. Write an $A^*$ grid path planner for the 2R robot with obstacles, and display found paths on the C-space. (See Exercise 13 and Figure 10.10.)

15. Write a program to implement a virtual potential field for a 2R robot in an environment with point obstacles. The two links of the robot are line segments, and the user specifies the goal configuration of the robot, the start configuration of the robot, and the location of the point obstacles in the workspace. Put two control points on each link of the robot and transform the workspace potential forces to configuration space potential forces. In one workspace figure, draw an example environment consisting of a few point obstacles, and the robot at its start and goal configurations. In a second C-space figure, plot the potential function as a contour plot over $(\theta_1, \theta_2)$, and overlay a planned path from a start configuration to a goal configuration. The robot uses the kinematic control law $\dot{q} = F(q)$.

See if you can create a planning problem that results in convergence to a local minimum for some initial arm configurations, but succeeds in finding a path to the goal for other initial arm configurations.
Chapter 11

Robot Control

A robot arm can exhibit a number of different behaviors, depending on the task and its environment. It can act as a source of programmed motions for tasks such as moving an object from one place to another, or tracing a trajectory for a spray paint gun. It can act as a source of forces, as when applying a polishing wheel to a workpiece. In tasks such as writing on a chalkboard, it must control forces in some directions (the force pressing the chalk against the board) and motions in others (motion in the plane of the board). When the purpose of the robot is to act as a haptic display, rendering a virtual environment, we may want it to act like a spring, damper, or mass, yielding in response to forces applied to it.

In each of these cases, it is the job of the robot controller to convert the task specification to forces and torques at the actuators. Control strategies to achieve the behaviors described above are known as motion control, force control, hybrid motion-force control, and impedance control. Which of these behaviors is appropriate depends on both the task and the environment. For example, a force control goal makes sense when the end-effector is in contact with something, but not when it is moving in free space. We also have a fundamental constraint imposed by mechanics, irrespective of the environment: the robot cannot independently control the motion and force in the same direction. If the robot imposes a motion, then dynamics and the environment will determine the force, and if the robot imposes a force, then dynamics and the environment will determine the motion.

Once we have chosen a control goal consistent with the task and environment, we can use feedback control to achieve it. Feedback control uses position, velocity, and force sensors to measure the actual behavior of the robot, compare it to the desired behavior, and modulate the control signals sent to the actuators. Feedback is used in nearly all robot systems.

In this chapter we focus on feedback control for motion control, force control, hybrid motion-force control, and impedance control.
11.1 Control System Overview

A typical control block diagram is shown in Figure 11.1(a). Sensors are typically potentiometers, encoders, or resolvers for joint position/angle sensing; tachometers for joint velocity sensing; strain gauge joint force/torque sensors; and/or multi-axis force-torque sensors at the “wrist” between the end of the arm and the end-effector. The controller samples the sensors and updates its control signals to the actuators at a rate of hundreds to a few thousands of Hz. In most robotic applications, higher control update rates are of limited benefit, given time constants associated with the dynamics of the robot and environment. In our analysis, we ignore the nonzero sampling time and treat controllers as if they are implemented in continuous time.

While tachometers can be used for direct velocity sensing, a common approach is to use a digital filter to numerically difference position signals at successive time steps. A low-pass filter is often used in combination with the differencing filter to reduce high-frequency signal content due to the quantization of the differenced position signals.

As discussed in Chapter 8.9, there are a number of different technologies for creating mechanical power, transforming the speeds and forces, and transmitting to the robot joints. In this chapter, we lump each joint’s amplifier, actuator, and transmission together and treat them as a transformer from low-power control signals to forces and torques. This assumption, along with the assumption of perfect sensors, allows us to simplify the block diagram of Figure 11.1(a) to the one shown in Figure 11.1(b), where the controller produces forces and torques directly. The rest of this chapter deals with the control algorithms that go inside the “controller” box in Figure 11.1(b).

Real robot systems are subject to flexibility and vibrations in the joints and links, backlash at gears and transmissions, actuator saturation limits, and limited resolution of the sensors. These raise significant issues in design and control, but they are beyond the scope of this chapter.

11.2 Motion Control

A motion controller can be defined in joint space or task space. In joint space, the controller is fed a steady stream of joint positions $\theta_d(t)$ and velocities $\dot{\theta}_d(t)$ to track, and the goal is to drive the robot to track this trajectory. When the trajectory is expressed in task space, the controller is fed a steady stream of end-effector configurations $X_d(t)$ and velocities $V_d(t)$. If the end-effector configuration is represented by a minimum set of coordinates, then $X_d(t) \in \mathbb{R}^m$ and $V_d(t) = \dot{X}_d(t)$. If the end-effector configuration is represented as an element of $SE(3)$, then $X_d(t) \in SE(3)$ and $V_d(t) \in \mathbb{R}^6$, where $V_d = X_d^{-1} \dot{X}_d$ is the twist expressed in the end-effector frame and $V_d = \dot{X}_d X_d^{-1}$ is the twist expressed in the space frame.
11.2. Motion Control

11.2.1 Motion Control of a Multi-Joint Robot with Velocity Input

As discussed in Chapter 8, we typically assume direct control of the forces or torques at robot joints, and the robot’s dynamics transform those controls to joint accelerations. In relatively rare cases, however, we can assume direct control of the joint velocities. An example is when the actuators are stepper motors. In this case, the velocity of a joint is determined directly by the frequency of the pulse train sent to the stepper.\(^1\) Another example is when the amplifier for an electric motor is placed in velocity control mode—the amplifier attempts to achieve the joint velocity requested by the user, rather than a joint force or

\(^{1}\)Assuming that the torque requirements are low enough that the stepper motor can keep up with the pulse train.
torque.

Under the assumption of perfect velocity control, motion control of a multi-
joint robot would be trivial: at all times, simply command the velocity
\[ \dot{\theta}_{\text{com}}(t) = \dot{\theta}_d(t), \]
where \( \dot{\theta}_d(t) \) comes from the desired trajectory. This is called a **feedforward** or
open-loop controller, since no feedback (sensor data) is needed to implement
it.

In practice, however, position errors will accumulate over time. An alter-
native strategy is to continually measure the actual position of each joint and
implement a simple feedback controller,
\[ \dot{\theta}_{\text{com}}(t) = K_p(\theta_d(t) - \theta(t)), \]
where \( K_p \) is a diagonal matrix of positive gains. This controller is called a
"proportional controller" because it creates a corrective velocity proportional
to the current position error \( \theta_e(t) = \theta_d(t) - \theta(t) \). This prevents the error from
growing large over time, which could happen with a purely open-loop controller.
A drawback, however, is that the robot only moves when there is error.

We can use feedforward control, which commands motion even when there
is no error, in combination with feedback control to limit the accumulation of
error:
\[ \dot{\theta}_{\text{com}}(t) = \dot{\theta}_d(t) + K_p(\theta_d(t) - \theta(t)), \]  
(11.1)
This feedforward-feedback controller is the preferred velocity control law.

We can write an analogous control law in task space. Let \( X(t) \in SE(3) \)
be the configuration of the end-effector as a function of time, and \( V(t) \) be the
end-effector twist expressed in the end-effector frame, i.e., \( V = X^{-1} \dot{X} \). The
task space version of the control law (11.1) is
\[ V_{\text{com}}(t) = V_d(t) + K_p X_e(t), \]  
(11.2)
where \( X_e(t) \) is not simply \( X_d(t) - X(t) \), since it does not make sense to subtract
elements of \( SE(3) \). Instead, as we saw in Chapter 6.2, \( X_e \) should refer to the
twist which, if followed for unit time, takes \( X \) to \( X_d \). The matrix representation
of this twist, expressed in the end-effector frame, is \( [X_e] = \log(X^{-1} X_d) \).\(^2\)

If \( X \) represents the end-effector configuration using a minimum set of coor-
dinates, then the control law (11.2) can be used with \( V = \dot{X} \) and \( X_e = X_d - X \).

The commanded joint velocities \( \dot{\theta}_{\text{com}} \) corresponding to \( V_{\text{com}} \) in Equation (11.2)
can be calculated using the inverse velocity kinematics from Chapter 6.3,
\[ \dot{\theta}_{\text{com}} = J^\dagger(\theta) V_{\text{com}}, \]
where \( J^\dagger(\theta) \) is the pseudoinverse of the appropriate Jacobian.

\(^2\)Writing \( X \) as \( X_{sb} \) (the end-effector frame \( \{b\} \) in a space frame \( \{s\} \)), and \( X_d \) as \( X_{sd} \), we
see that \( X_{sb}^{-1} X_{sd} = X_{bs} X_{sd} = X_{bsd} \), the configuration \( X_d \) expressed in the end-effector frame.
Therefore \( \log(X^{-1} X_d) \) is the matrix representation of the twist that takes \( X \) to \( X_d \) in unit
time, expressed in the end-effector frame.
11.2. Motion Control

11.2.2 Motion Control of a Single Joint with Torque or Force Input

Stepper-motor-controlled robots are generally limited to applications with low or predictable force/torque requirements. Also, robot control engineers do not rely on the velocity-control modes of off-the-shelf amplifiers for electric motors, because these velocity-control algorithms do not make use of the dynamic model of the robot. Instead, robot control engineers use amplifiers in torque-control mode: the input to the amplifier is the desired torque (or force). This allows the robot control engineer to use a dynamic model of the robot in the design of the control law. The ideas are well illustrated by a robot with a single joint, so we begin there, then generalize to a multi-joint robot.

Consider a single motor attached to a single link, as shown in Figure 11.2. Let \( \tau \) be the motor’s torque and \( \theta \) be the angle of the link. The dynamics can be written as

\[
\tau = M \ddot{\theta} + mgr \cos \theta, \tag{11.3}
\]

where \( M \) is the scalar inertia of the link about the axis of rotation, \( m \) is the mass of the link, \( r \) is the distance from the axis to the center of mass of the link, and \( g \geq 0 \) is gravitational acceleration.

According to the model (11.3), there is no dissipation: if the link is made to move and \( \tau \) is then set to zero, the link would move forever. This is unrealistic, of course; there is friction at the various bearings, gears, and transmissions. Friction modeling is an active research area, but a simple model of rotational friction is viscous friction,

\[
\tau_{\text{fric}} = b \dot{\theta}, \tag{11.4}
\]

where \( b > 0 \). Adding the friction torque, our final model is

\[
\tau = M \ddot{\theta} + mgr \cos \theta + b \dot{\theta}, \tag{11.5}
\]

which we may write more compactly as

\[
\tau = M \ddot{\theta} + h(\theta, \dot{\theta}), \tag{11.6}
\]

where \( h \) contains all terms that depend only on the state, not the acceleration.
For concreteness in the following simulations, we set $M = 0.5$ kg m$^2$, $m = 1$ kg, $r = 0.1$ m, and $b = 0.1$ Nm/rad. In some examples, the link moves in a horizontal plane, so $g = 0$. In other examples, the link moves in a vertical plane, so $g = 9.81$ m/s$^2$.

11.2.2.1 Feedback Control: PID Control

The most common feedback control algorithm is linear PID (proportional-integral-derivative) control. Defining the error between the desired angle $\theta_d$ and actual angle $\theta$ as

$$\theta_e = \theta_d - \theta,$$

the PID controller is simply

$$\tau = K_p \theta_e + K_i \int \theta_e(t) dt + K_d \dot{\theta}_e,$$  \hspace{1cm} (11.8)

where the control gains $K_p$, $K_i$, and $K_d$ are nonnegative. The proportional gain $K_p$ acts as a virtual spring that tries to reduce the position error $\theta_e = \theta_d - \theta$, and the derivative gain $K_d$ acts as a virtual damper that tries to reduce the velocity error $\dot{\theta}_e = \dot{\theta}_d - \dot{\theta}$. The integral gain, as we will see later, can be used to eliminate steady-state errors when the joint is at rest. See the block diagram in Figure 11.3.

For now let’s consider the case where $K_i = 0$. This is known as PD control. (We can similarly define PI, P, I, and D control by setting other gains to zero. PD and PI control are the most common variants of PID control.) Let’s also assume the robot moves in a horizontal plane, so $g = 0$. Plugging the control law (11.8) into the dynamics (11.5), we get

$$M \ddot{\theta} + b \dot{\theta} = K_p(\theta_d - \theta) + K_d(\dot{\theta}_d - \dot{\theta}).$$  \hspace{1cm} (11.9)
If the goal state is rest at a constant $\theta_d$, then $\dot{\theta}_d = \ddot{\theta}_d = 0$. This is called setpoint control. In setpoint control, $\theta_e = \theta_d - \theta$, $\dot{\theta}_e = -\dot{\theta}$, and $\ddot{\theta}_e = -\ddot{\theta}$, Equation (11.9) can be rewritten as the linear mass-spring-damper error dynamics

$$M \ddot{\theta}_e + (b + K_d) \dot{\theta}_e + K_p \theta_e = 0. \quad (11.10)$$

**Stability** Error dynamics, such as Equation (11.10), are an important concept in the study of control systems. A desirable property is that the error dynamics be stable, i.e., initial errors tend to zero with time. A linear homogeneous ordinary differential equation of the form

$$a_n \ddot{\theta}_e^{(n)} + a_{n-1} \dot{\theta}_e^{(n-1)} + \ldots + a_2 \dot{\theta}_e + a_1 \theta_e + a_0 \theta_e = 0$$

is stable if and only if all of the complex roots $s_1, \ldots, s_n$ of its characteristic equation

$$a_n s^n + a_{n-1} s^{n-1} + \ldots + a_2 s^2 + a_1 s + a_0 = 0$$

have real components less than zero, i.e., $\text{Re}(s_i) < 0$ for all $i = 1 \ldots n$. A necessary condition for stability, regardless of the order $n$ of the dynamics, is that $a_i > 0$ for all $i$. This condition is also sufficient for second-order dynamics such as Equation (11.10). For third-order dynamics, stability is assured if $a_i > 0$ for all $i$ and $a_2 a_1 > a_3 a_0$.

**PD Control and Second-Order Error Dynamics** To study the second-order error dynamics (11.10) more formally, we assume stability and rewrite in the standard second-order form

$$\ddot{\theta}_e + \frac{b + K_d}{M} \dot{\theta}_e + \frac{K_p}{M} \theta_e = 0 \quad \rightarrow \quad \ddot{\theta}_e + 2\zeta \omega_n \dot{\theta}_e + \omega_n^2 \theta_e = 0, \quad (11.11)$$

where the damping ratio $\zeta$ and the natural frequency $\omega_n$ are

$$\zeta = \frac{b + K_d}{2\sqrt{K_p M}}, \quad \omega_n = \sqrt{\frac{K_p}{M}}.$$

The characteristic equation of Equation (11.11) is

$$s^2 + 2\zeta \omega_n s + \omega_n^2 = 0, \quad (11.12)$$

with complex roots

$$s_{1,2} = -\zeta \omega_n \pm \omega_n \sqrt{\zeta^2 - 1}.$$

There are three types of solutions to the differential equation (11.11), depending on whether the roots $s_{1,2}$ are real and unequal ($\zeta > 1$), real and equal ($\zeta = 1$), or complex conjugates ($\zeta < 1$):

- **Overdamped**: $\zeta > 1$. The roots $s_{1,2}$ are real and distinct, and the solution is
  $$\theta_e(t) = c_1 e^{s_1 t} + c_2 e^{s_2 t},$$
where \( c_1 \) and \( c_2 \) depend on the initial conditions. The response is the sum of two decaying exponentials, with time constants \( \tau_{1,2} = -1/s_{1,2} \), where the time constant is the time it takes the exponential to decay to 37% of its original value. The “slower” time constant in the solution is given by the less negative root, \( s_1 = -\zeta \omega_n + \omega_n \sqrt{\zeta^2 - 1} \).

- **Critically damped**: \( \zeta = 1 \). The roots \( s_{1,2} = -\zeta \omega_n \) are equal and real, and the solution is
  \[
  \theta_e(t) = (c_1 + c_2 t)e^{-\zeta \omega_n t},
  \]
  i.e., a decaying exponential multiplied by a linear function of time. The time constant of the decaying exponential is \( \tau = 1/(\zeta \omega_n) \).

- **Underdamped**: \( \zeta < 1 \). The roots \( s_{1,2} \) are complex conjugates at \( s_{1,2} = -\zeta \omega_n \pm j \omega_d \), where \( \omega_d = \omega_n \sqrt{1-\zeta^2} \) is the damped natural frequency. The solution is
  \[
  \theta_e(t) = (c_1 \cos(\omega_d t) + c_2 \sin(\omega_d t))e^{-\zeta \omega_n t},
  \]
  i.e., a decaying exponential (time constant \( \tau = 1/(\zeta \omega_n) \)) multiplied by a sinusoid.

To see how to apply these solutions, imagine that the link is originally at rest at \( \theta = 0 \). At time \( t = 0 \), the desired position is suddenly changed from \( \theta_d = 0 \) to \( \theta_d = 1 \). This is called a step input and the resulting motion of the system \( \theta(t) \) is called the step response. Our interest is in the error response \( \theta_e(t) \). We can solve for \( c_{1,2} \) in the specific solution by solving \( \theta_e(0) = 1 \) (the error immediately becomes 1) and \( \dot{\theta}_e(0) = 0 \) (both \( \dot{\theta}_d(0) \) and \( \dot{\theta}(0) \) are zero).

The error response can be described by a transient response and a steady-state response (Figure 11.4). The steady-state response is characterized by the steady-state error \( e_{ss} \), which is the asymptotic error \( \theta_e(t) \) as \( t \to \infty \). For the link in zero gravity with a stable PD controller, \( e_{ss} = 0 \). The transient response is characterized by the overshoot and (2%) settling time. The 2% settling time is the first time \( T \) such that \( |\theta_e(t) - e_{ss}| \leq 0.02(1 - e_{ss}) \) for all \( t \geq T \), and is approximately equal to \( 4 \tau \), where \( \tau \) is the slowest time constant in the solution. Overshoot occurs if the error response initially overshoots the final steady-state error, and in this case overshoot is defined as

\[
\text{overshoot} = \left| \frac{\theta_{e,\min} - e_{ss}}{1 - e_{ss}} \right| \times 100\% \]

where \( \theta_{e,\min} \) is the least positive value achieved by the error. The overshoot can be calculated to be

\[
\text{overshoot} = e^{-\pi \zeta / \sqrt{1-\zeta^2}} \times 100\%, \quad 0 \leq \zeta < 1.
\]

A good transient response is characterized by a low settling time and little or no overshoot.
11.2. Motion Control

Figure 11.4: The error response to a step input for an underdamped second-order system, showing steady-state error $e_{ss}$, overshoot, and 2% settling time.

Figure 11.5 shows the relationship of the location of the roots of Equation (11.12) to the transient response. For a fixed $K_d$ and a small $K_p$, we have $\zeta > 1$, the system is overdamped, and the response is sluggish due to the “slow” root. As $K_p$ is increased, the damping ratio decreases. The system is critically damped ($\zeta = 1$) at $K_p = (b + K_d)^2/(4M)$, and the two roots are coincident on the negative real axis. This situation corresponds to a relatively fast response and no overshoot. As $K_p$ continues to increase, $\zeta$ drops below 1, the roots move off the negative real axis, and we begin to see overshoot and oscillation in the response. The settling time is unaffected as $K_p$ is increased beyond critical damping, as $\zeta \omega_n$ is unchanged. In general, critical damping is desirable.

Practical Bounds on Feedback Gains According to our simple model, we could increase $K_p$ and $K_d$ without bound to make the real components of the roots more and more negative, achieving arbitrarily fast response. In practice, however, large gains lead to actuator saturation, rapid torque changes (chattering), vibrations of the structure due to unmodeled flexibility in the joints and links, and possibly even instability due to the finite servo rate frequency. Thus there are practical limits on the set of useful gains.

PID Control and Third-Order Error Dynamics Now consider the case of setpoint control where the link moves in a vertical plane, i.e., $g > 0$. With the PD control law above, the system can now be written

$$M \ddot{\theta}_e + (b + K_d) \dot{\theta}_e + K_p \theta_e = m gr \cos \theta.$$  \hspace{1cm} (11.13)
This implies that the system comes to rest at a configuration $\theta$ satisfying $K_p\theta_e = mg_r \cos \theta$, i.e., the final error $\theta_e$ is not zero when $\theta_d \neq \pm \pi/2$. This is because the robot must provide a nonzero torque to hold the link at rest at $\theta \neq \pm \pi/2$, but the PD control law only creates a nonzero torque at rest if $\theta_e \neq 0$. We can make this steady-state error small by increasing the gain $K_p$, but as discussed above, there are practical limits.

To eliminate the steady-state error, we return to the PID controller by setting $K_i > 0$. This allows a nonzero steady-state torque even with zero position error; only the integrated error must be nonzero. Figure 11.6 demonstrates the addition of the integral term to the controller.

To see how this works, write the setpoint error dynamics

$$M \ddot \theta_e + (b + K_d) \dot \theta_e + K_p \theta_e + K_i \int \theta_e(t) dt = \tau_{\text{dist}}, \quad (11.14)$$

where $\tau_{\text{dist}}$ is a disturbance torque substituted for the gravity term $mg_r \cos \theta$. Taking derivatives of both sides, we get the third-order error dynamics

$$M \theta_e^{(3)} + (b + K_d) \ddot \theta_e + K_p \dot \theta_e + K_i \theta_e = \tau_{\text{dist}}. \quad (11.15)$$

If $\tau_{\text{dist}}$ is constant, then the right-hand side of Equation (11.15) is zero. If the PID controller is stable, then (11.15) shows that $\theta_e$ converges to zero. (While the disturbance torque due to gravity is not constant as the link rotates, it approaches a constant as $\dot \theta$ approaches zero, and therefore similar reasoning holds close to the equilibrium.)

Integral control is useful to eliminate steady-state error in setpoint control, but it may adversely affect the transient response. This is because integral
11.2. Motion Control

Figure 11.6: (Left) The tracking errors for a PD controller with $K_d = 2$ Nms/rad, $K_p = 2.205$ Nm/rad for critical damping, and a PID controller with the same PD gains and $K_i = 1$ Nm/(rad s). The arm starts at $\theta(0) = \pi / 2$, $\dot{\theta}(0) = 0$, with a goal state $\theta_d = 0$, $\dot{\theta}_d = 0$. (Middle) The individual contributions of the terms in the PD and PID control laws. (Right) The initial and final configurations.

control essentially responds to delayed information—it takes time for the system to respond to error as it integrates. It is well known in control theory that delayed feedback can cause instability. To see this, consider the characteristic equation of Equation (11.15) when $\tau_{\text{dist}}$ is constant:

$$Ms^3 + (b + K_d)s^2 + K_p s + K_i = 0.$$  (11.16)

For all roots to have negative real part, we require $b + K_d > 0$ and $K_p > 0$ as before, but there is also an upper bound on the new gain $K_i$ (Figure 11.7):

$$0 \leq K_i < \frac{(b + K_d)K_p}{M}.$$  

Thus a reasonable design strategy is to choose $K_p$ and $K_d$ for a good transient response, then choose $K_i$ small so as not to adversely affect stability. In the example of Figure 11.6, the relatively large $K_i$ worsens the transient response, giving significant overshoot. In practice, $K_i = 0$ for many robot controllers.

Pseudocode for the PID control algorithm is given in Figure 11.8.

While our analysis has focused on setpoint control, the PID controller applies perfectly well to trajectory following, where $\dot{\theta}_d(t) \neq 0$. Integral control will not eliminate tracking error along arbitrary trajectories, however.

11.2.2.2 Feedforward Control

Another strategy for trajectory following is to use a model of the robot’s dynamics to proactively generate torques, instead of waiting for errors. Let the controller’s model of the dynamics be

$$\tau = \tilde{M}(\theta)\ddot{\theta} + \tilde{h}(\theta, \dot{\theta}),$$  (11.17)
where the model is perfect if $\tilde{M}(\theta) = M(\theta)$ and $\tilde{h}(\theta, \dot{\theta}) = h(\theta, \dot{\theta})$. Note that the inertia model $\tilde{M}(\theta)$ is written as a function of the configuration $\theta$. While the inertia of our simple one-joint robot is not a function of configuration, writing this way allows us to re-use Equation (11.17) for multi-joint systems in Section 11.2.3.

Given $\theta_d$, $\dot{\theta}_d$, and $\ddot{\theta}_d$ from the trajectory generator, the commanded torque is calculated as

$$\tau = \ddot{\theta}_d \ddot{\theta}_d + \tilde{h}(\theta_d, \dot{\theta}_d).$$

(11.18)

If the model of the robot dynamics is exact, and there are no initial state errors, then the robot exactly follows the desired trajectory. This is called feedforward control, as no feedback is used.

A pseudocode implementation of feedforward control is given in Figure 11.9. Figure 11.10 shows two examples of trajectory following for the link in gravity. Here, the controller’s dynamic model is correct except that it has $\tilde{r} = 0.08$ m, when actually $r = 0.1$ m. In Task 1, the error stays small, as unmodeled gravity effects provide a spring-like force to $\theta = -\pi/2$, accelerating the robot at the beginning and decelerating it at the end. In Task 2, unmodeled gravity effects act against the desired motion, resulting in a larger tracking error. Because there
11.2. Motion Control

\[
\begin{align*}
\text{time} &= 0 \quad // \ dt = \text{servo cycle time} \\
eint &= 0 \quad \quad // \ \text{error integral} \\
qprev &= \text{senseAngle} \quad // \ \text{initial joint angle } q \\
\text{loop} &:\\\n[qd,qdotd] &= \text{trajectory}(\text{time}) \quad // \ \text{from trajectory generator} \\
q &= \text{senseAngle} \quad // \ \text{sense actual joint angle} \\
qdot &= (q - qprev)/dt \quad // \ \text{simple velocity calculation} \\
qprev &= q \\
\text{e} &= qd - q \\
edot &= qdotd - qdot \\
eint &= eint + e*dt \\
\tau &= Kp * e + Kd * edot + Ki * eint \\
\text{commandTorque}(\tau) \\
\text{time} &= \text{time} + dt \\
\text{end loop}
\end{align*}
\]

Figure 11.8: Pseudocode for PID control.

\[
\begin{align*}
\text{time} &= 0 \quad // \ dt = \text{servo cycle time} \\
\text{loop} &:\\\n[qd,qdotd,qdotdotd] &= \text{trajectory}(\text{time}) \quad // \ \text{trajectory generator} \\
\tau &= M\tilde{\text{t}}(qd)*qdotdotd + h\tilde{\text{t}}(qd,qdotd) \quad // \ \text{calculate dynamics} \\
\text{commandTorque}(\tau) \\
\text{time} &= \text{time} + dt \\
\text{end loop}
\end{align*}
\]

Figure 11.9: Pseudocode for feedforward control.

are always modeling errors, feedforward control is always used in conjunction with feedback, as discussed next.

11.2.2.3 Feedforward Plus Feedback Linearization

All practical controllers use feedback, as no model of robot and environment dynamics will be perfect. Nonetheless, a good model can be used to improve performance and simplify analysis.

Let’s combine PID control with a model of the robot dynamics \{\tilde{M}, \tilde{h}\} to
achieve the error dynamics

\[ \ddot{\theta}_e + K_d \dot{\theta}_e + K_p \theta_e + K_i \int \theta_e(t) dt = 0 \quad (11.19) \]

along arbitrary trajectories, not just to a setpoint. The error dynamics (11.19) and proper choice of PID gains ensure exponential decay of trajectory error.

Since \( \ddot{\theta}_e = \ddot{\theta}_d - \ddot{\theta} \), to achieve the error dynamics (11.19), we choose the robot’s commanded acceleration to be

\[
\ddot{\theta}_{\text{com}} = \ddot{\theta}_d - \ddot{\theta}_e \quad \text{then plug in Equation (11.19) to get}
\]

\[
= \ddot{\theta}_d + K_d \dot{\theta}_e + K_p \theta_e + K_i \int \theta_e(t) dt.
\]

(11.20)

Plugging \( \ddot{\theta}_{\text{com}} \) into a model of the robot dynamics \( \{ \ddot{M}, \ddot{h} \} \), we get the feedforward plus feedback linearizing controller, also called the computed torque controller:

\[
\tau = \ddot{M}(\theta) \left( \ddot{\theta}_d + K_p \dot{\theta}_e + K_i \int \theta_e(t) dt + K_d \dot{\theta}_e \right) + \ddot{h}(\theta, \dot{\theta}).
\]

(11.21)
11.2. Motion Control

Figure 11.11: Computed torque control. The feedforward acceleration $\ddot{\theta}_d$ is added to the acceleration $\ddot{\theta}_{fb}$ computed by the PID feedback controller to create the commanded acceleration $\ddot{\theta}_{com}$.

Figure 11.12: Performance of feedforward only (ff), feedback only (fb), and computed torque control (ff+fb). PID gains are taken from Figure 11.6, and the feedforward modeling error is taken from Figure 11.10. The desired motion is Task 2 from Figure 11.10. The middle plot shows the tracking performance of the three controllers. Also plotted is $\int \tau^2(t) dt$, a standard measure of control effort, for each of the three controllers. These plots show typical behavior: the computed torque controller yields better tracking than either feedforward or feedback alone, with less control effort than feedback alone.

This controller includes a feedforward component, due to the use of the planned acceleration $\ddot{\theta}_d$, and is called feedback linearizing because feedback of $\dot{\theta}$ and $\ddot{\theta}$ is used to generate linear error dynamics. The $h(\theta, \dot{\theta})$ term cancels dynamics that depend nonlinearly on the state, and the inertia model $\tilde{M}(\theta)$ converts desired joint accelerations into joint torques, realizing the simple linear error dynamics (11.19).

A block diagram of the computed torque controller is shown in Figure 11.11. The gains $K_p$, $K_i$, and $K_d$ are chosen to place the roots of the characteristic equation as desired to achieve good transient response. Under the assumption of a perfect dynamic model, we would choose $K_i = 0$.

Figure 11.12 shows typical behavior of feedback linearizing control relative to feedforward and feedback only. Pseudocode is given in Figure 11.13.
time = 0 // dt = cycle time
eint = 0 // error integral
qprev = senseAngle // initial joint angle q
loop
  [qd,qdotd,qdotdotd] = trajectory(time) // from trajectory generator
  q = senseAngle // sense actual joint angle
  qdot = (q - qprev)/dt // simple velocity calculation
  qprev = q
  e = qd - q
  edot = qdotd - qdot
  eint = eint + e*dt
  tau = Mtilde(q)*(qdotdotd + Kp*e + Kd*edot + Ki*eint) + htilde(q,qdot)
  commandTorque(tau)
  time = time + dt
end loop

Figure 11.13: Pseudocode for the computed torque controller.

11.2.3 Motion Control of a Multi-joint Robot with Torque or Force Input

The methods applied above for a single-joint robot carry over directly to $n$-joint robots. The difference is that the dynamics (11.6) now take the more general vector-valued form

$$\tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}),$$

where the $n \times n$ positive-definite mass matrix $M$ is now a function of the configuration $\theta$. We will sometimes find it convenient to explicitly write the components of the term $h(\theta, \dot{\theta})$,

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + g(\theta) + b(\dot{\theta}),$$  \hspace{1cm} (11.22)

where $C(\theta, \dot{\theta})\dot{\theta}$ are Coriolis and centripetal terms, $g(\theta)$ are potential (e.g., gravity) terms, and $b(\dot{\theta})$ are friction terms. In general, the dynamics (11.22) are coupled—the acceleration of a joint is a function of the positions, velocities, and torques at other joints.

We distinguish between two types of control of multi-joint robots: decentralized control, where each joint is controlled separately with no sharing of information between joints, and centralized control, where full state information for each of the $n$ joints is available to calculate the controls for each joint.
11.2. Motion Control

11.2.3.1 Decentralized Multi-Joint Control

The simplest method for controlling a multi-joint robot is to apply an independent controller at each joint, such as the single-joint controllers discussed in Section 11.2.2. Decentralized control is appropriate when the dynamics are decoupled, or at least approximately decoupled. The dynamics are decoupled when the acceleration of each joint depends only on the torque, position, and velocity of that joint. This requires that the mass matrix be diagonal, as in Cartesian or gantry robots, where the first three axes are prismatic and orthogonal along the x-y-z axes. This kind of robot is equivalent to three single-joint systems.

Approximate decoupling is also achieved in highly geared robots in the absence of gravity. The mass matrix \( M(\theta) \) is nearly diagonal, as it is dominated by the apparent inertias of the motors themselves (see Chapter 8.9.2). Significant friction at the individual joints also contributes to the decoupling of the dynamics.

11.2.3.2 Centralized Multi-Joint Control

When gravity forces and torques are significant and coupled, or when the mass matrix \( M(\theta) \) is not well approximated by a diagonal matrix, decentralized control may not yield acceptable performance. In this case, the computed torque control law (11.21) of Figure 11.11 can be generalized. The configurations \( \theta \) and \( \theta_d \) and the error \( \theta_e = \theta_d - \theta \) are now \( n \)-vectors, and the positive scalar gains become positive-definite matrices \( K_p, K_i, K_d \):

\[
\tau = \tilde{M}(\theta) \left( \dot{\theta}_d + K_p \theta_e + K_i \int \theta_e(t) dt + K_d \dot{\theta}_e \right) + \tilde{h}(\theta, \dot{\theta}).
\] (11.23)

Typically we choose the gain matrices as \( k_p I, k_i I, k_d I \), where \( I \) is the \( n \times n \) identity matrix and \( k_p, k_i, k_d \) are nonnegative scalars. Commonly \( K_i \) is chosen to be zero. In the case of an exact dynamics model \( \tilde{M} \) and \( \tilde{h} \), the dynamics of each joint reduces to the linear dynamics (11.19). The block diagram and pseudocode for this control algorithm are found in Figures 11.11 and 11.13, respectively.

Implementing the control law (11.23) requires calculating potentially complex dynamics. We may not have a good model of these dynamics, or the equations may be too computationally expensive to calculate at servo rate. In this case, if the desired velocities and accelerations are small, an approximation to (11.23) can be obtained using only PID control and gravity compensation:

\[
\tau = K_p \theta_e + K_i \int \theta_e(t) dt + K_d \dot{\theta}_e + \tilde{g}(\theta).
\] (11.24)
With zero friction, perfect gravity compensation, and PD setpoint control \( (K_i = 0 \text{ and } \dot{\theta}_d = \ddot{\theta}_d = 0) \), the controlled dynamics can be written

\[
M(\theta) \ddot{\theta} + C(\theta, \dot{\theta}) \dot{\theta} = K_p \theta_e - K_d \dot{\theta},
\]

(11.25)

where the Coriolis and centripetal terms are written \( C(\theta, \dot{\theta}) \dot{\theta} \). We can now define a virtual “error energy,” which is the sum of an “error potential energy” stored in the virtual spring \( K_p \) and an “error kinetic energy”:

\[
V(\theta_e, \dot{\theta}_e) = \frac{1}{2} \theta_e^T K_p \theta_e + \frac{1}{2} \dot{\theta}_e^T M(\theta) \dot{\theta}_e.
\]

(11.26)

Since \( \dot{\theta}_d = 0 \), this reduces to

\[
V(\theta_e, \dot{\theta}_e) = \frac{1}{2} \theta_e^T K_p \theta_e + \frac{1}{2} \dot{\theta}_e^T M(\theta) \dot{\theta}_e.
\]

(11.27)

Taking the time derivative and plugging in (11.25), we get

\[
\dot{V} = -\dot{\theta}_e^T K_p \theta_e + \dot{\theta}_e^T M(\theta) \dot{\theta} + \frac{1}{2} \dot{\theta}_e^T \dot{M}(\theta) \dot{\theta}
\]

\[
= -\dot{\theta}_e^T K_p \theta_e + \dot{\theta}_e^T \left( K_p \theta_e - K_d \dot{\theta} - C(\theta, \dot{\theta}) \dot{\theta} \right) + \frac{1}{2} \dot{\theta}_e^T \dot{M}(\theta) \dot{\theta}.
\]

(11.28)

Rearranging, and using the fact that \( \dot{M} - 2C \) is skew-symmetric (Proposition 8.1.2), we get

\[
\dot{V} = -\dot{\theta}_e^T K_p \theta_e + \dot{\theta}_e^T \left( K_p \theta_e - K_d \dot{\theta} - C(\theta, \dot{\theta}) \dot{\theta} \right) + \frac{1}{2} \dot{\theta}_e^T \left( \dot{M}(\theta) - 2C(\theta, \dot{\theta}) \right) \dot{\theta}
\]

\[
= -\dot{\theta}_e^T K_d \dot{\theta} \leq 0.
\]

(11.29)

This shows that the error energy is decreasing when \( \dot{\theta} \neq 0 \). If \( \dot{\theta} = 0 \) and \( \theta \neq \theta_d \), the virtual spring ensures that \( \dot{\theta} \neq 0 \), so \( \dot{\theta}_e \) will again become nonzero and more energy will be dissipated. Thus by the Krasovskii-LaSalle invariance principle (Exercise 12), the total error energy decreases monotonically and the robot converges to rest at \( \theta_d (\theta_e = 0) \) from any initial state.

### 11.2.4 Task Space Motion Control

In Section 11.2.3, we focused on motion control in joint space. This is convenient because joint limits are easily expressed in this space, and the robot should be able to execute any joint-space path respecting these limits. Trajectories are naturally described by the joint variables, and there are no issues of singularities or redundancy.

On the other hand, since the robot interacts with the external environment and objects in it, it may be more convenient to express the motion as a trajectory of the end-effector in task space. Let the end-effector trajectory be specified by
(X(t), V(t)), where X ∈ SE(3) or X ∈ ℜ^n, for example, with V ∈ ℜ^n the velocity. Provided the corresponding trajectory in joint space is feasible, we now have two options for control: (1) convert to a joint-space trajectory and proceed with control as in Section 11.2.3 or (2) express the robot dynamics and control law in the task space.

The first option is to convert the trajectory to joint space. The forward kinematics are \( X = f(\theta) \) and \( V = J(\theta) \dot{\theta} \), where \( J(\theta) \) is the appropriate Jacobian based on the chosen velocity representation \( V \). Then the joint space trajectory is obtained from the task space trajectory using inverse kinematics (Chapter 6):

\[
\begin{align*}
\theta(t) &= f^{-1}(X(t)) \\
\dot{\theta}(t) &= J^T(\theta(t))V(t) \\
\ddot{\theta}(t) &= J^T(\theta(t)) \left( \ddot{V}(t) - J(\theta(t)) \dot{\theta}(t) \right).
\end{align*}
\]

(11.30) (11.31) (11.32)

A drawback of this approach is that we must calculate the inverse kinematics, which may require significant computing power. The second option is to express the robot’s dynamics in task-space coordinates, as discussed in Chapter 8.6. Recall the task-space dynamics

\[
F = \Lambda(\theta) \dot{V} + \eta(\theta, V).
\]

The joint forces and torques \( \tau \) are related to the forces \( F \) expressed in the end-effector frame by \( \tau = J^T(\theta)F \).

We can now write a control law in task coordinates inspired by the computed torque control law in joint coordinates (11.23),

\[
\tau = J^T(\theta) \left( \dot{\Lambda}(\theta) \left( \dot{V}_d + K_p X_c + K_i \int X_c(t) dt + K_d V_c \right) + \dot{\eta}(\theta, V) \right),
\]

(11.33)

where \( \dot{V}_d \) is the desired acceleration and \( \{\dot{\Lambda}, \dot{\eta}\} \) represents the controller’s dynamics model.

The task-space control law (11.33) makes use of the configuration error \( X_c \) and velocity error \( V_c \). When \( X \) is expressed in a minimal set of coordinates \( (X \in \mathbb{R}^m) \) and \( V = \dot{X} \), a natural choice is \( X_c = X_d - X \), \( V_c = \dot{X}_d - \dot{X} \). When \( X = (R, p) \in SE(3) \), however, there are a number of possible choices, including the following:

- \( V = V_b \) and \( J(\theta) = J_b(\theta) \): twists are represented in the end-effector frame \( \{b\} \). A natural choice would use \( X_c \) such that \( [X_c] = \log(X^{-1}X_d) \) and \( V_c = [\operatorname{Ad}_{X^{-1}X_d}]V_d - V \). The twist \( X_c \in \mathbb{R}^6 \) is the velocity, expressed in the end-effector frame, which, if followed for unit time, would move the current configuration \( X \) to the desired configuration \( X_d \). The transform \( [\operatorname{Ad}_{X^{-1}X_d}] \) expresses the desired twist \( V_d \), which is expressed in the frame \( X_d \), as a twist in the end-effector frame at \( X \), in which the actual velocity \( V \) is represented, so the two can be differenced.
• \( \mathcal{V} \) and \( J(\theta) \) chosen so that \( \mathcal{V} = (\omega, v) \), where \( \omega \) is the angular velocity of the end-effector expressed in the \( \{s\} \) frame and \( v = \dot{p} \). The matrix representation of the angular velocity \( \omega_b \) that takes \( R \) to \( R_d \) in unit time, expressed in the end-effector frame \( \{b\} \), is \( [\omega_b] = \log(R^T R_d) \), so a natural choice for error coordinates would be

\[
X_e = \begin{bmatrix} R\omega_b \\ p_d - p \end{bmatrix} \quad \text{and} \quad \mathcal{V}_e = \mathcal{V}_d - \mathcal{V},
\]

where \( R\omega_b \) is the angular velocity that takes \( R \) to \( R_d \) in unit time, expressed in \( \{s\} \).

These choices lead to different behaviors of the robot. In particular, the second choice decouples the rotational and linear corrective terms.

### 11.3 Force Control

When the goal is not to create motions at the end-effector, but to apply forces and torques to the environment, the task requires force control. Pure force control is only possible if the environment provides resistance forces in every direction (e.g., when the end-effector is embedded in concrete or attached to a spring providing resistance in every motion direction). Pure force control is a bit of an abstraction, as robots are usually able to move freely in at least some direction. It is a useful abstraction, however. It also leads to hybrid motion-force control in Section 11.4.

In ideal force control, the force applied by the end-effector is unaffected by disturbance motions applied to the end-effector. This is dual to the case of ideal motion control, where the motion is unaffected by disturbance forces. Force control is dual to motion control in the sense that forces are dual to velocities, with their product being power, an intrinsic, coordinate-free concept.

Let \( \mathcal{F}_{\text{tip}} \) be the wrench that the manipulator applies to the environment. The manipulator dynamics can be written

\[
M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) + b(\dot{\theta}) + J^T(\theta)\mathcal{F}_{\text{tip}} = \tau, \tag{11.34}
\]

where the Jacobian \( J(\theta) \) satisfies \( \mathcal{V} = J(\theta)\dot{\theta} \). Since the robot typically moves slowly (or not at all) during a force control task, we ignore the acceleration and velocity terms to get

\[
g(\theta) + J^T(\theta)\mathcal{F}_{\text{tip}} = \tau. \tag{11.35}
\]

In the absence of any direct measurements of the force-torque at the robot end-effector, joint angle feedback alone can be used to implement the force control law

\[
\tau = \tilde{g}(\theta) + J^T(\theta)\mathcal{F}_d, \tag{11.36}
\]

where \( \tilde{g}(\theta) \) is the model of gravitational torques and \( \mathcal{F}_d \) is the desired wrench. This control law requires a good model for gravity compensation as well as precise control of the torques produced at the robot joints. In the case of a DC
11.3. Force Control

electric motor without gearing, torque control can be achieved by current control of the motor. In the case of a highly geared actuator, however, large friction torque in the gearing can degrade the quality of torque control achieved using only current control. In this case, the output of the gearing can be instrumented with strain gauges to directly measure the joint torque, which is fed back to a local controller that modulates the motor current to achieve the desired output torque.

Another solution is to equip the robot arm with a six-axis force-torque sensor between the arm and the end-effector to directly measure the end-effector wrench \( F_{\text{tip}} \) (Figure 11.14). Force-torque measurements are often noisy, so the time derivative of these measurements may not be meaningful. In addition, the desired wrench \( F_d \) is typically constant or only slowly changing. These characteristics suggest a PI force controller with a feedforward term and gravity compensation,

\[
\tau = \dot{g}(\theta) + J^T(\theta) \left( F_d + K_{fp} F_e + K_{fi} \int F_e(t) dt \right),
\]

where \( F_e = F_d - F_{\text{tip}} \) and \( K_{fp} \) and \( K_{fi} \) are positive-definite proportional and integral gain matrices, respectively. In the case of perfect gravity modeling, plugging the force controller (11.37) into the dynamics (11.35), we get the error dynamics

\[
K_{fp} F_e + K_{fi} \int F_e(t) dt = 0.
\]

In the case of a constant force disturbance on the right-hand side of (11.38), arising from an incorrect model of \( \dot{g}(\theta) \), for example, we take the derivative to get

\[
K_{fp} \dot{F}_e + K_{fi} F_e = 0,
\]
showing that $\mathcal{F}_c$ converges to zero for positive-definite $K_{fp}$ and $K_{fi}$.

The control law (11.37) is simple and appealing, but potentially dangerous if incorrectly applied. If there is nothing for the robot to push against, the robot will accelerate in a failing attempt to create end-effector forces. Since a typical force control task requires little motion, we can limit this acceleration by adding velocity damping. This gives the modified control law

$$\tau = \ddot{\theta} + J^T(\theta) \left( \mathcal{J}_d + K_{fp} \mathcal{F}_c + K_{fi} \int \mathcal{F}_c(t) dt - K_{damp} \mathcal{V} \right),$$

where $K_{damp}$ is positive definite.

### 11.4 Hybrid Motion-Force Control

Most tasks requiring the application of controlled forces also require the application of controlled motions. Control to achieve this is called hybrid motion-force control. If the task space is $n$-dimensional, then we are free to specify $n$ of the $2n$ forces and motions at any time $t$; the other $n$ are determined by the environment. Apart from this constraint, we also should not specify forces and motions in the “same direction,” as they are not independent.

As an example, consider a two-dimensional environment modeled by a damper, $\mathcal{F} = B_{env} \mathcal{V}$, where

$$B_{env} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}.$$  

Defining the components of $\mathcal{V}$ and $\mathcal{F}$ as $(V_1, V_2)$ and $(F_1, F_2)$, we have $F_1 = 2V_1 + V_2, F_2 = V_1 + V_2$. We have $n = 2$ freedoms to choose among the $2n = 4$ velocities and forces at any time. For example, we can specify both $F_1$ and $V_1$ independently, because $B_{env}$ is not diagonal. Then $V_2$ and $F_2$ are determined by $B_{env}$. We cannot independently control both $F_1$ and $2V_1 + V_2$, however, as these are in the “same direction” according to the damper.

#### 11.4.1 Natural and Artificial Constraints

A particularly interesting case is when the environment is infinitely stiff (rigid constraints) in $k$ directions and unconstrained in $n - k$ directions. In this case, we cannot choose which of the $2n$ motions and forces to specify—the contact with the environment chooses the $k$ directions in which the robot can freely apply forces and the $n - k$ directions of free motion. As an example, consider the task space to have the $n = 6$ dimensions of $SE(3)$. Then a robot opening a cabinet door has $6 - k = 1$ motion freedom, rotation about the cabinet hinges, and therefore $k = 5$ force freedoms—the robot can apply any wrench that has zero moment about the axis of the hinges without moving the door.

As another example, a robot writing on a chalkboard may freely control the force into the board ($k = 1$), but it cannot penetrate the board; and it may
freely move with $6 - k = 5$ degrees of freedom (two specifying the motion of the
tip of the chalk in the plane of the board, and three describing the orientation
of the chalk), but it cannot independently control forces in these directions.

The chalk example comes with two caveats. The first is due to friction—
the chalk-wielding robot can actually control forces tangent to the plane of the
board, provided the requested motion in the plane of the board is zero and the
requested tangential forces do not exceed the static friction limit determined by
the friction coefficient and the normal force into the board (see friction modeling
in Chapter 12). Within this regime, the robot has three motion freedoms
and three force freedoms. Second, the robot could decide to pull away from the
board. In this regime, the robot has six motion freedoms and no force freedoms.

Thus the configuration of the robot is not the only factor determining the
directions of motion and force freedoms. Nonetheless, in this section we consider
the simplified case where the motion and force freedoms are determined solely
by the robot’s configuration, and all constraints are equality constraints. For
example, the inequality velocity constraint of the board (the chalk cannot pen-
eterate the board) is treated as an equality constraint (the robot also does not
pull the chalk away from the board).

As a final example, consider a robot erasing a frictionless chalkboard using
an eraser modeled as a rigid block (Figure 11.15). Let the configuration $X(t) \in
SE(3)$ be the configuration of the block’s frame $\{b\}$ relative to a space frame
$\{s\}$. The body-frame twist and wrench are written
$V_b = (\omega_x, \omega_y, \omega_z, v_x, v_y, v_z)^T$
and $F_b = (m_x, m_y, m_z, f_x, f_y, f_z)^T$, respectively. Maintaining contact with the
board puts $k = 3$ constraints on the twist:

$$\begin{align*}
\omega_x &= 0 \\
\omega_y &= 0 \\
v_z &= 0.
\end{align*}$$

In the language of Chapter 2, these velocity constraints are holonomic—the
differential constraints can be integrated to give configuration constraints.

These constraints are called natural constraints, specified by the en-
vironment. There are $6 - k = 3$ natural constraints on the wrench, too:
$m_z = f_x = f_y = 0$. In light of the natural constraints, we can freely specify
any twist of the eraser satisfying the $k = 3$ velocity constraints and any wrench
satisfying the $6 - k = 3$ wrench constraints (provided $f_z < 0$, to maintain con-
tact with the board). These motion and force specifications are called artificial
constraints. Below is an example set of artificial constraints corresponding to
the natural constraints:

\begin{align*}
\text{natural constraints} & \quad \text{artificial constraints} \\
\omega_x &= 0 & m_x &= 0 \\
\omega_y &= 0 & m_y &= 0 \\
m_z &= 0 & \omega_z &= 0 \\
f_x &= 0 & v_x &= k_1 \\
f_y &= 0 & v_y &= 0 \\
v_z &= 0 & f_z &= k_2 < 0
\end{align*}

The artificial constraints cause the eraser to move with \( v_x = k_1 \) while applying a constant force \( k_2 \) against the board.

\section*{11.4.2 A Hybrid Controller}

We now return to the problem of designing a hybrid motion-force controller. If the environment is rigid, then we express the \( k \) natural constraints on velocity in task space as the Pfaffian constraints

\[ A(X)\dot{V} = 0, \quad (11.41) \]

where \( A(X) \in \mathbb{R}^{k \times m} \) and \( m = 6 \) for \( X \in SE(3) \). (Alternatively these constraints could be written in terms of constraints on the joint velocities, \( A(\theta)\dot{\theta} = 0 \).) This formulation includes holonomic and nonholonomic constraints.

If the task-space dynamics of the robot, in the absence of constraints, are

\[ \mathcal{F} = \Lambda(\theta)\dot{V} + \eta(\theta, V), \]
then the constrained dynamics, following Chapter 8.7, are

\[
\mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \eta(\theta, \mathcal{V}) + A^T(X)\lambda,
\]

(11.42)

where \( \lambda \in \mathbb{R}^k \) are Lagrange multipliers and \( \mathcal{F}_{\text{tip}} \) is the set of forces (or wrench) that the robot applies against the constraints. The requested force \( \mathcal{F}_d \) must lie in the column space of \( A^T(X) \).

Since Equation (11.41) must be satisfied at all times, we can replace (11.41) by the time derivative

\[
A(X)\dot{\mathcal{V}} + \dot{A}(X)\mathcal{V} = 0.
\]

(11.43)

To ensure that Equation (11.43) is satisfied when the system state already satisfies \( A(X)\mathcal{V} = 0 \), any requested acceleration \( \dot{\mathcal{V}}_d \) should satisfy

\[
A(X)\dot{\mathcal{V}}_d = 0.
\]

Now, solving Equation (11.42) for \( \dot{\mathcal{V}} \), plugging the result into (11.43), and solving for \( \lambda \), we get

\[
\lambda = (A\Lambda^{-1}A^T)^{-1}(A\dot{\mathcal{V}} + A\Lambda^{-1}(\eta - \mathcal{F})),
\]

(11.44)

where we have plugged in \( -A\dot{\mathcal{V}} = \dot{A}\mathcal{V} \) by Equation (11.43). With Equation (11.44), we can calculate the force \( \mathcal{F}_{\text{tip}} = A^T(q)\lambda \) the robot applies against the constraints.

Plugging Equation (11.44) into Equation (11.42) and manipulating, the \( n \) equations of the constrained dynamics (11.42) can be expressed as the \( n - k \) independent motion equations

\[
P(X)\mathcal{F} = P(X)(\Lambda(\theta)\dot{\mathcal{V}} + \eta(\theta, \mathcal{V}))
\]

(11.45)

where

\[
P = I - A^T(A\Lambda^{-1}A^T)^{-1}A\Lambda^{-1}
\]

(11.46)

and \( I \) is the identity matrix. The \( n \times n \) matrix \( P(X) \) has rank \( n - k \) and projects an arbitrary manipulator force \( \mathcal{F} \) onto the subspace of forces that move the end-effector tangent to the constraints. The rank \( k \) matrix \( I - P(X) \) projects an arbitrary force \( \mathcal{F} \) to the subspace of forces against the constraints. Thus \( P \) partitions the \( n \)-dimensional force space into forces that address the motion control task and forces that address the force control task.

Our hybrid motion-force controller is simply the sum of a task-space motion controller, derived from the computed torque control law (11.33), and a task-space force controller (11.37), each projected to generate forces in its appropriate
subspace:

\[ \tau = J^T(\theta) \left( P(X) \left( \dot{\Lambda}(\theta) \left[ \dot{X}_d + K_p X_e + K_i \int X_e(t) dt + K_d \dot{X}_e \right] \right) \right) \]  

\[ + \left( I - P(X) \right) \left( \mathcal{F}_d + K_f \mathcal{F}_e + K_f \int \mathcal{F}_e(t) dt \right) \]

\[ + \hat{\eta}(\theta, \mathcal{V}) \]

(11.47)

Because the dynamics of the two controllers are decoupled by the orthogonal projections \( P \) and \( I - P \), the controller inherits the error dynamics and stability analyses of the individual force and motion controllers on their respective subspaces.

A difficulty in implementing the hybrid control law (11.47) in rigid environments is knowing precisely the constraints \( A(X)\mathcal{V} = 0 \) active at any time. This is necessary to specify the desired motion and force and to calculate the projections, but any model of the environment will have some uncertainty. One approach to dealing with this issue is to use a real-time estimation algorithm to identify the constraint directions based on force feedback. Another is to sacrifice some performance by choosing low feedback gains, which makes the motion controller “soft” and the force controller more tolerant of force error. We can also build passive compliance into the structure of the robot itself to achieve a similar effect. In any case, some passive compliance is unavoidable, due to flexibility in the joints and links.

### 11.5 Impedance Control

Ideal hybrid motion-force control in rigid environments demands extremes in robot impedance, which characterizes the change in endpoint motion as a function of disturbance forces. Ideal motion control corresponds to high impedance (little change in motion due to force disturbances) while ideal force control corresponds to low impedance (little change in force due to motion disturbances). In practice, there are limits to a robot’s achievable impedance range.

In this section, we consider the problem of impedance control, where the robot end-effector renders particular mass, spring, and damper properties. \(^3\) For example, a robot used as a haptic surgical simulator could be tasked with mimicking the mass, stiffness, and damping properties of a virtual surgical instrument in contact with virtual tissue.

A one-degree-of-freedom robot rendering an impedance can be written

\[ m\ddot{x} + b\dot{x} + kx = f, \]  

(11.48)

\(^3\) A popular subcategory of impedance control is stiffness control or compliance control, where the robot renders a virtual spring only.
11.5. Impedance Control

Figure 11.16: A robot creating a one-degree-of-freedom mass-spring-damper virtual environment. The human hand applies a force \( f \) to the haptic interface.

where \( x \) is the position, \( m \) is the mass, \( b \) is the damping, \( k \) is the stiffness, and \( f \) is the force applied by the user (Figure 11.16). Loosely, we say that the robot renders high impedance if one or more of the \( \{m, b, k\} \) parameters, usually including \( b \) or \( k \), is large. Similarly, we say that the impedance is low if all of these parameters are small.

More formally, taking the Laplace transform of Equation (11.48), we get

\[
(m s^2 + bs + k)X(s) = F(s),
\]

and the impedance is defined by the transfer function from position perturbations to forces, \( Z(s) = F(s)/X(s) \). Thus impedance is frequency dependent, with low-frequency response dominated by the spring and high-frequency response dominated by the mass. Admittance is the inverse of impedance, \( Y(s) = Z^{-1}(s) = X(s)/F(s) \).

A good motion controller is characterized by high impedance (low admittance), since \( \Delta X = Y \Delta F \). If the admittance \( Y \) is small, then force perturbations \( \Delta F \) produce only small position perturbations \( \Delta X \). Similarly, a good force controller is characterized by low impedance (high admittance), since \( \Delta F = Z \Delta X \), and small \( Z \) implies that motion perturbations produce only small force perturbations.

The goal of impedance control is to implement the task-space behavior

\[
M \ddot{V} + B \dot{V} + KV = F_{\text{ext}},
\]

where \( X \in \mathbb{R}^n \) and \( V = \dot{X} \) are the task-space position and velocity\(^4\); \( M, B, \) and \( K \) are the positive-definite virtual mass, damping, and stiffness to be created by the robot; and \( F_{\text{ext}} \) is a force applied to the robot, perhaps by a user. The values of \( M, B, \) and \( K \) may change depending on the location in the virtual environment, to represent distinct objects for instance, but we will focus on the case of constant values. We could also replace \( \dot{V}, V, \) and \( X \) with small displacements \( \Delta \dot{V}, \Delta V, \) and \( \Delta X \) from reference values in a controlled motion of the robot, but we dispense with the extra notation here.

There are two common ways to achieve the behavior (11.50):

\(^4\)In this section we consider a simple coordinate description of the configuration of the virtual mass, such as \( X \in \mathbb{R}^3 \). The stiffness matrix \( K \) for a rigid body whose position is represented as \( X \in SE(3) \) is addressed in [42, 68].
The robot senses the motion $X(t)$ and commands joint torques and forces to create $-F_{\text{ext}}$, the force to display to the user. Such a robot is called impedance controlled, as it implements a transfer function $Z(s)$ from motions to forces. Theoretically, an impedance-controlled robot should only be coupled to an admittance-type environment.

The robot senses $F_{\text{ext}}$ using a wrist force-torque sensor and controls motions in response. Such a robot is called admittance controlled, as it implements a transfer function $Y(s)$ from forces to motions. Theoretically, an admittance-controlled robot should only be coupled to an impedance-type environment.

### 11.5.1 Impedance Control Algorithm

In an impedance control algorithm, encoders, tachometers, and possibly accelerometers are used to estimate the joint and endpoint positions, velocities, and possibly accelerations. Often impedance-controlled robots are not equipped with a wrist force-torque sensor, and instead rely on their ability to precisely control joint torques to display the appropriate end-effector force $-F_{\text{ext}}$ (from (11.50)) to the user. A good control law might be

$$\tau = J^T(\theta) \left( \begin{array}{c} \dot{\Lambda}(\theta) \dot{V} + \dot{\eta}(\theta, V) \\ \text{arm dynamics compensation} \\ \frac{M \dot{V} + BV + KX}{F_{\text{ext}}} \end{array} \right).$$

Addition of an end-effector force-torque sensor allows the use of feedback terms to more closely achieve the desired interaction force $-F_{\text{ext}}$.

In the control law (11.51), it is assumed that $\dot{V}$, $V$, and $X$ are measured directly. Measurement of the acceleration $\ddot{V}$ is likely to be noisy, however, and there is the problem of attempting to compensate for the robot’s mass after the acceleration has been sensed. Therefore, it is not uncommon to eliminate the mass compensation term $\dot{\Lambda}(\theta) \dot{V}$ and to set $M = 0$. The mass of the arm will be apparent to the user, but impedance-controlled manipulators are often designed to be lightweight. It is also not uncommon to assume small velocities and replace the nonlinear dynamics compensation $\dot{\eta}(\theta, V)$ with a simpler gravity compensation model.

Problems can arise when (11.51) is used to simulate stiff environments. Small changes in position, as measured by encoders for example, lead to large changes in motor torques. This effective high gain, coupled with delays, sensor quantization, and sensor errors, can lead to oscillatory behavior or instability. On the other hand, the effective gains are low when emulating low impedance environments. A lightweight backdrivable manipulator can excel at emulating low impedance environments.
11.5.2 Admittance Control Algorithm

In an admittance control algorithm, the force $F_{\text{ext}}$ applied by the user is sensed by the wrist load cell, and the robot responds with an end-effector acceleration satisfying Equation (11.50). A simple approach is to calculate the desired end-effector acceleration $\dot{V}_d$ according to

$$M\ddot{V}_d + BV + KX = F_{\text{ext}},$$

where $(X, V)$ is the current state. Solving, we get

$$\dot{V}_d = M^{-1}(F_{\text{ext}} - BV - KX).$$

(11.52)

Given $\dot{V}_d$, $V$, and $\theta$, the dynamics in the task space (8.88) can be used to determine the commanded wrench $F$. To make the response smoother in the face of noisy force measurements, the force readings can be low-pass filtered.

Simulating a low impedance environment is challenging for an admittance-controlled robot, as small forces produce large accelerations. The effective large gains can produce instability. On the other hand, admittance control by a highly geared robot can excel at emulating stiff environments.

11.6 Other Topics

Robust Control While all stable feedback controllers confer some amount of robustness to uncertainty, the field of robust control deals with designing controllers that explicitly guarantee the performance of a robot subject to bounded parametric uncertainties, such as in its inertial properties.

Adaptive Control Adaptive control of robots involves estimating the robot’s inertial or other parameters during execution and updating the control law in real time to incorporate those estimates.

Iterative Learning Control Iterative learning control (ILC) generally focuses on repetitive tasks. If a robot performs the same pick-and-place operation over and over, the trajectory errors from the previous execution can be used to modify the feedforward control for the next execution. In this way, the robot improves its performance over time, driving execution error toward zero. ILC differs from adaptive control in that the “learned” information is generally nonparametric and useful only for a single trajectory. On the other hand, ILC can account for effects that have not been parametrized in a model.

Passive Compliance and Flexible Manipulators All robots unavoidably have some passive compliance. Models of this compliance can be as simple as assuming torsional springs at each revolute joint (e.g., to account for finite stiffness in the flexspline of harmonic drive gearing) or as complicated as treating links as flexible beams. Two significant effects of flexibility are (1) a mismatch
between the motor angle reading, the true joint angle, and the endpoint of the attached link, and (2) increased order of the dynamics of the robot. These issues raise challenging problems in control, particularly when vibration modes are at low frequencies.

Some robots are specifically designed for passive compliance, particularly those meant for contact interactions with humans or the environment. Such robots may sacrifice motion control performance in favor of safety. One passively compliant actuator is the series elastic actuator, which consists of an electric motor, a gearbox, and a torsional or linear spring connecting the gearbox output to the link. This spring provides passive compliance between the stiff, highly geared motor and the robot link.

**Variable Impedance Actuators** The impedance of a joint is typically controlled using a feedback control law, as described in Section 11.5. There are limits to the bandwidth of this control, however; a joint that is actively controlled to behave as a spring will only achieve spring-like behavior to low-frequency perturbations.

A new class of actuators, called *variable impedance actuators* or *variable stiffness actuators*, aims to give actuators the desired passive mechanical impedance without the bandwidth limitations of an active control law. As an example, a variable stiffness actuator may consist of two motors, with one motor independently controlling the mechanical stiffness of the joint (e.g., based on the setpoint of an internal nonlinear spring) while the other motor produces a torque.

### 11.7 Summary

- A PID joint-space feedback controller takes the form

  $$\tau = K_p \theta_e + K_i \int \theta_e(t) dt + K_d \dot{\theta}_e,$$

  where $$\theta_e = \theta_d - \theta$$ and $$\theta_d$$ is the vector of desired joint angles. The proportional term acts like a virtual spring, to pull the robot joints to the desired positions; the derivative term acts like a virtual damper, acting to reduce velocity differences between the actual and desired velocities; and the integral term can be used to eliminate steady-state error in setpoint control.

- The linear error dynamics

  $$a_n \ddot{\theta}_e^{(n)} + a_{n-1} \theta_e^{(n-1)} + \ldots + a_2 \dot{\theta}_e + a_1 \theta_e + a_0 \theta_e = 0$$

  are stable, i.e., initial errors converge exponentially to zero, if and only if all of the complex roots $$s_1, \ldots, s_n$$ of the characteristic equation

  $$a_n s^n + a_{n-1} s^{n-1} + \ldots + a_2 s^2 + a_1 s + a_0 = 0$$

  have real components less than zero, i.e., $$\text{Re}(s_i) < 0$$ for all $$i = 1 \ldots n.$$
11.7. Summary

- Stable second-order linear error dynamics can be written in the standard form
  \[ \ddot{\theta}_e + 2\zeta\omega_n\dot{\theta}_e + \omega_n^2\theta_e = 0, \]
  where \( \zeta \) is the damping ratio and \( \omega_n \) is the natural frequency. The roots of the characteristic equation are
  \[ s_{1,2} = -\zeta\omega_n \pm \omega_n\sqrt{\zeta^2 - 1}. \]
  The system is overdamped if \( \zeta > 1 \), critically damped if \( \zeta = 1 \), and underdamped if \( \zeta < 1 \). The step response of a stable second-order system is often characterized by its overshoot, 2% settling time, and steady-state error.

- The joint-space computed torque controller is
  \[ \tau = \tilde{M}(\theta)\left(\ddot{\theta}_d + K_p\dot{\theta}_e + K_i\int \theta_e(t)dt + K_d\dot{\theta}_e\right) + \tilde{h}(\theta, \dot{\theta}). \]
  This controller cancels nonlinear terms, uses feedforward control to anticipate the desired acceleration, and uses linear feedback control for stabilization.

- For robots without joint friction and a perfect model of gravity forces, joint-space PD setpoint control plus gravity compensation
  \[ \tau = K_p\theta_e + K_d\dot{\theta} + \tilde{g}(\theta) \]
  yields global convergence to \( \theta_e = 0 \) by the Krasovskii-LaSalle invariance principle.

- The task-space computed torque motion controller is
  \[ \tau = J^T(\theta)\left(\ddot{\lambda}(\theta)\left(\dot{V}_d + K_pX_e + K_i\int X_e(t)dt + K_dV_e\right) + \tilde{\eta}(\theta, V)\right). \]
  There are several possible choices for calculating \( X_e \) and \( V_e \).

- Task-space force control can be achieved by the controller
  \[ \tau = \tilde{g}(\theta) + J^T(\theta)\left(\dot{F}_d + K_{fp}\dot{F}_e + K_{fi}\int F_e(t)dt - K_{damp}V\right), \]
  consisting of gravity compensation, feedforward force control, PI force feedback, and damping to prevent fast motion.

- In an \( m \)-dimensional task space (where typically \( m = 6 \)), rigid constraints specify \( m - k \) free motion directions and \( k \) constraint directions in which forces can be applied. These constraints can be represented as \( A(X)V = 0 \). A force \( F \) can be partitioned as \( F = P(X)F + (I - P(X))F \), where \( P(X) \)
projects to forces that move the end-effector tangent to the constraints and \( I - P(X) \) projects to forces against the constraints. The projection matrix \( P(X) \) is written in terms of the task-space mass matrix \( \Lambda(\theta) \) and constraints \( A(X) \) as

\[
P = I - A^T(A\Lambda^{-1}A^T)^{-1}A\Lambda^{-1}.
\]

- A hybrid motion-force controller using the projection \( P(X) \) can be written

\[
\tau = J^T(\theta) \left( P(X) \left( \dot{\Lambda}(\theta) \left[ \dot{\theta}_d + K_p X_e + K_i \int X_e(t)dt + K_d V_e \right] \right) \right)
\]

motion control

\[
+ (I - P(X)) \left( \mathcal{F}_d + K_{fp} F_e + K_{fi} \int F_e(t)dt \right)
\]

force control

\[
+ \tilde{\eta}(\theta, \mathcal{V})
\]

nonlinear compensation

- An impedance controller measures end-effector motions and creates endpoint forces to mimic a mass-spring-damper system. An admittance controller measures end-effector forces and creates endpoint motions to achieve the same purpose.

### 11.8 Software

```matlab
taulist = ComputedTorque(thetalist, dthetalist, eint, g, Mlist, Glist, Slist, thetalistd, dthetalistd, ddthetalistd, Kp, Ki, Kd)
```

This function computes the joint controls \( \tau \) for the computed torque control law (11.21) at a particular time instant. The inputs are the \( n \)-vectors of joint variables, joint velocities, and joint error integrals; the gravity vector \( g \); a list of transforms \( M_{i-1,i} \) describing the link center of mass locations; a list of link spatial inertia matrices \( G_i \); a list of joint screw axes \( S_i \) expressed in the base frame; the \( n \)-vectors \( \dot{\theta}_d \), \( \ddot{\theta}_d \), and \( \dot{\theta}_d \) describing the desired motion; and the scalar PID gains \( k_{fp}, k_i, \) and \( k_d \), where the gain matrices are just \( K_p = k_p I \), \( K_i = k_i I \), and \( K_d = k_d I \).

```matlab
[taumat, themat] = SimulateControl(thetalist, dthetalist, g, Ftipmat, Mlist, Glist, Slist, thetalistd, dthetalistd, ddthetalistd, gtilde, Mtildelist, Mtildelist, Kp, Ki, Kd, dt, intRes)
```

This function simulates the performance of the computed torque control law (11.21) over a given desired trajectory. Inputs include the initial state of the robot, \( \theta(0) \) and \( \dot{\theta}(0) \); the gravity vector \( g \); an \( N \times 6 \) matrix of wrenches applied by the end-effector, where each of the \( N \) rows is an instant in time in the trajectory; a list of transforms \( M_{i-1,i} \) describing the link center of mass locations; a list of link
spatial inertia matrices $G_i$; a list of joint screw axes $S_i$ expressed in the base frame; the $N \times n$ matrices of desired joint positions, velocities, and accelerations, where each of the $N$ rows corresponds to an instant in time; a (possibly incorrect) model of the gravity vector; a (possibly incorrect) model of the transforms $M_{i-1,i}$; a (possibly incorrect) model of the link inertia matrices; the scalar PID gains $k_p$, $k_i$, and $k_d$, where the gain matrices are just $K_p = k_p I$, $K_i = k_i I$, and $K_d = k_d I$; the timestep between each of the $N$ rows in the matrices defining the desired trajectory; and the number of integration steps to take during each timestep.

11.9 Notes and References

The computed torque controller originates from research in the 1970's [98, 79, 7, 103], and issues with its practical implementation (e.g., its computational complexity and modeling errors) have driven much of the subsequent research in nonlinear control, robust control, iterative learning control, and adaptive control. PD control plus gravity compensation was suggested and analyzed in [134], and subsequent analysis and modification of the basic controller is reviewed in [50].

The task space approach to motion control, also called operational space control, was originally outlined in [72, 52]. A geometric approach to tracking control for mechanical systems is presented in [15], where the configuration space for the system can be a generic manifold, including $SO(3)$ and $SE(3)$.

The notion of natural and artificial constraints in hybrid motion-force control was first described by Mason [80], and an early hybrid motion-force controller based on these concepts is reported in [102]. As pointed out by Duffy [28], we must take care in specifying the subspaces in which motions and forces can be controlled. The approach to hybrid motion-force control in this chapter mirrors the geometric approach of Liu and Li [67]. Impedance control was first described in a series of papers by Hogan [38, 39, 40].

Robot control builds on the well-established field of linear control (e.g., [33, 3]) and the growing field of nonlinear control. General references on robot control include the edited volume [23] and the textbooks by Spong et al. [129], Siciliano et al. [124], Craig [22], Murray et al. [90], the Handbook of Robotics chapters on Motion Control [20] and Force Control [137], the Robot Motion Control chapter in the Encyclopedia of Systems and Control [128], and, for underactuated and nonholonomic robots, the chapters in the Control Handbook [74] and the Encyclopedia of Systems and Control [73].
11.10 Exercises

1. Classify the following robot tasks as motion control, force control, hybrid motion-force control, impedance control, or some combination. Justify your answer.

   (i) Tightening a screw with a screwdriver.
   (ii) Pushing a box on the floor.
   (iii) Pouring a glass of water.
   (iv) Shaking hands with a human.
   (v) Throwing a baseball to hit a target.
   (vi) Shoveling snow.
   (vii) Digging a hole.
   (viii) Giving a back massage.
   (ix) Vacuuming the floor.
   (x) Carrying a tray of glasses.

2. The 2% settling time of an underdamped second-order system is approximately \( t = \frac{4}{(\zeta \omega_n)} \), based on \( e^{-\zeta \omega_n t} = 0.02 \). What is the 5% settling time?

3. Solve for any constants and give the specific equation for an underdamped second-order system with \( \omega_n = 4 \), \( \zeta = 0.2 \), \( \theta_e(0) = 1 \), and \( \dot{\theta}_e(0) = 0 \) (a step response). Calculate the damped natural frequency, approximate overshoot, and 2% settling time. Plot the solution on a computer and measure the exact overshoot and settling time.

4. Solve for any constants and give the specific equation for an underdamped second-order system with \( \omega_n = 10 \), \( \zeta = 0.1 \), \( \theta_e(0) = 0 \), and \( \dot{\theta}_e(0) = 1 \). Calculate the damped natural frequency. Plot the solution on a computer.

5. Consider a pendulum in gravity \( g = 10 \text{ m/s}^2 \). The pendulum consists of a 2 kg mass at the end of a 1 m massless rod. The pendulum joint has a viscous friction coefficient of \( b = 0.1 \text{ Nms/rad} \).

   (i) Write the equation of motion of the pendulum in terms of \( \theta \), where \( \theta = 0 \) corresponds to the “hanging down” configuration.
(ii) Linearize the equation of motion about the stable “hanging down” equilibrium. To do this, replace any trigonometric terms in \( \theta \) with the linear term in the Taylor expansion. Give the effective mass and spring constants \( m \) and \( k \) in the linearized dynamics \( m\ddot{\theta} + b\dot{\theta} + k\theta = 0 \). At the stable equilibrium, what is the damping ratio? Is the system underdamped, critically damped, or overdamped? If it is underdamped, what is the damped natural frequency? What is the time constant of convergence to the equilibrium and the 2% settling time?

(iii) Now write the linearized equations of motion for \( \theta = 0 \) at the balanced upright configuration. What is the effective spring constant \( k \)?

(iv) You add a motor at the joint of the pendulum to stabilize the upright position, and you choose a P controller \( \tau = K_p \theta \). For what values of \( K_p \) is the upright position stable?

6. You will develop a controller for a one-degree-of-freedom mass-spring-damper system of the form \( m\ddot{x} + b\dot{x} + kx = f \), where \( f \) is the control force and \( m = 4 \) kg, \( b = 2 \) Ns/m, and \( k = 0.1 \) N/m.

(i) What is the damping ratio of the uncontrolled system? Is the uncontrolled system overdamped, underdamped, or critically damped? If it is underdamped, what is the damped natural frequency? What is the time constant of convergence to the origin?

(ii) Choose a P controller \( f = K_p x_e \), where \( x_e = x_d - x \) is the position error and \( x_d = 0 \). What value of \( K_p \) yields critical damping?

(iii) Choose a D controller \( f = K_d \dot{x}_e \), where \( \dot{x}_d = 0 \). What value of \( K_d \) yields critical damping?

(iv) Choose a PD controller that yields critical damping and a 2% settling time of 0.01 s.

(v) For the PD controller above, if \( x_d = 1 \) and \( \dot{x}_d = \ddot{x}_d = 0 \), what is the steady-state error \( x_e(t) \) as \( t \) goes to infinity? What is the steady-state control force?

(vi) Now plug a PID controller in for \( f \). Assume \( x_d \neq 0 \) and \( \dot{x}_d = \ddot{x}_d = 0 \). Write the error dynamics in terms of \( \dot{x}_e, \ddot{x}_e, x_e, \) and \( \int x_e(t)dt \) on the left-hand side and a constant forcing term on the right-hand side. (Hint: You can write \( kx \) as \( -k(x_d - x) + kx_d \).) Take the time derivative of this equation and give the conditions on \( K_p, K_i, \) and \( K_d \) for stability. Show that zero steady-state error is possible with a PID controller.

7. Simulation of a one-degree-of-freedom robot and robot controller.
(i) Write a simulator for a one-joint robot consisting of a motor rotating a link in gravity using the model parameters given in Section 11.2.2. The simulator should consist of (1) a dynamics function that takes as input the current state of the robot and the torque applied by the motor and gives as output the acceleration of the robot; and (2) a numerical integrator that uses the dynamics function to calculate the new state of the system over a series of timesteps $\Delta t$. A first-order Euler integration method suffices for this problem (e.g., $\theta(k + 1) = \theta(k) + \dot{\theta}(k)\Delta t$, $\theta(k + 1) = \dot{\theta}(k) + \ddot{\theta}(k)\Delta t$). Test the simulator in two ways: (a) starting the robot at rest at $\theta = -\pi/2$ and applying a constant torque of 0.5 Nm and (b) starting the robot at rest at $\theta = -\pi/4$ and applying a constant torque of 0 Nm. For both examples, plot the position as a function of time for sufficient duration to see the basic behavior. Make sure the behavior makes sense. A reasonable choice of $\Delta t$ is 1 ms.

(ii) Add two more functions to your simulator: (1) a trajectory generator function that takes the current time and returns the desired state and acceleration of the robot, and (2) a control function that takes the current state of the robot and information from the trajectory generator and returns a control torque. The simplest trajectory generator would return $\theta = \theta_{d1}$ and $\dot{\theta} = \ddot{\theta} = 0$ for all time $t < T$, and $\theta = \theta_{d2} \neq \theta_{d1}$ and $\dot{\theta} = \ddot{\theta} = 0$ for all time $t \geq T$. This trajectory is a step function in position. Use PD feedback control for the control function and set $K_p = 10$ Nm/rad. For a well-tuned choice of $K_d$, give $K_d$ (including units) and plot the position as a function of time over 2 seconds for an initial state at rest at $\theta = -\pi/2$ and a step trajectory with $\theta_{d1} = -\pi/2$ and $\theta_{d2} = 0$. The step occurs at $T = 1$ s.

(iii) Demonstrate two different choices of $K_d$ that yield (1) overshoot and (2) sluggish response with no overshoot. Give the gains and the position plots.

(iv) Add a nonzero $K_i$ to your original well-tuned PD controller to eliminate steady-state error. Give the PID gains and plot the results of the step test.

8. Modify the simulation of the one-joint robot in Exercise 7 to model a flexible transmission from the motor to the link with a stiffness of 500 Nm/rad. Tune a PID controller to give a good step response from $\theta = -\pi/2$ to $\theta = 0$. Give the gains and plot the response.

9. Simulation of a two-degree-of-freedom robot and robot controller (Figure 11.17).

(i) Dynamics. Derive the dynamics of a 2R robot in gravity (Figure 11.17). The mass of link $i$ is $m_i$, the center of mass is a distance $r_i$ from the joint, the scalar inertia of link $i$ about the joint is $I_i$, and the length of link $i$ is $L_i$. There is no friction at the joints.
(ii) **Direct drive.** Assume each joint is directly driven by a DC motor with no gearing. Each motor comes with specifications of the mass $m_{\text{stator}}^i$ and inertia $I_{\text{stator}}^i$ of the stator and the mass $m_{\text{rotor}}^i$ and inertia $I_{\text{rotor}}^i$ of the rotor (the spinning portion). For the motor at joint $i$, the stator is attached to link $i - 1$ and the rotor is attached to link $i$. The links are thin uniform-density rods of mass $m_i$ and length $L_i$.

In terms of the quantities given above, for each link $i \in \{1, 2\}$ give equations for the total inertia $I_i$ about the joint, mass $m_i$, and distance $r_i$ from the joint to the center of mass. Think about how to assign the mass and inertia of the motors to the different links.

(iii) **Geared robot.** Assume motor $i$ has gearing with gear ratio $G_i$, and that the gearing itself is massless. As in the part above, for each link $i \in \{1, 2\}$, give equations for the total inertia $I_i$ about the joint, mass $m_i$, and distance $r_i$ from the joint to the center of mass.

(iv) **Simulation and control.** As in Exercise 7, write a simulator with (at least) four functions: a dynamics function, a numerical integrator, a trajectory generator, and a controller. Assume zero friction at the joints, $g = 9.81 \text{ m/s}^2$ in the direction indicated, $L_1 = 1 \text{ m}$, $r_i = 0.5 \text{ m}$, $m_1 = 3 \text{ kg}$, $m_2 = 2 \text{ kg}$, $I_1 = 2 \text{ kgm}^2$, and $I_2 = 1 \text{ kgm}^2$. Write a PID controller, find gains that give a good response, and plot the joint angles as a function of time for a trajectory with a reference at rest at $(\theta_1, \theta_2) = (-\pi/2, 0)$ for $t < 1 \text{ s}$ and $(\theta_1, \theta_2) = (0, -\pi/2)$ for $t \geq 1 \text{ s}$. The initial state of the robot is at rest with $(\theta_1, \theta_2) = (-\pi/2, 0)$.

(v) **Torque limits.** Real motors have limits on torque. While these limits are generally velocity dependent, here we assume that each motor’s torque limit is independent of velocity, $\tau_i \leq |\tau_{\text{max}}^i|$. Assume $\tau_{\text{max}}^1 = 100 \text{ Nm}$ and $\tau_{\text{max}}^2 = 20 \text{ Nm}$. If the control law requests greater torque, the actual torque is saturated to these values. Rerun the previous PID control simulation and plot the torques as well as the position as a function of time.

(vi) **Friction.** Add a viscous friction coefficient of $b_i = 1 \text{ Nms/rad}$ to each joint and rerun the previous PID control simulation.

10. For the two-joint robot of Exercise 9, write a more sophisticated trajectory generator function. The trajectory generator should take as input

- the desired initial position, velocity, and acceleration of each joint;
- the desired final position, velocity, and acceleration; and
- the total time of motion $T$.

A call of the form...
Figure 11.17: A two-link robot arm. The length of link $i$ is $L_i$ and its inertia about the joint is $I_i$. Gravity is $g = 9.81$ m/s$^2$.

\[
[q_d, q_{dotd}, q_{dotdotd}] = \text{trajectory}(\text{time})
\]
returns the desired position, velocity, and acceleration of each joint at time $\text{time}$. The trajectory generator should provide a trajectory that is a smooth function of time.

As an example, each joint could follow a fifth-order polynomial trajectory of the form

\[
\theta_d(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5.
\]  

(11.53)

Given the desired positions, velocities, and accelerations of the joints at times $t = 0$ and $t = T$, you can uniquely solve for the six coefficients $a_0 \ldots a_5$ by evaluating Equation (11.53) and its first and second derivatives at $t = 0$ and $t = T$.

Tune a PID controller to track a fifth-order polynomial trajectory moving from rest at $(\theta_1, \theta_2) = (-\pi/2, 0)$ to rest at $(\theta_1, \theta_2) = (0, -\pi/2)$ in $T = 2$ s. Give the values of your gains and plot the reference position of both joints and the actual position of both joints. You are free to ignore torque limits and friction.

11. For the two-joint robot of Exercise 9 and fifth-order polynomial trajectory of Exercise 10, simulate a computed torque controller to stabilize the trajectory. The robot has no joint friction or torque limits. The model of the link masses should be 20% greater than their actual values to create error in the feedforward model. Give the PID gains and plot the reference and actual joint angles for both the computed torque controller as well as PID control only.

12. The Krasovskii-LaSalle invariance principle states the following: Given a system $\dot{x} = f(x), x \in \mathbb{R}^n$ such that $f(0) = 0$, and any energy-like function $V(x)$ such that

- $V(x) > 0$ for all $x \neq 0$;
11.10. Exercises

- \( V(x) \to \infty \) as \( x \to \infty \);
- \( V(0) = \dot{V}(0) = 0 \); and
- \( \dot{V}(x) \leq 0 \) along all trajectories of the system,

let \( S \) be the largest set of \( \mathbb{R}^n \) such that \( \dot{V}(x) = 0 \) and trajectories beginning in \( S \) remain in \( S \) for all time. Then if \( S \) contains only the origin, the origin is globally asymptotically stable—all trajectories converge to the origin.

Show how the Krasovskii-LaSalle principle is violated for centralized multi-joint PD setpoint control with gravity compensation, using the energy function \( V(x) \) from Equation (11.26), if \( K_p = 0 \) or \( K_d = 0 \). For a practical robot system, is it possible to use the Krasovskii-LaSalle invariance principle to demonstrate global asymptotic stability even if \( K_d = 0 \)? Explain your answer.

13. The two-joint robot of Exercise 9 can be controlled in task space using the endpoint task coordinates \( X = (x,y) \), as shown in Figure 11.17. The task space velocity is \( \dot{X} = \dot{V} \). Give the Jacobian \( J(\theta) \) and the task-space dynamics model \( \{\ddot{\Lambda}(\theta), \ddot{\eta}(\theta, V)\} \) in the computed torque control law (11.33).

14. Choose appropriate space and end-effector reference frames \( \{s\} \) and \( \{b\} \) and express natural and artificial constraints, six each, that achieve the following tasks: (a) opening a cabinet door; (b) turning a screw that advances linearly a distance \( p \) for every revolution; and (c) drawing a circle on a chalkboard with a piece of chalk.

15. Assume the end-effector of the two-joint robot in Figure 11.17 is constrained to move on the line \( x - y = 1 \). The robot’s link lengths are \( L_1 = L_2 = 1 \). (a) Write the constraint as \( A(X)\dot{V} = 0 \), where \( X = (x,y) \) and \( V = \dot{X} \). (b) Write the constraint as \( A(\theta)\dot{\theta} = 0 \).

16. Derive the constrained motion equations (11.45) and (11.46). Show all the steps.

17. We have been assuming that each actuator delivers the torque requested by the control law. In fact, there is typically an inner control loop running at each actuator, typically at a higher servo rate than the outer loop, to try to track the torque requested. Figure 11.18 shows two possibilities for a DC electric motor, where the torque \( \tau \) delivered by the motor is proportional to the current \( I \) through the motor, \( \tau = k_t I \). The torque from the DC motor is amplified by the gearhead with gear ratio \( G \).

In one control scheme, the motor current is measured by a current sensor and compared to the desired current \( I_{\text{com}} \); the error is passed through a PI controller which sets the duty cycle of a low-power pulse-width-modulation (PWM) digital signal; and the PWM signal is sent to an H-bridge that generates the actual motor current. In the second scheme, a strain gauge torque sensor is inserted
between the output of the motor gearing and the link, and the measured torque is compared directly to the requested torque $\tau_{\text{com}}$. Since a strain gauge measures deflection, the element it is mounted on must have a finite torsional stiffness. Series elastic actuators are designed to have particularly flexible torsional elements, so much so that encoders are used to measure the larger deflection. The torque is estimated from the encoder reading and the torsional spring constant.

(i) For the current sensing scheme, what multiplicative factor should go in the block labeled $I_{\text{com}}/\tau_{\text{com}}$? Even if the PI current controller does its job perfectly ($I_{\text{error}} = 0$) and the torque constant $k_t$ is perfectly known, what effect may contribute to error in the generated torque?

(ii) For the strain gauge measurement method, explain the drawbacks, if any, of having a flexible element between the gearhead and the link.

18. Modify the SimulateControl function to allow initial state errors.
Chapter 12

Grasping and Manipulation

Most of the book so far has been concerned with kinematics, dynamics, motion planning, and control of the robot itself. Only in Chapter 11, on the topics of force control and impedance control, did the robot finally begin interacting with an environment other than free space. This is when a robot really becomes valuable—when it can perform useful work on objects in the environment.

In this chapter our focus moves outward, from the robot itself to the interaction between the robot and its environment. The desired behavior of the robot hand or end-effector, whether motion control, force control, hybrid motion-force control, or impedance control, is assumed to be achieved perfectly using the methods discussed so far. Our focus is on the contact interface between the robot and objects, as well as contacts among objects and between objects and constraints in the environment. In short, our focus is on manipulation rather than the manipulator. Examples of manipulation include grasping, pushing, rolling, throwing, catching, tapping, etc. To limit our scope, we will assume that the manipulator, objects, and obstacles in the environment are rigid.

To simulate, plan, and control robotic manipulation tasks, we need an understanding of (at least) three elements: contact kinematics; forces applied through contacts; and the dynamics of rigid bodies. Contact kinematics studies how rigid bodies can move relative to each other without penetration, and classifies these feasible motions according to whether the contacts are rolling, sliding, or separating. Contact force models address the normal and frictional forces that can be transmitted through rolling and sliding contacts. Finally, the actual motions of the bodies are those that simultaneously satisfy the kinematic constraints, contact force model, and rigid-body dynamics. The environment is assumed to consist of one or more rigid movable parts, and perhaps stationary constraints such as floors, walls, and fixtures. The manipulator consists of one or more controlled bodies (e.g., “fingers”) which could be under motion, force, hybrid motion-force, or impedance control.

The following definitions from linear algebra will be useful in this chapter:

Definition 12.1. Given a set of $j$ vectors $A = a_1, \ldots, a_j \in \mathbb{R}^n$, we define the
Grasping and Manipulation

Figure 12.1: (a) Three vectors in $\mathbb{R}^2$, drawn as arrows from the origin. (b) The linear span of the vectors is the entire plane. (c) The positive linear span is the cone shaded gray. (d) The convex span is the polygon and its interior.

linear span, or the set of linear combinations, of the vectors to be

$$\text{span}(\mathcal{A}) = \left\{ \sum_{i=1}^{j} k_i a_i \mid k_i \in \mathbb{R} \right\};$$

the nonnegative linear combinations, sometimes called the positive span, to be

$$\text{pos}(\mathcal{A}) = \left\{ \sum_{i=1}^{j} k_i a_i \mid k_i \geq 0 \right\};$$

and the convex span to be

$$\text{conv}(\mathcal{A}) = \left\{ \sum_{i=1}^{j} k_i a_i \mid k_i \geq 0 \text{ and } \sum_{i} k_i = 1 \right\}.$$

Clearly $\text{conv}(\mathcal{A}) \subseteq \text{pos}(\mathcal{A}) \subseteq \text{span}(\mathcal{A})$ (see Figure 12.1). The following facts from linear algebra will also be useful.

1. The space $\mathbb{R}^n$ can be linearly spanned by $n$ vectors, but no fewer.
2. The space $\mathbb{R}^n$ can be positively spanned by $n + 1$ vectors, but no fewer.

The first fact is implicit in our use of $n$ coordinates to represent $n$-dimensional Euclidean space. Fact 2 follows from the fact that for any choice of $n$ vectors $\mathcal{A} = \{a_1, \ldots, a_n\}$, there exists a vector $c \in \mathbb{R}^n$ such that $a_i^T c \leq 0$ for all $i$. In other words, no nonnegative combination of vectors in $\mathcal{A}$ can create a vector in the direction $c$. On the other hand, if we choose $a_1, \ldots, a_n$ to be orthogonal coordinate bases of $\mathbb{R}^n$, then choose $a_{n+1} = -\sum a_i$, we see that this set of $n + 1$ vectors positively spans $\mathbb{R}^n$.

12.1 Contact Kinematics

Contact kinematics is the study of how two or more rigid bodies can move relative to each other while respecting the impenetrability constraint. It also
classifies motion in contact as either rolling or sliding. Let’s start by looking at a single contact between two rigid bodies.

### 12.1.1 First-Order Analysis of a Single Contact

Consider two rigid bodies whose configuration is given by the local coordinate column vectors $q_1$ and $q_2$, respectively. Writing the composite configuration as $q = [q_1^T, q_2^T]^T$, we define a distance function $d(q)$ between the parts that is positive when they are separated, zero when they are touching, and negative when they are in penetration. When $d(q) > 0$, there are no constraints on the motions of the parts; each is free to move with six degrees of freedom. When the parts are in contact ($d(q) = 0$), we look at the time derivatives $\dot{d}$, $\ddot{d}$, etc., to determine if the parts stay in contact or break apart as they follow a particular trajectory $q(t)$. This can be determined by the following table of possibilities:

<table>
<thead>
<tr>
<th>$d$</th>
<th>$\dot{d}$</th>
<th>$\ddot{d}$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&gt;0$</td>
<td></td>
<td></td>
<td>no contact</td>
</tr>
<tr>
<td>$&lt;0$</td>
<td></td>
<td></td>
<td>infeasible (penetration)</td>
</tr>
<tr>
<td>$=0$</td>
<td>$&gt;0$</td>
<td></td>
<td>in contact, but breaking free</td>
</tr>
<tr>
<td>$=0$</td>
<td>$&lt;0$</td>
<td></td>
<td>infeasible (penetration)</td>
</tr>
<tr>
<td>$=0$</td>
<td>$=0$ $&gt;0$</td>
<td></td>
<td>in contact, but breaking free</td>
</tr>
<tr>
<td>$=0$</td>
<td>$=0$ $&lt;0$</td>
<td></td>
<td>infeasible (penetration)</td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The contact is maintained only if all time derivatives are zero.

Now let’s assume that the two bodies are initially in contact ($d = 0$) at a single point. The first two time derivatives of $d$ are written

\[
\dot{d} = \frac{\partial d}{\partial q} \dot{q},
\]

\[
\ddot{d} = \dot{q}^T \frac{\partial^2 d}{\partial q^2} \ddot{q} + \frac{\partial d}{\partial q} \ddot{q}.
\]

The terms $\partial d/\partial q$ and $\partial^2 d/\partial q^2$ carry information about the local contact geometry. The gradient vector $\partial d/\partial q$ corresponds to the separation direction in $q$ space associated with the **contact normal** (Figure 12.2). The matrix $\partial^2 d/\partial q^2$ encodes information about the relative curvature of the parts at the contact point.

In this chapter we assume that only contact normal information $\partial d/\partial q$ is available at contacts. Other information about the local contact geometry, including the contact curvature $\partial^2 d/\partial q^2$ and higher derivatives, is unknown. With this assumption, we truncate our analysis at Equation (12.1) and assume bodies remain in contact if $\dot{d} = 0$. Since we deal only with the first-order contact derivative $\partial d/\partial q$, we refer to our analysis as a **first-order** analysis. By a first-order analysis, the contacts in Figure 12.2 are treated identically, since they have the same contact normal.
Figure 12.2: (Left) The parts $A$ and $B$ in single-point contact define a contact tangent plane and a contact normal vector $\hat{n}$ perpendicular to the tangent plane. By default, the positive direction of the normal is chosen into body $A$. Since contact curvature is not addressed in this chapter, the contact places the same restrictions on the motions of the rigid bodies in the middle and right panels.

As indicated in the table above, a second-order analysis incorporating contact curvature $\partial^2 d / \partial q^2$ may indicate that the contact is actually breaking or penetrating even when $d = \dot{d} = 0$. We will see examples of this, but a detailed analysis of second-order contact conditions is beyond the scope of this chapter.

### 12.1.2 Contact Types: Rolling, Sliding, and Breaking Free

Given two bodies in single-point contact, they undergo a roll-slide motion if the contact is maintained. The constraint that contact is maintained is a holonomic constraint. A necessary condition for maintaining contact is $\dot{d} = 0$.

Let’s write the velocity constraint $\dot{d} = 0$ in a form that does not require an explicit distance function, based on the contact normal (Figure 12.2). Let $\hat{n} \in \mathbb{R}^3$ be a unit vector aligned with the contact normal, expressed in a world frame. Let $p_A \in \mathbb{R}^3$ be a representation of the point in contact on part $A$ in the world frame, and $p_B \in \mathbb{R}^3$ be the point in contact on part $B$. The condition $\dot{d} = 0$ is written

$$\hat{n}^T (\dot{p}_A - \dot{p}_B) = 0. \tag{12.3}$$

Since the contact normal is defined as into body $A$, the impenetrability constraint $\dot{d} \geq 0$ is written

$$\hat{n}^T (\dot{p}_A - \dot{p}_B) \geq 0. \tag{12.4}$$

Let us rewrite the constraint (12.4) in terms of the twists $\mathcal{V}_A = (\omega_A, v_A)$ and $\mathcal{V}_B = (\omega_B, v_B)$ of parts $A$ and $B$ in a space frame, respectively.\footnote{All twists and wrenches are expressed in a space frame in this chapter.} Note that

$$\dot{p}_A = v_A + \omega_A \times p_A = v_A + [\omega_A] p_A$$

$$\dot{p}_B = v_B + \omega_B \times p_B = v_B + [\omega_B] p_B.$$
We can also write the wrench \( F = (m, f) \) corresponding to a unit force applied
along the contact normal:
\[
F = (p_A \times \hat{n}, \hat{n}) = ([p_A] \hat{n}, \hat{n}).
\]
While it is not necessary to appeal to forces in a purely kinematic analysis of
rigid bodies, we will find it convenient to adopt this notation now in anticipation
of contact forces in Section 12.2.

With these expressions, the inequality constraint (12.4) can be written
\[
\text{(impenetrability constraint)} \quad F^T (v_A - v_B) \geq 0. \tag{12.5}
\]
(See Exercise 1). If
\[
\text{(active constraint)} \quad F^T (v_A - v_B) = 0, \tag{12.6}
\]
then, to first order, the constraint is active and the parts remain in contact.

In the case that \( B \) is a stationary fixture, the impenetrability constraint
(12.5) simplifies to
\[
F^T v_A \geq 0. \tag{12.7}
\]
If the condition (12.7) is satisfied, \( F \) and \( v_A \) are said to be repelling. If
\( F^T v_A = 0, F \) and \( v_A \) are said to be reciprocal and the constraint is active.

Twists \( v_A \) and \( v_B \) satisfying (12.6) are called first-order roll-slide mo-
tions—the contact may be either sliding or rolling. Roll-slide contacts may
be further separated into rolling contacts and sliding contacts. The contact
is rolling if the parts have no motion relative to each other at the contact:
\[
\text{(rolling constraint)} \quad \dot{p}_A = v_A + [\omega_A] p_A = v_B + [\omega_B] p_B = \dot{p}_B. \tag{12.8}
\]
Note that “rolling” contacts include those where the two parts remain stationary
relative to each other, i.e., no relative rotation. Thus “sticking” is another term
for these contacts.

If the twists satisfy Equation (12.6) but not the rolling equations of (12.8),
then they are sliding.

We assign a rolling contact the contact label \( R \), a sliding contact the label
\( S \), and a contact that is breaking free (the impenetrability constraint (12.5) is
satisfied but not the active constraint (12.6)) the label \( B \).

The distinction between rolling and sliding contacts becomes especially im-
portant when we consider friction forces in Section 12.2.

Example 12.1. Consider the contact shown in Figure 12.3. Parts \( A \) and \( B \) are
in contact at \( p_A = p_B = (1, 2, 0)^T \) with contact normal direction \( \hat{n} = (0, 1, 0)^T \).
The impenetrability constraint (12.5) is
Figure 12.3: Example 12.1. (Left) The body $B$ makes contact with $A$ at $p_A = p_B = (1, 2, 0)^T$ with normal $\hat{n} = (0, 1, 0)^T$. (Middle) The velocities $V_A$ and their corresponding contact labels for $B$ stationary and $A$ confined to a plane. The contact normal force $F$ is $(m_z, m_y, m_z, f_x, f_y, f_z)^T = (0, 0, 1, 0, 1, 0)^T$. (Right) Looking down the $-v_{Ax}$ axis.

and therefore roll-slide twists satisfy

$$\omega_{Az} - \omega_{Bz} + v_{Ay} - v_{By} = 0.$$  \hspace{1cm} (12.9)

Equation (12.9) defines an 11-dimensional hyperplane in the 12-dimensional space of twists ($V_A, V_B$).

The rolling constraints (12.8) are

$$v_{Ax} + \omega_{Az} p_{Ay} - \omega_{Ay} p_{Az} = v_{Bx} + \omega_{Bz} p_{By} - \omega_{By} p_{Bz}$$
$$v_{Ay} + \omega_{Az} p_{Ax} - \omega_{Ax} p_{Az} = v_{By} + \omega_{Bz} p_{Bx} - \omega_{Bx} p_{Bz}$$
$$v_{Az} + \omega_{Ax} p_{Ay} - \omega_{Ay} p_{Ax} = v_{Bz} + \omega_{Bx} p_{By} - \omega_{By} p_{Bx},$$

and plugging in for $p_A$ and $p_B$, we get

$$v_{Ax} + 2\omega_{Az} = v_{Bx} + 2\omega_{Bz}$$ \hspace{1cm} (12.10)
$$v_{Ay} + \omega_{Az} = v_{By} + \omega_{Bz}$$ \hspace{1cm} (12.11)
$$v_{Az} + 2\omega_{Ax} - \omega_{Ay} = v_{Bz} + 2\omega_{Bx} - \omega_{By}.$$ \hspace{1cm} (12.12)

The constraint equations (12.10)–(12.12) define a 9-dimensional hyperplane subset of the 11-dimensional hyperplane of roll-slide twists.

To visualize the constraints in a low-dimensional space, let’s assume that $B$ is stationary ($V_B = 0$) and $A$ is confined to the $z = 0$ plane, i.e., $V_A = (\omega_{Ax}, \omega_{Ay}, \omega_{Az}, v_{Ax}, v_{Ay}, v_{Az})^T = (0, 0, \omega_{Az}, v_{Ax}, v_{Ay}, 0)^T$. The wrench $F$ is written $(m_z, f_x, f_y)^T = (1, 0, 1)^T$. The roll-slide constraint (12.9) reduces to

$$v_{Ay} + \omega_{Az} = 0,$$

while the rolling constraints simplify to

$$v_{Ax} + 2\omega_{Az} = 0$$
$$v_{Ay} + \omega_{Az} = 0.$$
The single roll-slide constraint yields a plane in the \((\omega_{Az}, v_{Ax}, v_{Ay})\) space, and the two rolling constraints yield a line in that plane. Because \(V_B = 0\), the constraint surfaces pass through the origin \(V_A = 0\). If \(V_B \neq 0\), this is no longer the case in general.

Figure 12.3 graphically shows that nonpenetrating twists \(V_A\) must have a nonnegative dot product with the constraint wrench \(F\) when \(V_B = 0\).

### 12.1.3 Multiple Contacts

Now suppose that \(A\) is subject to several contacts, from \(B\) and perhaps other parts. Each impenetrability constraint (12.5) constrains \(V_A\) to a half-space of its six-dimensional twist space bounded by a five-dimensional hyperplane of the form \(F_i^T V_A = F_i^T V_B\). Unioning the set of constraints from all the contacts, we get a polyhedral convex set (polytope\(^2\) for short) \(V\) of feasible twists in the \(V_A\) space, written

\[
V = \{ V_A \mid F_i^T (V_A - V_i) \geq 0 \ \forall i \},
\]

where \(F_i\) corresponds to the \(i\)th contact normal and \(V_i\) is the twist of the other part in contact at contact \(i\). A contact constraint \(i\) is redundant if the half-space constraint contributed by \(F_i\) does not change the feasible twist polytope \(V\). In general, the feasible twist polytope for a part can consist of a six-dimensional interior (where no contact constraint is active), five-dimensional faces where one constraint is active, four-dimensional faces where two constraints are active, and so on, down to one-dimensional edges and zero-dimensional points. A twist \(V_A\) on an \(n\)-dimensional facet of the polytope indicates that \(6 - n\) independent (non-redundant) contact constraints are active.

If all of the bodies providing constraints are stationary, such as fixtures, then each constraint hyperplane defined by (12.5) passes through the origin of the \(V_A\) space. We call such a constraint homogeneous. The feasible twist set becomes a cone rooted at the origin, called a (homogeneous) polyhedral convex cone. Let \(F_i\) be the constraint wrench of stationary contact \(i\). Then the feasible twist cone \(V\) is

\[
V = \{ V_A \mid F_i^T V_A \geq 0 \ \forall i \}.
\]

If the \(F_i\) positively span the six-dimensional wrench space, or, equivalently, the convex hull of the \(F_i\) contains the origin in the interior, then the feasible twist polytope reduces to a point at the origin, the stationary contacts completely constrain the motion of the part, and we have form closure, discussed in more detail in Section 12.1.7.

As mentioned in Section 12.1.2, each point contact \(i\) can be given a label corresponding to the type of contact: \(B\) if the contact is breaking, \(R\) if the contact is rolling, and \(S\) if the contact is sliding, i.e., (12.6) is satisfied but (12.8) is not. The contact mode for the entire system can be written as the

\(^2\)We use the term “polytope” to refer generally to a convex set bounded by hyperplanes in an arbitrary vector space. The set need not be finite; it could be a cone with infinite volume. It could also be a point at the origin, or the null set if the constraints are incompatible with the rigid-body assumption.
concatenation of the contact labels at the contacts. Since we have three distinct contact labels, a system of bodies with \( k \) contacts can have a maximum of \( 3^k \) contact labels. Some of these contact modes may not be feasible, however, as their corresponding kinematic constraints may not be compatible.

**Example 12.2.** Figure 12.4 shows triangular fingers contacting a hexagonal part \( A \). To more easily visualize the contact constraints, the hexagon is restricted to translational motion in a plane only, so that its twist can be written \( \mathbf{V}_A = (0, 0, 0, v_{Ax}, v_{Ay}, 0) \). In Figure 12.4(a), the single stationary finger creates a contact wrench \( \mathcal{F}_1 \) that can be drawn in the \( \mathbf{V}_A \) space. All feasible twists have a nonnegative component in the direction of \( \mathcal{F}_1 \). Roll-slide twists satisfying \( \mathcal{F}_1^T \mathbf{V}_A = 0 \) lie on the constraint line. Since no rotations are allowed, the only twist yielding a rolling contact is \( \mathbf{V}_A = 0 \). In Figure 12.4(b), the union of the constraints due to two stationary fingers creates a cone of feasible twists. Figure 12.4(c) shows three fingers in contact, one of which is moving with twist \( \mathbf{V}_3 \). Because the moving finger has nonzero velocity, its constraint half-space is displaced from the origin by \( \mathbf{V}_3 \). The result is a closed polygon of feasible twists.

**Example 12.3.** Figure 12.5 shows the contact normals of three stationary contacts with a planar part \( A \), not shown. The part moves in a plane, so \( v_Az = \omega_Ax = \omega_Ay = 0 \). In this example we do not distinguish between rolling and sliding motions, so the locations of the contacts along the normals are irrelevant. The three contact wrenches, written \( (m_z, f_x, f_y) \), are \( \mathcal{F}_1 = (0, 1, -2), \mathcal{F}_2 = (-1, 0, 1) \), and \( \mathcal{F}_3 = (1, 0, 1) \), yielding the motion constraints

\[
\begin{align*}
v_{Ay} - 2\omega_{Az} & \geq 0 \\
-v_{Ax} + \omega_{Az} & \geq 0 \\
v_{Az} + \omega_{Az} & \geq 0.
\end{align*}
\]

These constraints describe a polyhedral convex cone of feasible twists rooted at the origin, as illustrated in Figure 12.5.

### 12.1.4 Collections of Parts

The discussion above can be generalized to find the feasible twists of multiple parts in contact. If parts \( i \) and \( j \) make contact at a point \( p \), where \( \hat{n} \) points into part \( i \) and \( \mathcal{F} = ([p] \hat{n}, \hat{n}) \), then their spatial twists \( \mathcal{V}_i \) and \( \mathcal{V}_j \) must satisfy the constraint

\[
\mathcal{F}^T(\mathcal{V}_i - \mathcal{V}_j) \geq 0
\]

to avoid penetration. This is a homogeneous half-space constraint in the composite \( (\mathcal{V}_i, \mathcal{V}_j) \) twist space. In an assembly of multiple parts, each pairwise contact contributes another constraint in the composite twist space, and the result is a polyhedral convex cone of kinematically feasible twists rooted at the origin of the composite twist space. The contact mode for the entire assembly is the concatenation of the contact labels at each contact in the assembly.

If there are parts whose motion is controlled, e.g., robot fingers, the constraints on the motion of the remaining parts are no longer homogeneous. As a
12.1. Contact Kinematics

Figure 12.4: Motion-controlled fingers contacting a hexagon that is constrained to translate in a plane only (Example 12.2). (a) A single stationary finger provides a single half-space constraint on the hexagon’s twist \( \mathcal{V}_A \). The feasible motion half-space is shaded gray. The two-dimensional set of twists corresponding to breaking contact \( \mathcal{B} \), the one-dimensional set corresponding to sliding contact \( \mathcal{S} \), and the zero-dimensional set corresponding to rolling (fixed) contact \( \mathcal{R} \) are shown. (b) The union of constraints from two stationary fingers creates a cone of feasible twists. This cone corresponds to four possible contact modes: \( \mathcal{R}\mathcal{R} \), \( \mathcal{S}\mathcal{B} \), \( \mathcal{B}\mathcal{S} \), and \( \mathcal{B}\mathcal{B} \). The contact label for the leftmost finger is given first. (c) Three fingers, one of which is moving with a linear velocity \( \mathcal{V}_3 \), create a closed polygon of feasible twists. There are seven possible contact modes corresponding to the feasible twists: a two-dimensional set where all contacts are breaking, three one-dimensional sets where one contact constraint is active, and three zero-dimensional sets where two contact constraints are active. Note that rolling contact at the moving finger is not feasible, since translation of the hexagon to “follow” the moving finger, as indicated by the \( \circ \) at the lower right of the figure, would violate one of the impenetrability constraints. If the third finger were stationary, the only feasible motion of the hexagon would be zero velocity, with contact mode \( \mathcal{R}\mathcal{R}\mathcal{R} \).

result, the convex polyhedral set of feasible twists of the uncontrolled parts, in their composite twist space, is no longer a cone rooted at the origin.

12.1.5 Other Types of Contacts

We have been considering point contacts of the type shown in Figure 12.6(a), where at least one of the bodies in contact uniquely defines the contact normal. Figures 12.6(b)-(e) show other types of contact. The kinematic constraints provided by the convex-concave vertex, line, and plane contacts of Figures 12.6(b)-(e)
Figure 12.5: Example 12.3. (Left) The lines of force corresponding to three stationary contacts on a planar body. If we are only concerned with feasible motions, and do not distinguish between rolling and sliding, contacts anywhere along the lines, with the contact normals shown, are equivalent. (Right) The three constraint half-spaces define a polyhedral convex cone of feasible twists. In the figure, the cone is truncated at the plane $v_{Ay} = 2$. The outer faces of the cone are shaded white and the inner faces are gray. Twists in the interior of the cone correspond to all contacts breaking, while twists on the faces of the cone correspond to one active constraint, and twists on one of the three edges of the cone correspond to two active constraints.

(d) are, to first order, identical to those provided by finite collections of single-point contacts. Constraints provided by other points of contact are redundant. The degenerate case in Figure 12.6(e) is ignored, as there is no unique definition of a contact normal.

The impenetrability constraint (12.5) derives from the fact that arbitrarily large contact forces can be applied in the normal direction to prevent penetration. In Section 12.2, we will see that tangential forces may also be applied due to friction, and these forces may prevent slipping between two objects in...
12.1 Contact Kinematics

Normal and tangential contact forces are subject to constraints: the normal force must be pushing into a part, not pulling, and the maximum friction force is proportional to the normal force.

If we wish to apply a kinematic analysis that can approximate the effects of friction without explicitly modeling forces, we can define three purely kinematic models of point contacts: a frictionless point contact, a point contact with friction, and a soft contact, also called a soft-finger contact. A frictionless point contact enforces only the roll-slide constraint (12.5). A point contact with friction also enforces the rolling constraints (12.8), implicitly modeling friction sufficient to prevent slip at the contact. A soft contact enforces the rolling constraints (12.8) as well as one more constraint: the two bodies in contact may not spin relative to each other about the contact normal axis. This models deformation and the resulting friction moment resisting spin due to the nonzero contact area between the two bodies. For planar problems, a point contact with friction and a soft contact are identical. See Table 12.1.

Similar to joints, which also specify the number of constraints and freedoms, these kinematic models of contact allow the application of Gröbler’s formula to determine the number of degrees of freedom of a set of bodies in contact. Because these models do not encode the unilateral nature of contact (contacts can “push” but not “pull”), however, we do not use them in the remainder of the chapter.

### 12.1.6 Planar Graphical Methods

Planar problems allow the possibility of using graphical methods to visualize the feasible motions for a single body, since the space of twists is three-dimensional. An example planar twist cone is shown in Figure 12.5. Such a figure would be very difficult to draw for a system with more than three degrees of freedom.

A convenient way to represent a planar twist $V = (\omega_z, v_x, v_y)$ is as a center of rotation (CoR) at $(-v_y/\omega_z, v_x/\omega_z)$ plus the angular velocity $\omega_z$. The CoR is the point in the (projective) plane that remains stationary under the motion, i.e., the point where the screw axis intersects the plane.\(^3\) In the case that the

\(^3\)Note that the case $\omega_z = 0$ must be treated with care, as it corresponds to a CoR at infinity.

<table>
<thead>
<tr>
<th>contact type</th>
<th>planar constraints</th>
<th>freedoms</th>
<th>spatial constraints</th>
<th>freedoms</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPC</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>PCWF</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>SC</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 12.1: The number of constraints and freedoms allowed by a kinematic model of a frictionless point contact (FPC), a point contact with friction (PCWF), and a soft contact (SC).
Grasping and Manipulation

Figure 12.7: Given the velocity of two points on the part, the lines normal to the velocities intersect at the CoR. The CoR shown is labeled + corresponding to the (counterclockwise) positive angular velocity of the part.

Figure 12.8: Mapping a planar twist $\mathcal{V}$ to a CoR. The ray containing a vector $\mathcal{V}$ intersects either the plane of $+$ CoRs at $\omega_z = 1$, the plane of $-$ CoRs at $\omega_z = -1$, or the circle of translation directions.

speed of motion is immaterial, we may simply label the CoR with a $+,$ $-$, or 0 sign representing the direction of rotation (Figure 12.7). The mapping from planar twists to CoRs is illustrated in Figure 12.8, which shows that the space of CoRs consists of a plane of $+$ CoRs (counterclockwise), a plane of $-$ CoRs (clockwise), and a circle of translation directions.

Given two distinct twists $\mathcal{V}_1$ and $\mathcal{V}_2$, the set of linear combinations of these velocities $k_1 \mathcal{V}_1 + k_2 \mathcal{V}_2$, where $k_1, k_2 \in \mathbb{R}$, maps to the line of CoRs containing CoR($\mathcal{V}_1$) and CoR($\mathcal{V}_2$). Since $k_1$ and $k_2$ can have either sign, if either $\omega_{1z}$ or $\omega_{2z}$ is nonzero, the CoRs on this line can have either sign. If $\omega_{1z} = \omega_{2z} = 0$, then this set corresponds to the set of all translation directions.

A more interesting case is when $k_1, k_2 \geq 0$. Given two twists $\mathcal{V}_1$ and $\mathcal{V}_2$, the
12.1. Contact Kinematics

The intersection of a twist cone with the unit twist sphere, and the representation of the cone as a set of CoRs.

Figure 12.10: (a) Positive linear combination of two CoRs labeled +. (b) Positive linear combination of a + CoR and a − CoR. (c) Positive linear combination of three + CoRs. (d) Positive linear combination of two + CoRs and a − CoR.

A nonnegative linear combination of these two velocities is written

\[ V = \text{pos}(\{V_1, V_2\}) = \{k_1V_1 + k_2V_2 \mid k_1, k_2 \geq 0\}, \]

a planar twist cone rooted at the origin, with \( V_1 \) and \( V_2 \) defining the edges of the cone. If \( \omega_1 \) and \( \omega_2 \), have the same sign, then the CoRs of their nonnegative linear combinations \( \text{CoR}(\text{pos}(\{V_1, V_2\})) \) all have that sign, and lie on the line segment between the two CoRs. If \( \text{CoR}(V_1) \) and \( \text{CoR}(V_2) \) are labeled + and −, respectively, then \( \text{CoR}(\text{pos}(\{V_1, V_2\})) \) consists of the line containing the two CoRs, minus the segment between the CoRs. This set consists of a ray of CoRs labeled + attached to \( \text{CoR}(V_1) \), a ray of CoRs labeled − attached to \( \text{CoR}(V_2) \), and a point at infinity labeled 0, corresponding to translation. This collection should be considered as a single line segment (though one passing through infinity), just like the first case. Figures 12.9 and 12.10 show examples of CoR regions corresponding to positive linear combinations of planar twists.

The CoR representation of planar twists is particularly useful for representing the feasible motions of one movable body in contact with stationary bodies. Since the constraints are stationary, as noted in Section 12.1.3, the feasible twists form a polyhedral convex cone rooted at the origin. Such a cone can be represented uniquely by a set of CoRs with +, −, and 0 labels. A general twist polytope, as generated by moving constraints, cannot be uniquely represented.
Grasping and Manipulation

Figure 12.11: The stationary triangle makes contact with a movable part. CoRs to the left of the contact normal are labeled $+$, to the right are labeled $-$, and on the normal labeled $\pm$. Also given are the contact types for the CoRs. For points on the contact normal, the sign assigned to $S_l$ and $S_r$ CoRs switches at the contact point. Three CoRs and labels are illustrated.

by a set of CoRs with $+$, $-$, and 0 labels.

Given a contact between a stationary body and a movable body, we can plot the CoRs that do not violate the impenetrability constraint. Label all points on the contact normal $\pm$, points to the left of the inward normal $+$, and points to the right $-$. All points labeled $+$ can serve as CoRs with positive angular velocity for the movable body, and all points labeled $-$ can serve as CoRs with negative angular velocity, without violating the first-order contact constraint. We can further assign contact labels to each CoR corresponding to the first-order conditions for breaking contact $B$, sliding contact $S$, and rolling contact $R$. For planar sliding, we subdivide the label $S$ into two subclasses: $S_r$, where the moving body slips right relative to the fixed constraint, and $S_l$, where the moving body slips to the left. Figure 12.11 illustrates the labeling.

If there is more than one contact, we simply union the constraints and contact labels from the individual contacts. The unioning of the constraints implies that the feasible CoR region is convex, as is the homogeneous polyhedral twist cone.

Example 12.4. Figure 12.12(a) shows a planar part standing on a table while being contacted by a stationary robot finger. The finger defines an inequality constraint on the part’s motion and the table defines two more. The cone of twists that do not violate the impenetrability constraints is represented by the CoRs that are consistently labeled for each contact (Figure 12.12(b)). Each feasible CoR is labeled with a contact mode that concatenates the labels for the individual contacts (Figure 12.12(c)).

Now look more closely at the CoR labeled $(+, S_r S_r)$ in Figure 12.12(c). Is this motion really possible? It should be apparent that it is, in fact, not
12.1. Contact Kinematics

Figure 12.12: Example 12.4. (a) A part resting on a table, with two contact constraints provided by the table and a single contact constraint provided by the stationary finger. (b) The feasible twists represented as CoRs, shown in gray. Note that the lines that extend off to the left and to the bottom “wrap around” at infinity and come back in from the right and the top, respectively, so this CoR region should be interpreted as a single connected convex region. (c) The contact modes assigned to each feasible motion. The zero velocity contact mode is $RRR$.

possible: the part would immediately penetrate the stationary finger. Our incorrect conclusion is due to the fact that our first-order analysis ignores the local contact curvature. A second-order analysis would show that this motion is impossible. On the other hand, if the radius of curvature of the part at the contact were sufficiently small, then the motion would be possible.

Thus a first-order analysis of a contact indicating roll-slide motion might be classified as penetrating or breaking by a second-order analysis. Similarly, if our second-order analysis indicates a roll-slide motion, a third or higher-order analysis may indicate penetration or breaking free. In any case, if an $n$th-order analysis indicates that the contact is breaking or penetrating, then no analysis of order greater than $n$ will change the conclusion.

12.1.7 Form Closure

Form closure of an object is achieved if a set of stationary constraints prevents all motion of the object. If these constraints are robot fingers, we call this a form-closure grasp. An example is shown in Figure 12.13.
12.1.7.1 Number of Contacts Needed for First-Order Form Closure

Each stationary contact $i$ provides a half-space twist constraint of the form

$$ F_i^T V \geq 0. $$

Form closure holds if the only twist $V$ satisfying the constraints is the zero twist. For $j$ contacts, this condition is equivalent to

$$ \text{pos}([F_1, \ldots, F_j]) = \mathbb{R}^6 $$

for parts in three dimensions. Therefore, by Fact 2 from the beginning of the chapter, at least $6 + 1 = 7$ contacts are needed for first-order form closure of spatial parts. For planar parts, the condition is

$$ \text{pos}([F_1, \ldots, F_j]) = \mathbb{R}^3, $$

and $3 + 1 = 4$ contacts are needed for first-order form closure. These results are summarized in the following theorem.

**Theorem 12.1.** For a planar part, at least four point contacts are needed for first-order form closure. For a spatial part, at least seven point contacts are needed.

Now consider the problem of grasping a circular disk in the plane. It should be clear that kinematically preventing motion of the disk is impossible regardless of the number of contacts; it will always be able to spin about its center. Such objects are called *exceptional*—the positive span of the contact normal forces at all points on the object is not equal to $\mathbb{R}^n$, where $n = 3$ in the planar case and $n = 6$ in the spatial case. Examples of such objects in three dimensions include surfaces of revolution, such as spheres and ellipsoids.

Figure 12.14 shows example planar grasps. The graphical methods of Section 12.1.6 indicate that the four contacts in Figure 12.14(a) immobilize the
12.1. Contact Kinematics

Figure 12.14: (a) Four fingers yielding planar form closure. The first-order analysis treats (b) and (c) identically, saying the triangle can rotate about its center in each case. A second-order analysis shows this is not possible for (b). The grasps in (d), (e), and (f) are identical by a first-order analysis, which says that rotation about any center on the vertical line is possible. This is true for (d), while only some of these centers are possible for (e). No motion is possible in (f).

To summarize, our first-order analysis always correctly labels breaking and penetrating motions, but second- and higher-order effects may change first-order roll-slide motions to breaking or penetrating. If an object is in form closure by the first-order analysis, it is in form closure for any analysis; if only roll-slide motions are feasible by the first-order analysis, the object could be in form closure by a higher-order analysis; and otherwise the object is not in form closure by any analysis.

12.1.7.2 A Linear Programming Test for First-Order Form Closure

Let \( F = [F_1 | F_2 | \ldots | F_j] \in \mathbb{R}^{n \times j} \) be a matrix whose columns are formed by the \( j \) contact wrenches. For spatial parts, \( n = 6 \), and for planar parts, \( n = 3 \) with \( F_i = (m_{ix}, f_{ix}, f_{iy})^T \). The contacts yield form closure if there exists a vector of weights \( k \in \mathbb{R}^j, k \geq 0 \) such that \( Fk + F_{\text{ext}} = 0 \) for all \( F_{\text{ext}} \in \mathbb{R}^n \).

Clearly the part is not in form closure if the rank of \( F \) is not full (\( \text{rank}(F) < n \)). If \( F \) is full rank, the form-closure condition is equivalent to the existence of strictly positive coefficients \( k > 0 \) such that \( Fk = 0 \). We can formulate this test...
as the following linear program:

\[
\begin{align*}
\text{find} & \quad k \\
\text{minimizing} & \quad 1^T k \\
\text{such that} & \quad Fk = 0 \\
& \quad k_i \geq 1, \quad i = 1, \ldots, j,
\end{align*}
\]

where \(1\) is a \(j\)-vector of ones. If \(F\) is full rank and there exists a solution \(k\) to (12.14), the part is in first-order form closure. Otherwise it is not. Note that the objective function \(1^T k\) is not necessary to answer the binary question, but it is included to make sure the problem is well posed, depending on the LP solver.

**Example 12.5.** The planar object in Figure 12.15 has a hole in the center. Two fingers each touch two different edges of the hole, creating four contact normals. The matrix \(F = [ F_1 \mid F_2 \mid F_3 \mid F_4 ]\) is

\[
F = \begin{bmatrix}
0 & 0 & -1 & 2 \\
-1 & 0 & 1 & 0 \\
0 & -1 & 0 & 1
\end{bmatrix}.
\]

The matrix \(F\) is clearly rank 3. The linear program of (12.14) returns a solution with \(k_1 = k_3 = 2, k_2 = k_4 = 1\), so the grasp is form closure. If the circular finger were moved to the bottom right corner of the hole, the new \(F\) matrix

\[
F = \begin{bmatrix}
0 & 0 & 0 & -2 \\
-1 & 0 & 1 & 0 \\
0 & -1 & 0 & -1
\end{bmatrix}
\]

is still full rank, but there is no solution to the linear program. This grasp is not form closure.

### 12.1.7.3 Measuring the Quality of a Form-Closure Grasp

Consider the two form-closure grasps shown in Figure 12.16. Which is a better grasp?
Answering this question requires a metric measuring the quality of a grasp. A grasp metric takes the set of contacts \( \{F_i\} \) and returns a single value \( \text{Qual}(\{F_i\}) \), where \( \text{Qual}(\{F_i\}) < 0 \) indicates that the grasp is not form closure, and larger positive values indicate better grasps.

There are many reasonable choices of grasp metric. As an example, suppose that to avoid damaging the object, we require the magnitude of the force at contact \( i \) be less than or equal to \( f_{i,\text{max}} > 0 \), which may be set to 1. Then the total set of contact wrenches that can be applied by the \( j \) contacts is given by

\[
CF = \left\{ \sum_{i=1}^j f_i F_i \mid f_i \in [0, f_{i,\text{max}}] \right\}.
\]  

(12.15)

See Figure 12.17 for an example in two dimensions. This is the convex set of wrenches that the contacts can apply to resist disturbance wrenches applied to the part. If the grasp is form closure, the set includes the origin of the wrench space in its interior.

Now the problem is to turn this polytope into a single number representing the quality of the grasp. Ideally this process would use some idea of the disturbance wrenches the part can be expected to experience. A simpler choice is to set \( \text{Qual}(\{F_i\}) \) to be the radius of the largest ball of wrenches, centered at the origin of the wrench space, that fits inside the convex polytope. In evaluating this radius, two caveats should be considered: (1) moments and forces have different units, so there is no obvious way to equate force and moment magnitudes, and (2) the moments due to contact forces depend on the choice of the location of the origin of the space frame. To address (1), it is common to choose a characteristic length \( r \) of the grasped part and convert contact moments \( m \) to forces \( m/r \). To address (2), the origin can be chosen somewhere near the geometric center of the part, or at its center of mass.

Given the choice of the space frame and the characteristic length \( r \), we simply calculate the signed distance from the origin of the wrench space to each hyperplane on the boundary of \( CF \). The minimum of these distances is \( \text{Qual}(\{F_i\}) \) (Figure 12.17).

Returning to our original example in Figure 12.16, we can see that if each finger is allowed to apply the same force, then the grasp on the left may be
considered the better grasp, as the contacts can resist greater moments about the center of the object.

### 12.1.7.4 Choosing Contacts for Form Closure

Many methods have been suggested for choosing form-closure contacts for fixturing or grasping. One approach is to sample candidate grasp points on the surface of the object (four for planar parts or seven for spatial parts) until a set is found yielding form closure. From there, the candidate grasp points may be incrementally repositioned according to gradient ascent using the grasp metric, i.e., $\frac{\partial \text{Qual}(p)}{\partial p}$, where $p$ is a vector of all the coordinates of the contact locations.\(^4\)

### 12.2 Contact Forces and Friction

#### 12.2.1 Friction

A commonly used model of friction in robotic manipulation is **Coulomb friction**. This experimental law states that the tangential friction force magnitude $f_t$ is related to the normal force magnitude $f_n$ by $f_t \leq \mu f_n$, where $\mu$ is called the **friction coefficient**. If the contact is sliding, or currently rolling but with incipient slip (i.e., at the next instant the contacts are sliding), then $f_t = \mu f_n$, and

---

\(^4\)The gradient vector $\frac{\partial \text{Qual}(p)}{\partial p}$ must be projected to the tangent planes at the points of contact to keep the contact locations on the surface of the object.
12.2. Contact Forces and Friction

Figure 12.18: (a) A friction cone illustrating all possible forces that can be transmitted through the contact. (b) A side view of the same friction cone showing the friction coefficient $\mu$ and the friction angle $\alpha = \tan^{-1} \mu$. (c) An inscribed polyhedral convex cone approximation to the circular friction cone.

The direction of the friction force is opposite of the sliding direction, i.e., friction dissipates energy. The friction force is independent of the speed of sliding.

Often two friction coefficients are defined, a static friction coefficient $\mu_s$ and a kinetic (or sliding) friction coefficient $\mu_k$, where $\mu_s \geq \mu_k$. This implies that a larger friction force is available to resist initial motion, but once motion has begun, the resisting force is smaller. Many other friction models have been developed with different functional dependencies on factors such as the speed of sliding and the duration of static contact before sliding. All of these are aggregate models of complex microscopic behavior. For simplicity, we use the simplest Coulomb friction model with a single friction coefficient $\mu$. This model is reasonable for hard, dry materials. The friction coefficient depends on the two materials in contact, and typically ranges from 0.1 to 1.

For a contact normal in the $+\hat{z}$ direction, the set of forces that can be transmitted through the contact satisfies

$$\sqrt{f_x^2 + f_y^2} \leq \mu f_z, \quad f_z \geq 0. \quad (12.16)$$

Figure 12.18(a) shows that this set of forces forms a friction cone. The set of forces that the finger can apply to the plane lies inside the cone shown. Figure 12.18(b) shows the same cone from a side view, illustrating the friction angle $\alpha = \tan^{-1} \mu$, which is the half-angle of the cone. If the contact is not sliding, the force may be anywhere inside the cone. If the finger slides to the right, the force it applies lies on the right edge of the friction cone, with a magnitude determined by the normal force. Correspondingly, the plane applies the opposing force to the finger, and the tangential (frictional) portion of this force opposes the sliding direction.

To allow linear formulations of contact mechanics problems, it is often convenient to represent the convex circular cone by a polyhedral convex cone. Fig-
Figure 12.18(c) shows an inscribed four-sided pyramidal approximation of the friction cone, defined by the positive span of the \((f_x, f_y, f_z)\) cone edges \((\mu, 0, 1), (-\mu, 0, 1), (0, \mu, 1),\) and \((0, -\mu, 1)\). We can obtain a tighter approximation to the circular cone by using more edges. An inscribed cone underestimates the friction forces available, while a circumscribed cone overestimates the friction forces. The choice of which to use depends on the application. For example, if we want to ensure that a robot hand can grasp an object, it is a good idea to underestimate the friction forces available.

For planar problems, no approximation is necessary—a friction cone is exactly represented by the positive span of the two edges of the cone, similar to the side view illustrated in Figure 12.18(b).

Once we choose a coordinate frame, any contact force can be expressed as a wrench \(F = ([p], f)\), where \(p\) is the contact location. This turns the friction cone into a wrench cone. A planar example is shown in Figure 12.19. The two edges of the planar friction cone give two rays in the wrench space, and the wrenches that can be transmitted to the part through the contact is the positive span of basis vectors along these edges. If \(F_1\) and \(F_2\) are basis vectors for these wrench cone edges, we write the wrench cone as \(WC = \text{pos}\{F_1, F_2\}\).

If multiple contacts act on a part, then the total set of wrenches that can be transmitted to the part through the contacts is the positive span of all the individual wrench cones \(WC_i\),

\[
WC = \text{pos}\{WC_i\} = \left\{ \sum k_i F_i \mid F_i \in WC_i, k_i \geq 0 \right\}.
\]

This composite wrench cone is a convex cone rooted at the origin. An example composite wrench cone is shown in Figure 12.19(d) for a planar object with the two friction cones shown in Figure 12.19(c). For planar problems, the composite wrench cone in the three-dimensional wrench space is polyhedral. For spatial problems, wrench cones in the six-dimensional wrench space are not polyhedral, unless the individual friction cones are approximated by polyhedral cones, as in Figure 12.18(c).

If a contact or set of contacts acting on a part is ideally force-controlled, the wrench \(F_{\text{cont}}\) specified by the controller must lie within the composite wrench cone corresponding to those contacts. If there are other non-force-controlled contacts acting on the part, then the cone of possible wrenches on the part is equivalent to the wrench cone from the non-force-controlled contacts, but translated to be rooted at \(F_{\text{cont}}\).

12.2.2 Planar Graphical Methods

12.2.2.1 Representing Wrenches

Any planar wrench \(F = (m_z, f_x, f_y)\) with a nonzero linear component can be represented as an arrow drawn in the plane, where the base of the arrow is at
12.2. Contact Forces and Friction

Figure 12.19: (a) A planar friction cone with friction coefficient $\mu$ and corresponding friction angle $\alpha = \tan^{-1}\mu$. (b) The corresponding wrench cone. (c) Two friction cones. (d) The corresponding composite wrench cone.

The point
\[(x, y) = \frac{1}{f_x^2 + f_y^2}(m_z f_y, -m_z f_x),\]
and the head of the arrow is at \((x + f_x, y + f_y)\). The moment is unchanged if we slide the arrow anywhere along the line of the arrow, so any arrow of the same direction and length along the line represents the same wrench (Figure 12.20). If $f_x = f_y = 0$ and $m_z \neq 0$, the wrench is a pure moment, and we do not try to represent it graphically.

Two wrenches, represented as arrows, can be summed graphically by sliding the arrows along their lines until the bases of the arrows are coincident. The arrow corresponding to the sum of the two wrenches is obtained as shown in Figure 12.20. The approach can be applied sequentially to sum multiple wrenches represented as arrows.
Grasping and Manipulation

Figure 12.20: (Left) The planar wrench $F = (m_z, f_x, f_y) = (2.5, -1, 2)$ represented as an arrow in the $\hat{x}$-$\hat{y}$ plane. (Middle) The same wrench can be represented by an arrow anywhere along the line of action. (Right) Two wrenches are summed by sliding their arrows along their lines of action until the bases of the arrows are coincident, then doing a vector sum by the parallelogram construction.

12.2.2.2 Representing Wrench Cones

In the previous section, each wrench had a specified magnitude. But a rigid-body contact implies that the contact normal force can be unbounded; it is whatever is needed to prevent two bodies from penetrating. Therefore it is useful to have a representation of all wrenches of the form $kF$, where $k \geq 0$ and $F \in \mathbb{R}^3$ is a basis vector.

One such representation is moment labeling. The arrow for the basis wrench $F$ is drawn as in Section 12.2.2.1. Then all points in the plane to the left of the line of the arrow are labeled ‘+,’ indicating that any positive scaling of $F$ creates a positive moment $m_z$ about those points, and all points in the plane to the right of the arrow are labeled ‘−,’ indicating that any positive scaling of $F$ creates a negative moment about those points. Points on the line are labeled ‘±.’

Generalizing, moment labels can represent any homogeneous convex planar wrench cone, much like a homogeneous convex planar twist cone can be represented as a convex CoR region. Given a collection of directed force lines corresponding to wrenches $k_iF_i$ for all $k_i \geq 0$, the wrench cone $\text{pos}([F_i])$ can be represented by labeling each point in the plane with a ‘+’ if each of the $F_i$ makes nonnegative moment about that point, a ‘−’ if each $F_i$ makes nonpositive moment about that point, a ‘±’ if each makes zero moment about that point, or a blank label if at least one wrench makes positive moment and at least one
12.2. Contact Forces and Friction

Figure 12.21: (a) Representing a line of force by moment labels. (b) Representing the positive span of two lines of force by moment labels. (c) The positive span of three lines of force.

wrench makes negative moment about that point.

The idea is best illustrated by an example. In Figure 12.21(a), the basis wrench $F_1$ is represented by labeling the points to the left of the line with a $+$ and points to the right of the line with a $-$. Points on the line are labeled $\pm$. In Figure 12.21(b), another basis wrench is added, which could represent the other edge of a planar friction cone. Only the points in the plane that are consistently labeled for both lines of force retain their labels; inconsistently labeled points lose their labels. Finally, a third basis wrench is added in Figure 12.21(c). The result is a single region labeled $+$. A nonnegative linear combination of the three basis wrenches can create any line of force in the plane that passes around this region in a counterclockwise sense. No other wrench can be created.

If an additional basis wrench were added passing clockwise around the region labeled $+$ in Figure 12.21(c), then there would be no consistently labeled point in the plane: the positive linear span of the four wrenches is the entire wrench space $\mathbb{R}^3$.

The moment-labeling representation is equivalent to a homogeneous convex wrench cone representation. The moment labeling regions in each of Figure 12.21(a), (b), and (c) is properly interpreted as a single convex region, much like the CoR regions of Section 12.1.6.

12.2.3 Force Closure

Consider a single movable object and a number of frictional contacts. We say the contacts result in **force closure** if the composite wrench cone contains the entire wrench space—any external wrench $F_{\text{ext}}$ on the object can be balanced by contact forces.

We can derive a simple linear test for force closure which is exact for planar
Grasping and Manipulation

Figure 12.22: An equilateral triangle can be force-closure grasped by two fingers on edges of the triangle if \( \mu \geq \tan 30^\circ \approx 0.577 \). (a) This grasp with \( \mu = 0.25 \) is not force closure, as indicated by the consistently labeled moment-labeling region. (b) This grasp is force closure with \( \mu = 1 \).

cases and approximate for spatial cases. Let \( F_i, i = 1 \ldots j \), be the wrenches corresponding to the edges of the friction cones for all the contacts. For planar problems, each friction cone contributes two edges, and for spatial problems, each friction cone contributes three or more edges, depending on the polyhedral approximation chosen (see Figure 12.18(c)). The columns of the \( n \times j \) matrix \( F \) are the \( F_i \), where \( n = 3 \) for planar problems and \( n = 6 \) for spatial problems. Now the test for force closure is identical to that for form closure. The contacts yield force closure if

- \( \text{rank}(F) = n \), and
- there exists a solution to the linear programming problem (12.14).

In the case of \( \mu = 0 \), each contact can provide forces only along the normal direction, and force closure is equivalent to first-order form closure.

12.2.3.1 Number of Contacts Needed for Force Closure

For planar problems, four contact wrenches are sufficient to positively span the three-dimensional wrench space, which means that as few as two frictional contacts (with two friction cone edges each) are sufficient for force closure. Using moment labeling, we see that force closure is equivalent to having no consistent moment labels. For example, if the two contacts can “see” each other by a line inside both friction cones, we have force closure (Figure 12.22).

It is important to note that force closure simply means that the contact friction cones can generate any wrench. It does not necessarily mean that the object will not move in the presence of an external wrench, however. For the example of Figure 12.22(b), whether the triangle falls under gravity or not depends on the internal forces between the fingers. If the motors powering the fingers cannot provide sufficient forces, or if they are restricted to only generate forces in certain directions, the triangle may fall despite force closure.
Two frictional point contacts are insufficient to yield force closure for spatial parts, as there is no way to generate moment about the axis joining the two contacts. A force-closure grasp can be obtained with as few as three frictional contacts, however. A particularly simple and appealing result due to Li et al. [65] reduces the force closure analysis of spatial frictional grasps to a planar force closure problem. Referring to Figure 12.23, suppose a rigid body is constrained by three point contacts with friction. If the three contact points happen to be collinear, then obviously any moment applied about this line cannot be resisted by the three contacts. We can therefore exclude this case, and assume that the three contact points are not collinear. The three contacts then define a unique plane \( S \), and at each contact point, three possibilities arise (see Figure 12.24):

- The friction cone intersects \( S \) in a planar cone;
- The friction cone intersects \( S \) in a line;
- The friction cone intersects \( S \) in a point.

The object is in force closure if and only if each of the friction cones intersects \( S \) in a planar cone, and \( S \) is also in planar force closure.

**Theorem 12.2.** Given a spatial rigid body restrained by three point contacts with friction, the body is in force closure if and only if the friction cone at each of the contacts intersects the plane \( S \) of the contacts in a cone, and the plane \( S \) is in planar force closure.
Proof. First, the necessity condition—if the spatial rigid body is in force closure, then each of the friction cones intersects \( S \) in a planar cone and \( S \) is also in planar force closure—is easily verified: if the body is in spatial force closure, then \( S \) (which is a part of the body) must also be in planar force closure. Moreover, if even one friction cone intersects \( S \) in a line or point, then there will be external moments (e.g., about the line between the remaining two contact points) that cannot be resisted by the grasp.

To prove the sufficiency condition—if each of the friction cones intersects \( S \) in a planar cone and \( S \) is also in planar force closure, then the spatial rigid body is in force closure—choose a fixed reference frame such that \( S \) lies in the \( x-y \) plane, and let \( r_i \in \mathbb{R}^3 \) denote the vector from the fixed frame origin to contact point \( i \) (see Figure 12.23). Denoting the contact force at \( i \) by \( f_i \in \mathbb{R}^3 \), the contact wrench \( F_i \in \mathbb{R}^6 \) is then of the form

\[
F_i = \begin{bmatrix} m_i \\ f_i \end{bmatrix},
\]

where each \( m_i = r_i \times f_i \), \( i = 1, 2, 3 \). Denote the arbitrary external wrench \( F_{\text{ext}} \in \mathbb{R}^6 \) by

\[
F_{\text{ext}} = \begin{bmatrix} m_{\text{ext}} \\ f_{\text{ext}} \end{bmatrix} \in \mathbb{R}^6.
\]

Force closure then requires that there exist contact wrenches \( F_i, i = 1, 2, 3 \), each lying inside its respective friction cone, such that for any external disturbance wrench \( F_{\text{ext}} \), the following equality is satisfied:

\[
F_1 + F_2 + F_3 + F_{\text{ext}} = 0,
\]

or equivalently,

\[
f_1 + f_2 + f_3 + f_{\text{ext}} = 0 \quad (12.20)
\]

\[
(r_1 \times f_1) + (r_2 \times f_2) + (r_3 \times f_3) + m_{\text{ext}} = 0 \quad (12.21)
\]

If each of the contact forces and moments, as well as the external force and moment, is orthogonally decomposed into components lying on the plane spanned by \( S \) (corresponding to the \( x-y \) plane in our chosen reference frame) and its normal subspace \( N \) (corresponding to the \( z \)-axis in our chosen reference frame), then the previous force closure equality relations can be written

\[
f_{1S} + f_{2S} + f_{3S} = -f_{\text{ext},S} \quad (12.22)
\]

\[
(r_1 \times f_{1S}) + (r_2 \times f_{2S}) + (r_3 \times f_{3S}) = -m_{\text{ext},S} \quad (12.23)
\]

\[
f_{1N} + f_{2N} + f_{3N} = -f_{\text{ext},N} \quad (12.24)
\]

\[
(r_1 \times f_{1N}) + (r_2 \times f_{2N}) + (r_3 \times f_{3N}) = -m_{\text{ext},N} \quad (12.25)
\]

In what follows we shall use \( S \) to refer both to the slice of the rigid body corresponding to the \( x-y \) plane, as well as the \( x-y \) plane itself. \( N \) will always be identified with the \( z \)-axis.
Proceeding with the proof of sufficiency, we now show that if \( S \) is in planar force closure, then the body is in spatial force closure. In terms of Equations (12.24)-(12.25) we wish to show that, for any arbitrary forces \( f_{\text{ext},S} \in S \), \( f_{\text{ext},N} \in N \) and arbitrary moments \( m_{\text{ext},S} \in S \), \( m_{\text{ext},N} \in N \), there exist contact forces \( f_{iS} \in S \), \( f_{iN} \in N \), \( i = 1, 2, 3 \), that satisfy (12.24)-(12.25), and such that for each \( i = 1, 2, 3 \), the contact force \( f_i = f_{iS} + f_{iN} \) lies in friction cone \( i \).

First consider the force closure equations (12.24)-(12.25) in the normal direction \( N \). Given an arbitrary external force \( f_{\text{ext},N} \in N \) and external moment \( m_{\text{ext},S} \in S \), Equations (12.24)-(12.25) constitute a set of three linear equations in three unknowns. From our assumption that the three contact points are never collinear, these equations will always have a unique solution set \( \{ f^*_1N, f^*_2N, f^*_3N \} \) in \( N \).

Since \( S \) is assumed to be in planar force closure, for any arbitrary \( f_{\text{ext},S} \in S \) and \( m_{\text{ext},N} \in N \), there will exist planar contact forces \( f_{iS} \in S \), \( i = 1, 2, 3 \), that lie inside their respective planar friction cones and also satisfy Equations (12.22)-(12.23). This solution set is not unique: one can always find a set of internal forces \( \eta_i \in S \), \( i = 1, 2, 3 \), each lying inside its respective friction cone, satisfying

\[
\eta_1 + \eta_2 + \eta_3 = 0 \quad (12.26)
\]

\[
(r_1 \times \eta_1) + (r_2 \times \eta_2) + (r_3 \times \eta_3) = 0. \quad (12.27)
\]

(To see why such \( \eta \) exist, recall that since \( S \) is assumed to be in planar force closure, solutions to (12.22)-(12.23) must exist for \( f_{\text{ext},S} = \mu_{\text{ext},N} = 0 \); these solutions are precisely the internal forces \( \eta_i \)). Note that these two equations constitute three linear equality constraints involving six variables, so that there exists a three-dimensional linear subspace of solutions for \( \{ \eta_1, \eta_2, \eta_3 \} \).

Now if \( \{ f_{1S}, f_{2S}, f_{3S} \} \) satisfy (12.22)-(12.23), then so will \( \{ f_{1S} + \eta_1, f_{2S} + \eta_2, f_{3S} + \eta_3 \} \). The internal forces \( \{ \eta_1, \eta_2, \eta_3 \} \) can in turn be chosen to have sufficiently large magnitude so that the contact forces

\[
f_1 = f^*_1N + f_{1S} + \eta_1 \quad (12.28)
\]

\[
f_2 = f^*_2N + f_{2S} + \eta_2 \quad (12.29)
\]

\[
f_3 = f^*_3N + f_{3S} + \eta_3 \quad (12.30)
\]

all lie inside their respective friction cone. This completes the proof of the sufficiency condition.

12.2.3.2 Measuring the Quality of a Force-Closure Grasp

Friction forces are not always repeatable. For example, try putting a coin on a book and tilting the book. The coin should begin to slide when the book is at an angle \( \alpha = \tan^{-1} \mu \) with respect to horizontal. If you do the experiment several times, you may find a range of measured values of \( \mu \), however, due to effects that are difficult to model. For that reason, when choosing between grasps, it is reasonable to choose finger locations that minimize the friction coefficient needed to achieve force closure.
12.2.4 Duality of Force and Motion Freedoms

Our discussion of kinematic constraints and friction should make apparent that, for any point contact and contact label, the number of equality constraints on the part’s motion caused by that contact is equal to the number of wrench freedoms it provides. For example, a breaking contact $B$ provides zero equality constraints on the part motion and also allows no contact force. A fixed contact $R$ provides 3 motion constraints (the motion of a point on the part is specified) and 3 freedoms on the contact force: any wrench in the interior of the contact wrench cone is consistent with the contact mode. Finally, a slipping contact $S$ provides 1 equality motion constraint (one equation on the part’s motion must be satisfied to maintain the contact), and for a given motion satisfying the constraint, the contact wrench has only 1 freedom, the magnitude of the contact wrench on the edge of the friction cone and opposing the slipping direction. In the planar case, the motion constraints and wrench freedoms for $B$, $S$, and $R$ contacts are 0, 1, and 2, respectively.

12.3 Manipulation

So far we have studied the feasible twists and contact forces due to a set of contacts. We have also considered two types of manipulation: form- and force-closure grasping.

Manipulation consists of much more than just grasping, however. It includes almost anything where manipulators impose motions or forces with the purpose of achieving motion or restraint of objects. Examples include carrying glasses on a tray without toppling them, pivoting a refrigerator about a foot, pushing a sofa on the floor, throwing and catching a ball, transporting parts on a vibratory conveyor, etc. Endowing a robot with methods of manipulation beyond grasp-and-carry allows it to manipulate several parts simultaneously; manipulate parts that are too large to be grasped or too heavy to be lifted; or even to send parts outside the workspace of the end-effector by throwing.

To plan such manipulation tasks, we use the contact kinematic constraints of Section 12.1, the Coulomb friction law of Section 12.2, and the dynamics of rigid bodies. Restricting ourselves to a single rigid body and using the notation of Chapter 8, the part’s dynamics are written

$$\mathbf{F}_{\text{ext}} + \sum k_i \mathbf{F}_i = \mathbf{G} \dot{\mathbf{V}} - [\mathbf{ady}]^T \mathbf{G} \mathbf{V}, \quad k_i \geq 0, \quad \mathbf{F}_i \in \mathbf{WC}_i,$$

(12.31)

where $\mathbf{V}$ is the part’s twist, $\mathbf{G}$ is its spatial inertia matrix, $\mathbf{F}_{\text{ext}}$ is the external wrench acting on the part due to gravity, etc., $\mathbf{WC}_i$ is the set of possible wrenches acting on the object due to contact $i$, and $\sum k_i \mathbf{F}_i$ is the wrench due to the contacts, and all wrenches are written in the part’s center of mass frame. Now, given a set of motion- or force-controlled contacts acting on the part, and the initial state of the system, one method for solving for the motion of the part is the following:
(i) Enumerate the set of possible contact modes considering the current state of the system (e.g., a contact that is currently sticking can transition to sliding or breaking). The contact modes consist of the contact labels R, S, and B at each contact.

(ii) For each contact mode, determine if there exists a contact wrench $\sum k_i F_i$ that is consistent with the contact mode and Coulomb’s law, and an acceleration $\dot{V}$ consistent with the kinematic constraints of the contact mode, such that Equation (12.31) is satisfied. If so, this contact mode, contact wrench, and part acceleration is a consistent solution to the rigid-body dynamics.

This kind of “case analysis” may sound unusual; we are not simply solving a set of equations. It also leaves open the possibility that we could find more than one consistent solution, or perhaps no consistent solution. This is, in fact, the case: we can define problems with multiple solutions (ambiguous) and problems with no solutions (inconsistent). This state of affairs is a bit unsettling; surely there is exactly one solution to any real mechanics problem! But this is the price we pay to use the assumptions of perfectly rigid bodies and Coulomb friction. Also, for many problems the method described above will yield a unique contact mode and motion.

In this chapter we do not provide tools to solve the general simulation problem. Instead, we focus on manipulation problems that permit solutions using our first-order kinematic analysis. We also consider quasistatic problems, where the velocities and accelerations of the parts are small so that inertial forces may be ignored. Contact wrenches and external wrenches are always in force balance, and Equation (12.31) reduces to

$$F_{\text{ext}} + \sum k_i F_i = 0, \quad k_i \geq 0, \quad F_i \in W_{C_i}. \quad (12.32)$$

Below we illustrate the methods of this chapter with several examples.

**Example 12.6. A block carried by two fingers.**

Consider a planar block in gravity supported by two fingers, as in Figure 12.25(a). The friction coefficient between one finger and the block is $\mu = 1$, and the other contact is frictionless. Thus the cone of wrenches that can be applied by the fingers is $\text{pos} \{ F_1, F_2, F_3 \}$, as shown using moment labeling in Figure 12.25(b).

Our first question is whether the stationary fingers can keep the block at rest. To do so, the fingers must provide a wrench $F = (m_x, f_x, f_y) = (0, 0, mg)$ to balance $F_{\text{ext}} = (0, 0, -mg)$ due to gravity, where $g > 0$. As shown in Figure 12.25(b), however, this wrench is not in the composite cone of possible contact wrenches. Therefore the contact mode RR is not feasible, and the block will move relative to the fingers.

Now consider the case where the fingers each accelerate to the left at $2g$. In this case, the contact mode RR requires that the block also accelerate to the left at $2g$. The wrench needed to cause this acceleration is $(0, -2mg, 0)$. Therefore the total wrench that the fingers must apply to the block is $(0, -2mg, 0) - F_{\text{ext}} =
Grasping and Manipulation

Figure 12.25: (a) A planar block in gravity supported by two robot fingers, one with a friction cone with $\mu = 1$ and one with $\mu = 0$. (b) The composite wrench cone that can be applied by the fingers represented using moment labels. To balance the block against gravity, the fingers must apply the line of force shown. This line makes positive moment with respect to some points labeled $-$, and therefore it cannot be generated by the two fingers. (c) For the block to match the fingers’ acceleration to the left, the contacts must apply the vector sum of the wrench to balance gravity plus the wrench needed to accelerate the block to the left. This total wrench lies inside the composite wrench cone, as the line of force makes positive moment with respect to the points labeled $+$ and negative moment with respect to the points labeled $-$. (d) The total wrench applied by the fingers in Figure (c) can be translated along the line of action without changing the wrench. This allows us to easily visualize the components $k_1 F_1 + k_2 F_2$ and $k_3 F_3$ provided by the fingers.

$(0, -2mg, mg)$. As shown in Figures 12.25(c) and (d), this wrench lies inside the composite wrench cone. Thus RR (the block stays stationary relative to the fingers) is a solution as the fingers accelerate to the left at $2g$.

This is called a dynamic grasp—inertial forces are used to keep the block pressed against the fingers while the fingers move. If we plan to manipulate the block using a dynamic grasp, we should make certain that no contact modes other than RR are feasible, for completeness.

Moment labels are convenient for understanding this problem graphically,
12.3. Manipulation

but we can also solve it algebraically. Finger one contacts the block at \((x, y) = (-3, -1)\) and finger 2 contacts the block at \((1, 1)\). This gives the basis contact wrenches

\[
\mathcal{F}_1 = \frac{1}{\sqrt{2}} (-4, -1, 1)^T \\
\mathcal{F}_2 = \frac{1}{\sqrt{2}} (-2, 1, 1)^T \\
\mathcal{F}_3 = (1, -1, 0)^T.
\]

Let the fingers’ acceleration in the \(x\) direction be written \(a_x\). Then, under the assumption that the block stays fixed to the fingers (\(RR\) contact mode), Equation (12.31) can be written

\[
k_1\mathcal{F}_1 + k_2\mathcal{F}_2 + k_3\mathcal{F}_3 + (0, 0, -mg) = (0, ma_x, 0).
\]

This yields three equations in the three unknowns, \(k_1, k_2, k_3\). Solving, we get

\[
k_1 = -\frac{1}{2\sqrt{2}} (a_x + g)m, \quad k_2 = \frac{1}{2\sqrt{2}} (a_x + 5g)m, \quad k_3 = -\frac{1}{2} (a_x - 3g)m.
\]

For the \(k_i\) to be nonnegative, we need \(-5g \leq a_x \leq -g\). For \(x\)-direction finger accelerations in this range, a dynamic grasp is a consistent solution.

Example 12.7. The meter stick trick.

Try this experiment: Get a meter stick (or any similar long smooth stick) and balance it horizontally on your two index fingers. Place your left finger near the 10 cm mark and your right finger near the 60 cm mark. The center of mass is closer to your right finger, but still between your fingers, so that the stick is supported. Now, keeping your left finger stationary, slowly move your right finger towards your left until they touch. What happens to the stick?

If you didn’t try the experiment, you might guess that your right finger passes under the center of mass of the stick, at which point the stick falls. If you did try the experiment, you saw something different. Let’s see why.

Figure 12.26 shows the stick supported by two frictional fingers. Since all motions are slow, we use the quasistatic approximation that the stick’s acceleration is zero, and the net contact wrench must balance the gravitational wrench. As the two fingers move together, the stick must slip on one or both fingers to accommodate the fact that the fingers are getting closer to each other. Figure 12.26 shows the moment-labeling representation of the composite wrench cone for three different contact modes where the stick remains stationary on the left finger (\(R\)) or slips left relative to it (\(S1\)) while either remaining stationary on the right finger (\(R\)) or slipping right relative to it (\(Sr\)). It is clear from the figure that only the \(S1R\) contact mode can provide a wrench that balances the gravitational wrench. In other words, the right finger, which supports more of the stick’s weight, remains sticking, while the left finger slides under the stick. Since the right finger is moving to the left in the world frame, this means the center of mass is moving to the left at the same speed. This continues until the
center of mass is halfway between the fingers, at which point the stick transitions to the \( S1Sr \) contact mode, and the center of mass stays centered between the fingers until they meet. The stick never falls.

Note that this analysis relies on the quasistatic assumption. It is easy to make the stick fall if you move your right finger quickly; the friction force at the right finger is not large enough to create the large stick acceleration needed to maintain a sticking contact. Also, in your experiment, you might notice that when the center of mass is nearly centered, the stick does not actually achieve the idealized \( S1Sr \) contact mode, but instead switches rapidly between the \( S1R \) and \( Rsr \) contact modes. This is because the static friction coefficient is larger than the kinetic friction coefficient.

**Example 12.8. Stability of an assembly.**
Consider the arch in Figure 12.27. Is it stable under gravity?

For a problem like this, graphical planar methods are difficult to use, since there are potentially multiple moving parts. Instead we test algebraically for consistency of the contact mode with all contacts labeled R. The friction cones are shown in Figure 12.27. With these labelings of the friction cone edges, the arch remaining standing is a consistent solution if there exist \( k_i \geq 0 \) for \( i = 1 \ldots 16 \) satisfying the following nine wrench-balance equations, three for each body:

\[
\begin{align*}
\sum_{i=1}^{8} k_i F_i + F_{ext1} &= 0 \\
\sum_{i=9}^{16} k_i F_i + F_{ext2} &= 0 \\
- \sum_{i=5}^{12} k_i F_i + F_{ext3} &= 0.
\end{align*}
\]

The last set of equations comes from the fact that wrenches that body 1 applies to body 3 are equal and opposite those that body 3 applies to body 1, and similarly for bodies 2 and 3.

This linear constraint satisfaction problem can be solved by a variety of methods, including linear programming.

**Example 12.9. Peg insertion.**

Figure 12.28 shows a force-controlled planar peg in two-point contact with a hole during insertion. Also shown are the contact friction cones acting on the peg and the corresponding composite wrench cone, illustrated using moment labels. If the force controller applies the wrench \( F_1 \) to the peg, it may jam—the hole may generate contact forces that balance \( F_1 \). Therefore the peg may get stuck in this position. If the force controller applies the wrench \( F_2 \), however, the contacts cannot balance the wrench and insertion proceeds.

If the friction coefficients at the two contacts are large enough that the two friction cones “see” each others’ base, the peg is in force closure, and the contacts may be able to resist any wrench (depending on the internal force between the two contacts). The peg is said to be wedged.
12.4 Summary

- Three ingredients are needed to solve rigid-body contact problems with friction: (1) contact kinematics, which describe the feasible motions of rigid bodies in contact; (2) a contact force model, which describes what forces can be transmitted through frictional contacts; and (3) rigid-body dynamics, as described in Chapter 8.

- Let two rigid bodies, $A$ and $B$, be in point contact at $p_A$ in a space frame. Let $\hat{n} \in \mathbb{R}^3$ be the unit contact normal, pointing into body $A$. Then the spatial contact wrench $\mathcal{F}$ associated with a unit force along the contact normal is $\mathcal{F} = (([p_A]\hat{n})^T \hat{n})^T$. The impenetrability constraint is $\mathcal{F}^T(V_A - V_B) \geq 0$, where $V_A$ and $V_B$ are the spatial twists of $A$ and $B$.

- A contact that is sticking or rolling is assigned the contact label $R$, a contact that is sliding is assigned the contact label $S$, and a contact that is breaking free is given the contact label $B$. For a body with multiple contacts, the contact mode is the concatenation of the labels of the individual contacts.

- A single rigid body subjected to multiple stationary point contacts has a homogeneous (rooted at the origin) polyhedral convex cone of twists that satisfy all the impenetrability constraints.

- A homogeneous polyhedral convex cone of planar twists in $\mathbb{R}^3$ can be equivalently represented by a convex region of signed rotation centers in the plane.

- If a set of stationary contacts prevents an object from moving, purely by a kinematic analysis considering only the contact normals, the object is
The contact wrenches $\mathcal{F}_i$ for contacts $i = 1 \ldots j$ positively span $\mathbb{R}^n$, where $n = 3$ for the planar case and $n = 6$ for the spatial case.

- At least four point contacts are required for first-order form closure of a planar part, and at least seven point contacts are required for first-order form closure of a spatial part.

- The Coulomb friction law states that the tangential frictional force magnitude $f_t$ at a contact satisfies $f_t \leq \mu f_n$, where $\mu$ is the friction coefficient and $f_n$ is the normal force. When the contact is sticking, the frictional force can be anything satisfying this constraint. When the contact is sliding, $f_t = \mu f_n$, and the direction of the friction force opposes the direction of sliding.

- Given a set of frictional contacts acting on a body, the wrenches that can be transmitted through these contacts is the positive span of the wrenches that can be transmitted through the individual contacts. These wrenches form a homogeneous convex cone. If the body is planar, or if the body is spatial but the contact friction cones are approximated by polyhedral cones, the wrench cone is also polyhedral.

- A homogeneous convex cone of planar wrenches in $\mathbb{R}^3$ can be represented as a convex region of moment labels in the plane.

- An object is in force closure if the homogeneous convex cone of contact wrenches from the stationary contacts is the entire wrench space ($\mathbb{R}^3$ or $\mathbb{R}^6$). If the contacts are frictionless, force closure is equivalent to first-order form closure.

### Notes and References

The kinematics of contact draw heavily from concepts in linear algebra (see, for example, the texts [131, 87]) and, more specifically, screw theory [4, 93, 13, 1, 86]. Graphical methods for analysis of planar constraints were introduced by Reuleaux [104], and Mason introduced graphical construction of contact labels for planar kinematics and moment labels for representation of homogeneous wrench cones [81, 82]. Polyhedral convex cones, and their application in representing feasible twist cones and contact wrench cones, are discussed in [82, 46, 30, 37]. The formalization of the friction law used in this chapter was given by Coulomb in 1781 [21]. Surprising consequences of Coulomb friction are problems of ambiguity and inconsistency [69, 82, 85] and that infinite friction does not necessarily prevent slipping at an active contact [75].

Form closure and force closure are discussed in detail in the Handbook of Robotics [100]. In particular, that reference uses the term “frictional form closure” to mean the same thing that “force closure” means in this chapter. According to [100], force closure additionally requires that the hand doing the
Grasping and Manipulation

grasping be sufficiently capable of controlling the internal “squeezing” forces. Similar distinctions are made in [9] and the reviews [11, 10]. In this chapter we do not consider the details of the robot hand and adopt a definition of force closure based solely on the geometry and friction of the contacts.

The numbers of contacts needed for planar and spatial form closure were established by Reuleaux [104] and Somoff [127], respectively. Other foundational results in form and force closure grasping are developed in [56, 89, 78] and are reviewed in [10, 100]. An overview of grasp quality metrics is given in [100]. The result that two friction cones that can “see” each other’s base are sufficient for planar force closure was first reported in [92], and the result reviewed in this chapter on three-finger force-closure grasps in 3D appeared in [65]. Salisbury applied Grüber’s formula to calculate the mobility of a grasped object using kinematic models of contact like those in Table 12.1 [84].

Second-order models of contact constraints were introduced by Rimon and Burdick [106, 105, 107, 108] and used to show that curvature effects allow form closure by fewer contacts.

Jamming and wedging in robotic insertion were described in [125, 91, 139], and the notion of a dynamic grasp was first introduced in [83].

An important class of methods for simulating systems of rigid bodies in frictional contact, not covered in this chapter, are based on solving linear and nonlinear complementarity problems [130, 94, 135]. These complementarity formulations directly encode the fact that if a contact is breaking, then no force is applied; if a contact is sticking, then the force can be anywhere inside the friction cone; and if a contact is sliding, the force is on the edge of the friction cone.

General references on contact modeling and manipulation include Handbook of Robotics chapters [46, 100] and the texts by Mason [82] and Murray et al. [90].
12.6 Exercises

1. Prove that the impenetrability constraint (12.4) is equivalent to the constraint (12.7).

2. (a) Consider the two planar twists $\mathcal{V}_1 = (\omega z_1, v_{x1}, v_{y1}) = (1, 2, 0)$ and $\mathcal{V}_2 = (\omega z_2, v_{x2}, v_{y2}) = (1, 0, -1)$. Draw the corresponding CoRs in a planar coordinate frame, and illustrate $\text{pos}(\{\mathcal{V}_1, \mathcal{V}_2\})$ as CoRs. (b) Draw the positive span of $\mathcal{V}_1 = (\omega z_1, v_{x1}, v_{y1}) = (1, 2, 0)$ and $\mathcal{V}_2 = (\omega z_2, v_{x2}, v_{y2}) = (-1, 0, -1)$ as CoRs.

3. A rigid body is contacted at $p = (1, 2, 3)$ with a contact normal into the object $\hat{n} = (0, 1, 0)$. Write the constraint on the body’s twist $\mathcal{V}$ due to this contact.

4. A space frame $\{s\}$ is defined at a contact between a stationary constraint and an object. The contact normal, into the object, is the $z$-axis of the $\{s\}$ frame. (a) Write the constraint on the object’s twist $\mathcal{V}$ if the contact is a frictionless point contact. (b) Write the constraints on $\mathcal{V}$ if the contact is a point contact with friction. (c) Write the constraints on $\mathcal{V}$ if the contact is a soft contact.

5. Figure 12.29 shows five stationary “fingers” contacting an object. The object is in first-order form closure and therefore force closure. If we take away one finger, the object may still be in form closure. For which subsets of four fingers is the object still in form closure? Prove your answers using graphical methods.

6. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 1 and 2. Label the feasible CoRs with their contact.
Grasping and Manipulation

7. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 2 and 3. Label the feasible CoRs with their contact labels.

8. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 2 and 3. Label the feasible CoRs with their contact labels.

9. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 1 and 5. Label the feasible CoRs with their contact labels.

10. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 1, 2, and 3.

11. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 1, 2, and 4.

12. Draw the set of feasible twists as CoRs when the triangle of Figure 12.29 is contacted only by fingers 1, 3, and 5.

13. Refer to the triangle of Figure 12.29. (a) Draw the wrench cone from contact 5, assuming a friction angle $\alpha = 22.5^\circ$ (a friction coefficient $\mu = 0.41$), using moment labeling. (b) Add contact 2 to the moment-labeling drawing. The friction coefficient at contact 2 is $\mu = 1$.

14. Refer to the triangle of Figure 12.29. Draw the moment-labeling region corresponding to contact 1 with $\mu = 1$ and contact 4 with $\mu = 0$.

15. (a) The planar grasp of Figure 12.30 consists of five frictionless point contacts. The square is of size $4 \times 4$. Show that this grasp is not force closure. (b) The grasp of part (a) can be made force closure by adding one frictionless point contact. Draw all the possible locations for this contact.

16. Assume all contacts shown in Figure 12.31 are frictionless point contacts. Determine whether the grasp is force closure. If it is not, how many additional frictionless point contacts are needed to construct a force closure grasp?

17. (a) In Figure 12.32-(a), the planar object with two rectangular holes is restrained from the interior by four frictionless point contacts. Determine whether this grasp is force closure.
12.6. Exercises

439

Figure 12.30: A $4 \times 4$ planar square restrained by five frictionless point contacts.

Figure 12.31: A planar disk restrained by three frictionless point contacts.

(b) In Figure 12.32-(b), the same planar object with two rectangular holes is now restrained from the interior by three frictionless point contacts. Determine whether this grasp is force closure.

18. The planar object of Figure 12.33 is restrained by four frictionless point contacts.

(a) Determine if this grasp is force closure.

(b) Suppose the contacts A, B, C, D are now allowed to slide along the half-circle (without crossing each other). Describe the set of all possible force closure grasps.

19. (a) Determine whether the grasp of Figure 12.34-(a) is force closure. Assume all contacts are frictionless point contacts. If the grasp is not force closure, slide the position of one of the contacts in order to construct a force closure...
(a) Now place two frictionless point contacts at the corners as shown in Figure 12.34-(b). Determine if this grasp is force closure.

(b) In the grasp of Figure 12.34-(c), contact A is a point contact with friction (its friction cone is 90° as indicated in the figure), while contacts B and C are frictionless point contacts. Determine whether this grasp is force closure.

20. Determine whether the grasp of Figure 12.35 is force closure. Assume all contacts are point contacts with a friction coefficient $\mu = 1$. 

Figure 12.32: A planar rigid body with two square holes.

Figure 12.33: A planar wedge-like object with a hole.
21. (a) In the planar triangle of Figure 12.36-(a), contacts A, B, and C are all frictionless point contacts. Is this grasp force closure? If not, what type of external force or moment would cause the object to slip out of the grasp? 
(b) In the planar triangle of Figure 12.36-(a), suppose now that contact A is a point contact with friction coefficient \( \mu = 1 \), while contacts B and C are frictionless point contacts. Determine whether the grasp shown is force closure. 
(c) Now suppose contact point A can be moved to anywhere on the hypotenuse of the triangle as shown in Figure 12.36-(b). Determine the range of all contact points A for which the grasp is force closure. 
(d) Contact points B and C are now moved as shown in Figure 12.37. Bill, a clever student, argues that the two contacts B and C can be replaced by a virtual point contact with friction (point D) with the given friction cone, and
Nguyen’s condition for force closure can now be applied to A and D, as shown in the right Figure 12.37. Is Bill correct? Justify your answer with a derivation of the appropriate force closure condition.

22. Consider the L-shaped planar object of Figure 12.38.
(a) Suppose both contacts are point contacts with friction coefficient $\mu = 1$. Determine whether this grasp is force closure.
(b) Now suppose point contact 1 is a point contact with friction coefficient $\mu = 1$, while point contact 2 is frictionless. Determine whether this grasp is
12.6. Exercises

Figure 12.37: A planar triangle constrained by three point contacts.

Figure 12.38: An L-shaped planar object restrained by two point contacts with friction.

(c) Suppose now that the vertical position of contact 1 is allowed to vary; denote its height by $x$. Find all positions $x$ such that the grasp of part (b) is force closure.

23. A square is restrained by three point contacts as shown in Figure 12.39: contact $f_1$ is a point contact with friction coefficient $\mu$, while contacts $f_2$ and $f_3$ are frictionless point contacts. If $c = \frac{1}{4}$ and $h = \frac{1}{2}$, find the range of values of $\mu$ such that grasp is force closure.

24. In the grasp of Figure 12.40, contacts $f_1$ and $f_2$ on the left are frictionless point contacts, while contact $f_3$ on the right is a point contact with friction coefficient $\mu = 0.2$. Determine whether this grasp is force closure.

25. A single point contact with friction coefficient $\mu = 1$ is applied to the left side of the square doughnut as shown in Figure 12.41. A force closure grasp can
be constructed by adding another point contact with friction, also with \( \mu = 1 \). Draw all possible locations for this additional point contact.

**26.** (a) In the planar grasp of Figure 12.42(a), contacts A and B are frictionless, while contact C has friction, with friction cone as shown given by the angle \( \beta \). Find the smallest value of \( \beta \) that makes this grasp a force closure grasp.

(b) Now suppose contacts A and B have friction, with 90° friction cones as
27. The square object of Figure 12.43 has four holes as shown. The four holes form parts of a circle.
(a) In the planar grasp of Figure 12.43(a), contacts $A$, $B$, and $C$ are all frictionless point contacts. Suppose contact $B$ is allowed to slide along any of the three sides of the hole as shown in Figure 12.43(a). Find all possible locations for $B$ that make this grasp force closure.
(b) In the planar grasp of Figure 12.43(b), contacts $A$ and $B$ are frictionless point contacts, while contact $C$ is a point contact with friction cone (defined by angle $\beta$) as shown. Determine the range of angle $\beta$ so that the grasp is force closure.
(c) In the planar grasp of Figure 12.43(c), contacts $A$ and $B$ are frictionless point
contacts, while contact \( C \) is a point contact with friction cone defined by angle \( \beta \). \( \beta \) lies in the range \( 0 < \beta < \frac{\pi}{4} \). Angles \( \alpha \) and \( \theta \) are defined as shown in Figure 12.43(c), with \( \theta \) satisfying \( \alpha < \theta < \frac{\pi}{2} - \alpha \). Find the range of values for angles \( \beta \) and \( \theta \) such that the grasp is force closure. (Hint: Nguyen’s Theorem is useful).

28. Consider a planar donut-like rigid object shown in Figure 12.44(a).
(a) Does there exist any set of four frictionless point contacts that makes this grasp force closure?
(b) Point contacts B and C, both with friction coefficient \( \mu > 0 \), are placed as shown in Figure 12.44(b). The position of contact C is variable, so that the angle \( \theta = \angle BOC \) can vary. Find the range of \( \theta \) that makes this grasp force closure in terms of \( \mu \).

29. In Figure 12.45, the cross-shaped object with four circular holes is re-
strained from the interior by four point contacts \((A, B, C, D)\). The distance from the center of the cross to each hole is \(L\), and the radius of each hole is \(r\). Angles \(\theta_1, \theta_2, \theta_3, \theta_4\) are defined as shown.

(a) Let \(\theta_1 = \frac{7}{3}\pi, \theta_2 = \frac{1}{2}\pi, \theta_3 = \frac{3}{2}\pi, \theta_4 = \frac{3}{2}\pi\) and assume \(A, B, C, D\) are frictionless point contacts. Is this grasp form closure?

(b) Suppose \(\theta_1 = 0, \theta_3 = 0\), and \(A, B, C, D\) are frictionless point contacts. For what values of \(\theta_2\) and \(\theta_4\) is the grasp form closure?

(c) Now suppose there is no contact at point \(D\) (there are only 3 contact points).
A and C are frictionless point contacts, whereas B is a point contact with friction coefficient $\mu = 1$ as shown in Figure 12.45(b). Suppose $L$ is large enough to ignore $r$ (i.e., $L \gg r$) and $\theta_2 = 0$. Find all values of $\theta_1$ and $\theta_3$ that makes the grasp force closure.

30. (a) For the planar grasp of Figure 12.46(a), assume contact C is frictionless, while contacts A and B have friction (assume $\mu=1$). Determine whether this grasp is force closure.

(b) For the planar grasp of Figure 12.46(b), assume contacts A and B are frictionless, while contact C has a friction cone of angle $\beta$. Find the range of values of $\beta$ that makes this grasp force closure.
31. The object of Figure 12.47 consists of 4 semicircular petals with a circular hole in the middle. Assume the radius is $R$ for all semicircles.
(a) In the planar grasp of Figure 12.47(a), contacts A, B, C and D are frictionless. Determine whether the grasp is force closure.
(b) In the planar grasp of Figure 12.47(b), contacts C and D are frictionless, while contact E has friction with friction coefficient $\mu = 1$. Find the range of angles $\alpha$ and $\beta$ such that the grasp is force closure.

Figure 12.48: A coin with a square hole.

32. Figure 12.48 shows a coin with a square hole in the middle grasped by several point contacts as shown.
(a) Suppose the coin is grasped by two frictionless point contacts and one point contact with friction coefficient $\mu = 1$ placed on the inside edge as shown in Figure 12.48(a). Determine whether the grasp is force closure.
(b) Suppose the coin is now grasped by four frictionless point contacts as shown in Figure 12.48(b), three of which are placed on the inside edge at the locations shown. The fourth point contact can be placed anywhere on the outside edge. Determine all the contact point locations on the outside edge that makes this grasp force closure.

(c) Three frictionless point contacts are placed on the inside edge as shown in Figure 12.48(c). Suppose any number of frictionless point contacts can be placed on the outside edge. Find the minimum number of outside edge point contacts that makes this grasp force closure.

(d) Is it possible to construct a force closure grasp using exactly two frictionless point contacts? Explain your answer.

---

33. A pie-shaped planar rigid body of radius $R = 1$ is grasped by four point contacts $A(t_1), B(t_2), C(\theta_1), \text{ and } D(\theta_2)$ as shown in Figure 12.49. The contact locations are constrained as follows:

\[ 0 < t_1, t_2 < 1, \quad 0 < \theta_1 < \frac{\pi}{4}, \quad \frac{\pi}{4} < \theta_2 < \frac{\pi}{2}. \]

(a) Suppose all the point contacts are frictionless, and $t_1 = 1/2$, $t_2 = 1/2$, $\theta_1 = \pi/6$, $\theta_2 = \pi/3$. Use the convex hull test to determine if this grasp is force closure.

(b) Suppose all the point contacts are frictionless, but that the contact locations for $A(t_1), B(t_2), C(\theta_1)$ are arbitrarily given. Does there always exist a feasible
contact location for $D(\theta_2)$ that makes the grasp force closure? If yes, explain your answer based on the convex hull test. If no, provide a counterexample and carefully explain your reasoning.

(c) Contact $D$ is now eliminated, so that the body is now grasped by the three contacts $A$, $B$, and $C$ only. Assume that point contacts $A$ and $B$ have friction coefficient $\mu = \frac{1}{2}$, with locations fixed at $t_1 = 1/2$ and $t_2 = 1/2$, respectively. Assume point contact $C$ is frictionless, with its location determined by angle $\theta_1$. Find the range of values for $\theta_1$ such that the grasp is always force closure.

Figure 12.50: A Planar object.

34. The planar object of Figure 12.50 is restrained by several point contacts. $A$, $B$, $D$, and $E$ are frictionless point contacts, while $C$ is a frictional point contact with $\mu = 1$. Let $0 < L < 2$.
   (a) Suppose the object is restrained by point contacts $A$ and $E$ only. Is this grasp force closure?
   (b) Suppose the object is restrained by point contacts $B$, $C$, and $D$ only. For what range of values of $L$ is the grasp force closure?
   (c) Suppose the object is restrained by point contacts $A$, $B$, and $C$ only. Is this grasp force closure?

35. Come up with a formula, as a function of $n$, for the minimum friction coefficient needed for a two-fingered force-closure grasp of an $n$-sided regular polygon, where $n$ is odd. Assume the fingers can only make contact with the edges, not the vertices. If the fingers could also contact the vertices, how does your answer change? You can assume the fingers are circular.

36. Consider a table at rest, supported by four legs in frictional contact with the floor. The normal forces provided by each leg are not unique; there is an infinite
Figure 12.51: A zero thickness rod supported by a single contact.

Figure 12.52: A frictionless finger pushes a box to the right. Gravity acts downward. Does the box slide flat against the table, does it tip over the right corner, or does it slide and tip on the right corner?

set of solutions to the normal forces yielding force balance with gravity. What is the dimension of the space of normal force solutions? What is the dimension of the space of contact force solutions if we include tangential frictional forces?

37. A thin rod in gravity is supported from below by a single stationary contact, shown in Figure 12.51. You can place one more contact anywhere else on the top or the bottom of the rod. Indicate all places you can put this contact while balancing the gravitational force. Use moment labeling to justify your answer. Prove the same using algebraic force balance, and comment on the magnitude of the normal forces depending on the location of the second contact.

38. A frictionless finger begins pushing a box over a table (Figure 12.52). There is friction between the box and the table, as indicated in the figure. There are three possible contact modes between the box and the table: either the box slides to the right flat against the table, or it tips over the right corner, or it tips over the right corner while the the corner also slides to the right. Which actually occurs? Assume quasistatic force balance and answer the following questions.
(a) For each of the three contact modes, draw the moment labeling regions corresponding to the table friction cone edges active in that contact mode.
(b) For each moment labeling drawing, determine whether the pushing force plus the gravitational force can be quasistatically balanced by the support forces. From this, determine which contact mode actually occurs.
(c) Graphically show a support friction cone for which your answer changes.
39. In Figure 12.53, body 1, of mass $m_1$ with center of mass at $(x_1, y_1)$, leans on body 2, of mass $m_2$ with center of mass at $(x_2, y_2)$. Both are supported by a horizontal line, and gravity acts downward. The friction coefficient at all four contacts (one at $(0, 0)$, one at $(x_L, 0)$, one at $(x_L, y)$, and one at $(x_R, 0)$) is $\mu > 0$. We want to know if it is possible for the assembly to stay standing by some choice of contact forces within the friction cones. Write the six equations of force balance for the two bodies in terms of the gravitational forces and the contact forces, and express the conditions that must be satisfied for it to be possible for this assembly to stay standing. How many equations and unknowns are there?

40. Write a program that accepts a set of contacts acting on a planar part and determines whether the part is in first-order form closure.

41. Write a program that accepts a set of contacts acting on a spatial part and determines whether the part is in first-order form closure.

42. Write a program that accepts a friction coefficient and a set of contacts acting on a planar part and determines whether the part is in force closure.

43. Write a program that accepts a friction coefficient and a set of contacts acting on a spatial part and determines whether the part is in force closure. Use a polyhedral approximation to the friction cone at each contact point that
underestimates the friction cone and that has four facets.

44. Write a program to simulate the quasistatic meter stick trick of Example 12.7. The program takes as input the initial \( x \)-position of the left finger, the right finger, and the stick’s center of mass; the constant speed \( \dot{x} \) of the right finger (toward the left finger); and the static and kinetic friction coefficients, where \( \mu_s \geq \mu_k \). The program should simulate until the two fingers touch, or until the stick falls. The program should plot the position of the left finger (which is constant), the right finger, and the center of mass as a function of time. Include an example where \( \mu_s = \mu_k \), an example where \( \mu_s \) is only slightly larger than \( \mu_k \), and an example where \( \mu_s \) is much larger than \( \mu_k \).

45. Write a program that determines if a given assembly of planar parts can remain standing in gravity. Gravity \( g \) acts in the \(-\hat{y}\) direction. The assembly is described by \( m \) bodies, \( n \) contacts, and the friction coefficient \( \mu \), all entered by the user. Each of the \( m \) bodies is described by its mass \( m_i \) and the \((x_i, y_i)\) location of its center of mass. Each contact is described by the index \( i \) of each of the two bodies involved in the contact and the unit normal direction (defined as into the first body). If the contact has only one body involved, the second body is assumed to be stationary (e.g., ground). The program should look for a set of coefficients \( k_j \geq 0 \) multiplying the friction cone edges at the contacts (if there are \( n \) contacts, then there are \( 2n \) friction cone edges and coefficients) such that each of the \( m \) bodies is in force balance, considering gravity. Except in degenerate cases, if there are more force balance equations (\( 3m \)) than unknowns (\( 2n \)), there is no solution. In the usual case where \( 2n > 3m \), there is a family of solutions, meaning that the force at each contact cannot be known with certainty.

One approach is to have your program generate an appropriate linear program, and use the programming language’s built-in linear programming solver.

46. This is a generalization of the previous exercise. Now, instead of simply deciding whether the assembly stays standing for a stationary base, the base moves according to a trajectory specified by the user, and the program determines whether the assembly can stay together during the trajectory (i.e., whether sticking contact at all contacts allows each body to follow the specified trajectory). The three-dimensional trajectory of the base can be specified as a polynomial in \((x(t), y(t), \theta(t))\), for a base reference frame defined at a particular position. For this problem, you also need to specify the scalar moment of inertia about the center of mass for each part. You may find it convenient to express motion and forces (gravitational, contact, inertial) in the frame of each part and solve for the dynamics in the body frames. Your program should check for stability (all contact normal forces are nonnegative while satisfying the dynamics) at finely spaced discrete points along the trajectory. It should return a binary result: the assembly can be maintained at all points along the trajectory, or not.
Chapter 13

Wheeled Mobile Robots

A kinematic model of a mobile robot governs how wheel speeds map to robot velocities, while a dynamic model governs how wheel torques map to robot accelerations. In this chapter, we ignore dynamics and focus on kinematics. We also assume that the robots roll on hard, flat, horizontal ground without skidding (i.e., no tanks or skid-steered vehicles). The mobile robot is assumed to have a single rigid-body chassis (not articulated like tractor-trailers) with a configuration $T_{sb} \in SE(2)$ representing a chassis-fixed frame $\{b\}$ relative to a fixed space frame $\{s\}$ in the horizontal plane. In this chapter we represent $T_{sb}$ by the three coordinates $q = (\phi, x, y)$. We also usually represent the velocity of the chassis as the time derivative of the coordinates, $\dot{q} = (\dot{\phi}, \dot{x}, \dot{y})$. Occasionally it will be convenient to refer to the chassis’ planar twist $V_b = (\omega_{bz}, v_{bx}, v_{by})$ expressed in $\{b\}$, where

$$V_b = \begin{bmatrix} \omega_{bz} \\ v_{bx} \\ v_{by} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{x} \\ \dot{y} \end{bmatrix}$$

$$\dot{q} = \begin{bmatrix} \dot{\phi} \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix} \begin{bmatrix} \omega_{bz} \\ v_{bx} \\ v_{by} \end{bmatrix}.$$ (13.1)

This chapter covers kinematic modeling, motion planning, and feedback control for wheeled mobile robots, and concludes with a brief introduction to mobile manipulation.

13.1 Types of Wheeled Mobile Robots

Wheeled mobile robots may be classified in two major categories, omnidirectional and nonholonomic. Omnidirectional mobile robots have no equality constraints on the chassis velocity $\dot{q} = (\dot{\phi}, \dot{x}, \dot{y})$, while nonholonomic robots are called “nonholonomic” because the chassis is subject to a single Pfaffian velocity constraint $A(q)\dot{q} = 0$ (see Chapter 2.4 for an introduction to Pfaffian
constraints). For a car-like robot, this constraint is that the car cannot move directly sideways. Despite this velocity constraint, the car can reach any \((\phi, x, y)\) configuration in the obstacle-free plane. In other words, the velocity constraint cannot be integrated to an equivalent configuration constraint, and therefore it is a nonholonomic constraint.

Whether a wheeled mobile robot is omnidirectional or nonholonomic depends on the type of wheels it employs (Figure 13.1). Nonholonomic mobile robots employ “typical” wheels, such as one you might find on your car: it rotates about an axle perpendicular to the plane of the wheel at the wheel’s center, and optionally it can be steered by spinning the wheel about an axis perpendicular to the ground at the contact point. The wheel rolls without sideways slip, which is the source of the nonholonomic constraint on the robot’s chassis.

Omnidirectional wheeled mobile robots typically employ either omniwheels or mecanum wheels. An omniwheel is a typical wheel augmented with rollers on the outer circumference of the wheel. These rollers spin freely about axes in the plane of the wheel and tangent to the wheel’s outer circumference, and they allow sideways sliding while the wheel drives forward and backward without slip in that direction. Mecanum wheels are similar, except the spin axes of the circumferential rollers are not in the plane of the wheel (see Figure 13.1). The sideways sliding allowed by omniwheels and mecanum wheels ensure that there are no velocity constraints on the robot chassis.

Omnwheels and mecanum wheels are not steered, only driven forward or backward. Because of the small diameter rollers, omniwheels and mecanum wheels work best on hard, flat ground.

Because the issues in modeling, motion planning, and control of wheeled mobile robots depend intimately on whether the robot is omnidirectional or nonholonomic, we treat these two cases separately in the following sections.

---

1 These types of wheels are often called “Swedish wheels,” as they were invented by Bengt Ilon working at the Swedish company Mecanum AB. The usage of, and the differentiation between, the terms “omniwheel,” “mecanum wheel,” and “Swedish wheel” is not completely standard, but here we use one popular choice.
13.2 Omnidirectional Wheeled Mobile Robots

13.2.1 Modeling

To achieve an arbitrary three-dimensional chassis velocity \( \dot{q} = (\dot{\phi}, \dot{x}, \dot{y}) \), an omnidirectional mobile robot must have at least three wheels, since each wheel has only one motor (controlling its forward/backward velocity). Figure 13.2 shows two omnidirectional mobile robots, one with three omniwheels and one with four mecanum wheels. Also shown are the motions that are obtained by driving the wheel motors as well as the free sliding motions allowed by the rollers.

Two important questions in kinematic modeling are:

(i) Given a desired chassis velocity \( \dot{q} \), at what speeds must the wheels be driven?

(ii) Given limits on the individual wheel driving speeds, what are the limits on the chassis velocity \( \dot{q} \)?

To answer these questions, we need to understand the wheel kinematics.
Figure 13.3: (Left) The direction of the wheel motion due to driving the wheel’s motor and the direction that its rollers allow it to slide freely. For an omniwheel, \( \gamma = 0 \), and for a mecanum wheel, typically \( \gamma = \pm 45^\circ \). (Right) The driven and free-sliding speeds for the wheel velocity \( v = (v_x, v_y) \) expressed in the wheel frame \( \hat{x}_w, \hat{y}_w \), where the \( \hat{x}_w \)-axis is aligned with the forward driving direction.

In a frame \( \hat{x}_w, \hat{y}_w \) at the center of the wheel, the linear velocity of the center of the wheel is written \( v = (v_x, v_y) \), which satisfies

\[
\begin{bmatrix}
  v_x \\
  v_y
\end{bmatrix} = v_{\text{drive}} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + v_{\text{slide}} \begin{bmatrix} -\sin \gamma \\ \cos \gamma \end{bmatrix},
\]

where \( \gamma \) denotes the angle at which free “sliding” occurs (allowed by the passive rollers on the circumference of the wheel), \( v_{\text{drive}} \) is the driving speed, and \( v_{\text{slide}} \) is the sliding speed. For an omniwheel, \( \gamma = 0 \), and for a mecanum wheel, typically \( \gamma = \pm 45^\circ \). Solving Equation (13.3), we get

\[
v_{\text{drive}} = v_x + v_y \tan \gamma \\
v_{\text{slide}} = v_y / \cos \gamma.
\]

Letting \( r \) be the radius of the wheel and \( u \) be the driving angular speed of the wheel,

\[
u = \frac{v_{\text{drive}}}{r} = \frac{1}{r} (v_x + v_y \tan \gamma).
\]

To derive the full transformation from a chassis velocity \( \dot{q} = (\dot{\phi}, \dot{x}, \dot{y}) \) to the driving angular speed \( u_i \) for wheel \( i \), refer to the notation illustrated in Figure 13.4. The chassis frame \( \{b\} \) is at \( q = (\phi, x, y) \) in the fixed space frame \( \{s\} \). The center of the wheel and its driving direction are given by \( (\beta_i, x_i, y_i) \) expressed in \( \{b\} \), the wheel’s radius is \( r_i \), and the wheel’s sliding direction is given by \( \gamma_i \). Then \( u_i \) is related to \( \dot{q} \) by

\[
u_i = h_i(\phi)\dot{q} =
\begin{bmatrix}
  1, \tan \gamma_i \\
  r_i, -\sin \beta_i
\end{bmatrix}
\begin{bmatrix}
  \cos \beta_i & \sin \beta_i \\
  -\sin \beta_i & \cos \beta_i
\end{bmatrix}
\begin{bmatrix}
  1 \\ 0 \\
  0 \\ 1
\end{bmatrix}
\begin{bmatrix}
  1 & 0 & 0 \\
  0 & \cos \phi & \sin \phi \\
  0 & -\sin \phi & \cos \phi
\end{bmatrix}
\begin{bmatrix}
  \dot{\phi} \\
  \dot{x} \\
  \dot{y}
\end{bmatrix}.
\]
13.2. Omnidirectional Wheeled Mobile Robots

Reading from right to left, the first transformation expresses \( \dot{q} \) as \( V_b \); the next transformation produces the linear velocity at the wheel in \( \{b\} \); the next transformation expresses this linear velocity in the wheel frame \( \hat{x}_w-\hat{y}_w \); and the final transformation calculates the driving angular velocity using Equation (13.4).

Evaluating Equation (13.5) for \( h_i(\phi) \), we get

\[
\frac{1}{r_i \cos \gamma_i} \begin{bmatrix} x_i \sin(\beta_i + \gamma_i) - y_i \cos(\beta_i + \gamma_i) \\ \cos(\beta_i + \gamma_i + \phi) \\ \sin(\beta_i + \gamma_i + \phi) \end{bmatrix}^T.
\]

(13.6)

For an omnidirectional robot with \( m \geq 3 \) wheels, the matrix \( H(\phi) \in \mathbb{R}^{m \times 3} \) mapping a desired chassis velocity \( \dot{q} \in \mathbb{R}^3 \) to the vector of wheel driving speeds \( u \in \mathbb{R}^m \) is constructed by stacking the \( m \) rows \( h_i(\phi) \):

\[
u = H(\phi)\dot{q} = \begin{bmatrix} h_1(\phi) \\ h_2(\phi) \\ \vdots \\ h_m(\phi) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{x} \\ \dot{y} \end{bmatrix}.
\]

(13.7)

We can also express the relationship between \( u \) and the body twist \( V_b \). This mapping does not depend on the chassis orientation \( \phi \):

\[
u = H(0)V_b = \begin{bmatrix} h_1(0) \\ h_2(0) \\ \vdots \\ h_m(0) \end{bmatrix} \begin{bmatrix} \omega_{bz} \\ v_{bx} \\ v_{by} \end{bmatrix}.
\]

(13.8)
The wheel positions and headings \((\beta_i, x_i, y_i)\) in \(\{b\}\), and their free sliding directions \(\gamma_i\), must be chosen so that \(H(0)\) is rank 3. For example, if we constructed a mobile robot of omniwheels whose driving directions and sliding directions were all aligned, the rank of \(H(0)\) would be two, and there would be no way to controllably generate translational motion in the sliding direction.

In the case \(m > 3\), such as the four-wheeled youBot of Figure 13.2, choosing \(u\) such that Equation (13.8) is not satisfied for any \(V_b \in \mathbb{R}^3\) implies that the wheels must skid in their driving directions.

Using the notation in Figure 13.5, the kinematic model of the mobile robot with three omniwheels is

\[
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3
\end{bmatrix} = H(0) V_b = \frac{1}{r} \begin{bmatrix}
  -d & 1 & 0 \\
  -d & -1/2 & -\sin(\pi/3) \\
  -d & -1/2 & \sin(\pi/3)
\end{bmatrix}
\begin{bmatrix}
  \omega_{bz} \\
  v_{bx} \\
  v_{by}
\end{bmatrix}
\]  \tag{13.9}

and the kinematic model of the mobile robot with four mecanum wheels is

\[
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3 \\
  u_4
\end{bmatrix} = H(0) V_b = \frac{1}{r} \begin{bmatrix}
  -\ell - w & 1 & -1 \\
  \ell + w & 1 & 1 \\
  \ell + w & 1 & -1 \\
  -\ell - w & 1 & 1
\end{bmatrix}
\begin{bmatrix}
  \omega_{bz} \\
  v_{bx} \\
  v_{by}
\end{bmatrix}.
\]  \tag{13.10}

For the mecanum robot, to move in \(+\hat{x}_b\), all wheels drive forward at the same speed; to move in \(+\hat{y}_b\), wheels 1 and 3 drive backward and wheels 2 and 4 drive forward at the same speed; and to rotate in the counterclockwise direction, wheels 1 and 4 drive backward and wheels 2 and 3 drive forward at the same speed. Note that the robot chassis is capable of the same speeds in the forward and sideways directions.
13.2. Omnidirectional Wheeled Mobile Robots

Figure 13.6: (Top row) Regions of feasible body twists $V$ for the three-wheeled (left) and four-wheeled (right) robots of Figure 13.5. Also shown for the three-wheeled robot is the intersection with the $\omega_{bz} = 0$ plane. (Bottom row) The bounds in the $\omega_{bz} = 0$ plane (translational motions only).

If the driving angular velocity of wheel $i$ is subject to bounds $|u_i| \leq u_{i,\text{max}}$, i.e.,

$$-u_{i,\text{max}} \leq u_i = h_i(0)\mathcal{V}_b \leq u_{i,\text{max}},$$

two parallel constraint planes defined by $-u_{i,\text{max}} = h_i(0)\mathcal{V}_b$ and $u_{i,\text{max}} = h_i(0)\mathcal{V}_b$ are generated in the three-dimensional space of body twists. Any $\mathcal{V}_b$ between these two planes is a feasible chassis body twist according to wheel $i$, while any $\mathcal{V}_b$ outside this slice is infeasible. The normal direction to the constraint planes is $h_i^T(0)$, and the points on the planes closest to the origin are $-u_{i,\text{max}}h_i^T(0)/\|h_i(0)\|_2$ and $u_{i,\text{max}}h_i^T(0)/\|h_i(0)\|_2$.

If the robot has $m$ wheels, then the region of feasible body twists $V$ is bounded by the $m$ pairs of parallel constraint planes. The region $V$ is therefore a convex three-dimensional polyhedron, with the origin (zero twist) in the center and $2m$ faces. Visualizations of the 6-sided and 8-sided regions $V$ for the three-wheeled and four-wheeled models in Figure 13.5 are shown in Figure 13.6.

13.2.2 Motion Planning

Since omnidirectional mobile robots are free to move in any direction, any of the trajectory planning methods for kinematic systems in Chapter 9, and most
of the motion planning methods of Chapter 10, can be adapted.

13.2.3 Feedback Control

Given a desired trajectory $q_d(t)$, we can adopt the feedforward plus proportional feedback linear controller (11.1) of Chapter 11 to track the trajectory:

$$
\dot{q}_{\text{com}}(t) = \dot{q}_d(t) + K_p(\dot{q}_d(t) - q(t)), \quad (13.11)
$$

where $K_p \in \mathbb{R}^{3 \times 3}$ is positive definite and $q(t)$ is an estimate of the actual configuration derived from sensors. Then $\dot{q}_{\text{com}}(t)$ can be converted to commanded wheel driving velocities $u_{\text{com}}(t)$ using Equation (13.7).

13.3 Nonholonomic Wheeled Mobile Robots

In Chapter 2.4, the $k$ Pfaffian velocity constraints acting on a system with configuration $q \in \mathbb{R}^n$ were written as $A(q)\dot{q} = 0$, where $A(q) \in \mathbb{R}^{k \times n}$. Instead of specifying the $k$ directions in which velocities are not allowed, we can write the allowable velocities of a kinematic system as a linear combination of $n-k$ velocity directions. This representation is equivalent, and it has the advantage that the coefficients of the linear combinations are precisely the controls available to us. We will see this representation in the kinematic models below.

By the title of this section, we are implying that the velocity constraints are not integrable to equivalent configuration constraints. We will establish this formally in Section 13.3.2.

13.3.1 Modeling

13.3.1.1 The Unicycle

The simplest wheeled mobile robot is a single upright rolling wheel, or unicycle. Let $r$ be the radius of the wheel. We write the configuration of the wheel as $q = (\phi, x, y, \theta)^T$, where $(x, y)$ is the contact point, $\phi$ is the heading direction, and $\theta$ is the rolling angle of the wheel (Figure 13.7). The configuration of the “chassis” (e.g., the seat of the unicycle) is $(\phi, x, y)$. The kinematic equations of motion are

$$
\dot{q} = \begin{bmatrix}
\phi \\
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} = \begin{bmatrix}
0 & 1 \\
r \cos \phi & 0 \\
r \sin \phi & 0 \\
1 & 0
\end{bmatrix} \begin{bmatrix}
u_1 \\
u_2
\end{bmatrix}
= G(q)u = g_1(q)u_1 + g_2(q)u_2. \quad (13.12)
$$

The control inputs are $u = (u_1, u_2)^T$, with $u_1$ the forward-backward angular speed and $u_2$ the angular turning speed. The controls are subject to the constraints $-u_{1,\text{max}} \leq u_1 \leq u_{1,\text{max}}$ and $-u_{2,\text{max}} \leq u_2 \leq u_{2,\text{max}}$. 
13.3. Nonholonomic Wheeled Mobile Robots

The vector-valued functions \( g_i(q) \in \mathbb{R}^4 \) are the columns of the matrix \( G(q) \), and these are called the tangent vector fields (also called the control vector fields or simply velocity vector fields) over \( q \) associated with the controls \( u_i = 1 \).Evaluated at a specific configuration \( q \), \( g_i(q) \) is a tangent vector (or velocity vector) of the tangent vector field.

An example of a vector field on \( \mathbb{R}^2 \) is illustrated in Figure 13.8.

All of our kinematic models of nonholonomic mobile robots will have the form \( \dot{q} = G(q)u \), as in (13.12). Three things to notice about these models are: (1) there is no drift—zero controls mean zero velocity; (2) the vector fields \( g_i(q) \) are generally functions of the configuration \( q \); and (3) \( \dot{q} \) is linear in the controls.

Since we are not usually concerned with the rolling angle of the wheel, we
can drop the fourth row from (13.12) to get the simplified control system
\[
\dot{q} = \begin{bmatrix}
\dot{\phi} \\
\dot{x} \\
\dot{y} \\
\dot{\theta}_L \\
\dot{\theta}_R
\end{bmatrix} = \begin{bmatrix}
0 & 1 \\
r \cos \phi & 0 \\
r \sin \phi & 0 \\
\frac{r}{d} \cos \phi & \frac{r}{d} \cos \phi \\
\frac{r}{d} \sin \phi & \frac{r}{d} \sin \phi
\end{bmatrix} \begin{bmatrix}
u_L \\
u_R
\end{bmatrix}.
\] (13.13)

13.3.1.2 The Differential-Drive Robot

The differential-drive robot, or diff-drive, is perhaps the simplest wheeled mobile robot architecture. A diff-drive robot consists of two independently driven wheels of radius \(r\) that rotate about the same axis, as well as one or more caster wheels, ball casters, or low-friction sliders that keep the robot horizontal. Let the distance between the wheels be \(2d\) and choose the \((x, y)\) reference point halfway between the wheels (Figure 13.9). Writing the configuration as \(q = (\phi, x, y, \theta_L, \theta_R)^T\), where \(\theta_L\) and \(\theta_R\) are the rolling angles of the left and right wheels, respectively, the kinematic equations are
\[
\dot{q} = \begin{bmatrix}
\dot{\phi} \\
\dot{x} \\
\dot{y} \\
\dot{\theta}_L \\
\dot{\theta}_R
\end{bmatrix} = \begin{bmatrix}
\frac{-r}{d} \\
\frac{r}{d} \cos \phi \\
\frac{r}{d} \sin \phi \\
1 & 0 \\
0 & 1
\end{bmatrix} \begin{bmatrix}
u_L \\
u_R
\end{bmatrix},
\] (13.14)

where \(u_L\) is the angular speed of the left wheel and \(u_R\) is the angular speed of the right wheel. Positive angular speed of each wheel corresponds to forward motion at that wheel. The controls at each wheel are taken from the interval \([-u_{\text{max}}, u_{\text{max}}]\).

Since we are not usually concerned with the rolling angles of the two wheels, we can drop the last two rows to get the simplified control system
\[
\dot{q} = \begin{bmatrix}
\dot{\phi} \\
\dot{x} \\
\dot{y}
\end{bmatrix} = \begin{bmatrix}
\frac{-r}{d} \\
\frac{r}{d} \cos \phi \\
\frac{r}{d} \sin \phi
\end{bmatrix} \begin{bmatrix}
u_L \\
u_R
\end{bmatrix}.
\] (13.15)
13.3. Nonholonomic Wheeled Mobile Robots

The two front wheels of a car are steered at different angles using Ackermann steering so that all wheels roll without slipping. The car is shown executing a turn at its minimum turning radius $r_{\text{min}}$.

Two advantages of a diff-drive robot are its simplicity (typically a motor attached directly to the axle of each wheel) and high maneuverability (the robot can spin in place by rotating the wheels in opposite directions). Casters are often not appropriate for outdoor use, however.

### 13.3.1.3 The Car-Like Robot

The most familiar wheeled vehicle is a car, with two steered front wheels and two fixed-heading rear wheels. To prevent slipping of the front wheels, they are steered using Ackermann steering, as illustrated in Figure 13.10. The center of rotation of the car’s chassis lies on the line passing through the rear wheels at the intersection with the perpendicular bisectors of the front wheels.

To define the configuration of the car, we ignore the rolling angles of the four wheels and write $q = (\phi, x, y, \psi)^T$, where $(x, y)$ is the location of the midpoint between the rear wheels, $\phi$ is the car’s heading direction, and $\psi$ is the steering angle of the car, defined at the virtual wheel at the midpoint between the front wheels. The controls are the forward speed $v$ of the car at its reference point and the angular speed $w$ of the steering angle. The car’s kinematics are

$$
\dot{q} = \begin{bmatrix}
\dot{\phi} \\
\dot{x} \\
\dot{y} \\
\dot{\psi}
\end{bmatrix} = \begin{bmatrix}
tan \psi & 0 \\
0 & \cos \phi \\
0 & \sin \phi \\
0 & 1
\end{bmatrix} \begin{bmatrix}
v \\
w
\end{bmatrix},
$$

(13.16)

where $\ell$ is the wheelbase between the front and rear wheels. The control $v$ is limited to a closed interval $[v_{\text{min}}, v_{\text{max}}]$ where $v_{\text{min}} < 0 < v_{\text{max}}$, the steering rate is limited to $[-w_{\text{max}}, w_{\text{max}}]$ with $w_{\text{max}} > 0$, and the steering angle $\psi$ is limited to $[\psi_{\text{min}}, \psi_{\text{max}}]$. 

Figure 13.10: The two front wheels of a car are steered at different angles using Ackermann steering so that all wheels roll without slipping. The car is shown executing a turn at its minimum turning radius $r_{\text{min}}$. 
The kinematics (13.16) can be simplified if the steering control is actually the steering angle \( \psi \) and not its rate \( w \). This assumption is justified if the steering rate limit \( w_{\text{max}} \) is high so that the steering angle can be changed nearly instantaneously by a lower-level controller. In this case, \( \psi \) is eliminated as a state variable, and the car’s configuration is simply \( q = (\phi, x, y)^T \). We use the control inputs \( (v, \omega) \), where \( v \) is still the car’s forward speed and \( \omega \) is now its rate of rotation. These can be converted to the controls \( (v, \psi) \) by the relations

\[
v = v, \quad \psi = \tan^{-1} \left( \frac{\ell \omega}{v} \right).
\]

The constraints on the controls \( (v, \omega) \) due to the constraints on \( (v, \psi) \) take a somewhat complicated form, as we see shortly.

The simplified car kinematics can now be written

\[
\dot{q} = \begin{bmatrix}
\dot{\phi} \\
\dot{x} \\
\dot{y}
\end{bmatrix} = G(q)u = \begin{bmatrix}
0 & 1 \\
\cos \phi & 0 \\
\sin \phi & 0
\end{bmatrix} \begin{bmatrix}
v \\
\omega
\end{bmatrix}.
\]

(13.18)

The nonholonomic constraint associated with this system can be derived by using one of the equations from (13.18)

\[
\dot{x} = v \cos \phi \\
\dot{y} = v \sin \phi
\]

to solve for \( v \), then plugging the result into the other equation to get

\[
A(q)\dot{q} = [0, \sin \phi, -\cos \phi] \dot{q} = \dot{x} \sin \phi - \dot{y} \cos \phi = 0.
\]

13.3.1.4 Canonical Simplified Model for Nonholonomic Mobile Robots

The kinematics (13.18) are a canonical simplified model for nonholonomic mobile robots. Using control transformations such as (13.17), the simplified unicycle kinematics (13.13) and the simplified differential-drive kinematics (13.15) can also be expressed in this form. The control transformation for the simplified unicycle kinematics (13.13) is

\[
u_1 = \frac{v}{r}, \quad u_2 = \omega
\]

and the transformation for the simplified diff-drive kinematics (13.15) is

\[
u_L = \frac{v - \omega d}{r}, \quad u_R = \frac{v + \omega d}{r}.
\]

With these input transformations, the only difference between the simplified unicycle, diff-drive robot, and car kinematics are the control limits on \( (v, \omega) \). These are illustrated in Figure 13.11.

We can use the two control inputs \( (v, \omega) \) in the canonical model (13.18) to directly control the two components of the linear velocity of a reference point...
Figure 13.11: The \((v, \omega)\) control sets for the simplified unicycle, diff-drive robot, and car kinematics. The car’s control set illustrates that it is incapable of turning in place. The angle of the sloped lines in the car’s bowtie control set is determined by the car’s minimum turning radius. If the car has no reverse gear, only the right half of the bowtie is available.

Figure 13.12: The point \(P\) is located at \((x_r, y_r)\) in the chassis-fixed frame \(\{b\}\).

\(P\) fixed to the robot chassis. This is useful when a sensor is located at \(P\), for example. Let \((x_P, y_P)\) be the coordinates of \(P\) in the world frame, and \((x_r, y_r)\) be the (constant) coordinates in the chassis frame \(\{b\}\), with \(\dot{x}_b\) in the robot’s heading direction (Figure 13.12). To find the controls \((v, \omega)\) needed to achieve a desired world-frame motion \((\dot{x}_P, \dot{y}_P)\), we first write

\[
\begin{bmatrix}
  x_P \\
  y_P 
\end{bmatrix} =
\begin{bmatrix}
  x \\
  y 
\end{bmatrix} +
\begin{bmatrix}
  \cos \phi & -\sin \phi \\
  \sin \phi & \cos \phi 
\end{bmatrix}
\begin{bmatrix}
  x_r \\
  y_r 
\end{bmatrix}.
\] (13.19)

Differentiating, we get

\[
\begin{bmatrix}
  \dot{x}_P \\
  \dot{y}_P 
\end{bmatrix} =
\begin{bmatrix}
  \dot{x} \\
  \dot{y} 
\end{bmatrix} + \dot{\phi}
\begin{bmatrix}
  -\sin \phi & -\cos \phi \\
  \cos \phi & -\sin \phi 
\end{bmatrix}
\begin{bmatrix}
  x_r \\
  y_r 
\end{bmatrix}.
\] (13.20)

Substituting \(\omega\) for \(\dot{\phi}\) and \((v \cos \phi, v \sin \phi)\) for \((\dot{x}, \dot{y})\) and solving, we get

\[
\begin{bmatrix}
  v \\
  \omega 
\end{bmatrix} = \frac{1}{x_r}
\begin{bmatrix}
  x_r \cos \phi - y_r \sin \phi & x_r \sin \phi + y_r \cos \phi \\
  -\sin \phi & \cos \phi 
\end{bmatrix}
\begin{bmatrix}
  \dot{x}_P \\
  \dot{y}_P 
\end{bmatrix}.
\] (13.21)

This equation may be read as \((v, \omega)^T = J^{-1}(q)(\dot{x}_P, \dot{y}_P)^T\), where \(J(q)\) is the Jacobian relating \((v, \omega)\) to the world-frame motion of \(P\). Note that the Jacobian \(J(q)\) is singular when \(P\) is chosen on the line \(x_r = 0\). Points on this line, such as at the midway point between the wheels of a diff-drive robot or the rear wheels of a car, can only move in the heading direction of the vehicle.
13.3.2 Controllability

Feedback control for an omnidirectional robot is simple, as there is a set of wheel driving speeds for any desired chassis velocity \( \dot{q} \) (Equation (13.7)). In fact, if the goal of the feedback controller is simply to stabilize the robot to the origin \( q = (0, 0, 0) \), not trajectory tracking as in the control law (13.11), we could use the even simpler feedback controller

\[
\dot{q}_{\text{com}}(t) = -Kq(t)
\]  

(13.22)

for any positive definite \( K \). The feedback gain matrix \(-K\) acts like a spring to pull \( q \) to the origin, and Equation (13.7) is used to transform \( \dot{q}_{\text{com}}(t) \) to \( u_{\text{com}}(t) \). The same type of “linear spring” controller could be used to stabilize the point \( P \) on the canonical nonholonomic robot (Figure 13.12) to \((x_P, y_P) = (0, 0)\) since, by Equation (13.21), any desired \((\dot{x}_P, \dot{y}_P)\) can be achieved by controls \((v, \omega)\).\(^2\)

In short, the kinematics of the omnidirectional robot, as well as the kinematics of the point \( P \) for the nonholonomic robot, can be rewritten in the single-integrator form

\[
\dot{x} = \nu,
\]  

(13.23)

where \( x \) is the configuration we are trying to control and \( \nu \) is a “virtual control” that is actually implemented using the transformations in Equation (13.7) for an omnidirectional robot or Equation (13.21) for the control of \( P \) by a nonholonomic robot. Equation (13.23) is a simple example of the more general class of linear control systems

\[
\dot{x} = Ax + B\nu
\]  

(13.24)

which are known to be **linearly controllable** if the **Kalman rank condition** is satisfied:

\[
\text{rank}([B \ AB \ A^2B \ ... \ A^{n-1}B]) = \dim(x) = n,
\]

where \( x \in \mathbb{R}^n, \nu \in \mathbb{R}^m, A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m} \). In Equation (13.23), \( A = 0 \) and \( B \) is the identity matrix, trivially satisfying the rank condition for linear controllability, since \( m = n \). This implies the existence of the simple linear control law

\[
\nu = -Kx,
\]

as in Equation (13.22), to stabilize the origin.

There is no linear controller that can stabilize the full chassis configuration to \( q = 0 \) for a nonholonomic robot, however; the nonholonomic robot is not linearly controllable. More than that, there is no controller that is a continuous function of \( q \) that can stabilize \( q = 0 \). This fact is embedded in the following well known result, which we state without proof.

**Theorem 13.1.** A system \( \dot{q} = G(q)u \) with \( \text{rank}(G(0)) < \dim(q) \) cannot be stabilized to \( q = 0 \) by a continuous time-invariant feedback control law.

\(^2\)For the moment we ignore the different constraints on \((v, \omega)\) for the unicycle, diff-drive robot, and car-like robot, as they do not change the main point.
13.3. Nonholonomic Wheeled Mobile Robots

This theorem applies to our canonical nonholonomic robot model, since the rank of $G(q)$ is two everywhere (there are only two control vector fields), while the chassis configuration is three dimensional.

For nonlinear systems of the form $\dot{q} = G(q)u$, there are other notions of controllability, however. We consider a few of these next, and show that even though the canonical nonholonomic robot is not linearly controllable, it still satisfies other important notions of controllability. In particular, the velocity constraint does not integrate to a configuration constraint.

13.3.2.1 Definitions of Controllability

We begin with some standard definitions of nonlinear controllability.

**Definition 13.1.** A robot is **controllable** from $q$ if, for any $q_{\text{goal}}$, there exists a control trajectory $u(t)$ that drives the robot from $q$ to $q_{\text{goal}}$ in finite time $T$. The robot is **small-time locally accessible** from $q$ if, for any time $T > 0$ and any neighborhood $3W(q)$, the reachable set in time less than $T$ without leaving $W(q)$ is a full-dimensional subset of the configuration space. The robot is **small-time locally controllable** (STLC) from $q$ if, for any time $T > 0$ and any neighborhood $W(q)$, the reachable set in time less than $T$ without leaving $W(q)$ is a neighborhood of $q$.

Small-time local accessibility (STLA) and small-time local controllability (STLC) are illustrated in Figure 13.13 for a two-dimensional configuration space. Clearly STLC at $q$ is a stronger condition than STLA at $q$. If a system is STLC at all $q$, then it is controllable from any $q$, by patching together paths in neighborhoods from $q$ to $q_{\text{goal}}$.

For all the examples in this chapter, if a controllability property holds for any $q$, then it holds for all $q$, since the maneuverability of the robot does not change with configuration.

Consider the examples of a car and a forward-only car with no reverse gear. A forward-only car is STLA, as we will see shortly, but it is not STLC; if it is confined to a tight space (a small neighborhood $W$), it cannot reach configurations directly behind its initial configuration. On the other hand, a car with

---

Figure 13.13: Illustrations of small-time local accessibility (STLA) and small-time local controllability (STLC) in a two-dimensional space. The shaded regions are the reachable sets without leaving the neighborhood $W(q)$.

---

A neighborhood $W(q)$ of a configuration $q$ is any full-dimensional set of configuration space containing $q$ in the interior. For example, the set of configurations in a ball of radius $r > 0$ centered at $q$ (i.e., all $q_b$ satisfying $\|q_b - q\| < r$) is a neighborhood of $q$. 
Wheeled Mobile Robots

a reverse gear is STLC. Both cars are controllable in the obstacle-free plane, because even a forward-only car can drive anywhere.

If there are obstacles in the plane, there may be some free space configurations the forward-only car cannot reach that the STLC car can reach. (Consider an obstacle directly in front of the car, for example.) If the obstacles are all defined as closed subsets of the plane containing their boundaries, the STLC car can reach any configuration in its connected component of the free space.

It is worth thinking about this last statement for a moment. All free configurations have collision-free neighborhoods, since the free space is defined as open and the obstacles are defined as closed (containing their boundaries). Therefore it is always possible to maneuver in any direction. This means that if the length of your car is shorter than the available parking space, you can parallel park into it, even if it takes a long time!

If any of the controllability properties holds (controllable, STLA, or STLC), then the reachable configuration space is full dimensional, and therefore any velocity constraints on the system are nonholonomic.

13.3.2.2 Controllability Tests

Consider a driftless, linear-in-the-control ("control affine") system

$$\dot{q} = G(q)u = \sum_{i=1}^{m} g_i(q)u_i, \quad q \in \mathbb{R}^n, u \in \mathcal{U} \subset \mathbb{R}^m, m < n, \quad (13.25)$$

generalizing the canonical nonholonomic model where $n = 3$ and $m = 2$. The set of feasible controls is $\mathcal{U} \subset \mathbb{R}^m$. For example, the control sets $\mathcal{U}$ for the unicycle, diff-drive, car-like, and forward-only car-like robots are shown in Figure 13.11. In this chapter we consider two types of control sets $\mathcal{U}$: those whose positive linear span is $\mathbb{R}^m$, i.e., $\text{pos}(\mathcal{U}) = \mathbb{R}^m$, such as the control sets for the unicycle, diff-drive robot, and car in Figure 13.11, and those whose positive linear span does not cover $\mathbb{R}^m$ but whose linear span does, i.e., $\text{span}(\mathcal{U}) = \mathbb{R}^m$, such as the control set for the forward-only car in Figure 13.11.

Local controllability properties (STLA or STLC) of (13.25) depend on the non-commutativity of motions along the vector fields $g_i$. Let $F^\epsilon_{g_i}(q)$ be the configuration reached by following the vector field $g_i$ for time $\epsilon$ starting from $q$. Then $g_i(q)$ and $g_j(q)$ commute if $F^\epsilon_{g_j}(F^\epsilon_{g_i}(q)) = F^\epsilon_{g_i}(F^\epsilon_{g_j}(q))$, i.e., the order of following the vector fields does not matter. If they do not commute, i.e., $F^\epsilon_{g_j}(F^\epsilon_{g_i}(q)) - F^\epsilon_{g_i}(F^\epsilon_{g_j}(q)) \neq 0$, then the order of application of the vector fields affects the final configuration. In addition, if we define the noncommutativity as

$$\Delta q = F^\epsilon_{g_j}(F^\epsilon_{g_i}(q)) - F^\epsilon_{g_i}(F^\epsilon_{g_j}(q))$$

for small $\epsilon$, if $\Delta q$ is in a direction that cannot be achieved directly by any of the other vector fields $g_k$, then switching between $g_i$ and $g_j$ can create motion in a direction not present in the original set of vector fields. A familiar example is parallel parking a car: there is no vector field corresponding to direct sideways translation, but
by alternating forward and backward motion along two different vector fields, it is possible to create a net motion to the side.

To approximately calculate \( q(2\epsilon) = F_{g_j}^\epsilon(F_{g_i}^\epsilon(q(0))) \) for small \( \epsilon \), we use a Taylor expansion and truncate the expansion at \( O(\epsilon^3) \). We start by following \( g_i \) for time \( \epsilon \), and use the fact that \( \dot{q} = g_i(q) \) and \( \ddot{q} = \frac{\partial g_i}{\partial q} \dot{q} = \frac{\partial g_i}{\partial q} g_i(q) \):

\[
q(\epsilon) = q(0) + \epsilon \dot{q}(0) + \frac{1}{2} \epsilon^2 \ddot{q}(0) + O(\epsilon^3)
\]

\[
= q(0) + \epsilon g_i(q(0)) + \frac{1}{2} \epsilon^2 \frac{\partial g_i}{\partial q} g_i(q(0)) + O(\epsilon^3).
\]

Now, after following \( g_j \) for time \( \epsilon \):

\[
q(2\epsilon) = q(\epsilon) + \epsilon g_j(q(\epsilon)) + \frac{1}{2} \epsilon^2 \frac{\partial g_j}{\partial q} g_j(q(\epsilon)) + O(\epsilon^3)
\]

\[
= q(0) + \epsilon g_i(q(0)) + \frac{1}{2} \epsilon^2 \frac{\partial g_i}{\partial q} g_i(q(0)) + \epsilon g_j(q(0)) + \frac{1}{2} \epsilon^2 \frac{\partial g_j}{\partial q} g_j(q(0)) + O(\epsilon^3).
\]

\[ (13.26) \]

Note the presence of the \( \epsilon^2 \frac{\partial g_j}{\partial q} g_i \) term, the only term that depends on the order of the vector fields. Using the expression \( (13.26) \), we can calculate the noncommutativity

\[
\Delta q = F_{g_j}^\epsilon(F_{g_i}^\epsilon(q)) - F_{g_i}^\epsilon(F_{g_j}^\epsilon(q)) = \epsilon^2 \left( \frac{\partial g_i}{\partial q} g_j - \frac{\partial g_j}{\partial q} g_i \right)(q(0)) + O(\epsilon^3). \]

\[ (13.27) \]

In addition to measuring noncommutativity, \( \Delta q \) is also the net motion (to order \( \epsilon^2 \)) obtained by following \( g_i \) for time \( \epsilon \), then \( g_j \) for time \( \epsilon \), then \(-g_i\) for time \( \epsilon \), then \(-g_j\) for time \( \epsilon \).

The term \( \frac{\partial g_j}{\partial q} g_i - \frac{\partial g_i}{\partial q} g_j \) in Equation \( (13.27) \) is important enough that we give it its own name:

**Definition 13.2.** The **Lie bracket** of the vector fields \( g_i(q) \) and \( g_j(q) \) is

\[
[g_i, g_j](q) = \left( \frac{\partial g_j}{\partial q} g_i - \frac{\partial g_i}{\partial q} g_j \right)(q).
\]

\[ (13.28) \]

This Lie bracket is the same as the Lie bracket for rigid-body motions introduced in Chapter 8.2.2. The only difference is that the Lie bracket in Chapter 8.2.2 was thought of as the noncommutativity of two constant twists \( V_i, V_j \) defined at a given instant, rather than of two velocity vector fields defined over
all configurations $q$. The Lie bracket $[\mathcal{V}_i, \mathcal{V}_j] = [\mathcal{V}_i][\mathcal{V}_j] - [\mathcal{V}_j][\mathcal{V}_i]$ from Chapter 8.2.2 would be identical to the expression in Equation (13.28) if the constant twists were represented as vector fields $g_i(q), g_j(q)$ in local coordinates $q$. See Exercise 1, for example.

The Lie bracket of two vector fields $g_i(q)$ and $g_j(q)$ should itself be thought of as a vector field $[g_i, g_j](q)$, where approximate motion along the Lie bracket vector field can be obtained by switching between the original two vector fields. As we saw in our Taylor expansion, however, motion along the Lie bracket vector field is slow relative to motions along the original vector fields—for small times $\epsilon$, motion of order $\epsilon$ can be obtained in the directions of the original vector fields, while motion in the Lie bracket direction is only of order $\epsilon^2$. This agrees with our common experience that moving a car sideways by parallel parking motions is slow relative to forward and backward or turning motions, as in the next example.

**Example 13.1.** Consider the canonical nonholonomic robot with vector fields $g_1(q) = (0, \cos \phi, \sin \phi)^T$ and $g_2(q) = (1, 0, 0)^T$. The Lie bracket vector field $g_3(q) = [g_1, g_2](q)$ is

$$g_3(q) = [g_1, g_2](q) = \left( \frac{\partial g_2}{\partial q} g_1 - \frac{\partial g_1}{\partial q} g_2 \right)(q) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & \cos \phi \\ \sin \phi & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & \sin \phi & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \cos \phi \end{bmatrix} = \begin{bmatrix} 0 \\ -\cos \phi \\ 0 \end{bmatrix}.$$  

The Lie bracket direction is a sideways “parallel parking” motion, as illustrated in Figure 13.14. The net motion obtained by following $g_1$ for $\epsilon$, $g_2$ for $\epsilon$, $-g_1$ for $\epsilon$, and $-g_2$ for $\epsilon$ is a motion of order $\epsilon^2$ in this Lie bracket direction, plus a term of order $\epsilon^3$.

Based on the result of Example 13.1, no matter how small the maneuvering space is for a car with a reverse gear, it can generate sideways motion. Thus we have shown that the Pfaffian velocity constraint implicit in the kinematics $\dot{q} = G(q)u$ for the canonical nonholonomic mobile robot is not integrable to a configuration constraint.

A Lie bracket $[g_i, g_j]$ is called a **Lie product** of degree 2, because the original vector fields appear two times in the bracket. For the canonical nonholonomic model, it is only necessary to consider the degree 2 Lie product to show that there are no configuration constraints. To test whether there are configuration constraints for more general systems of the form (13.25), however, it may be necessary to consider even deeper Lie brackets, such as $[g_i, [g_j, g_k]]$ or $[g_i, [g_i, [g_i, g_j]]]$, which are Lie products of degree 3 and 4, respectively. Just as it is possible to generate motions in Lie bracket directions by switching between
the original vector fields, it is possible to generate motion in Lie product directions of degree greater than 2. Generating motions in these directions is even slower than for degree 2 Lie products, however.

The Lie algebra of a set of vector fields is defined by all Lie products of all degrees, including Lie products of degree 1 (the original vector fields themselves):

Definition 13.3. The Lie algebra of a set of vector fields $G = \{g_1, \ldots, g_m\}$, written $\text{Lie}(G)$, is the linear span of all Lie products of degree $1 \ldots \infty$ of the vector fields $G$.

For example, for $G = \{g_1, g_2\}$, $\text{Lie}(G)$ is given by the linear combinations of the following Lie products:

- degree 1: $g_1, g_2$
- degree 2: $[g_1, g_2]$
- degree 3: $[g_1, [g_1, g_2]], [g_2, [g_1, g_2]]$
- degree 4: $[g_1, [g_1, [g_1, g_2]]], [g_1, [g_2, [g_1, g_2]]], [g_2, [g_1, [g_1, g_2]]], [g_2, [g_2, [g_1, g_2]]], [g_2, [g_2, [g_1, g_2]]]$
- : 

Since Lie products obey the following identities,

- $[g_i, g_i] = 0$
- $[g_i, g_j] = -[g_j, g_i]$
- $[g_i, [g_j, g_k]] + [g_k, [g_i, g_j]] + [g_j, [g_k, g_i]] = 0$ (the Jacobi identity),

Figure 13.14: The Lie bracket of the forward-backward vector field $g_1(q)$ and the spin-in-place vector field $g_2(q)$ is a sideways vector field.
not all bracket combinations need to be considered at each degree level.

In practice, there will be a finite degree $k$ beyond which higher-degree Lie products yield no more information about the Lie algebra. This happens, for example, when the dimension of the Lie algebra is $n$ at all $q$, i.e., $\dim(\mathfrak{Lie}(G)(q)) = n$ for all $q$; no further Lie brackets can yield new motion directions, as all motion directions have already been obtained. If this never happens, however, in general there is no way to know when to stop trying higher-degree Lie products. For a class of systems (13.25) called regular, however, if there is a degree $k$ that yields no new motion directions not included at lower degrees, then there is no need to look deeper.

With all of this as background, we are finally ready to state our main theorem on controllability.

**Theorem 13.2.** The control system (13.25), with vector fields $\mathcal{G} = \{g_1(q), \ldots, g_m(q)\}$, is small-time locally accessible from $q$ if $\dim(\mathfrak{Lie}(\mathcal{G})(q)) = n$ and $\text{span}(U) = \mathbb{R}^m$. If additionally $\text{pos}(U) = \mathbb{R}^m$, then the system is small-time locally controllable from $q$.

We omit the proof, but the intuition is as follows. If the Lie algebra is full rank, then the vector fields (followed both forward and backward) locally permit motion in any direction. If $\text{pos}(U) = \mathbb{R}^m$ (like a car with a reverse gear), then it is possible to directly follow all vector fields forward or backward, or it is possible to switch between feasible controls to follow any vector field forward and backward arbitrarily closely, and therefore the Lie algebra rank condition implies STLC. If the controls only satisfy $\text{span}(U) = \mathbb{R}^m$ (like a forward-only car), then some vector fields may only be followed forward or backward. Still, the Lie algebra rank condition ensures there are no equality constraints on the reachable set, so the system is STLA.

For any system of the form (13.25), the question of whether the velocity constraints are integrable is finally answered by Theorem 13.2. If the system is STLA at any $q$, the constraints are not integrable.

Let us apply Theorem 13.2 to a few examples.

**Example 13.2 (Controllability of the canonical nonholonomic mobile robot).** In Example 13.1, we computed the Lie bracket $g_3 = [g_1, g_2] = (0, \sin \phi, -\cos \phi)^T$ for the canonical nonholonomic robot. Putting the column vectors $g_1(q), g_2(q),$ and $g_3(q)$ side-by-side in a matrix and calculating its determinant, we find

$$\det[ g_1(q) \ g_2(q) \ g_3(q)] = \det \begin{bmatrix} 0 & 1 & 0 \\ \cos \phi & 0 & \sin \phi \\ \sin \phi & 0 & -\cos \phi \end{bmatrix} = \cos^2 \phi + \sin^2 \phi = 1,$$

i.e., the three vector fields are linearly independent at all $q$, and therefore the dimension of the Lie algebra is three at all $q$. By Theorem 13.2 and the control sets illustrated in Figure 13.11, the unicycle, diff-drive, and car are all STLC at all $q$, while the forward-only car is only STLA at all $q$. Each of the unicycle, diff-drive, car, and forward-only car is controllable in the obstacle-free plane.
Example 13.3 (Controllability of the full configuration of the unicycle). We already know from the previous example that the unicycle is STLC on its \((\phi, x, y)\) subspace; what if we include the rolling angle \(\theta\) in the description of the configuration? According to Equation (13.12), for \(q = (\phi, x, y, \theta)^T\), the two vector fields are \(g_1(q) = (0, r \cos \phi, r \sin \phi, 1)^T\) and \(g_2(q) = (1, 0, 0, 0)^T\).

Calculating the degree 2 and 3 Lie brackets

\[
g_3(q) = [g_1, g_2](q) = (0, r \sin \phi, -r \cos \phi, 0)^T
\]

\[
g_4(q) = [g_2, g_3](q) = (0, r \cos \phi, r \sin \phi, 0)^T,
\]

we see that these directions correspond to sideways translation and forward/backward motion without changing the wheel rolling angle \(\theta\), respectively. These are clearly linearly independent of \(g_1(q)\) and \(g_2(q)\), but we can confirm that by evaluating

\[
\det\begin{bmatrix}
g_1(q) & g_2(q) & g_3(q) & g_4(q)
ge_1(q) & g_2(q) & g_3(q) & g_4(q)
ge_1(q) & g_2(q) & g_3(q) & g_4(q)
ge_1(q) & g_2(q) & g_3(q) & g_4(q)
\end{bmatrix} = -r^2,
\]

i.e., \(\dim(\text{Lie}(G)(q)) = 4\) for all \(q\). Since \(\text{pos}(U) = \mathbb{R}^2\) for the unicycle by Figure 13.11, the unicycle is STLC at all points in its four-dimensional configuration space.

Think about a short “parallel parking” type maneuver which results in a net change in the rolling angle \(\theta\) with zero net change in the other configuration variables.

Example 13.4 (Controllability of the full configuration of the diff-drive). The full configuration of the diff-drive is five dimensional, \(q = (\phi, x, y, \theta_L, \theta_R)^T\), including the angles of both wheels. The two control vector fields are given in Equation (13.14). Proceeding to take Lie brackets of these vector fields, we would find that we can never create more than four linearly independent vector fields, i.e.,

\[
\dim(\text{Lie}(G)(q)) = 4
\]

at all \(q\). This is because there is a fixed relationship between the two wheel angles \((\theta_L, \theta_R)\) and the angle of the robot chassis \(\phi\). The three velocity constraints \((\dim(q) = 5, \dim(u) = 2)\) implicit in the kinematics (13.14) therefore can be viewed as two nonholonomic constraints and one holonomic constraint. On the full five-dimensional configuration space, the diff-drive is nowhere STLA.

Since we usually only worry about the configuration of the chassis, this negative result is not of much concern.

13.3.3 Motion Planning

13.3.3.1 Obstacle-Free Plane

It is easy to find feasible motions between any two chassis configurations \(q_0\) and \(q_{\text{goal}}\) in the obstacle-free plane for any of the four nonholonomic robot models (unicycle, diff-drive, car, or forward-only car). The problem gets more interesting when we try to optimize an objective function. Below we consider shortest paths for the forward-only car, shortest paths for the car, and fastest...
paths for the diff-drive. The solutions to these problems depend on optimal control theory, and the proofs are left to the original references (see the Notes and References at the end of the chapter).

**Shortest Paths for the Forward-Only Car** The shortest path problem is to find a path from \( q_0 \) to \( q_{\text{goal}} \) that minimizes the length of the path followed by the robot’s reference point. This is not an interesting question for the unicycle or the diff-drive; a shortest path for each of them is a rotation to point toward the goal position \((x_{\text{goal}}, y_{\text{goal}})\), a translation, then a rotation to the goal orientation. The total path length is \( \sqrt{(x_0 - x_{\text{goal}})^2 + (y_0 - y_{\text{goal}})^2} \).

The problem is more interesting for the forward-only car, sometimes called the Dubins car in honor of the mathematician who first studied the structure of the shortest planar curves with bounded curvature between two oriented points.

**Theorem 13.3.** For a forward-only car with the control set shown in Figure 13.11, shortest paths consist only of arcs at the minimum turning radius and straight-line segments. Denoting a circular arc segment as \( C \) and a straight-line segment as \( S \), the shortest path between any two configurations follows either the sequence (a) \( CSC \) or (b) \( CC_\alpha C \), where \( C_\alpha \) indicates a circular arc of angle \( \alpha > \pi \). Any of the \( C \) or \( S \) segments can be of length zero.

The two optimal path classes for a forward-only car are illustrated in Figure 13.15. We can calculate the shortest path by enumerating the possible \( CSC \) and \( CC_\alpha C \) paths. Construct the two minimum-turning-radius circles of the vehicle at both \( q_0 \) and \( q_{\text{goal}} \) and solve for (a) the points where lines (with the correct heading direction) are tangent to one of the circles at \( q_0 \) and one of the circles at \( q_{\text{goal}} \), and (b) the points where a minimum-turning-radius circle (with the correct heading direction) is tangent to one of the circles at \( q_0 \) and one of the circles at \( q_{\text{goal}} \). The solutions to (a) correspond to \( CSC \) paths and the solutions to (b) correspond to \( CC_\alpha C \) solutions. The shortest of all the solutions is the optimal path. The shortest path may not be unique.

If we break the \( C \) segments into two categories, \( L \) (when the steering wheel is pegged to the left) and \( R \) (when the steering wheel is pegged to the right),
we see there are four types of CSC paths (LSL, LSR, RSL, and RSR) and two types of CCαC paths (RLαR and LRαL).

A shortest path is also a minimum-time path for the forward-only car control set illustrated in Figure 13.11, where the only controls \((v, \omega)\) ever used are the three controls \((v_{\text{max}}, 0)\) (an S segment) and \((v_{\text{max}}, \pm \omega_{\text{max}})\) (a C segment).

**Shortest Paths for the Car** The shortest paths for a car with a reverse gear, sometimes called the **Reeds-Shepp car** in honor of the mathematicians who first studied the problem, again only use straight-line segments and minimum-turning-radius arcs. Using the notation \(C\) for a minimum-turning-radius arc, \(C_{a}\) for an arc of angle \(a\), \(S\) for a straight-line segment, and \(|\) for a cusp (a reversal of the linear velocity), Theorem 13.4 enumerates all the possible shortest path sequences.

**Theorem 13.4.** For a car with the control set shown in Figure 13.11, the shortest path between any two configurations is in one of the following nine classes:

\[
\begin{align*}
C|C|C & \quad CC|C & \quad C|CC & \quad CC_{a}|C_{a}C & \quad C|C_{a}C_{a}|C \\
C|C_{\pi/2}SC & \quad CSC_{\pi/2}|C_{a}C_{a} | C_{\pi/2}SC_{\pi/2}C & \quad CSC & \quad CSC_{\pi/2}SC_{\pi/2}C
\end{align*}
\]

Any of the \(C\) or \(S\) segments can be of length zero.

Three of the nine shortest path classes are illustrated in Figure 13.16. Again, the actual shortest path may be found by enumerating the finite set of possible solutions in the path classes in Theorem 13.4. The shortest path may not be unique.

If we break the \(C\) segments into four categories, \(L^{+}, L^{-}, R^{+}, \text{and} \ R^{-}\), where \(L\) and \(R\) mean the steering wheel is turned all the way to the left or right and + and – indicate the gear shift (forward or reverse), then the nine path classes of Theorem 13.4 can be expressed as \((6 \times 4) + (3 \times 8) = 48\) different types:

- **4 types (6 classes):** \(C|C|C, \ CC|C, \ C|CC, \ CC_{a}|C_{a}C, \ C|C_{a}C_{a}|C, \ C|C_{\pi/2}SC_{\pi/2}|C\)
- **8 types (3 classes):** \(C|C_{\pi/2}SC, \ CSC_{\pi/2}|C, \ CSC\)
Table 13.1: The 40 time-optimal trajectory types for the diff-drive. The notation \( R_\pi \) and \( L_\pi \) indicate spins of angle \( \pi \).

<table>
<thead>
<tr>
<th>motion segments</th>
<th>number of types</th>
<th>motion sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>( F, B, R, L )</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>( FR, FL, BR, BL, RF, RB, LF, LB )</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>( FRB, FLB, FR\pi B, FL\pi B, BRF, BLF, BR\pi F, BL\pi F, RFR, RFL, RBR, RBL, LFR, LFL, LBR,LBL )</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>( FRBL, FLBR, BRFL, BLFR, RFLB, RBLF, LFRB, LBRF )</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>( FRBLF, FLBRF, BRFLB, BLFRB )</td>
</tr>
</tbody>
</table>

The four types for six of the path classes are determined by the four different initial motion directions, \( L^+ \), \( L^- \), \( R^+ \), and \( R^- \). The eight types for three of the path classes are determined by the four initial motion directions and whether the turn is to the left or the right after the straight-line segment. There are only four types in the \( C|C_{\pi/2}SC_{\pi/2}|C \) class because the turn after the \( S \) segment is always opposite the turn before the \( S \) segment.

If it takes zero time to reverse the linear velocity, a shortest path is also a minimum-time path for the car control set illustrated in Figure 13.11, where the only controls \((v, \omega)\) ever used are the two controls \((\pm v_{\text{max}}, 0)\) (an \( S \) segment) or the four controls \((\pm v_{\text{max}}, \pm \omega_{\text{max}})\) (a \( C \) segment).

**Minimum-Time Motions for the Diff-Drive**  For a diff-drive robot with the diamond-shaped control set in Figure 13.11, any minimum-time motion consists of only translational motions and spins in place.

**Theorem 13.5.** For a diff-drive robot with the control set illustrated in Figure 13.11, minimum-time motions consist of forward and backward translations (\( F \) and \( B \)) at maximum speed \( \pm v_{\text{max}} \) and spins in place (\( R \) and \( L \) for right turns and left turns) at maximum angular speed \( \pm \omega_{\text{max}} \). There are 40 types of time-optimal motions, categorized in Table 13.1 by the number of motion segments. The notations \( R_\pi \) and \( L_\pi \) indicate spins of angle \( \pi \).

Note that Table 13.1 includes both \( FR_\pi B \) and \( FL_\pi B \), which are equivalent, as well as \( BR_\pi F \) and \( BL_\pi F \). Each of the trajectory types is time optimal for some pair \({q_0, q_{\text{goal}}} \), and the time-optimal trajectory may not be unique. Notably absent are three-segment sequences where the first and last motions are translations in the same direction (i.e., \( FRF, FLF, BRB, \) and \( BLB \)).

While any reconfiguration of the diff-drive can be achieved by spinning, translating, and spinning, in some cases other three-segment sequences have a shorter travel time. For example consider a diff-drive with \( v_{\text{max}} = \omega_{\text{max}} = 1 \)
13.3. Nonholonomic Wheeled Mobile Robots

Figure 13.17: (Top) A motion planning problem specified as a motion from \( q_0 = (0, 0, 0) \) to \( q_{\text{goal}} = (-7\pi/8, 1.924, 0.383) \). (Bottom left) A non-optimal LFR solution taking time 5.103. (Bottom right) The time-optimal FRB solution, through a “via point,” taking time 4.749.

and \( q_0 = 0, q_{\text{goal}} = (-7\pi/8, 1.924, 0.383) \) as shown in Figure 13.17. The time needed for a spin of angle \( \alpha \) is \( |\alpha|/\omega_{\text{max}} = |\alpha| \) and the time for a translation of \( d \) is \( |d|/v_{\text{max}} = |d| \). Therefore, the time needed for the LFR sequence is

\[
\frac{\pi}{16} + 1.962 + \frac{15\pi}{16} = 5.103
\]

while the time needed for the FRB sequence through a “via point” is

\[
1 + \frac{7\pi}{8} + 1 = 4.749.
\]

13.3.3.2 With Obstacles

If there are obstacles in the plane, the grid-based motion planning methods of Chapter 10.4.2 can be applied to the unicycle, diff-drive, car, or forward-only car using discretized versions of the control sets in Figure 13.11. See, for example, the discretizations in Figure 10.14 that use the extremal controls from Figure 13.11. Using the extremal controls takes advantage of our observation that shortest paths for cars and the diff-drive consist of minimum-turning-radius turns and straight-line segments. Also, because the C-space is only three-dimensional, the grid size should be manageable for reasonable resolutions along each dimension.
We can also apply the sampling methods of Chapter 10.5. For RRTs, we can again use a discretization of the control set, as mentioned above, or, for both PRMs and RRTs, the local planner that attempts to connect two configurations could use the shortest paths from Theorem 13.3, Theorem 13.4, or Theorem 13.5.

Another option for a car is to use any efficient obstacle-avoiding path planner, even if it ignores the motion constraints of the vehicle. Since the car is STLC, and since the free configuration space is defined to be open (obstacles are closed, containing their boundaries), the car can follow the path arbitrarily closely. To follow the path closely, however, the motion may be slow—imagine using parallel parking to travel a kilometer down the road.

Instead, the initial constraint-free path can be quickly transformed into a fast, feasible path that respects the car’s motion constraints. To do this, represent the initial path as \( q(s), s \in [0, 1] \). Then try to connect \( q(0) \) to \( q(1) \) using a shortest path from Theorem 13.4. If this path is in collision, then divide the original path in half and try to connect \( q(0) \) to \( q(1/2) \) and \( q(1/2) \) to \( q(1) \) using shortest paths. If either of these are in collision, divide that path, and so on. Because the car is STLC and the initial path lies in open free space, the process will eventually terminate, and the new path consists of a sequence of subpaths from Theorem 13.4. The process is illustrated in Figure 13.18.

### 13.3.4 Feedback Control

We can consider three types of feedback control problems for the canonical nonholonomic mobile robot (13.18) with controls \((v, \omega)\):

(i) **Stabilization of a configuration.** As we saw in Theorem 13.1, no time-invariant feedback law continuous in the state variables can stabilize a configuration for a nonholonomic mobile robot. There do exist time-varying and discontinuous feedback laws that accomplish the task, but we will not consider this problem further in this chapter.
(ii) **Trajectory tracking.** Given a desired trajectory $q_d(t)$, drive the error $q(t) - q_d(t)$ to zero as time goes to infinity.

(iii) **Path tracking.** Given a path $q(s)$, follow the geometric path, without regard to the time of the motion. This provides more control freedom than the trajectory tracking problem; essentially, we can choose the reference configuration along the path to help reduce tracking error, in addition to choosing $(v, \omega)$.

Path tracking and trajectory tracking are “easier” than stabilizing a configuration, in the sense that there exist continuous time-invariant feedback laws to stabilize the desired motions. In this section we consider the problem of trajectory tracking.

Assume that the reference trajectory is specified as $q_d(t) = (\phi_d(t), x_d(t), y_d(t))$ for $t \in [0, T]$, with a corresponding nominal control $(v_d(t), \omega_d(t)) \in \text{int}(U)$ for $t \in [0, T]$. The requirement that the nominal control be in the interior of the feasible control set $U$ is to ensure that some control effort is “left over” to accommodate small errors. This implies that the reference trajectory is neither a shortest path nor a time-optimal trajectory, since optimal motions saturate the controls. The reference trajectory could be planned using not-quite-extremal controls.

A simple first controller idea is to choose a reference point $P$ on the chassis of the robot (not on the axis of the two driving wheels), as in Figure 13.12. The desired trajectory $q_d(t)$ is then represented by the desired trajectory of the reference point $(x_{P_d}(t), y_{P_d}(t))$. To track this reference point trajectory, we can use a proportional feedback controller

$$
\begin{bmatrix}
\dot{x}_P \\
\dot{y}_P
\end{bmatrix} =
\begin{bmatrix}
k_p(x_{P_d} - x_P) \\
k_p(y_{P_d} - y_P)
\end{bmatrix}
$$

(13.29)

where $k_p > 0$. This simple linear control law is guaranteed to pull the actual point position along with the moving desired point position. The velocity $(\dot{x}_P, \dot{y}_P)$ calculated by the control law (13.29) is converted to $(v, \omega)$ by Equation (13.21).

The idea is that, as long as the reference point is moving, over time the entire robot chassis will line up with the desired orientation of the chassis. The problem is that the controller may choose the opposite orientation of what is intended; there is nothing in the control law to prevent this. Figure 13.19 shows two simulations, one where the control law (13.29) produces the desired chassis motion and one where the control law causes an unintended reversal in the sign of the driving velocity $v$. In both simulations, the controller succeeds in causing the reference point to track the desired motion.

To fix this, let us explicitly incorporate chassis angle error in the control law. The fixed space frame is written $\{s\}$, the chassis frame $\{b\}$ is at the point between the two wheels of the diff-drive (or the two rear wheels for a car) with the forward driving direction along the $\hat{x}_b$-axis, and the frame corresponding to
Wheeled Mobile Robots

Figure 13.19: (Left) A nonholonomic mobile robot with a reference point. (Middle) A scenario where the linear control law (13.29) tracking a desired reference point trajectory yields the desired trajectory tracking behavior for the entire chassis. (Right) A scenario where the point-tracking control law causes an unintended cusp in the robot motion. The reference point converges to the desired path but the robot’s orientation is opposite the intended orientation.

Figure 13.20: The space frame \( \{s\} \), the robot frame \( \{b\} \), and the desired configuration \( \{d\} \) driving forward along the planned path. The heading direction error is \( \phi_e = \phi - \phi_d \).

\( q_d(t) \) is written \( \{d\} \). We define the error coordinates

\[
q_e = \begin{bmatrix} \phi_e \\ x_e \\ y_e \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi_d & \sin \phi_d \\ 0 & -\sin \phi_d & \cos \phi_d \end{bmatrix} \begin{bmatrix} \phi - \phi_d \\ x - x_d \\ y - y_d \end{bmatrix},
\]

as illustrated in Figure 13.20. The vector \( (x_e, y_e) \) is the \( \{s\} \)-coordinate error vector \( (x - x_d, y - y_d) \) expressed in the reference frame \( \{d\} \).

Consider the nonlinear feedforward plus feedback control law

\[
\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} (v_d - k_1|v_d|(x_e + y_e \tan \phi_e))/\cos \phi_e \\ \omega_d - (k_2v_dy_e + k_3|v_d|\tan \phi_e)\cos^2 \phi_e \end{bmatrix},
\]

where \( k_1, k_2, k_3 > 0 \). Note two things about this control law: (1) if the error is zero, the control is simply the nominal control \( (v_d, \omega_d) \), and (2) the controls grow without bound as \( \phi_e \) approaches \( \pi/2 \) or \( -\pi/2 \). In practice, we assume that \( |\phi_e| \) is less than \( \pi/2 \) during trajectory tracking.

In the controller for \( v \), the second term, \(-k_1|v_d|x_e/\cos \phi_e\), attempts to reduce \( x_e \) by driving to catch up or slow down to the reference frame. The third term, \(-k_1|v_d|y_e\tan \phi_e/\cos \phi_e\), attempts to reduce \( y_e \) based on the component of the forward/backward velocity that impacts \( y_e \).
13.4 Odometry

Odometry is the process of estimating the chassis configuration $q$ from wheel motions, essentially integrating the wheel velocities. Since wheel rotation sensing is available on all mobile robots, odometry is cheap and convenient. Estimation errors tend to accumulate over time, though, due to unexpected slipping and skidding of the wheels and due to numerical integration error. Therefore, it is common to supplement odometry with other position sensors, like GPS, visual recognition of landmarks, ultrasonic beacons, laser or ultrasonic range sensing, etc. Those sensing modalities have their own measurement uncertainty, but errors do not accumulate over time. As a result, odometry generally gives superior results on short time scales, but the odometric estimates should either (a) be periodically corrected by other sensing modalities or, preferably, (b) integrated with other sensing modalities in an estimation framework based on a Kalman filter, particle filter, or similar.

In this section we focus on odometry. We assume each wheel of an omni-directional robot, and each rear wheel of a diff-drive or car, has an encoder that senses how far the wheel has rotated in its driving direction. If the wheels are driven by stepper motors, then we know the driving rotation of each wheel.

Figure 13.21: A mobile robot implementing the nonlinear control law (13.31).

In the controller for the turning velocity $\omega$, the second term, $-k_2 v_d y_e \cos^2 \phi_e$, attempts to reduce $y_e$ in the future by turning the heading direction of the robot toward the reference frame origin. The third term, $-k_3 |v_d| \tan \phi_e \cos^2 \phi_e$, attempts to reduce the heading error $\phi_e$.

A simulation of the control law is shown in Figure 13.21.

The control law (13.31) requires $|v_d| \neq 0$, so it is not appropriate for stabilizing “spin-in-place” motions for a diff-drive. The proof of the stability of the control law requires methods beyond the scope of this book. In practice, the gains should be chosen large enough to provide significant corrective action, but not so large that the controls chatter at the boundary of the feasible control set $U$. 
based on the steps we have commanded to it.

The goal is to estimate the new chassis configuration \( q_{k+1} \) as a function of the previous chassis configuration \( q_k \) given the change in wheel angles from the instant \( k \) to the instant \( k+1 \).

Let \( \Delta \theta_i \) be the change in wheel \( i \)’s driving angle since the wheel angle was last queried \( \Delta t \) ago. Since we only know the net change in the wheel driving angle, not the time history of how the wheel angle evolved during the time interval, the simplest approximation is that the wheel’s angular velocity was constant during the time interval, \( \dot{\theta}_i = \Delta \theta_i / \Delta t \). The choice of units used to measure the time interval is not relevant (since we will eventually integrate the chassis body twist \( \mathcal{V}_b \) over the same time interval), so set \( \Delta t = 1 \), i.e., \( \dot{\theta}_i = \Delta \theta_i \).

For omnidirectional mobile robots, the vector of wheel speeds \( \dot{\theta}_i \), and therefore \( \Delta \theta_i \), is related to the body twist \( \mathcal{V}_b = (\omega_{bz}, v_{bx}, v_{by})^T \) of the chassis by Equation (13.8)

\[
\Delta \theta = H(0) \mathcal{V}_b,
\]

where \( H(0) \) for the three-omniwheel robot is given by Equation (13.9) and \( H(0) \) for the four-mecanum-wheel robot is given by Equation (13.10). Therefore, the body twist \( \mathcal{V}_b \) corresponding to \( \Delta \theta \) is

\[
\mathcal{V}_b = F \Delta \theta = \frac{r}{4} \begin{bmatrix}
-1/(3d) & -1/(3d) & -1/(3d) \\
2/3 & -1/3 & -1/3 \\
0 & -1/(2 \sin(\pi/3)) & 1/(2 \sin(\pi/3))
\end{bmatrix} \Delta \theta,
\]

(13.32)

and for the four-mecanum-wheel robot,

\[
\mathcal{V}_b = F \Delta \theta = \frac{r}{4} \begin{bmatrix}
-1/(\ell + w) & 1/(\ell + w) & 1/(\ell + w) & -1/(\ell + w) \\
1 & 1 & 1 & 1 \\
-1 & 1 & -1 & 1
\end{bmatrix} \Delta \theta.
\]

(13.33)

The relationship \( \mathcal{V}_b = F \dot{\theta} = F \Delta \theta \) also holds for the diff-drive robot and the car (Figure 13.22), where \( \Delta \theta = (\Delta \theta_L, \Delta \theta_R)^T \) (the increments of the left and right wheels) and

\[
\mathcal{V}_b = F \Delta \theta = r \begin{bmatrix}
-1/(2d) & 1/(2d) \\
1/2 & 1/2
\end{bmatrix} \begin{bmatrix}
\Delta \theta_L \\
\Delta \theta_R
\end{bmatrix}.
\]

(13.34)

Since the wheel speeds are assumed constant during the time interval, so is the body twist \( \mathcal{V}_b \). Calling \( \mathcal{V}_{b6} \) the six-dimensional version of the planar twist \( \mathcal{V}_b \) (i.e., \( \mathcal{V}_{b6} = (0, 0, \omega_{bz}, v_{bx}, v_{by}, 0)^T \)), \( \mathcal{V}_{b6} \) can be integrated to generate the displacement created by the wheel angle increment vector \( \Delta \theta \):

\[
T_{b^\prime b} = e^{[\mathcal{V}_{b6}]}.
\]
13.5 Mobile Manipulation

For a robot arm mounted on a mobile base, mobile manipulation is the coordination of the motion of the base and the robot joints to achieve a desired motion at the end-effector. Typically the motion of the arm can be controlled more precisely than the motion of the base, so the most popular type of mobile manipulation is to drive the base, park it, let the arm perform the precise motion task, then drive away.
Figure 13.23: The space frame \(\{s\}\) and the frames \(\{b\}\), \(\{0\}\), and \(\{e\}\) attached to the mobile manipulator.

In some cases, however, it is advantageous, or even necessary, for the end-effector motion to be achieved by a combination of motion of the base and motion of the arm. Defining the fixed space frame \(\{s\}\), the chassis frame \(\{b\}\), a frame at the base of the arm \(\{0\}\), and an end-effector frame \(\{e\}\), the configuration of \(\{e\}\) in \(\{s\}\) is

\[
X(q, \theta) = T_{se}(q, \theta) = T_{sb}(q) T_{b0} T_{0e}(\theta),
\]

where \(\theta \in \mathbb{R}^n\) is the set of arm joint variables, \(T_{0e}(\theta)\) is the forward kinematics of the arm, \(T_{b0}\) is the fixed offset of \(\{0\}\) from \(\{b\}\), \(q = (\phi, x, y)\) is the planar configuration of the mobile base, and

\[
T_{sb}(q) = \begin{bmatrix}
\cos \phi & -\sin \phi & 0 & x \\
\sin \phi & \cos \phi & 0 & y \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]

See Figure 13.23.

Let \(X(t)\) be a path for the end-effector as a function of time. Then \(\mathbf{V}_e(t) = X^{-1}(t) X(t)\) is the \(se(3)\) representation of the end-effector twist expressed in \(\{e\}\). Further, let the vector of wheel velocities, whether the robot is omnidirectional or nonholonomic, be written \(u \in \mathbb{R}^m\). For kinematic control of the end-effector frame using wheel and joint velocities, we need the Jacobian \(J_e(\theta) \in \mathbb{R}^{6 \times (m+n)}\) satisfying

\[
\mathbf{V}_e = J_e(\theta) \begin{bmatrix} u \\ \dot{\theta} \end{bmatrix} = [ J_{\text{base}}(\theta) \ J_{\text{arm}}(\theta) ] \begin{bmatrix} u \\ \dot{\theta} \end{bmatrix}.
\]

Note that the Jacobian \(J_e(\theta)\) does not depend on \(q\); the end-effector velocity expressed in \(\{e\}\) is independent of the configuration of the mobile base. Also, we can partition \(J_e(\theta)\) into \(J_{\text{base}}(\theta) \in \mathbb{R}^{6 \times m}\) and \(J_{\text{arm}}(\theta) \in \mathbb{R}^{6 \times n}\). The term \(J_{\text{base}}(\theta) u\) expresses the contribution of the wheel velocities \(u\) to the end-effector’s velocity, and the term \(J_{\text{arm}}(\theta) \dot{\theta}\) expresses the contribution of the joint velocities to the end-effector’s velocity.
13.5. Mobile Manipulation

In Chapter 5, we developed a method to derive $J_{\text{arm}}(\theta)$, which is called the body Jacobian $J_b(\theta)$ in that chapter. All that remains is to find $J_{\text{base}}(\theta)$. As we saw in Section 13.4, for any type of mobile base there exists an $F$ satisfying

$$V_b = Fu.$$ 

To create a six-dimensional twist $V_{b6}$ corresponding to the planar twist $V_b$, we can define the $6 \times m$ matrix

$$F_6 = \begin{bmatrix}
0_m \\
0_m \\
F \\
0_m
\end{bmatrix},$$

where two rows of $m$ zeros are stacked above $F$ and one row is placed below. Now

$$V_{b6} = F_6 u.$$ 

This chassis twist can be expressed in the end-effector frame as

$$[\text{Ad}_{T_b(t)}]V_{b6} = [\text{Ad}_{T_{0e}^{-1}(t)T_{b0}^{-1}}]V_{b6} = [\text{Ad}_{T_{0e}^{-1}(t)T_{b0}^{-1}}]F_6 u = J_{\text{base}}(\theta)u.$$ 

Therefore,

$$J_{\text{base}}(\theta) = [\text{Ad}_{T_{0e}^{-1}(t)T_{b0}^{-1}}]F_6.$$ 

Now that we have the complete Jacobian $J_e(\theta) = [J_{\text{base}}(\theta) \ J_{\text{arm}}(\theta)]$, we can perform numerical inverse kinematics (Chapter 6.2) or implement kinematic feedback control laws to track a desired end-effector trajectory. For example, given a desired end-effector trajectory $X_d(t), t \in [0, 1]$, is parameterized by $\alpha = -\pi t, x_e(t) = -3\cos(\pi t)$, and $y_e(t) = 3\sin(\pi t)$, where $\alpha$ indicates the planar angle from the $\hat{x}_e$-axis to the $\hat{x}_e$-axis (see Figure ??).
13.6 Summary

- The chassis configuration of a wheeled mobile robot moving in the plane is $q = (\phi, x, y)$. The velocity can be represented either as $\dot{q}$ or as the planar twist $\mathbf{V}_b = (\omega_{bz}, v_{bx}, v_{by})^T$ expressed in the chassis-fixed frame $\{b\}$, where

$$
\mathbf{V}_b = \begin{bmatrix}
\omega_{bz} \\
v_{bx} \\
v_{by}
\end{bmatrix} = 
\begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \phi & \sin \phi \\
0 & -\sin \phi & \cos \phi
\end{bmatrix}
\begin{bmatrix}
\dot{\phi} \\
\dot{x} \\
\dot{y}
\end{bmatrix}.
$$

- The chassis of a nonholonomic mobile robot is subject to a single non-integrable Pfaffian velocity constraint $A(q)\dot{q} = [0, \sin \phi, -\cos \phi] \dot{q} = \dot{x} \sin \phi - \dot{y} \cos \phi = 0$. An omnidirectional robot, employing omniwheels or mecanum wheels, has no such constraint.

- For a properly constructed omnidirectional robot with $m \geq 3$ wheels, there
exists a rank 3 matrix $H(\phi) \in \mathbb{R}^{m \times 3}$ that maps the chassis velocity $\dot{q}$ to the wheel driving velocities $u$:

$$u = H(\phi)\dot{q}.$$  

In terms of the body twist $V_b$,

$$u = H(0)V_b.$$  

The driving speed limits of each wheel place two parallel planar constraints on the feasible body twists, creating a polyhedron $V$ of feasible body twists.

- Motion planning and feedback control for omnidirectional robots is simplified by the fact that there are no chassis velocity equality constraints.

- Nonholonomic mobile robots are described as driftless linear-in-the-control systems

$$\dot{q} = G(q)u, \quad u \in \mathcal{U} \subset \mathbb{R}^m,$$

where $G(q) \in \mathbb{R}^{n \times m}, n > m$. The columns $g_i(q)$ of $G(q)$ are called the control vector fields.

- The canonical simplified nonholonomic mobile robot model is

$$\dot{q} = \begin{bmatrix} \dot{\phi} \\ \dot{x} \\ \dot{y} \end{bmatrix} = G(q)u = \begin{bmatrix} 0 & 1 & 0 \\ \cos \phi & 0 & \omega \\ \sin \phi & 0 & \omega \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}.$$  

The control sets $\mathcal{U}$ differ for the unicycle, diff-drive, car, and forward-only car.

- A control system is small-time locally accessible (STLA) from $q$ if, for any time $T > 0$ and any neighborhood $W(q)$, the reachable set in time less than $T$ without leaving $W(q)$ is a full-dimensional subset of the configuration space. A control system is small-time locally controllable (STLC) from $q$ if, for any time $T > 0$ and any neighborhood $W(q)$, the reachable set in time less than $T$ without leaving $W(q)$ is a neighborhood of $q$. If the system is STLC from a given $q$, it can locally maneuver in any direction.

- The Lie bracket of two vector fields $g_1$ and $g_2$ is the vector field

$$[g_1, g_2] = \left( \frac{\partial g_2}{\partial q} g_1 - \frac{\partial g_1}{\partial q} g_2 \right).$$

- A Lie product of degree $k$ is a Lie bracket term where the original vector fields appear $k$ times. A Lie product of degree 1 is just one of the original vector fields.
• The Lie algebra of a set of vector fields $\mathcal{G} = \{g_1, \ldots, g_m\}$, written $\text{Lie}(\mathcal{G})$, is the linear span of all Lie products of degree $1 \ldots \infty$ of the vector fields $\mathcal{G}$.

• A driftless control-affine system is small-time locally accessible from $q$ if $\dim(\text{Lie}(\mathcal{G})(q)) = n$ and $\text{span}(\mathcal{U}) = \mathbb{R}^m$. If additionally $\text{pos}(\mathcal{U}) = \mathbb{R}^m$, then the system is small-time locally controllable from $q$.

• For a forward-only car in the obstacle-free plane, shortest paths always follow a turn at the tightest turning radius ($C$) or straight-line motions ($S$). There are two classes of shortest paths: $CSC$ and $CC_\alpha C$, where $C_\alpha$ is a turn of angle $|\alpha| > \pi$. Any of the $C$ or $S$ segments can be of length zero.

• For a car with a reverse gear, shortest paths always consist of a sequence of straight-line segments or turns at the tightest turning radius. Shortest paths always belong to one of nine classes.

• For the diff-drive, minimum-time motions always consist of turn-in-place motions and straight-line motions.

• For the canonical nonholonomic robot, there is no time-invariant control law continuous in the configuration that stabilizes the origin configuration. Continuous time-invariant control laws exist to stabilize a trajectory, however.

• Odometry is the process of estimating the chassis configuration based on how far the robot’s wheels have rotated in their driving direction, assuming no skidding in the driving direction and, for typical wheels (not omniwheels or mecanum wheels), no slip in the orthogonal direction.

• For a mobile manipulator with $m$ wheels and $n$ joints in the robot arm, the end-effector twist $\mathcal{V}_e$ in the end-effector frame $\{e\}$ is written

$$
\mathcal{V}_e = J_e(\theta) \begin{bmatrix} u \\ \dot{\theta} \end{bmatrix} = [ J_{\text{base}}(\theta) \ J_{\text{arm}}(\theta) ] \begin{bmatrix} u \\ \dot{\theta} \end{bmatrix}.
$$

The $6 \times m$ Jacobian $J_{\text{base}}(\theta)$ maps the wheel velocities $u$ to a velocity at the end-effector, and the $6 \times n$ Jacobian is the body Jacobian derived in Chapter 5. The Jacobian $J_{\text{base}}(\theta)$ is given by

$$
J_{\text{base}}(\theta) = [Ad_{T_{\text{base}}^{-1}(\theta)T_{\text{base}}^{-1}}]F_6
$$

where $F_6$ is the transformation from the wheel velocities to the chassis twist, $\mathcal{V}_{66} = Fu$.  

13.7 Notes and References
13.8 Exercises

1.
Appendix A

Summary of Useful Formulas

Chapter 2

• $dof = (\text{sum of freedoms of bodies}) - (\text{number of independent configuration constraints})$

• Grüber’s formula is an expression of the previous formula for mechanisms with $N$ links (including ground) and $J$ joints, where joint $i$ has $f_i$ degrees of freedom and $m = 3$ for planar mechanisms or $m = 6$ for spatial mechanisms:

$$dof = m(N - 1 - J) + \sum_{i=1}^{J} f_i.$$

• Pfaffian velocity constraints take the form $A(\theta)\dot{\theta} = 0$.

Chapter 3

<table>
<thead>
<tr>
<th>Rotations</th>
<th>Rigid-Body Motions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R \in SO(3) : 3 \times 3$ matrices satisfying $R^T R = I$, $\det R = 1$</td>
<td>$T \in SE(3) : 4 \times 4$ matrices</td>
</tr>
<tr>
<td>$T = \begin{bmatrix} R &amp; p \ 0 &amp; 1 \end{bmatrix}$, where $R \in SO(3), p \in \mathbb{R}^3$</td>
<td></td>
</tr>
</tbody>
</table>

$R^{-1} = R^T$

$T^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix}$

change of coord frame:

$R_{ab} R_{bc} = R_{ac}, \ R_{ab} p_b = p_a$

change of coord frame:

$T_{ab} T_{bc} = T_{ac}, \ T_{ab} p_b = p_a$

continued...
### Rotations (cont.)

<table>
<thead>
<tr>
<th>Rotating a frame {b}:</th>
<th>( R = \text{Rot}(\hat{\omega}, \theta) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_{ab'} = RR_{ab} ): rotate ( \theta ) about ( \hat{\omega}_a = \hat{\omega} )</td>
<td>( T_{sb'} = TT_{sb} ): rotate ( \theta ) about ( \hat{\omega}_a = \hat{\omega} ) (moves {b} origin), translate ( p ) in {s}</td>
</tr>
<tr>
<td>( R_{ab'} = R_{ab}R ): rotate ( \theta ) about ( \hat{\omega}_b = \hat{\omega} )</td>
<td>( T_{sb'} = T_{sb}T ): translate ( p ) in {b}, rotate ( \theta ) about ( \hat{\omega} ) in new body frame</td>
</tr>
</tbody>
</table>

- Unit rotation axis is \( \hat{\omega} \in \mathbb{R}^3 \), where \( \| \hat{\omega} \| = 1 \)

### Rigid-Body Motions (cont.)

<table>
<thead>
<tr>
<th>Angular velocity can be written ( \omega = \hat{\omega} \theta )</th>
<th>for any 3-vector, e.g., ( \omega \in \mathbb{R}^3 ), ( [\omega] = \left[ \begin{array}{ccc} 0 &amp; -\omega_3 &amp; \omega_2 \ \omega_3 &amp; 0 &amp; -\omega_1 \ -\omega_2 &amp; \omega_1 &amp; 0 \end{array} \right] \in \text{so}(3) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \omega, \theta ) ∈ \mathbb{R} | \omega | \in [0, 1] ) for ( \omega = 0 ) and | v | = 1 )</td>
<td></td>
</tr>
</tbody>
</table>

| Moment change of coord frame: \( \omega, \theta \) ∈ \mathbb{R} \| \omega \| \in [0, 1] \) for \( \omega = 0 \) and \| v \| = 1 \) |
|---|---|
| \( \hat{\omega}_a = R_{ab'}\hat{\omega}_b, \ \omega_a = R_{ab}\omega_b \) | \( \hat{\omega}_a = R_{ab'}\hat{\omega}_b, \ \omega_a = R_{ab}\omega_b \) |

| Moment change of coord frame: \( m_a = R_{ab}m_b \) | Wrench change of coord frame: \( F_a = (m_a, f_a) = [\text{Ad}_{T_{ab}}]^T F_b \) |

<table>
<thead>
<tr>
<th>Summary of Useful Formulas</th>
<th>( \omega, \theta ) ∈ \mathbb{R} | \omega | \in [0, 1] ) for ( \omega = 0 ) and | v | = 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{\omega} \in \mathbb{R}^3 ) are exp coords for ( R \in \text{SO}(3) )</td>
<td>( \omega, \theta ) ∈ \mathbb{R} | \omega | \in [0, 1] ) for ( \omega = 0 ) and | v | = 1 )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Log ( R \in \text{SO}(3) \rightarrow [\omega] \theta \in \text{so}(3) )</th>
<th>Log ( T \in \text{SE}(3) \rightarrow [S] \theta \in \text{se}(3) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{algorithm in Chapter 3.2.3.3} )</td>
<td>( \text{algorithm in Chapter 3.3.3.2} )</td>
</tr>
</tbody>
</table>
Chapter 4

- The product of exponentials formula for a serial chain manipulator is
  \[
  T = e^{[S_1]\theta_1} \cdots e^{[S_n]\theta_n} M
  \]
  space frame:
  \[
  T = M e^{[B_1]\theta_1} \cdots e^{[B_n]\theta_n}
  \]
  body frame:

  where \( M \) is the frame of the end-effector in the space frame when the
  manipulator is at its home position, \( S_i \) is the spatial twist when joint
  \( i \) rotates (or translates) at unit speed while all other joints are at their zero
  position, and \( B_i \) is the body twist of the end-effector frame when joint \( i \)
  moves at unit speed and all other joints are at their zero position.

Chapter 5

- For a manipulator end-effector configuration written in coordinates \( x \), the
  forward kinematics is \( x = f(\theta) \), and the differential kinematics is given by
  \( \dot{x} = \frac{\partial f}{\partial \theta} \dot{\theta} \), where \( J(\theta) \) is the manipulator Jacobian.

- Written using twists, the relation is \( \mathcal{V}_* = J_*(\theta) \dot{\theta} \), where \( * \) is either \( s \) (space Jacobian) or \( b \) (body Jacobian). The columns \( \mathcal{V}_{si} \) of the space Jacobian are
  \[
  \mathcal{V}_{si}(\theta) = [\text{Ad}_{e^{[S_1]\theta_1} \cdots e^{[S_{i-1}]\theta_{i-1}}}][S_i]
  \]
  and the columns \( \mathcal{V}_{bi} \) of the body Jacobian are
  \[
  \mathcal{V}_{bi}(\theta) = [\text{Ad}_{e^{-[B_{i+1}]\theta_{i+1}} \cdots e^{-[B_n]\theta_n}}][B_i].
  \]
  The spatial twist caused by joint \( i \) motion is only altered by the config-
  urations of joints inboard from joint \( i \) (between the joint and the space
  frame), while the body twist caused by joint \( i \) is only altered by the con-
  figurations of joints outboard from joint \( i \) (between the joint and the body
  frame).

  The two Jacobians are related by
  \[
  J_b(\theta) = [\text{Ad}_{T_{bs}(\theta)}][J_s(\theta)], \quad J_s(\theta) = [\text{Ad}_{T_{sb}(\theta)}][J_b(\theta)].
  \]

- Generalized forces at the joints \( \tau \) are related to wrenches expressed in the
  space or end-effector body frame by
  \[
  \tau = J_s^T(\theta) \mathcal{F}_*,
  \]
  where \( * \) is \( s \) (space frame) or \( b \) (body frame).

- The manipulability ellipsoid is defined by
  \[
  \mathcal{V}^T(JJ^T)^{-1}\mathcal{V} = 1,
  \]
  where \( \mathcal{V} \) may be a set of task space coordinate velocities \( \dot{q} \), a spatial
  or body twist, or the angular or linear components of a twist, and \( J \) is the
appropriate Jacobian satisfying $V = J(\theta)\dot{\theta}$. The principal axes of the manipulability ellipsoid are aligned with the eigenvectors of $JJ^T$, and the semi-axis lengths are the square roots of the corresponding eigenvalues.

• The force ellipsoid is defined by
  \[ F^T JJ^T F = 1, \]
  where $J$ is a Jacobian (possibly in terms of a minimum set of task space coordinates, or in terms of the spatial or body wrench) and $F$ is an end-effector force or wrench satisfying $\tau = J^T F$. The principal axes of the manipulability ellipsoid are aligned with the eigenvectors of $(JJ^T)^{-1}$, and the semi-axis lengths are the square roots of the corresponding eigenvalues.

Chapter 6

• The law of cosines states that $c^2 = a^2 + b^2 - 2ab \cos \gamma$, where $a$, $b$, and $c$ are the lengths of the sides of a triangle and $\gamma$ is the interior angle opposite side $c$. This formula is often useful to solve inverse kinematics problems.

• Numerical methods are used to solve the inverse kinematics for systems for which closed-form solutions do not exist. A Newton-Raphson method using the Jacobian pseudoinverse $J^\dagger(\theta)$ is outlined below.

(i) **Initialization:** Given $T_{sd}$ and an initial guess $\theta_0 \in \mathbb{R}^n$. Set $i = 0$.

(ii) Set $[V_b] = \log \left( T_{sb}^{-1}(\theta_i)T_{sd} \right)$. While $\|\omega_b\| > \epsilon_\omega$ or $\|v_b\| > \epsilon_v$ for small $\epsilon_\omega$, $\epsilon_v$:
   - Set $\theta_{i+1} = \theta_i + J^\dagger_b(\theta_i)V_b$.
   - Increment $i$.

If $J$ is square and full rank, then $J^\dagger = J^{-1}$. If $J \in \mathbb{R}^{m \times n}$ is full rank (rank $m$ for $n > m$ or rank $n$ for $n < m$), i.e., the robot is not at a singularity, the pseudoinverse can be calculated as

- $J^\dagger = J^T (JJ^T)^{-1}$ if $J$ is fat, $n > m$ (called a right inverse, since $JJ^\dagger = I$)
- $J^\dagger = (J^T J)^{-1}J^T$ if $J$ is tall, $n < m$ (called a left inverse, since $J^\dagger J = I$).

Chapter 8

• The Lagrangian is the kinetic minus the potential energy, $L(\theta, \dot{\theta}) = K(\theta, \dot{\theta}) - P(\theta)$.

• The Euler-Lagrange equations are
  \[ \tau = \frac{d}{dt} \frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta}. \]
The equations of motion of a robot can be written in the following equivalent forms:

\[ \tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}) \]
\[ = M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) \]
\[ = M(\theta)\ddot{\theta} + \dot{\theta}^T \Gamma(\theta) \dot{\theta} + g(\theta) \]
\[ = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) \dot{\theta} + g(\theta), \]

where \( M(\theta) \) is the \( n \times n \) symmetric positive-definite mass matrix, \( h(\theta, \dot{\theta}) \) is the sum of generalized forces due to gravity and quadratic velocity terms, \( c(\theta, \dot{\theta}) \) are quadratic velocity forces, \( g(\theta) \) are gravitational forces, \( \Gamma(\theta) \) is an \( n \times n \times n \) matrix of Christoffel symbols of the first kind obtained from partial derivatives of \( M(\theta) \) with respect to \( \theta \), and \( C(\theta, \dot{\theta}) \) is the \( n \times n \) Coriolis matrix whose \((i, j)\) entry is given by

\[ c_{ij}(\theta, \dot{\theta}) = \sum_{k=1}^{n} \Gamma_{ijk}(\theta) \dot{\theta}_k. \]

If the end-effector of the robot is applying a wrench \( \mathcal{F}_{\text{tip}} \) to the environment, the term \( J^T(\theta) \mathcal{F}_{\text{tip}} \) should be added to the right-hand side of the robot’s dynamic equations.

The symmetric positive-definite rotational inertia matrix of a rigid body is

\[ \mathcal{I}_b = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix}, \]

where

\[ I_{xx} = \int_{\mathcal{B}} (y^2 + z^2) \rho(x, y, z) dV \]
\[ I_{yy} = \int_{\mathcal{B}} (x^2 + z^2) \rho(x, y, z) dV \]
\[ I_{zz} = \int_{\mathcal{B}} (x^2 + y^2) \rho(x, y, z) dV \]
\[ I_{xy} = \int_{\mathcal{B}} x y \rho(x, y, z) dV \]
\[ I_{xz} = \int_{\mathcal{B}} x z \rho(x, y, z) dV \]
\[ I_{yz} = \int_{\mathcal{B}} y z \rho(x, y, z) dV, \]

\( \mathcal{B} \) is the body, \( dV \) is a differential volume element, and \( \rho(x, y, z) \) is the density function.

If \( \mathcal{I}_b \) is defined in a frame \( \{b\} \) at the center of mass, with axes aligned with the principal axes of inertia, then \( \mathcal{I}_b \) is diagonal.

If \( \{b\} \) is at the center of mass but its axes are not aligned with the principal axes of inertia, there always exists a rotated frame \( \{c\} \) defined by the rotation matrix \( R_{bc} \) such that \( \mathcal{I}_c = R_{bc}^T \mathcal{I}_b R_{bc} \) is diagonal.

If \( \mathcal{I}_b \) is defined in a frame \( \{b\} \) at the center of mass, then \( \mathcal{I}_q \), the inertia in a frame \( \{q\} \) aligned with \( \{b\} \), but displaced from the origin of \( \{b\} \) by \( q \in \mathbb{R}^3 \) in \( \{b\} \) coordinates, is

\[ \mathcal{I}_q = \mathcal{I}_b + m(q^T q I_3 - qq^T) \]

by Steiner’s theorem.
Summary of Useful Formulas

- The spatial inertia matrix $\mathcal{G}_b$ expressed in a frame $\{b\}$ at the center of mass is defined as the $6 \times 6$ matrix
  \[
  \mathcal{G}_b = \begin{bmatrix}
  I_b & 0 \\
  0 & mI
  \end{bmatrix}.
  \]

- The Lie bracket of two twists $\mathcal{V}_1$ and $\mathcal{V}_2$ is
  \[
  [\mathcal{V}_1, \mathcal{V}_2] = \text{ad}_{\mathcal{V}_1}(\mathcal{V}_2) = \text{ad}_{\mathcal{V}_1}\mathcal{V}_2,
  \]
  where
  \[
  \text{ad}_{\mathcal{V}} = \begin{bmatrix}
  [\omega] & 0 \\
  [v] & [\omega]
  \end{bmatrix} \in \mathbb{R}^{6 \times 6}.
  \]

- The twist-wrench formulation of the rigid-body dynamics of a single rigid body is
  \[
  \mathcal{F}_b = \mathcal{G}_b \dot{\mathcal{V}}_b - \text{ad}_{\mathcal{V}_b}^T \mathcal{G}_b \mathcal{V}_b.
  \]

- The kinetic energy of a rigid body is $\frac{1}{2} \mathcal{V}_b^T \mathcal{G}_b \mathcal{V}_b$, and the kinetic energy of an open-chain robot is $\frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta}$.

- The forward-backward Newton-Euler inverse dynamics algorithm is the following:

  **Initialization:** Attach a frame $\{0\}$ to the base, frames $\{1\}$ to $\{n\}$ to the centers of mass of links $\{1\}$ to $\{n\}$, and a frame $\{n+1\}$ at the end-effector, fixed in the frame $\{n\}$. Define $M_{i-1,i}$ to be the configuration of $\{i\}$ in $\{i-1\}$ when $\theta_i = 0$. Let $\mathcal{A}_i$ be the screw axis of joint $i$ expressed in $\{i\}$, and $\mathcal{G}_i$ be the $6 \times 6$ spatial inertia matrix of link $i$. Define $\mathcal{V}_0$ to be the twist of the base frame $\{0\}$ expressed in base frame coordinates. (This quantity is typically zero.) Let $g \in \mathbb{R}^3$ be the gravity vector expressed in base frame $\{0\}$ coordinates, and define $\mathcal{V}_0 = (0, -g)$. (Gravity is treated as an acceleration of the base in the opposite direction.) Define $\mathcal{F}_{n+1} = \mathcal{F}_{\text{tip}} = (m_{\text{tip}}, f_{\text{tip}})$ to be the wrench applied applied to the environment by the end-effector expressed in the end-effector frame $\{n+1\}$.

  **Forward iterations:** Given $\theta, \dot{\theta}, \ddot{\theta}$, for $i = 1$ to $n$ do
  \[
  T_{i-1,i} = M_{i-1,i} e^{[\mathcal{A}_i] \theta_i},
  \]
  \[
  \mathcal{V}_i = \text{Ad}_{T_{i-1,i}}(\mathcal{V}_{i-1}) + \mathcal{A}_i \dot{\theta}_i,
  \]
  \[
  \dot{\mathcal{V}}_i = \text{Ad}_{T_{i-1,i}}(\dot{\mathcal{V}}_{i-1}) + [\mathcal{V}_i, \mathcal{A}_i] \dot{\theta}_i + \mathcal{A}_i \ddot{\theta}_i.
  \]

  **Backward iterations:** For $i = n$ to $1$ do
  \[
  \mathcal{F}_i = \text{Ad}_{T_{i+1,i}}^T(\mathcal{F}_{i+1}) + \mathcal{G}_i \dot{\mathcal{V}}_i - \text{ad}_{\mathcal{V}_i}^T(\mathcal{G}_i \mathcal{V}_i)
  \]
  \[
  \tau_i = \mathcal{F}_i^T \mathcal{A}_i.
  \]
• Let $J_{ib}(\theta)$ to be the Jacobian relating $\dot{\theta}$ to the body twist $\mathbf{V}_i$ in link $i$’s center-of-mass frame $\{i\}$. Then the mass matrix $M(\theta)$ of the manipulator can be expressed as

$$M(\theta) = \sum_{i=1}^{n} J_{ib}(\theta)^T \mathbf{G}_i J_{ib}(\theta).$$

• The robot’s dynamics $M(\theta)\ddot{\theta} + h(\theta, \dot{\theta})$ can be expressed in the task space as

$$\mathbf{F} = \Lambda(\theta) \dot{\mathbf{V}} + \eta(\theta, \mathbf{V}),$$

where $\mathbf{F}$ is the wrench applied to the end-effector, $\mathbf{V}$ is the twist of the end-effector, and $\mathbf{F}, \mathbf{V}$, and the Jacobian $J(\theta)$ are all defined in the same frame. The task-space mass matrix $\Lambda(\theta)$ and gravity and quadratic velocity forces $\eta(\theta, \mathbf{V})$ are

$$\Lambda(\theta) = J^{-T}M(\theta)J^{-1},$$

$$\eta(\theta, \mathbf{V}) = J^{-T}h(\theta, J^{-1}\mathbf{V}) - \Lambda(\theta) J J^{-1} \mathbf{V}.$$

• Define two $n \times n$ projection matrices of rank $n - k$

$$P(\theta) = I - A^T(AM^{-1}A^T)^{-1}AM^{-1},$$

$$P_b(\theta) = M^{-1}PM = I - M^{-1}A^T(AM^{-1}A^T)^{-1}A$$

corresponding to the $k$ Pfaffian constraints acting on the robot, $A(\theta)\dot{\theta} = 0$, $A \in \mathbb{R}^{k \times n}$. Then the $n + k$ constrained equations of motion

$$\tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}) + A^T(\theta)\lambda,$$

$$A(\theta)\dot{\theta} = 0$$

can be reduced to these equivalent forms by eliminating the Lagrange multipliers $\lambda$:

$$P \tau = P(M\ddot{\theta} + h),$$

$$P \ddot{\theta} = P_b M^{-1}(\tau - h).$$

The matrix $P$ projects away joint force/torque components that act on the constraints without doing work on the robot, and the matrix $P_{\dot{\theta}}$ projects away acceleration components that do not satisfy the constraints.

**Chapter 9**

• A straight-line path in joint space is given by $\theta(s) = \theta_{\text{start}} + s(\theta_{\text{end}} - \theta_{\text{start}})$ as $s$ goes from 0 to 1.

• A constant-screw-axis motion of the end-effector from $X_{\text{start}} \in SE(3)$ to $X_{\text{end}}$ is $X(s) = X_{\text{start}} \exp(\log(X_{\text{start}}^{-1}X_{\text{end}})s)$ as $s$ goes from 0 to 1.
• The path-constrained dynamics of a robot can be written

\[ m(s)\ddot{s} + c(s)\dot{s}^2 + g(s) = \tau \in \mathbb{R}^n \]

as \( s \) goes from 0 to 1.
Appendix B

Other Representations of Rotations

B.1 Euler Angles

As we established earlier, the orientation of a rigid body can be parametrized by three independent coordinates. For example, consider a rigid body with a body frame \( \{b\} \) attached to it, initially aligned with the space frame \( \{s\} \). Now rotate the body by \( \alpha \) about the body \( \hat{z}_b \)-axis, then by \( \beta \) about the body \( \hat{y}_b \)-axis, and finally by \( \gamma \) about the body \( \hat{x}_b \)-axis. Then \((\alpha, \beta, \gamma)\) are the ZYX Euler angles representing the final orientation of the body (see Figure B.1). Since the successive rotations are done in the body frame, this corresponds to the final rotation matrix

\[
R(\alpha, \beta, \gamma) = I \text{Rot}(\hat{z}, \alpha) \text{Rot}(\hat{y}, \beta) \text{Rot}(\hat{x}, \gamma),
\]

where

\[
\text{Rot}(\hat{z}, \alpha) = \begin{bmatrix}
\cos \alpha & -\sin \alpha & 0 \\
\sin \alpha & \cos \alpha & 0 \\
0 & 0 & 1
\end{bmatrix}, \quad \text{Rot}(\hat{y}, \beta) = \begin{bmatrix}
\cos \beta & 0 & \sin \beta \\
0 & 1 & 0 \\
-\sin \beta & 0 & \cos \beta
\end{bmatrix},
\]

\[
\text{Rot}(\hat{x}, \gamma) = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos \gamma & -\sin \gamma \\
0 & \sin \gamma & \cos \gamma
\end{bmatrix}.
\]

Writing out the entries explicitly, we get

\[
R(\alpha, \beta, \gamma) = \begin{bmatrix}
c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\
c_\alpha s_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\
-s_\beta & c_\beta s_\gamma & c_\beta c_\gamma
\end{bmatrix}, \quad (B.1)
\]

where \( s_\alpha \) is shorthand for \( \sin \alpha \), \( c_\alpha \) for \( \cos \alpha \), etc.
Figure B.1: To understand the ZYX Euler angles, use the corner of a box or a book as the body frame. The ZYX Euler angles are successive rotations of the body about the $\hat{z}_b$-axis by $\alpha$, the $\hat{y}_b$-axis by $\beta$, and the $\hat{x}_b$-axis by $\gamma$.

We now ask the following question: given an arbitrary rotation matrix $R$, does there exist $(\alpha, \beta, \gamma)$ satisfying Equation (B.1)? In other words, can the ZYX Euler angles represent all orientations? The answer is yes, and we prove this fact constructively as follows. Let $r_{ij}$ be the $ij$-th element of $R$. Then from Equation (B.1) we know that $r_{11}^2 + r_{21}^2 = \cos^2 \beta$; as long as $\cos \beta \neq 0$, or equivalently $\beta \neq \pm 90^\circ$, we have two possible solutions for $\beta$:

$$\beta = \operatorname{atan2}\left( -r_{31}, \sqrt{r_{11}^2 + r_{21}^2} \right)$$

and

$$\beta = \operatorname{atan2}\left( -r_{31}, -\sqrt{r_{11}^2 + r_{21}^2} \right).$$

(The $\operatorname{atan2}$ two-argument arctangent is described at the beginning of Chapter 6.) In the first case $\beta$ lies in the range $[-90^\circ, 90^\circ]$, while in the second case $\beta$ lies in the range $[90^\circ, 270^\circ]$. Assuming the $\beta$ obtained above is not $\pm 90^\circ$, $\alpha$ and $\gamma$ can then be determined from the following relations:

$$\alpha = \operatorname{atan2}(r_{21}, r_{11})$$

$$\gamma = \operatorname{atan2}(r_{32}, r_{33}).$$

In the event that $\beta = \pm 90^\circ$, there exists a one-parameter family of solutions for $\alpha$ and $\gamma$. This is most easily seen from Figure B.3. If $\beta = 90^\circ$, then $\alpha$ and $\gamma$ represent rotations (in the opposite direction) about the same vertical axis. Hence, if $(\alpha, \beta, \gamma) = (\bar{\alpha}, 90^\circ, \bar{\gamma})$ is a solution for a given rotation $R$, then any triple $(\bar{\alpha}', 90^\circ, \bar{\gamma}')$ where $\bar{\alpha}' - \bar{\gamma}' = \bar{\alpha} - \bar{\gamma}$ is also a solution.
B.1. Euler Angles

B.1.1 Algorithm for Computing the ZYX Euler Angles

Given $R \in SO(3)$, we wish to find angles $\alpha, \gamma \in [0, 2\pi)$ and $\beta \in [-\pi/2, \pi/2)$ that satisfy

$$R = \begin{bmatrix}
c_{\alpha}c_{\beta} & c_{\alpha}s_{\beta}s_{\gamma} - s_{\alpha}c_{\gamma} & c_{\alpha}s_{\beta}c_{\gamma} + s_{\alpha}c_{\gamma} \\
s_{\alpha}c_{\beta} & s_{\alpha}s_{\beta}s_{\gamma} + c_{\alpha}c_{\gamma} & s_{\alpha}s_{\beta}c_{\gamma} - c_{\alpha}s_{\gamma} \\
-s_{\beta} & c_{\beta}s_{\gamma} & c_{\beta}c_{\gamma}
\end{bmatrix}, \quad (B.2)$$

where $s_{\alpha}$ is shorthand for $\sin \alpha$, $c_{\alpha}$ for $\cos \alpha$, etc. Denote by $r_{ij}$ the $ij$-th entry of $R$.

(i) If $r_{31} \neq \pm 1$, set

$$\beta = \arctan2(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}) \quad (B.3)$$

$$\alpha = \arctan2(r_{21}, r_{11}) \quad (B.4)$$

$$\gamma = \arctan2(r_{32}, r_{33}), \quad (B.5)$$

where the square root is taken to be positive.

(ii) If $r_{31} = 1$, then $\beta = \pi/2$, and a one-parameter family of solutions for $\alpha$ and $\gamma$ exists. One possible solution is $\alpha = 0$ and $\gamma = \arctan2(r_{12}, r_{22})$.

(iii) If $r_{31} = -1$, then $\beta = -\pi/2$, and a one-parameter family of solutions for $\alpha$ and $\gamma$ exists. One possible solution is $\alpha = 0$ and $\gamma = -\arctan2(r_{12}, r_{22})$.

B.1.2 Other Euler Angle Representations

The ZYX Euler angles can be visualized using the wrist mechanism shown in Figure B.2. The ZYX Euler angles $(\alpha, \beta, \gamma)$ refer to the angle of rotation about the three joint axes of this mechanism. In the figure the wrist mechanism is shown in its zero position, i.e., when all three joints are set to zero.

Four reference frames are defined as follows: frame $\{0\}$ is the fixed frame, while frames $\{1\}$, $\{2\}$, and $\{3\}$ are attached to the three links of the wrist mechanism as shown. When the wrist is in the zero position, all four reference frames have the same orientation. At the joint angles $(\alpha, \beta, \gamma)$, frame $\{1\}$ relative to $\{0\}$ is $R_{01}(\alpha) = \text{Rot}(\hat{z}, \alpha)$, and similarly $R_{12}(\beta) = \text{Rot}(\hat{y}, \beta)$ and $R_{23}(\gamma) = \text{Rot}(\hat{x}, \gamma)$. Therefore $R_{03}(\alpha, \beta, \gamma) = \text{Rot}(\hat{z}, \alpha)\text{Rot}(\hat{y}, \beta)\text{Rot}(\hat{x}, \gamma)$ as in Equation (B.1).

It should be evident that the choice of zero position for $\beta$ is, in some sense, arbitrary. That is, we could just as easily have defined the home position of the wrist mechanism to be as in Figure B.3; this would then lead to another three-parameter representation $(\alpha, \beta, \gamma)$ for $SO(3)$. In fact, Figure B.3 illustrates the ZYZ Euler angles. The resulting rotation matrix can be obtained via the following sequence of rotations, equivalent to rotating the body in Figure B.1.
first about the body's $\hat{z}_b$-axis, then about the $\hat{y}_b$-axis, then about the $\hat{z}_b$-axis:

$$R(\alpha, \beta, \gamma) = R(\hat{z}, \alpha) R(\hat{y}, \beta) R(\hat{z}, \gamma)$$

$$= \begin{bmatrix} c_\alpha & -s_\alpha & 0 \\ s_\alpha & c_\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{bmatrix} \begin{bmatrix} c_\gamma & -s_\gamma & 0 \\ s_\gamma & c_\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_\alpha c_\beta c_\gamma - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ s_\alpha c_\beta c_\gamma + c_\alpha s_\gamma & -s_\alpha c_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{bmatrix}. \quad (B.6)$$

Just as before, we can show that for every rotation $R \in SO(3)$, there exists a triple $(\alpha, \beta, \gamma)$ that satisfies $R = R(\alpha, \beta, \gamma)$ for $R(\alpha, \beta, \gamma)$ as given in Equation (B.6). (Of course, the resulting formulas will differ from those for the ZYX Euler angles.)

From the wrist mechanism interpretation of the ZYX and ZYZ Euler angles, it should be evident that for Euler angle parametrizations of $SO(3)$, what really matters is that rotation axis 1 is orthogonal to rotation axis 2, and that rotation axis 2 is orthogonal to rotation axis 3 (axis 1 and axis 3 need not necessarily be orthogonal to each other). Specifically, any sequence of rotations of the form

$$\text{Rot}(\text{axis}_1, \alpha) \cdot \text{Rot}(\text{axis}_2, \beta) \cdot \text{Rot}(\text{axis}_3, \gamma), \quad (B.7)$$

where axis1 is orthogonal to axis2, and axis2 is orthogonal to axis3, can serve as a valid three-parameter representation for $SO(3)$. The angle of rotation for...
the first and third rotations ranges in value over a \( 2\pi \) interval, while that of the second rotation ranges in value over an interval of length \( \pi \).

## B.2 Roll-Pitch-Yaw Angles

While Euler angles refer to a sequence of rotations about a body-fixed frame, the **roll-pitch-yaw angles** refer to a sequence of rotations about axes fixed in the space frame. Referring to Figure B.4, given a frame at the identity configuration (that is, \( R = I \)), we first rotate this frame by an angle \( \gamma \) about the \( \hat{x} \)-axis of the fixed frame, followed by an angle \( \beta \) about the \( \hat{y} \)-axis of the fixed frame, and finally by an angle \( \alpha \) about the \( \hat{z} \)-axis of the fixed frame.

Since the three rotations are in the fixed frame, the final orientation is

\[
R(\alpha, \beta, \gamma) = \text{Rot}(\hat{x}, \alpha) \text{Rot}(\hat{y}, \beta) \text{Rot}(\hat{z}, \gamma) I
\]

\[
= \begin{bmatrix}
  c_\alpha & -s_\alpha & 0 \\
  s_\alpha & c_\alpha & 0 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  c_\beta & 0 & s_\beta \\
  0 & 1 & 0 \\
  -s_\beta & 0 & c_\beta
\end{bmatrix}
\begin{bmatrix}
  1 & 0 & 0 \\
  0 & c_\gamma & -s_\gamma \\
  0 & s_\gamma & c_\gamma
\end{bmatrix} I
\]

\[
= \begin{bmatrix}
  c_\alpha c_\beta & c_\alpha s_\gamma s_\beta - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\
  s_\alpha c_\beta & s_\alpha s_\gamma s_\beta + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\
  -s_\beta & s_\beta s_\gamma & c_\beta c_\gamma
\end{bmatrix}.
\]

This product of three rotations is exactly the same as that for the ZYZ Euler angles given in (B.2). We see that the same product of three rotations admits
two different physical interpretations: as a sequence of rotations with respect to the body frame (ZYX Euler angles), or, reversing the order of rotations, as a sequence of rotations with respect to the fixed frame (the XYZ roll-pitch-yaw angles).

The terms roll, pitch, and yaw are often used to describe the rotational motion of a ship or aircraft. In the case of a typical fixed-wing aircraft, for example, suppose a body frame is attached such that the \( \hat{x} \)-axis is in the direction of forward motion, the \( \hat{z} \)-axis is the vertical axis pointing downward toward ground (assuming the aircraft is flying level with respect to ground), and the \( \hat{y} \)-axis extends in the direction of the wing. The roll, pitch, and yaw motions are then defined according to the XYZ roll-pitch-yaw angles \((\alpha, \beta, \gamma)\) of Equation (B.8).

### B.3 Unit Quaternions

One disadvantage of the exponential coordinates on \( SO(3) \) is that because of the division by \( \sin \theta \) in the logarithm formula, the logarithm can be numerically sensitive to small rotation angles \( \theta \). The necessary singularity of the three-parameter representation occurs at \( R = I \). The unit quaternions are an alternative representation of rotations that alleviates this singularity, but at the cost of introducing an additional fourth parameter. We now illustrate the definition and use of these coordinates.

Let \( R \in SO(3) \) have the exponential coordinate representation \( \hat{\omega} \theta \), i.e., \( R = e^{[\hat{\omega}]\theta} \), where as usual \( \|\hat{\omega}\| = 1 \) and \( \theta \in [0, \pi] \). The unit quaternion representation of \( R \) is constructed as follows. Define \( q \in \mathbb{R}^4 \) according to

\[
q = \begin{bmatrix}
q_0 \\
q_1 \\
q_2 \\
q_3 \\
\end{bmatrix} = \begin{bmatrix}
\cos \frac{\theta}{2} \\
\frac{\hat{\omega}}{2} \sin \frac{\theta}{2} \\
\end{bmatrix} \in \mathbb{R}^4.
\]

(B.9)

\( q \) as defined clearly satisfies \( \|q\| = 1 \). Geometrically, \( q \) is a point lying on the
three-dimensional unit sphere in $\mathbb{R}^4$, and for this reason the unit quaternions are also identified with the three-sphere, denoted $S^3$. Naturally among the four coordinates of $q$, only three can be chosen independently. Recalling that $1 + 2\cos \theta = \text{tr} \, R$, and using the cosine double angle formula, i.e., $\cos 2\phi = 2\cos^2 \phi - 1$, the elements of $q$ can be obtained directly from the entries of $R$ as follows:

$$q_0 = \frac{1}{2} \sqrt{1 + r_{11} + r_{22} + r_{33}}$$  \hspace{1cm} (B.10)

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \frac{1}{4q_0} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - 2r_{12} \end{bmatrix}.$$ \hspace{1cm} (B.11)

Going the other way, given a unit quaternion $(q_0, q_1, q_2, q_3)$, the corresponding rotation matrix $R$ is obtained as a rotation about the unit axis in the direction of $(q_1, q_2, q_3)$, by an angle $2\cos^{-1} q_0$. Explicitly,

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) \\ 2(q_0 q_3 + q_1 q_2) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}.$$ \hspace{1cm} (B.12)

From the above explicit formula it should be apparent that both $q \in S^3$ and its antipodal point $-q \in S^3$ produce the same rotation matrix $R$. For every rotation matrix there exists two unit quaternion representations that are antipodal to each other.

The final property of the unit quaternions concerns the product of two rotations. Let $R_q, R_p \in SO(3)$ denote two rotation matrices, with unit quaternion representations $\pm q, \pm p \in S^3$, respectively. The unit quaternion representation for the product $R_q R_p$ can then be obtained by first arranging the elements of $q$ and $p$ in the form of the following $2 \times 2$ complex matrices:

$$Q = \begin{bmatrix} q_0 + i q_1 \\ -q_2 + i q_3 \\ q_0 - i q_1 \end{bmatrix}, \hspace{1cm} P = \begin{bmatrix} p_0 + i p_1 \\ -p_2 + i p_3 \\ p_0 - i p_1 \end{bmatrix},$$ \hspace{1cm} (B.13)

where $i$ denotes the imaginary unit. Now take the product $N = QP$, where the entries of $N$ are written

$$N = \begin{bmatrix} n_0 + in_1 \\ n_2 + in_3 \\ -n_2 + in_3 \hspace{0.5cm} n_0 - in_1 \end{bmatrix}.$$ \hspace{1cm} (B.14)

The unit quaternion for the product $R_q R_p$ is then given by $\pm(n_0, n_1, n_2, n_3)$ obtained from the entries of $N$:

$$\begin{bmatrix} n_0 \\ n_1 \\ n_2 \\ n_3 \end{bmatrix} = \begin{bmatrix} q_0 p_0 - q_1 p_1 - q_2 p_2 - q_3 p_3 \\ q_0 p_1 + q_0 p_1 + q_2 p_3 - q_3 p_2 \\ q_0 p_2 + q_0 p_2 - q_1 p_3 + q_3 p_1 \\ q_0 p_3 + q_0 p_3 + q_1 p_2 - q_2 p_1 \end{bmatrix}.$$ \hspace{1cm} (B.15)
B.4 Cayley-Rodrigues Parameters

Another set of widely used local coordinates for $SO(3)$ is the Cayley-Rodrigues parameters. These parameters can be obtained from the exponential representation on $SO(3)$ as follows: given $R = e^{[\hat{\omega}]\theta}$ for some unit vector $\hat{\omega}$ and angle $\theta$, the Cayley-Rodrigues parameter $r \in \mathbb{R}^3$ is obtained by setting

$$r = \hat{\omega} \tan \frac{\theta}{2}.$$  \hfill (B.16)

Referring again to the radius $\pi$ solid ball picture of $SO(3)$, the above parametrization has the effect of infinitely “stretching” the radius of this ball via the tangent half-angle function. These parameters can be derived from a general formula attributed to Cayley that is also valid for rotation matrices of arbitrary dimension: if $R \in SO(3)$ such that $\text{tr}(R) \neq -1$, then $(I - R)(I + R)^{-1}$ is skew-symmetric. Denoting this skew-symmetric matrix by $[r]$, it is known that $R$ and $[r]$ are related as follows:

$$R = (I - [r])(I + [r])^{-1}$$ \hfill (B.17)  

$$[r] = (I - R)(I + R)^{-1}.$$ \hfill (B.18)

The above two formulas establish a one-to-one correspondence between $so(3)$ and those elements of $SO(3)$ with trace not equal to $-1$. In the event that $\text{tr}(R) = -1$, the following alternative formulas can be used to relate $SO(3)$ (this time excluding those with unit trace) and $so(3)$ in a one-to-one fashion:

$$R = -(I - [r])(I + [r])^{-1}$$ \hfill (B.19)  

$$[r] = (I + R)(I - R)^{-1}.$$ \hfill (B.20)

Equation (B.18) can furthermore be explicitly computed as

$$R = \frac{(1 - r^T r)I + 2rr^T + 2[r]}{1 + r^T r}$$ \hfill (B.21)

with its inverse mapping given by

$$[r] = \frac{R - R^T}{1 + \text{tr}(R)}.$$ \hfill (B.22)

(This formula is valid when $\text{tr}(R) \neq -1$). The vector $r = 0$ therefore corresponds to the identity matrix, and $-r$ represents the inverse of the rotation corresponding to $r$.

The following two identities also follow from the above formulas:

$$1 + \text{tr}(R) = \frac{4}{1 + r^T r}$$ \hfill (B.23)  

$$R - R^T = \frac{4[r]}{1 + r^T r}.$$ \hfill (B.24)
B.4. Cayley-Rodrigues Parameters

One of the attractive features of the Cayley-Rodrigues parameters is the particularly simple form for the composition of two rotation matrices. If \( r_1 \) and \( r_2 \) denote the Cayley-Rodrigues parameter representations of two rotations \( R_1 \) and \( R_2 \), respectively, then the Cayley-Rodrigues parameter representation for \( R_3 = R_1 R_2 \), denoted \( r_3 \), is given by

\[
r_3 = \frac{r_1 + r_2 + (r_1 \times r_2)}{1 - r_1^T r_2}
\]

(B.25)

In the event that \( r_1^T r_2 = 1 \), or equivalently \( \text{tr}(R_1 R_2) = -1 \), the following alternative composition formula can be used. Define

\[
s = \frac{r}{\sqrt{1 + r^T r}}
\]

(B.26)

so that the rotation corresponding to \( r \) can be written

\[
R = I + 2 \sqrt{1 - s^T s} \left[ s \right] + 2 [s]^2.
\]

(B.27)

The direction of \( s \) coincides with that of \( r \), and \( \| s \| = \sin \frac{\theta}{2} \). The composition law now becomes

\[
s_3 = s_1 \sqrt{1 - s_2^T s_2} + s_2 \sqrt{1 - s_1^T s_1} + (s_1 \times s_2)
\]

(B.28)

Angular velocities and accelerations also admit a simple form in terms of the Cayley-Rodrigues parameters. If \( r(t) \) denotes the Cayley-Rodrigues parameter representation of the orientation trajectory \( R(t) \), then in vector form

\[
\omega_s = \frac{2}{1 + \|r\|^2} (r \times \dot{r} + \dot{r})
\]

(B.29)

\[
\omega_b = \frac{2}{1 + \|r\|^2} (-r \times \dot{r} + \dot{r}).
\]

(B.30)

The angular acceleration with respect to the inertial and body-fixed frames can now be obtained by time-differentiating the above expressions:

\[
\ddot{\omega}_s = \frac{2}{1 + \|r\|^2} (r \times \ddot{r} + \ddot{r} - r^T \dot{r} \omega_s)
\]

(B.31)

\[
\ddot{\omega}_b = \frac{2}{1 + \|r\|^2} (-r \times \ddot{r} + \ddot{r} - r^T \dot{r} \omega_b).
\]

(B.32)
Other Representations of Rotations
Appendix C

Denavit-Hartenberg Parameters and Their Relationship to the Product of Exponentials

C.1 Denavit-Hartenberg Representation

The basic idea underlying the Denavit-Hartenberg approach to forward kinematics is to attach reference frames to each link of the open chain, and to derive the forward kinematics based on knowledge of the relative displacements between adjacent link frames. Assume that a fixed reference frame has been established, and that a reference frame (the end-effector frame) has been attached to some point on the last link of the open chain. For a chain consisting of \( n \) one degree of freedom joints, the links are numbered sequentially from 0 to \( n \), in which the ground link is labeled 0, and the end-effector frame is attached to link \( n \). Reference frames attached to the links are also correspondingly labeled from \( \{0\} \) (the fixed frame) to \( \{n\} \) (the end-effector frame). The joint variable corresponding to the \( i \)-th joint is denoted \( \theta_i \). The forward kinematics of the \( n \)-link open chain can then be expressed as

\[
T_{0n}(\theta_1, \ldots, \theta_n) = T_{01}(\theta_1)T_{12}(\theta_2) \cdots T_{n-1,n}(\theta_n),
\]

where \( T_{i,i-1} \in SE(3) \) denotes the relative displacement between link frames \( \{i-1\} \) and \( \{i\} \). Depending on how the link reference frames have been chosen, each \( T_{i,i-1} \) can be obtained in a straightforward fashion.
C.1.1 Assigning Link Frames

Rather than attaching reference frames to each link in arbitrary fashion, in the Denavit-Hartenberg convention a set of rules for assigning link frames is observed. Figure C.1 illustrates the frame assignment convention for two adjacent revolute joints $i-1$ and $i$ that are connected by link $i-1$.

The first rule is that the $\hat{z}_i$-axis coincides with joint axis $i$, and $\hat{z}_{i-1}$ coincides with joint axis $i-1$. The direction of each link frame’s $\hat{z}$-axis is determined via the right-hand rule, i.e., such that positive rotations are counterclockwise about the $\hat{z}$-axis.

Once the $\hat{z}$-axis direction has been assigned, the next rule determines the origin of the link reference frame. First, find the line segment that orthogonally intersects both joint axes $\hat{z}_{i-1}$ and $\hat{z}_i$. For now let us assume that this line segment is unique; the case where it is not unique (i.e., when the two joint axes are parallel), or fails to exist (i.e., when the two joint axes intersect), is addressed later. Connecting joint axes $i-1$ and $i$ by a mutually perpendicular line, the origin of frame $\{i-1\}$ is then located at the point where this line intersects joint axis $i-1$.

Determining the remaining $\hat{x}$- and $\hat{y}$-axes of each link reference frame is now straightforward: the $\hat{x}$ axis is chosen to be in the direction of the mutually perpendicular line pointing from the $i-1$ axis to the $i$ axis. The $\hat{y}$-axis is then uniquely determined from the cross-product $\hat{x} \times \hat{y} = \hat{z}$. Figure C.1 depicts the
Having assigned reference frames in this fashion for links $i$ and $i-1$, we now define four parameters that exactly specify $T_{i-1,i}$:

- The length of the mutually perpendicular line, denoted by the scalar $a_{i-1}$, is called the **link length** of link $i-1$. Despite its name, this link length does not necessarily correspond to the actual length of the physical link.
- The **link twist** $\alpha_{i-1}$ is the angle from $\hat{z}_{i-1}$ to $\hat{z}_i$, measured about $\hat{x}_{i-1}$.
- The **link offset** $d_i$ is the distance from the intersection of $\hat{x}_{i-1}$ and $\hat{z}_i$ to the link $i$ frame origin (the positive direction is defined to be along the $\hat{z}_i$ axis).
- The **joint angle** $\phi_i$ is the angle from $\hat{x}_{i-1}$ to $\hat{x}_i$, measured about the $\hat{z}_i$-axis.

These parameters constitute the Denavit-Hartenberg parameters. For an open chain with $n$ one-degree-of-freedom joints, the $4n$ Denavit-Hartenberg parameters are sufficient to completely describe the forward kinematics. In the case of an open chain with all joints revolute, the link lengths $a_{i-1}$, twists $\alpha_{i-1}$, and offset parameters $d_i$ are all constant, while the joint angle parameters $\phi_i$ act as the joint variables.

We now consider the case where the mutually perpendicular line is undefined or fails to be unique, as well as when some of the joints are prismatic, and finally, how to choose the ground and end-effector frames.

**When Adjacent Revolute Joint Axes Intersect**

If two adjacent revolute joint axes intersect each other, then the mutually perpendicular line between the joint axes fails to exist. In this case the link length is set to zero, and we choose $\hat{x}_{i-1}$ to be perpendicular to the plane spanned by $\hat{z}_{i-1}$ and $\hat{z}_i$. There are two possibilities here, both of which are acceptable: one leads to a positive value of the twist angle $\alpha_{i-1}$, while the other leads to a negative value.

**When Adjacent Revolute Joint Axes are Parallel**

The second special case occurs when two adjacent revolute joint axes are parallel. In this case there exist many possibilities for a mutually perpendicular line, all of which are valid (more precisely, a one-dimensional family of mutual perpendicular lines is said to exist). A useful guide is to try to choose the mutually perpendicular line that is the most physically intuitive, and simplifies as many Denavit-Hartenberg parameters as possible (e.g., such that their values become zero).
Figure C.2: Link frame assignment convention for prismatic joints. Joint $i - 1$ is a revolute joint, while joint $i$ is a prismatic joint.

**Prismatic Joints**

For prismatic joints, the $\hat{z}$-direction of the link reference frame is chosen to be along the positive direction of translation. This convention is consistent with that for revolute joints, in which the $\hat{z}$-axis indicates the positive axis of rotation. With this choice the link offset $d_i$ is the joint variable and the joint angle $\phi_i$ is constant (see Figure C.2). The procedure for choosing the link frame origin, as well as the remaining $\hat{x}$ and $\hat{y}$-axes, remains the same as for revolute joints.

**Assigning the Ground and End-Effector Frames**

Our frame assignment procedure described thus far does not specify how to choose the ground and final link frames. Here as before, a useful guideline is to choose initial and final frames that are the most physically intuitive, and simplify as many Denavit-Hartenberg parameters as possible. This usually implies that the ground frame is chosen to coincide with the link 1 frame in its zero (rest) position; in the event that the joint is revolute, this choice forces $a_0 = \alpha_0 = d_1 = 0$, while for a prismatic joint we have $a_0 = \alpha_0 = \phi_1 = 0$. The end-effector frame is attached to some reference point on the end-effector, usually at a location that makes the description of the task intuitive and natural, and also simplifies as many of the Denavit-Hartenberg parameters as possible (e.g., their values become zero).

It is important to realize that arbitrary choices of the ground and end-effector frames may not always be possible, since there may not exist a valid set of Denavit-Hartenberg parameters to describe the relative transformation. We elaborate on this point below.
C.1. Denavit-Hartenberg Representation

515

Figure C.3: An example of three frames \{a\}, \{b\}, and \{c\}, in which the transformations \(T_{ab}\) and \(T_{ac}\) cannot be described by any set of Denavit-Hartenberg parameters.

C.1.2 Why Four Parameters are Sufficient

In our earlier study of spatial displacements, we argued that a minimum of six independent parameters were required to describe the relative displacement between two frames in space: three for the orientation, and three for the position. Based on this result, it would seem that for an \(n\)-link arm, a total of \(6n\) parameters would be required to completely describe the forward kinematics (each \(T_{i-1,i}\) in the above equation would require six parameters). Surprisingly, in the Denavit-Hartenberg parameter representation only four parameters are required for each transformation \(T_{i-1,i}\). Although this result may at first appear to contradict our earlier results, this reduction in the number of parameters is accomplished by the carefully stipulated rules for assigning link reference frames. If the link reference frames had been assigned in arbitrary fashion, then more parameters would have been required.

Consider, for example, the link frames shown in Figure C.3. The transformation from frame \{a\} to frame \{b\} is a pure translation along the \(\hat{y}\)-axis of frame \{a\}. If one were to try to express the transformation \(T_{ab}\) in terms of the Denavit-Hartenberg parameters \((\alpha, a, d, \theta)\) as prescribed above, it should become apparent that no such set of parameter values exist. Similarly, the transformation \(T_{ac}\) also does not admit a description in terms of Denavit-Hartenberg parameters, as only rotations about the \(\hat{x}\) - and \(\hat{z}\) - axes are permissible. Under our Denavit-Hartenberg convention, only rotations and translations along the \(\hat{x}\) and \(\hat{z}\) axes are allowed, and no combination of such motions can achieve the transformation shown in Figure C.3.

Given that the Denavit-Hartenberg convention uses exactly four parameters to describe the transformation between link frames, one might naturally wonder if the number of parameters can be reduced even further, by an even more clever set of link frame assignment rules. Denavit and Hartenberg show that this is not possible, and that four is the minimum number of parameters [24].

We end this section with a reminder that there are alternative conventions
Denavit-Hartenberg Parameters and Their Relationship to the Product of Exponentials

for assigning link frames. Whereas we chose the $\hat{z}$-axis to coincide with the joint axis, some authors choose the $\hat{x}$-axis, and reserve the $\hat{z}$-axis to be the direction of the mutually perpendicular line. To avoid ambiguities in the interpretation of the Denavit-Hartenberg parameters, it is essential to include a concise description of the link frames together with the parameter values.

C.1.3 Manipulator Forward Kinematics

Once all the transformations $T_{i-1,i}$ between adjacent link frames are known in terms of their Denavit-Hartenberg parameters, the forward kinematics is obtained by sequentially multiplying these link transformations. Each link frame transformation is of the form

$$
T_{i-1,i} = \text{Rot}(\hat{x}, \alpha_{i-1})\text{Trans}(\hat{x}, a_{i-1})\text{Trans}(\hat{z}, d_i)\text{Rot}(\hat{z}, \phi_i)
$$

where

$$
\text{Rot}(\hat{x}, \alpha_{i-1}) = 
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & \cos \alpha_{i-1} & -\sin \alpha_{i-1} & 0 \\
0 & \sin \alpha_{i-1} & \cos \alpha_{i-1} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
$$

$$
\text{Trans}(\hat{x}, a_{i-1}) = 
\begin{bmatrix}
1 & 0 & 0 & a_{i-1} \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
$$

$$
\text{Trans}(\hat{z}, d_i) = 
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & d_i \\
0 & 0 & 0 & 1
\end{bmatrix}
$$

$$
\text{Rot}(\hat{z}, \phi_i) = 
\begin{bmatrix}
\cos \phi_i & -\sin \phi_i & 0 & 0 \\
\sin \phi_i & \cos \phi_i & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
$$

A useful way to visualize $T_{i,i-1}$ is to transport frame $\{i-1\}$ to frame $\{i\}$ via the following sequence of four transformations:

(i) Rotate frame $\{i-1\}$ about its $\hat{x}$ axis by an angle $\alpha_{i-1}$.

(ii) Translate this new frame along its $\hat{x}$ axis by a distance $a_{i-1}$.

(iii) Translate this new frame along its $\hat{z}$ axis by a distance $d_i$.

(iv) Rotate this new frame about its $\hat{z}$ axis by an angle $\phi_i$. 

Note that switching the order of the first and second steps will not change the final form of $T_{i-1,i}$. Similarly, the order of the third and fourth steps can also be switched without affecting $T_{i-1,i}$.

C.1.4 Examples

We now derive the Denavit-Hartenberg parameters for some common spatial open chain structures.

Example: A 3R Spatial Open Chain

Consider the 3R spatial open chain of Figure 4.3, shown in its zero position (i.e., with all its joint variables set to zero). The assigned link reference frames are shown in the figure, and the corresponding Denavit-Hartenberg parameters listed in the following table:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\alpha_{i-1}$</th>
<th>$a_{i-1}$</th>
<th>$d_i$</th>
<th>$\phi_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$\theta_1$</td>
</tr>
<tr>
<td>2</td>
<td>$90^\circ$</td>
<td>$L_1$</td>
<td>0</td>
<td>$\theta_2 - 90^\circ$</td>
</tr>
<tr>
<td>3</td>
<td>$-90^\circ$</td>
<td>$L_2$</td>
<td>0</td>
<td>$\theta_3$</td>
</tr>
</tbody>
</table>

Note that frames \{1\} and \{2\} are uniquely specified from our frame assignment convention, but that we have some latitude in choosing frames \{0\} and \{3\}. Here we choose the ground frame \{0\} to coincide with frame \{1\} (resulting in $\alpha_0 = a_0 = d_1 = 0$), and frame \{3\} such that $\hat{x}_3 = \hat{x}_2$ (resulting in no offset to the joint angle $\theta_3$).

Example: A Spatial RRRP Open Chain

The next example we consider is the four degree-of-freedom RRRP spatial open chain of Figure C.4, here shown in its zero position. The link frame assignments
are as shown, and the corresponding Denavit-Hartenberg parameters are listed in the following table:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\alpha_{i-1}$</th>
<th>$a_{i-1}$</th>
<th>$d_i$</th>
<th>$\phi_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$\theta_1$</td>
</tr>
<tr>
<td>2</td>
<td>$90^\circ$</td>
<td>0</td>
<td>0</td>
<td>$\theta_2$</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>$L_2$</td>
<td>0</td>
<td>$\theta_3 + 90^\circ$</td>
</tr>
<tr>
<td>4</td>
<td>$90^\circ$</td>
<td>0</td>
<td>0</td>
<td>$\theta_4$</td>
</tr>
</tbody>
</table>

The four joint variables are $(\theta_1, \theta_2, \theta_3, \theta_4)$, where $\theta_4$ is the displacement of the prismatic joint. As in the previous example, the ground frame $\{0\}$ and final link frame $\{4\}$ have been chosen to make as many of the Denavit-Hartenberg parameters zero.

**Example: A Spatial 6R Open Chain**

The final example we consider is a widely used six 6R robot arm (Figure C.5). This open chain has six rotational joints: the first three joints function as a Cartesian positioning device, while the last three joints act as a $XYZ$ Euler angle-type wrist. The link frames are shown in the figure, and the corresponding Denavit-Hartenberg parameters are listed in the following table:

<table>
<thead>
<tr>
<th>$i$</th>
<th>$\alpha_{i-1}$</th>
<th>$a_{i-1}$</th>
<th>$d_i$</th>
<th>$\phi_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>$\theta_1$</td>
</tr>
<tr>
<td>2</td>
<td>$90^\circ$</td>
<td>0</td>
<td>0</td>
<td>$\theta_2$</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>$L_3$</td>
<td>0</td>
<td>$\theta_3 + 90^\circ$</td>
</tr>
<tr>
<td>4</td>
<td>$90^\circ$</td>
<td>0</td>
<td>$L_2$</td>
<td>$\theta_4 + 180^\circ$</td>
</tr>
<tr>
<td>5</td>
<td>$90^\circ$</td>
<td>0</td>
<td>0</td>
<td>$\theta_5 + 180^\circ$</td>
</tr>
<tr>
<td>6</td>
<td>$90^\circ$</td>
<td>0</td>
<td>0</td>
<td>$\theta_6$</td>
</tr>
</tbody>
</table>
C.1. Denavit-Hartenberg Representation

C.1.5 Relation between the Product of Exponential and the Denavit-Hartenberg Representations

The product of exponentials formula can be derived directly from the Denavit-Hartenberg parameter-based representation of the forward kinematics. As before, denote the relative displacement between adjacent link frames by

\[ T_{i-1,i} = \text{Rot}(\hat{x}, \alpha_{i-1}) \text{Trans}(\hat{x}, a_{i-1}) \text{Trans}(\hat{z}, d_i) \text{Rot}(\hat{z}, \phi_i). \]

If joint \( i \) is revolute, the first three matrices can be regarded as constant, and \( \phi_i \) becomes the revolute joint variable. Define \( \theta_i = \phi_i \), and

\[ M_i = \text{Rot}(\hat{x}, \alpha_{i-1}) \text{Trans}(\hat{x}, a_{i-1}) \text{Trans}(\hat{z}, d_i), \quad (C.6) \]

and write \( \text{Rot}(\hat{z}, \theta_i) \) as the following matrix exponential:

\[ \text{Rot}(\hat{z}, \theta_i) = e^{[A_i] \theta_i}, \quad [A_i] = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (C.7) \]

With the above definitions we can write

\[ T_{i-1,i} = M_i e^{[A_i] \theta_i}. \]

If joint \( i \) is prismatic, then \( d_i \) becomes the joint variable, \( \phi_i \) is a constant parameter, and the order of \( \text{Trans}(\hat{z}, d_i) \) and \( \text{Rot}(\hat{z}, \phi_i) \) in \( T_{i-1,i} \) can be reversed (recall that reversing translations and rotations taken along the same axis still results in the same motion). In this case we can still write \( T_{i-1,i} = M_i e^{[A_i] \theta_i} \), where \( \theta_i = d_i \) and

\[ M_i = \text{Rot}(\hat{x}, \alpha_{i-1}) \text{Trans}(\hat{x}, a_{i-1}) \text{Rot}(\hat{z}, \phi_i) \quad (C.8) \]

\[ [A_i] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \quad (C.9) \]

Based on the above, for an \( n \)-link open chain containing both revolute and prismatic joints, the forward kinematics can be written

\[ T_{0,n} = M_1 e^{[A_1] \theta_1} M_2 e^{[A_2] \theta_2} \cdots M_n e^{[A_n] \theta_n} \quad (C.10) \]

where \( \theta_i \) denotes joint variable \( i \), and \( [A_i] \) is either of the form (C.7) or (C.9) depending on whether joint \( i \) is revolute or prismatic.

We now make use of the matrix identity \( M_1e^P M_1^{-1} = e^{MPM^{-1}} \), which holds for any nonsingular \( M \in \mathbb{R}^{n \times n} \) and arbitrary \( P \in \mathbb{R}^{n \times n} \). The above can also be rearranged as \( M e^P = e^{MPM^{-1}} M \). Beginning from the left of Equation (C.10), if we repeatedly apply this identity, after \( n \) iterations we obtain the product of exponentials formula as originally derived:

\[ T_{0n} = e^{M_1[A_1] M_1^{-1} \theta_1 (M_1 M_2) [A_2] \theta_2 \cdots (M_1 M_2 M_3) [A_n] \theta_n} \]

\[ = e^{[S_1] \theta_1} \cdots e^{[S_n] \theta_n} M, \quad (C.11) \]
where

$$[S_i] = (M_1 \cdots M_{i-1})[A_i](M_1 \cdots M_{i-1})^{-1}, \quad i = 1, \ldots, n \quad (C.12)$$

$$M = M_1 M_2 \cdots M_n. \quad (C.13)$$

We now re-examine the physical meaning of the $S_i$ by recalling how a screw twist transforms under a change of reference frames. If $S_a$ represents the screw twist for a given screw motion with respect to frame $\{a\}$, and $S_b$ represents the screw twist for the same physical screw motion but this time with respect to frame $\{b\}$, then recall that $S_a$ and $S_b$ are related by

$$[S_b] = T_{ba}[S_a]T_{ba}^{-1}, \quad (C.14)$$

or using the adjoint notation $\text{Ad}_{T_{ba}}$,

$$S_b = \text{Ad}_{T_{ba}}(S_a). \quad (C.15)$$

Seen from the perspective of this transformation rule, Equation (C.13) suggests that $A_i$ is the screw twist for joint axis $i$ as seen from link frame $\{i\}$, while $S_i$ is the screw twist for joint axis $i$ as seen from the fixed frame $\{0\}$. 
Bibliography


Index

ambiguity, 30

center of rotation, 10

centre of rotation, 10

constraint
    active, 4
    impenetrability, 4
    rolling, 4

contact
    roll-slide, 4
    rolling, 4
    sliding, 4

contact kinematics, 2

contact label, 4

contact mode, 7

contact normal, 2

convex span, 15

CoR, 10

Coulomb friction, 21

dynamic grasp, 31

exceptional objects, 16

force closure, 25

form closure, 7, 14

form-closure grasp, 14

friction angle, 22

friction coefficient, 21

friction cone, 22

frictionless point contact, 10

grasp metric, 19

hard-finger contact, 10

homogeneous constraint, 7

inconsistent, 30

jammed, 35

linear span, 14

moment labeling, 24

nonnegative linear combinations, 14

polyhedral convex cone, 7

polyhedral convex set, 6

polyhedron, 6

positive span, 14

quasistatic, 31

reciprocal force and velocity, 4

repelling force and velocity, 4

roll-slide motion, 3, 4

soft-finger contact, 10

wedged, 35