INTRODUCTION TO ROBOTICS

MECHANICS, PLANNING, AND CONTROL

F. C. PARK AND K. M. LYNCH

Contents

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 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 refine complex motions. 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.

In contrast to the lofty goals set by robotics researchers, the aims of this textbook are more modest. Our focus will be on the mechanics, planning and control of **robot mechanisms**. Robot arms are one familiar example. So are wheeled platforms, as are robot arms mounted on wheeled platforms. 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 serial-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 a serial chain, all of the joints are actuated, while in the case of mechanisms with closed loops only a subset of the 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 by pneumatic or hydraulic cylinders, or even by 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
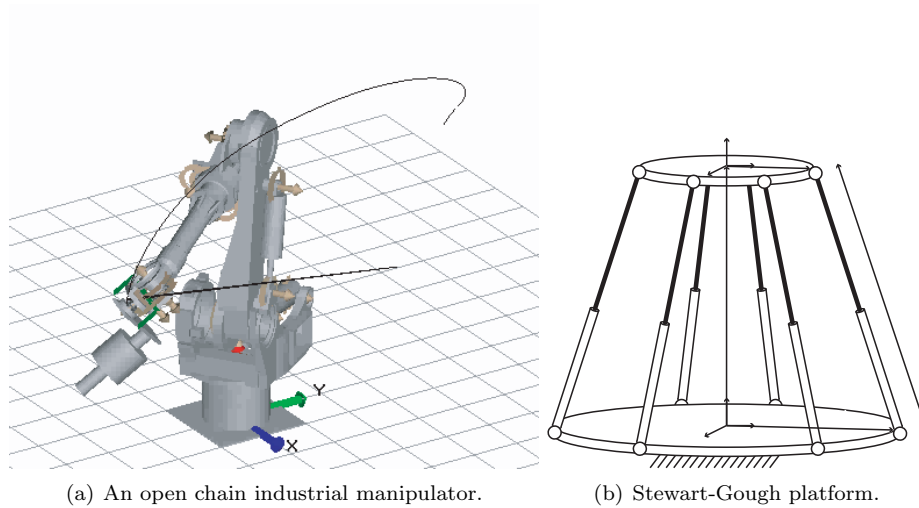
(a) An open chain industrial manipulator.   (b) Stewart-Gough platform.

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 ignore these effects 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 a serial chain robot such as the industrial manipulator of Figure 1.1(a), all of the 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 serial 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 a frame attached to 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 cover some preliminaries on the analysis of velocities and accelerations of particles with respect to moving frames. We then introduce the 3×3 matrix representation for describing a frame's orientation; such a matrix is referred to as a **rotation matrix**. Two well-known three-parameter representations for rotation matrices, the Euler angles and roll-pitch-yaw angles, are described.

We then introduce the exponential representation for rotations. This representation, which can also be identified with the familiar angle-axis representation for rotations, is derived in a somewhat roundabout way as the solution

to a certain linear vector differential equation. Doing so allows us to, among other things, proceed directly to the exponential description of general rigid body motions, which forms the cornerstone for our later kinematic analysis of serial chains.

The exponential description of rigid body motions can also be identified with classical screw theory. In addition to the basic rules for the matrix representation and manipulation of rigid body motions, we also cover in detail the linear algebraic constructs of screw theory, including the unified description of linear and angular velocities as six-dimensional **spatial velocities**. Analogously, it is also natural to combine three-dimensional forces and moments into a six-dimensional **spatial force**.

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 study two methods for describing the forward kinematics of open chains: the Denavit-Hartenberg (D-H) representation and the product-of-exponentials (PoE) formula. The D-H representation uses a fewer number of parameters, but requires that reference frames be attached to each link. The PoE formula requires more parameters, but there is no need to attach reference frames to each link. Instead it relies solely on information about the location of each joint axis—essentially, a line in space—making this the preferred forward kinematic representation for our subsequent analysis in the later chapters.

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 this chapter we show how to obtain the Jacobian for general serial chains, and its many practical uses in the above and other settings.

Chapter 6: Inverse Kinematics

The **inverse kinematics** problem is to determine the set of joint positions that achieves a desired end-effector configuration. For serial chain robots, the inverse kinematics is in general more involved than the forward kinematics: for a given set of joint values there usually exists 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 special class of six-dof serial 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 serial chains. We also examine the inverse kinematics of redundant serial chains (that is, those with more than seven 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 serial chains have unique forward kinematics solutions, closed chains often have multiple forward kinematics solutions. Also, because closed chains possess both actuated and passive joints, the kinematic singularity analysis of closed chains presents unusual subtleties not encountered in serial 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

This chapter derives the dynamic equations for serial chains. As a first step, the dynamic equations for a single rigid body are derived in terms of spatial velocities, accelerations, and forces. The dynamics for a serial chain robot are then derived by applying the single body equation to each link of the robot. Analogous to the notions of forward and inverse kinematics, the **forward dynamics** problem involves determining the resulting joint trajectory for a given input joint torque profile. Similarly, the **inverse dynamics** problem is concerned with determining the input joint torque profile for a desired joint trajectory. The main result of this chapter is a set of recursive algorithms for the forward and inverse dynamics problem. Unlike traditional methods that rely on a separate analysis of the linear and angular components of the dynamic equations, our algorithms are formulated entirely in terms of six-dimensional spatial quantities, and rely upon the product-of-exponentials formula to model the kinematics.

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. Often this input 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 smooth trajectory for each joint that satisfies various user-supplied conditions.

We study some popular algorithms for trajectory generation that were originally developed for computer-aided curve design applications. Algorithms that are of particular interest are cubic splines and Bézier curves. Versions of these algorithms are offered for the generation of trajectories in both joint space and end-effector space. In the latter case, algorithms are presented for interpolating through an ordered set of end-effector reference frames.

Chapter 10: Motion Planning

This chapter addresses the problem of finding a collision-free path for a robot through a cluttered workspace. The most intuitive approach is to work in configuration space, which because of the workspace obstacles have forbidden regions. We begin with the most basic of planning algorithms involving grid search. A convenient way to navigate through such cluttered regions is through the use of **artificial potential fields**. Briefly, in these methods the obstacles generate artificial forces that repel the robot should it venture too near. When they work, these methods have the advantage of being able to generate collision-free trajectories in real-time. There are a number of subtleties associated with these methods, however, e.g., the possibility of getting "stuck" in local equilibria without reaching the goal configuration, and we discuss some methods to overcome these difficulties. We also cover a basic randomized algorithm for planning collision-free paths, based on **rapidly-exploring random trees**.

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 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 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 equations 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 system. 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 assume a dynamic model of the robot is available; such **feedforward control** techniques use the dynamic model of the robot and its environment to determine 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. 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, which we considered earlier in the chapter on force closure, at least in the case for point contacts. We examine the equations of motion for

rigid body mechanics with friction, formulate the general problem of manipulation plannning and grasping, and examine simplifications and assumptions that lead to certain basic solutions.

Chapter 13: Wheeled Mobile 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 omnidirectional 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 typical robot is mechanically constructed from several bodies, or **links**, that are connected by various types of **joints**. The robot moves when **actuators** (such as electric motors) deliver forces or torques to the joints. Usually an **end-effector**, such as a gripper or hand for grasping and manipulating objects, is attached to some link of the robot. All of the robots considered in this book have links that can be modeled as rigid bodies.

Given a particular robot, perhaps the most fundamental question one can ask is "where is the robot?"[1] The answer to this question 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, only a few numbers are needed to represent the configuration.[2] For example, to represent the configuration of a door, we need only one number, the angle θ that the door rotates about its hinge. The configuration of a point on a plane can be described by two coordinates, (x, y). To represent the configuration of a coin lying heads up on a flat table, we need three coordinates: two specifying the location (x, y) on the table of a particular point on the coin, and one specifying the coin's orientation, θ. (See Figure 2.1.)

The minimum number of real-valued coordinates needed to represent the configuration is the number of **degrees of freedom** (**dof**) of the robot. Thus a coin (viewed as a robot) lying on a table has three degrees of freedom. If the coin could lie heads up or tails up, the configuration space still has only three degrees of freedom, since the fourth variable, representing which side of the coin is up, can only take values in the discrete set {heads, tails}; it does not take a continuous range of real values as required by our definition.

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

---

[1]In the sense of "where are the links of the robot situated?"
[2]Compare with trying to represent the configuration of a pillow.
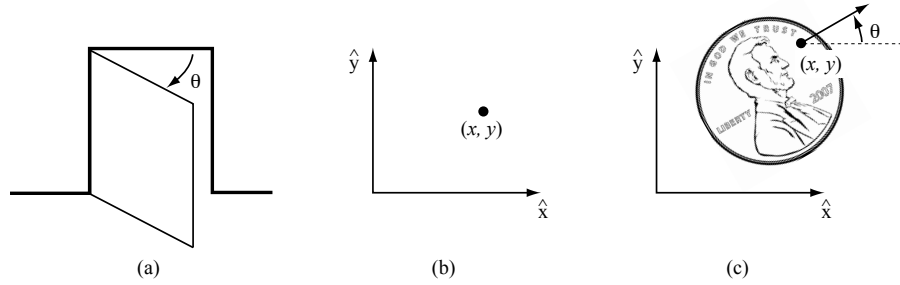
Figure 2.1: (a) The configuration of a door is described by its angle θ. (b) The configuration of a point in a plane is described by coordinates (x, y). (c) The configuration of a coin on a table is described by (x, y, θ).

configurations of the robot is called the **configuration space** (or **C-space** for short).

In this chapter we study the C-space and degrees of freedom of general robots. Since our robots are constructed of rigid bodies, we first examine the degrees of freedom of a rigid body. We then examine the number of degrees of freedom of general robots. Next we discuss the shape of curved C-spaces and general issues in representing these spaces. 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 rigid bodies.

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 fixed to the coin (Figure 2.2(a)). Once we attach a coordinate frame $\hat{x}$-$\hat{y}$ 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 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 the configuration of the coin; similarly, the distances $d(B, C)$ and $d(A, C)$ must be constant. The following equality constraints are therefore imposed on the coordinates (x_A, y_A), (x_B, y_B), and (x_C, y_C):

$$d(A, B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2} = d_{AB}$$
$$d(B, C) = \sqrt{(x_B - x_C)^2 + (y_B - y_C)^2} = d_{BC}$$
$$d(A, C) = \sqrt{(x_A - x_C)^2 + (y_A - y_C)^2} = d_{AC}.$$

To determine the number of degrees of freedom of the coin, we first choose the position of point A in the plane (Figure 2.2(b)). We may choose it to be
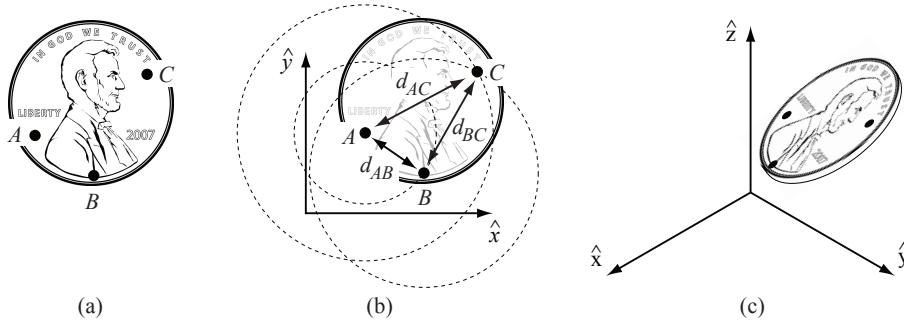
Figure 2.2: (a) Choosing three points fixed to the penny. (b) Once the location of A is chosen, B must lie on a circle of radius d_{AB} centered at A. Once the location of B is chosen, C must lie at the intersection of circles centered at A and B. Only one of these two intersections corresponds to the "heads up" configuration. (c) The configuration of a penny 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.

anything we want, so we have two degrees of freedom to specify, (x_A, y_A). Once this is specified, however, 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 ϕ_{AB}, and define it to be the angle that the vector $\overrightarrow{AB}$ makes with the $\hat{x}$-axis.

Finally, once we have chosen the location of point B, there are only two possible locations of 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, ϕ_{AB}).

Suppose we were to choose an additional point on the coin, D, thus introducing 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*, and adds no new information—only two of the three constraints are independent. The two freedoms apparently introduced by the coordinates (x_D, y_D) are immediately eliminated by the two independent constraints. The same would hold for any other point on the coin, so 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:

$$\text{Degrees of freedom} = (\text{Sum of freedoms of the points}) -$$
$$(\text{Number of independent constraints}). \qquad (2.1)$$

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

$$\text{Degrees of freedom} = (\text{Number of variables}) -$$
$$(\text{Number of independent equations}). \qquad (2.2)$$

We can use this general rule to determine the number of freedoms of a rigid body in three dimensions as well. For example, assume our coin is no longer confined to the table (Figure 2.2(c)). The three points A, B, and C are now defined by (x_A, y_A, z_A), (x_B, y_B, z_B), and (x_C, y_C, z_C). 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. The intersection of two spheres is a circle, and thus 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 six-parameter representations for the six freedoms 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, we express Equation (2.1) as follows:

$$\text{Degrees of freedom} = (\text{Sum of freedoms of the bodies}) -$$
$$(\text{Number of independent constraints}). \qquad (2.3)$$

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 θ. 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 (θ). Alternatively, we can view the door from above and regard it as a planar body, which has three degrees of freedom. The hinge joint then imposes two independent constraints, again leaving only one indepedent coordinate (θ). Its C-space is represented by some range in the interval $[0, 2\pi)$ over which θ is allowed to vary.

In both cases we see that joints have the effect of constraining the motion of the rigid body, and thus reducing the overall degrees of freedom. It seems plausible that a formula can be obtained for determining the degrees of freedom of a robot, simply by counting the number of rigid bodies and joints. This is in fact the case, and in this section we derive Grübler's formula for determining the degrees of freedom of planar and spatial robots.

2.2.1   Degrees of Freedom of a Robot

Robot Joints

Figure 2.3 illustrates the basic joints found in typical robots. Every joint connects exactly two links; we do not allow joints that simultaneously connect three or more links. 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 constructed by serially connecting a pair of revolute joints 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, or it can 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 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
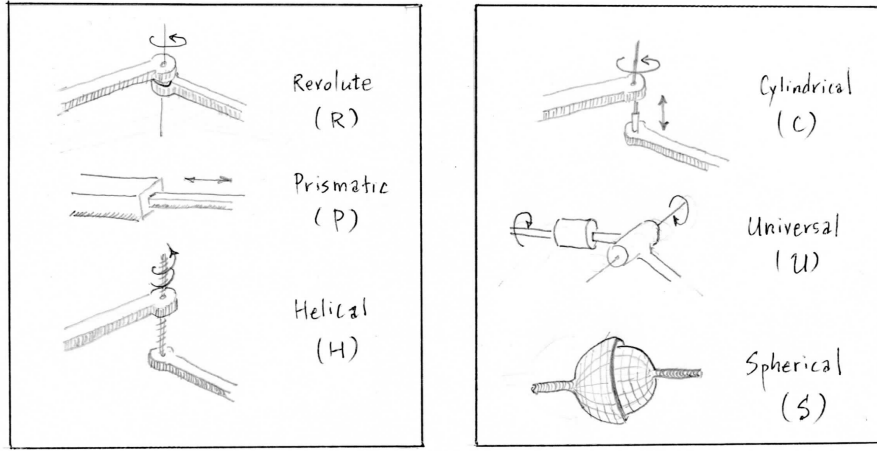
Figure 2.3: Typical robot joints.

joint type	dof	constraints between two planar rigid bodies	constraints between two spatial rigid bodies
revolute (R)	1	2	5
prismatic (P)	1	2	5
screw (H)	1	N/A	5
cylindrical (C)	2	N/A	4
universal (U)	2	N/A	4
spherical (S)	3	N/A	3

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

the joint.

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

Grübler's Formula

The number of degrees of freedom of a mechanism with links and joints can be calculated using **Grübler's formula**, an expression of Equation (2.3).

Proposition 2.1. *Consider a robot 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 (three for planar mechanisms and six for spatial mechanisms), f_i be the number of freedoms provided by joint i, and c_i be the*
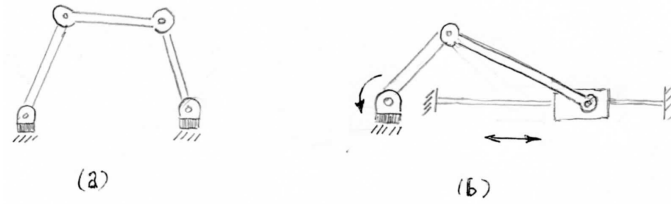
Figure 2.4: (a) Four-bar linkage. (b) Slider-crank mechanism.

number of constraints provided by joint i (where $f_i + c_i = m$). Then Grübler's formula for the number of degrees of freedom (dof) of the robot is

$$
\text{dof} = \underbrace{m(N-1)}_{\text{rigid body freedoms}} - \underbrace{\sum_{i=1}^{J} c_i}_{\text{joint constraints}}
$$

$$
= m(N-1) - \sum_{i=1}^{J}(m - f_i)
$$

$$
= m(N-1-J) + \sum_{i=1}^{J} f_i. \tag{2.4}
$$

This formula only holds if all joint constraints are independent. If they may not be 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 two types of mechanisms: **open-chain mechanisms** (also known as **serial mechanisms**) and **closed-chain mechanims** (also known as **parallel mechanisms**). A closed-chain mechanism is any mechanism that has a closed loop; an example is the closed loop from the ground, to your right leg, to your waist, to your left leg, to the ground when you are standing with both feet on the ground. An open-chain mechanism is any mechanism without a closed loop; an example is your arm when your hand is in free space.

Example 2.1. Four-bar linkage and slider-crank mechanism
Consider the four-bar linkage shown in Figure 2.4(a). This planar closed-chain mechanism consists of four links—one of them ground—connected in a single closed loop by four revolute joints. Since the mechanism and the rigid bodies are planar, $m = 3$. Subsituting $N = 4$, $J = 4$, and $f_i = 1, i = 1, \ldots, 4$, into Grübler's formula, it can be determined 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 joint ($J = 4$, and each $f_i = 1$) and four links ($N = 4$, including the ground
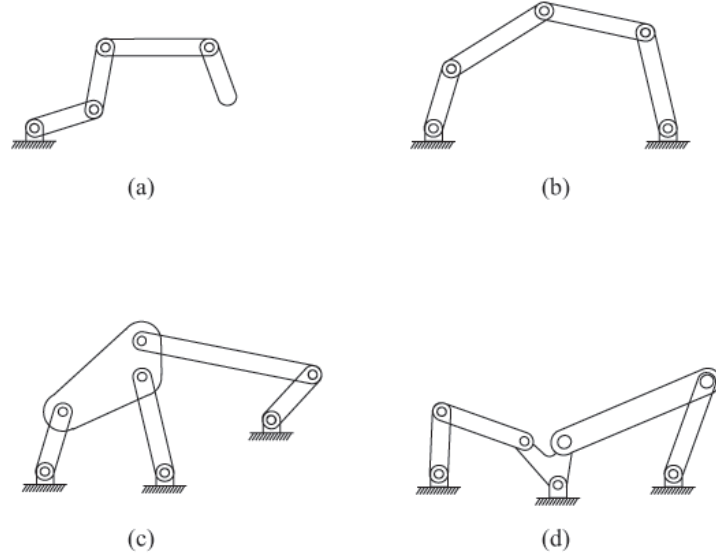
Figure 2.5: (a) k-link planar serial chain. (b) Five-bar planar linkage. (c) Stephenson six-bar linkage. (d) Watt six-bar linkage.

link), or (ii) the mechanism consists of two revolute joints ($f_i = 1$) and one RP joint (the RP joint is a concatenation of a revolute and prismatic joint, so that $f_i = 2$) and three links ($N = 3$; remember that each joint connects precisely two bodies). In both cases the mechanism has one degree of freedom.

Example 2.2. Some classical planar mechanisms
Let us now apply Grübler'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 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

$$\text{dof} = 3(6 - 1 - 7) + 7 = 1.$$
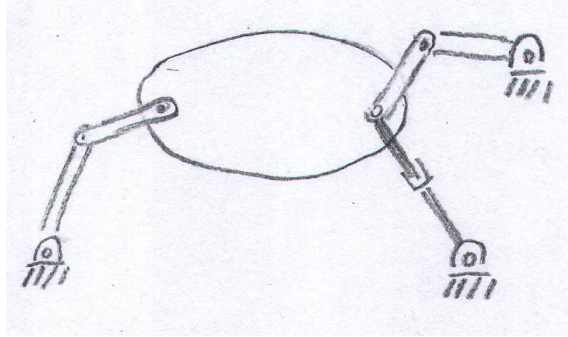
Figure 2.6: A planar mechanism with two overlapping joints.

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,

$$\text{dof} = 3(6 - 1 - 7) + 7 = 1.$$

Example 2.3. A planar mechanism with overlapping joints

Consider the planar mechanism illustrated in Figure 2.6. Again, there is more than one way to derive the number of degrees of freedom using Grübler's formula. If all the joints are regarded as either revolute or prismatic, then the mechanism consists of eight links ($N = 8$), eight revolute joints, and one prismatic joint. (Note that three bodies meet at a single point on the right. Recalling that a joint connects exactly two links, the joint at this intersection point should not be regarded as a single revolute joint. Rather, it should be correctly interpreted as two revolute joints overlapping each other.) Substituting into Grübler's formula,

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

Alternatively, the revolute-prismatic pair connected to the body 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. For this interpretation, Grübler's formula yields

$$\text{dof} = 3(7 - 1 - 8) + 7(1) + 1(2) = 3.$$

Example 2.4. Grübler's formula and singularities

Consider the parallelogram linkage of Figure 2.7(a). Here $N = 5$, $J = 6$, and $f_i = 1$ for each link, and from Grübler's formula, the number of 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 collinear as implied by the figure, the mechanism can in fact move, with one degree of freedom. Any one of the three parallel links is redundant with the other two.
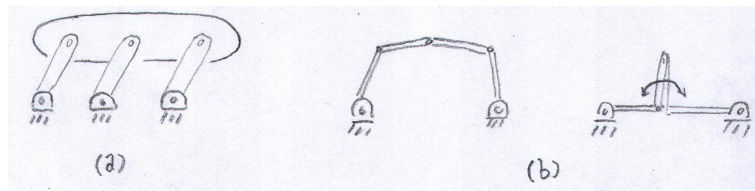
Figure 2.7: (a) A parallelogram linkage; (b) The five-bar linkage in a regular and singular configuration.
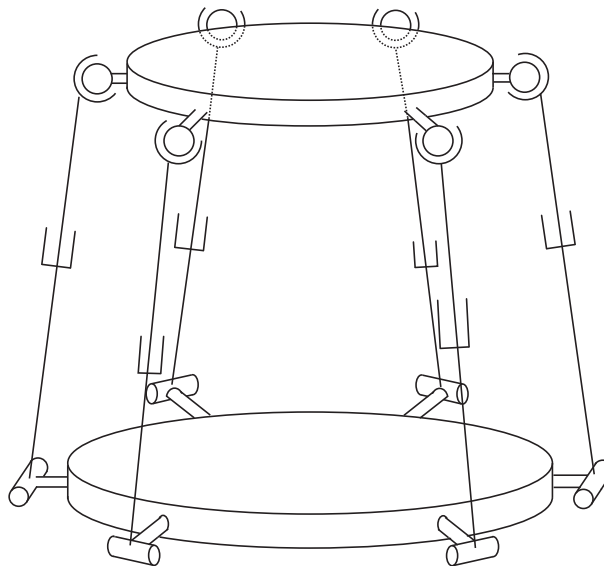
Figure 2.8: The Stewart-Gough Platform.

A similar situation occurs for the five-bar linkage of Figure 2.7(b). If the two joints connected to ground are fixed, then the five-bar linkage should become a rigid structure, given that the mechanism has two degrees of freedom by Grübler's formula. Note, however, that if the two middle links overlap each other as shown in the figure, then these links are able to rotate freely about the two overlapping joints. Of course, the link lengths need to be chosen in such a way that this configuration is feasible. If a different pair of joints is fixed, then the mechanism does become a rigid structure as expected.

Grübler's formula provides a lower bound for the number of degrees of freedom for singular cases like those just described. Configuration space singularities arising in closed chains are discussed in detail in the later chapter on closed-chain kinematics.

Example 2.5. Stewart-Gough Platform
The Stewart-Gough platform of Figure 2.8 consists of two platforms—the lower
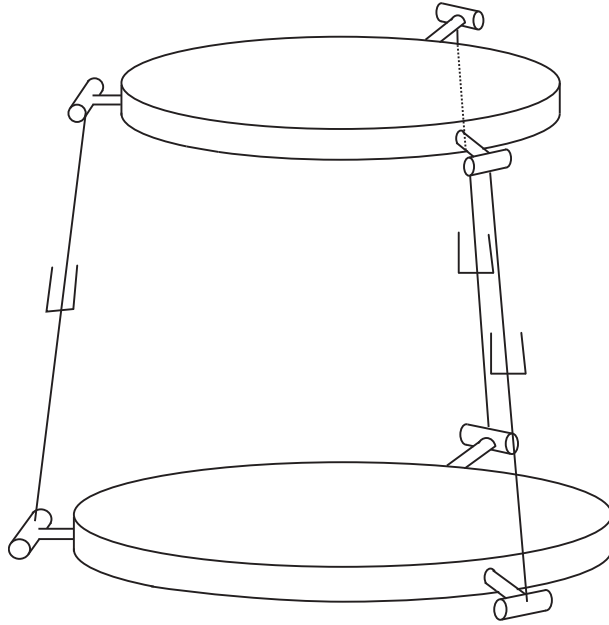
Figure 2.9: The $3 \times UPU$ platform.

one stationary, the upper one mobile—connected by six legs. Each leg consists of a universal-prismatic-spherical (*UPS*) joint arrangement. The total number of links (including the fixed lower platform, which is regarded as the ground link) in this closed-chain mechanism is 14 ($N = 14$). 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 Grübler'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. Grübler's formula then indicates that this mechanism has twelve degrees of freedom. This result may seem surprising but it is in fact correct; 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.

Example 2.6. $3 \times UPU$ platform
The next example, shown in Figure 2.9, is also a closed-chain platform structure, in which the two platforms are connected by three UPU legs. The total number of links is $N = 8$. There are six universal joints and three prismatic joints,

making for a total of $J = 9$ joints. Substituting these values into Grübler's formula leads to

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

The $3 \times UPU$ mechanism has three degrees of freedom according to Grübler's formula, but constructed prototypes of this mechanism reveal extra degrees of freedom not predicted by Grübler's formula. These and other subtle aspects of the $3 \times UPU$ platform are further discussed in Chapter 7.

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.

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[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. The topology of a space is independent of how we choose to represent the space.

Topologically distinct one-dimensional spaces include the line, a closed interval of the line, and a circle. The line can be written $\mathbb{E}$ or $\mathbb{E}^1$, indicating a one-dimensional Euclidean "flat" space. Since a point in $\mathbb{E}^1$ is so commonly represented by a real number (after choosing an origin and a length scale), it is

---

[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 space.

Figure 2.10: 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.

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 2.10. A closed interval is not topologically equivalent to a line, since a line does not contain endpoints.) The circle is written S or S^1.

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.

Some C-spaces can be expressed as the **Cartesian product** of lower-dimensional spaces, meaning that points in the 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 $(x, y) \in \mathbb{R}^2$ and $\theta \in S^1$.

- 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 $\mathbb{R}^3 \times S^2 \times S^1$.
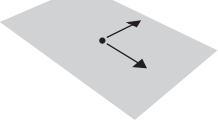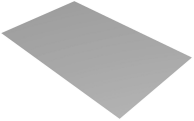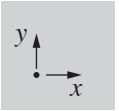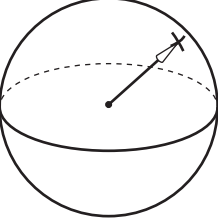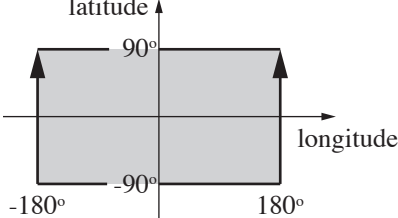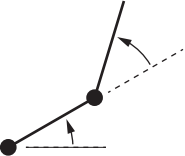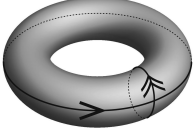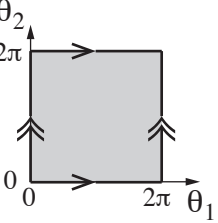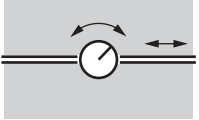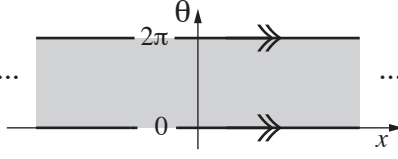
system	C-space	sample representation
point on a plane	$\mathbb{E}^2$	$\mathbb{R}^2$
tip of spherical pendulum	2-sphere S^2	$[-180°, 180°) \times [-90°, 90°]$
2R robot arm	2-torus $T^2 = S^1 \times S^1$	$[0, 2\pi) \times [0, 2\pi)$
rotating sliding knob	cylinder $\mathbb{R}^1 \times S^1$	$\mathbb{R}^1 \times [0, 2\pi)$

Table 2.2: Four topologically different two-dimensional C-spaces and example coordinate representations. In the latitude-longitude representation of the sphere, the latitudes $-90°$ and $90°$ each correspond to a single point (the South Pole and the North Pole, respectively), and the longitude parameter wraps around at $180°$ and $-180°$: 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 θ_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.

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, 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., $[-90°, 90°]$ for latitude and $[-180°, 180°)$ for longitude for a sphere, where, on Earth, negative values correspond to "South" and "West," respectively).

The latitude-longitude representatation of a sphere is dissatisfying if you are walking near the North Pole (latitude equals $90°$) or South Pole (latitude equals $-90°$), 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. The location of these singularities has nothing to do with the space 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 constant speed.

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 $\phi \in [-90°, 90°]$ and longitude $\psi \in [-180°, 180°)$, and alternative coordinates (ϕ', ψ') in a rotated coordinate frame, where the alternative latitude ϕ' is $90°$ at the "East Pole" and $-90°$ at the "West Pole." Then the first coordinate chart can be used when $-90° + \epsilon < \phi < 90° - \epsilon$, for some small $\epsilon > 0$, and the second coordinate chart can be used when $-90° + \epsilon < \phi' < 90° - \epsilon$.

  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 one-dimensional unit circle can be viewed as a curve embedded in a two-dimensional Euclidean space. An implicit representation uses the coordinates of the higher-dimensional space (e.g., (x, y) in the two-dimensional space), but subjects these coordinates to constraints that reduce the number of degrees of freedom (e.g., $x^2 + y^2 = 1$ for the unit circle).

  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 circle is represented by a smoothly changing (x, y), a single representation over the whole circle.

  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). C-spaces of closed-chain mechanisms can have complex geometry and may not even have constant dimension. An example is the mechanism of Figure 2.7(b), which gains a degree of freedom at the singularity shown.

  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[4]), 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.[5]

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.

---

[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.
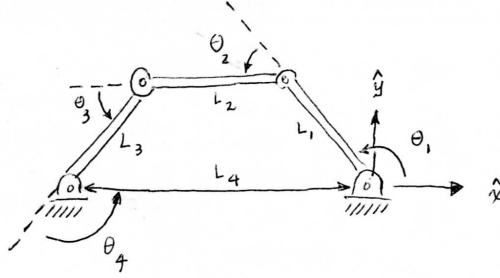
Figure 2.11: The four-bar linkage.

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 2.11, 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:

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

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 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

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

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 **holonomic constraints**, constraints that reduce the dimension of the C-space, such as the rigid-body constraints discussed earlier when deriving the degrees of freedom of a rigid body. The C-space can be viewed as a surface of dimension $n-k$ (assuming all constraints are independent) embedded in $\mathbb{R}^n$.
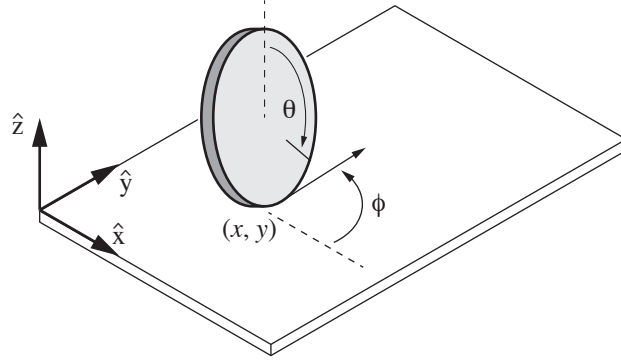
Figure 2.12: A coin rolling on a plane without slipping.

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

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

$$\begin{bmatrix} \frac{\partial g_1}{\partial \theta_1}(\theta)\dot\theta_1 + \ldots + \frac{\partial g_1}{\partial \theta_n}(\theta)\dot\theta_n \\ \vdots \\ \frac{\partial g_k}{\partial \theta_1}(\theta)\dot\theta_1 + \ldots + \frac{\partial g_k}{\partial \theta_n}(\theta)\dot\theta_n \end{bmatrix} = 0$$

$$\begin{bmatrix} \frac{\partial g_1}{\partial \theta_1}(\theta) & \cdots & \frac{\partial g_1}{\partial \theta_n}(\theta) \\ \vdots & \ddots & \vdots \\ \frac{\partial g_k}{\partial \theta_1}(\theta) & \cdots & \frac{\partial g_k}{\partial \theta_n}(\theta) \end{bmatrix} \begin{bmatrix} \dot\theta_1 \\ \vdots \\ \dot\theta_n \end{bmatrix} = 0$$

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

Here $\dot\theta_i$ denotes the time derivative of θ_i with respect to time t, $\frac{\partial g}{\partial \theta}(\theta) \in \mathbb{R}^{k\times n}$, 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

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

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 **Pfaffian constraints**. For the case of $A(\theta) = \frac{\partial g}{\partial \theta}(\theta)$, 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 **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 2.12. The configuration of the coin is given by the contact point (x, y) on the plane, the steering angle ϕ, and the angle of rotation (see Figure 2.12). 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 ϕ and θ. This C-space is four-dimensional.

Let us 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 $r\dot\theta$:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = r\dot\theta \begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}. \tag{2.9}$$

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

$$\begin{bmatrix} 1 & 0 & 0 & -r\cos q_3 \\ 0 & 1 & 0 & -r\sin q_3 \end{bmatrix} \dot{q} = 0. \tag{2.10}$$

These are Pfaffian constraints of the form $A(q)\dot{q} = 0$, $A(q) \in \mathbb{R}^{2\times4}$.

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:

$$
\begin{array}{rcllcl}
\frac{\partial g_1}{\partial q_1} & = & 1 & \longrightarrow & g_1(q) = q_1 + h_1(q_2, q_3, q_4) \\
\frac{\partial g_1}{\partial q_2} & = & 0 & \longrightarrow & g_1(q) = h_2(q_1, q_3, q_4) \\
\frac{\partial g_1}{\partial q_3} & = & 0 & \longrightarrow & g_1(q) = h_3(q_1, q_2, q_4) \\
\frac{\partial g_1}{\partial q_4} & = & -r\cos q_3 & \longrightarrow & g_1(q) = -rq_4\cos q_3 + h_4(q_1, q_2, q_3),
\end{array}
$$

for some h_i, $i = 1, \dots, 4$, 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.[6]

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.

---

[6]Some texts define the number of degrees of freedom of a system to be the dimension of 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.
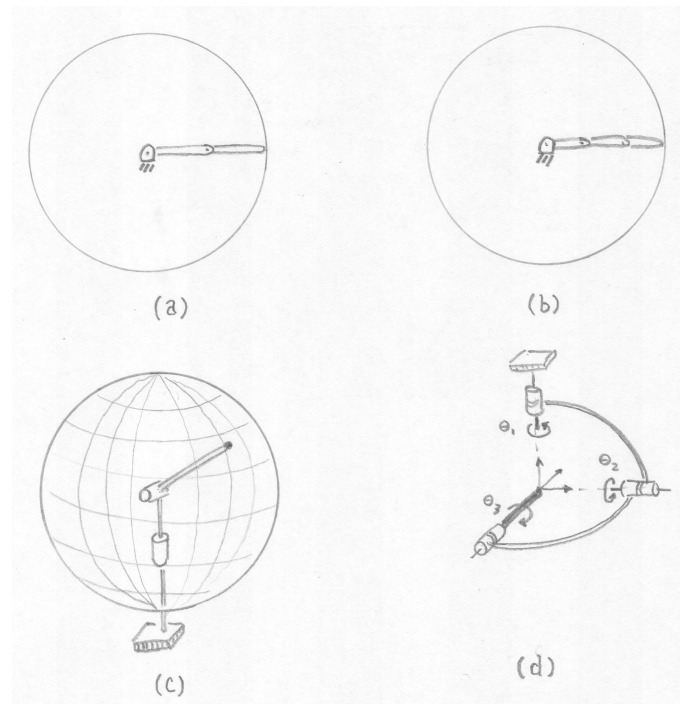
Figure 2.13: Examples of task spaces for various robots: (a) a planar $2R$ open chain; (b) a planar $3R$ open chain; (c) a spherical $2R$ open chain; (d) a $3R$ orienting mechanism.

2.5 Workspace and Task Space

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 the 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 rigid bodies, 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 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 cannot 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 2.13(a)) and the planar $3R$ open chain with links of equal length two (Figure 2.13(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 2.13(a) has a planar disk as its workspace, while the $2R$ open chain of Figure 2.13(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 2.13(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 2.14 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, ϕ), where (x, y, z) denotes the Cartesian position of the end-effector center point, and ϕ 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 2.15. 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
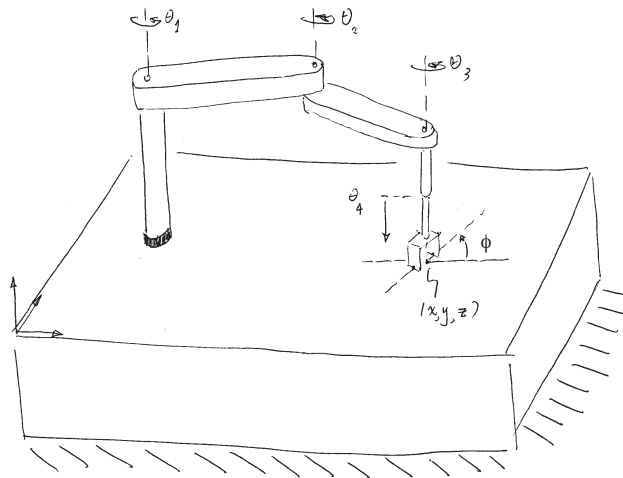
Figure 2.14: SCARA robot.
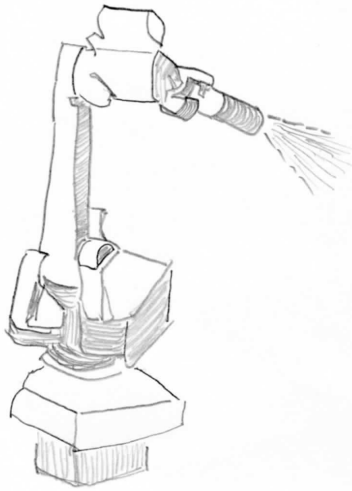
Figure 2.15: A spray-painting robot.

Cartesian position of the nozzle and spherical coordinates (θ, ϕ) to describe the direction in which the nozzle is pointing. The task space can be written as $\mathbb{R}^3 \times S^2$. The workspace could be the reachable points in $\mathbb{R}^3 \times S^2$, or, to simplify visualization, the user could define the workspace to be the subset of $\mathbb{R}^3$ corresponding to the reachable Cartesian positions of the nozzle.

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 is 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**,

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

  where m is three for planar mechanisms or six 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** 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 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 the C-space of its end-effector, or a subspace of the C-space as needed for the particular task.

Notes and References

In the classical kinematics literature, structures that consist of links connected by joints are called **mechanisms** or **linkages**. The degrees of freedom of mechanisms is treated in most texts on mechanism analysis and design, e.g., [9]. In general, a robot's kinematic configuration space has the mathematical structure of a differentiable manifold. Some accessible introductions to differential geometry are [22], [8]. Configuration spaces are further examined in a motion planning context in [14], [5].

2.7 Exercises

1. Use the planar version of Grübler's formula to determine the mobility of the mechanisms shown in Figure 2.16.

(a)

(b)

(c)

(d)

(e)

(f)
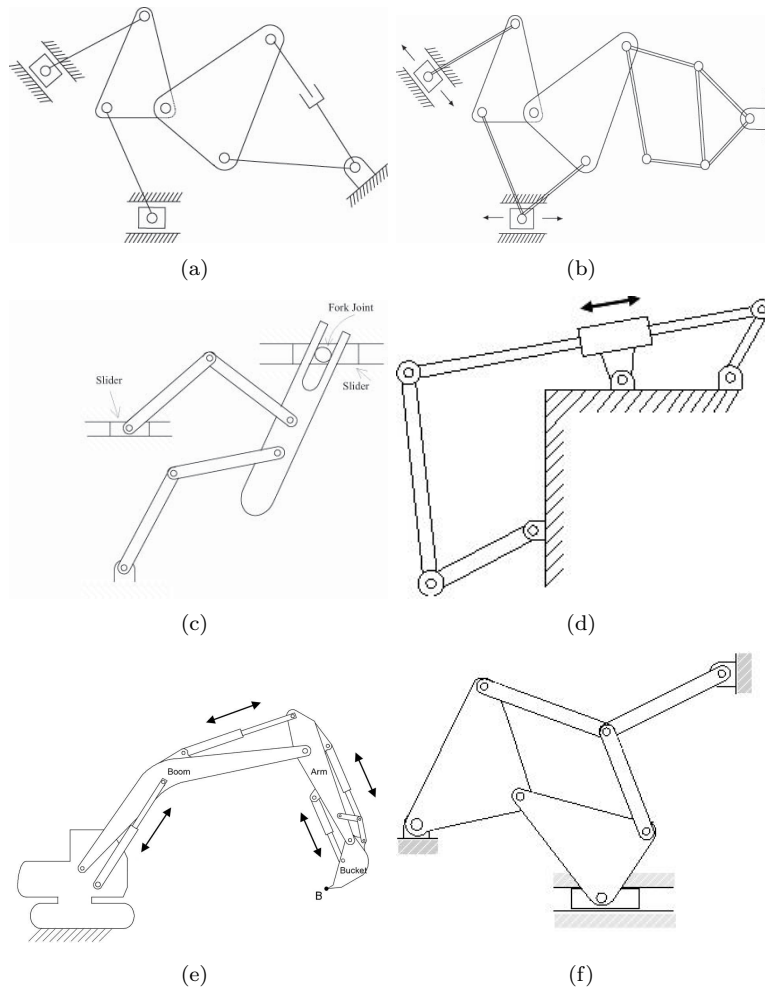
Figure 2.16: A collection of planar mechanisms.

2. (a) Use an appropriate version of Grübler's formula to determine the mobility of the $6 \times RUS$ platform of Figure 2.17.
(b) What happens to the mobility if the universal joints are replaced by spherical joints?
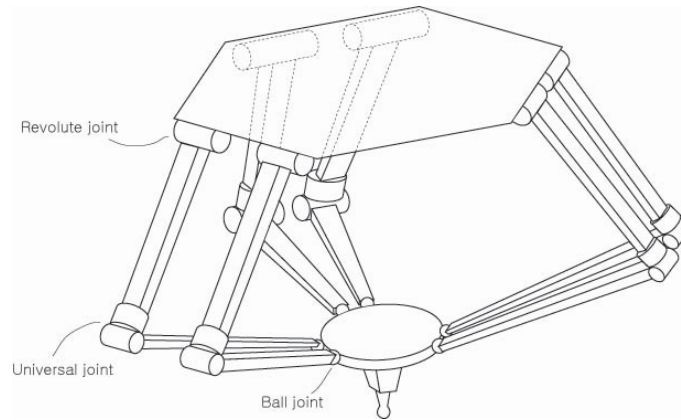
Figure 2.17: $6 \times RUS$ platform.

3. Use Grübler's formula to determine the mobility of the six parallel mechanisms shown in Figure 2.18.

4. Three identical open chain SRS arms are grasping an object as shown in Figure 2.19.
(a) Find the mobility of this system of cooperating robots.
(b) Suppose there are now a total of n 7-dof open chain arms grasping the object. What is the mobility of this system of robots?
(c) Suppose the spherical wrist joint in each of the n seven degree of freedom open chains is replaced by a universal joint. What is the mobility of the overall system?

5. Determine the mobility of the chain formed by the human arm and robot arm as shown in Figure 2.20.

6. Consider a spatial parallel mechanism consisting of a moving plate connected to the fixed plate by n identical open chains. In order for the moving plate to have mobility six, how many kinematic 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 the above for arbitrary n.

7. Consider the parallel mechanism shown in Figure 2.21, in which six legs of identical length are connected to a fixed and moving platform via spherical joints. Determine the mobility of this mechanism using Grübler's formula, and

(a)

(b)

(c)

(d)

(e)

(f)
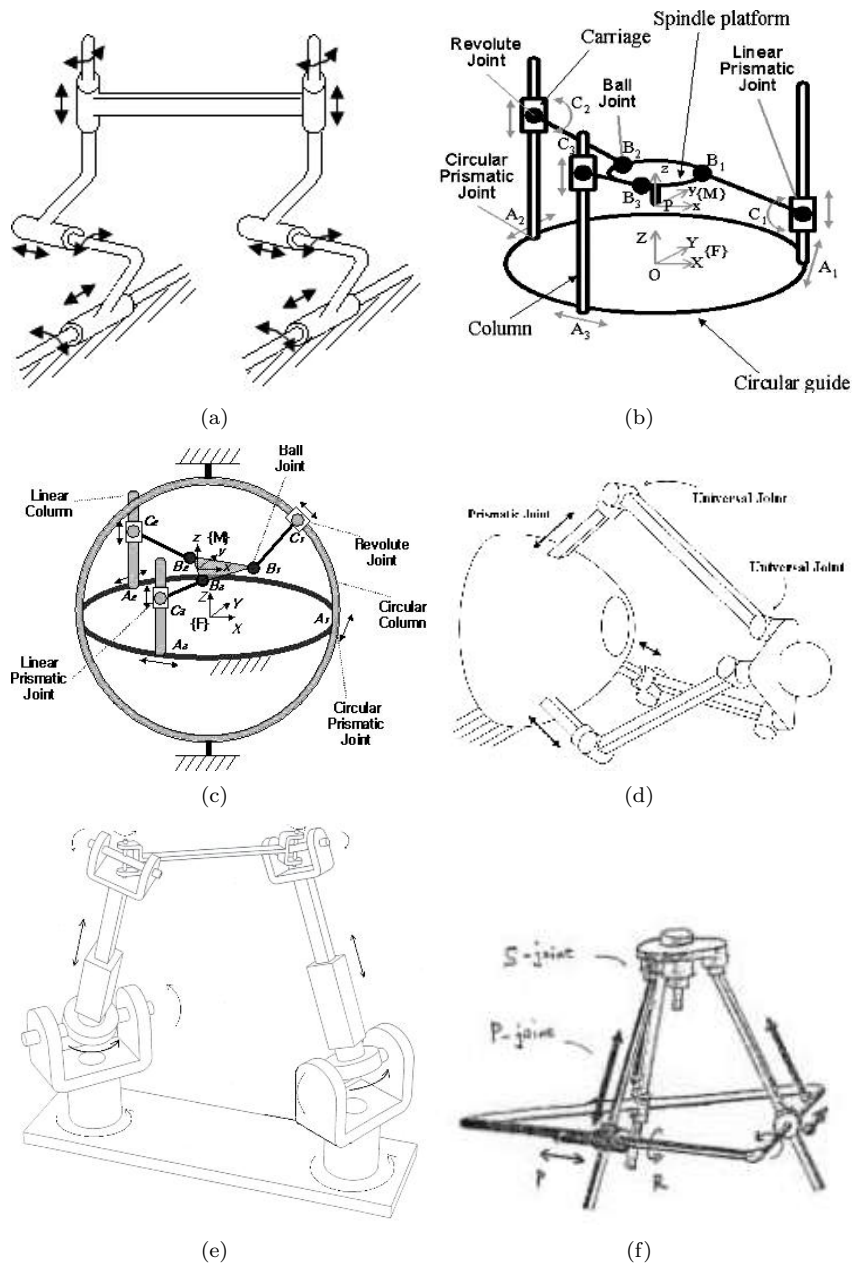
Figure 2.18: A collection of spatial parallel mechanisms.

illustrate all possible motion types of the upper platform.

8. The mobile manipulator of Figure 2.22 consists of a $6R$ arm and multifin-
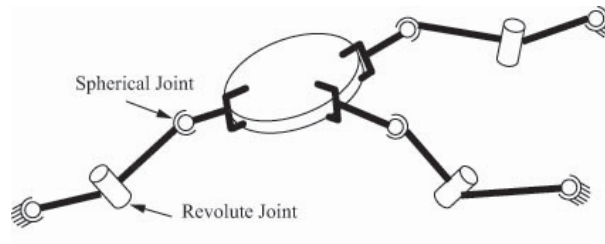
Figure 2.19: Three cooperating open chain arms grasping a common object.
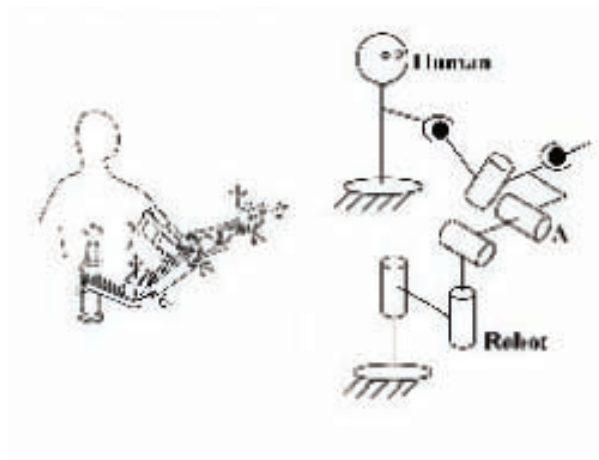
Figure 2.20: Human-robot cooperating device.

gered hand mounted on a mobile base with a single-wheel. The wheel rotates without slip, and its axis of rotation is always parallel to the ground.

(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 opens the door using only its arm. Assume the wheel is locked in place, so that the mobile base does not move. After the door has been opened, what is the **mobility** of the mechanism formed by the arm and open door?

(c) A second identical mobile manipulator comes along, and after parking its mobile base, also rigidly grasps the refrigerator door handle. What is the **mobility** of the mechanism formed by the two arms and the open referigerator door?

9. In this exercise we consider the configuration space of the four-bar linkage of Figure 2.23, this time projected onto the space of the two joints attached to the ground link. Label the joints and link lengths, and attach a fixed reference
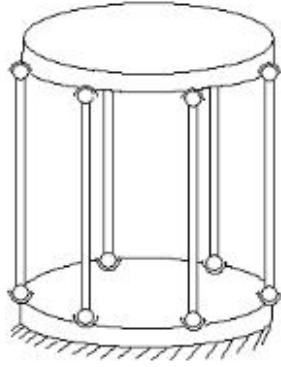
Figure 2.21: A $6 \times SS$ platform.
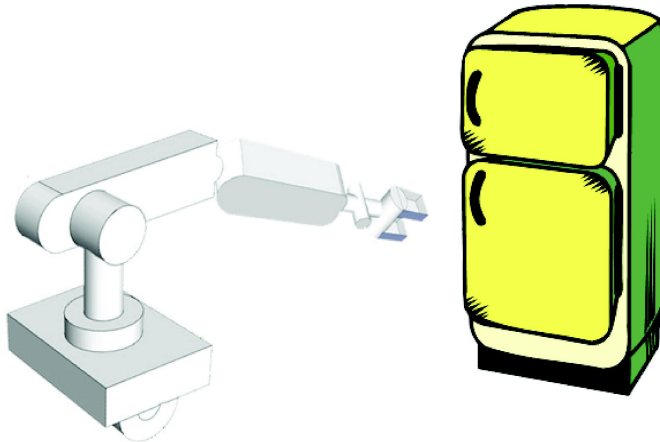
Figure 2.22: Mobile manipulator for Exercise 9.

frame as shown in the figure. It follows that

$$
\begin{aligned}
A(\theta) &= (a\cos\theta,\ a\sin\theta)^T \\
B(\psi) &= (g + b\cos\psi,\ b\sin\psi)^T
\end{aligned}
$$

From $\|A(\theta) - B(\psi)\|^2 = h^2$ for all values of θ and ψ, we get

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

Grouping the coefficients of $\cos\psi$ and $\sin\psi$, the above equation can be expressed
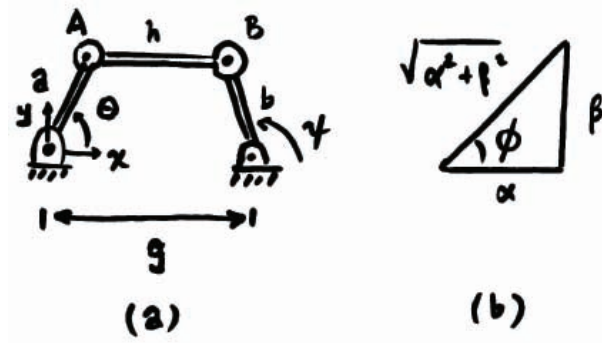
Figure 2.23: Planar four-bar linkage.

in the form

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

where

$$
\begin{aligned}
\alpha(\theta) &= 2gb - 2ab\cos\theta \\
\beta(\theta) &= -2ab\sin\theta \\
\gamma(\theta) &= h^2 - g^2 - b^2 - a^2 + 2ag\cos\theta.
\end{aligned}
$$

We wish to obtain ψ as a function of θ. To do so, first divide both sides of Equation 9 by $\sqrt{\alpha^2 + \beta^2}$ to obtain

$$\frac{\alpha}{\sqrt{\alpha^2 + \beta^2}}\cos\psi + \frac{\beta}{\sqrt{\alpha^2 + \beta^2}}\sin\psi = \frac{\gamma}{\sqrt{\alpha^2 + B^2}}$$

Now, referring to Figure 2.23-(b), the angle ϕ is given by $\phi = \tan^{-1}(\frac{\beta}{\alpha})$, and Equation 9 becomes

$$\cos(\psi - \phi) = \frac{\gamma}{\sqrt{\alpha^2 + \beta^2}}.$$

Therefore,

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

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

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

(c) Draw the configuration space of the mechanism in θ-ψ 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: $a = 1$, $b = 1$, $h = 1$, $g = \sqrt{3}$.
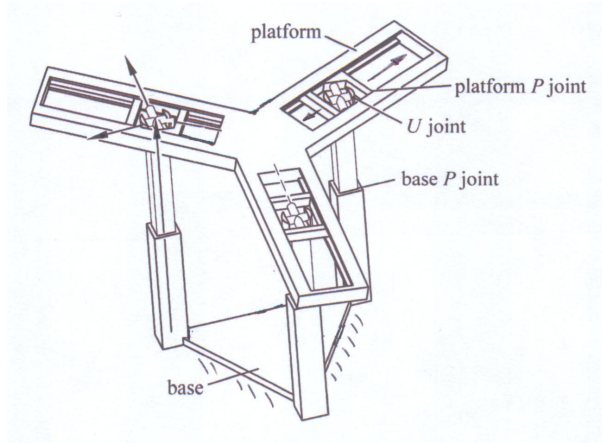
Figure 2.24: The $3 \times PUP$ platform.

10. Figure 2.24 shows a $3 \times PUP$ platform, in which three identical PUP legs connect a fixed base to a moving platform. The P joints on both the fixed base and moving platform are arranged symmetrically as shown. Recalling that the U joint consists of two revolute joints connected orthogonally, each R joint connected to the moving platform has its joint axis aligned in the same direction as the platform P joint. The R joint connected to the fixed base has its joint axis orthogonal to the base P joint.
(a) Use an appropriate version of Grübler's formula to find the mobility.
(b) Reasoning intuitively, determine whether the mobility you obtained in part (a) is correct. If not, try to explain the discrepancy without resorting to a detailed kinematic analysis.

11. Figure 2.25 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 appropriate version of Grübler's formula to find the mobility.
(b) Describe the configuration space.
(c) Assuming a reference frame is attached to the center link, describe its task space.

12. Consider the two-link planar open chain of Figure 2.26, whose tip coordinates are given by

$$x = 2\cos\theta_1 + \cos(\theta_1 + \theta_2)$$
$$y = 2\sin\theta_1 + \sin(\theta_1 + \theta_2).$$
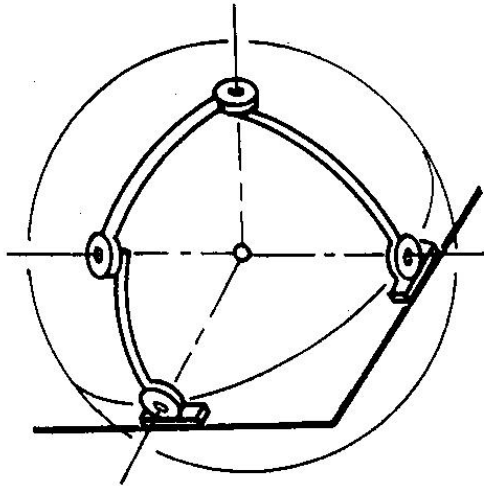
Figure 2.25: The spherical four-bar linkage.
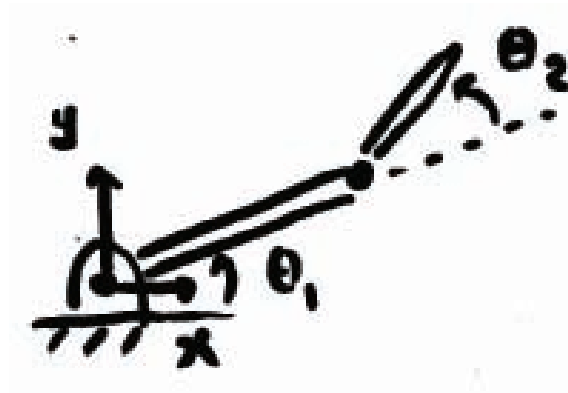
Figure 2.26: Two-link planar RR open chain.

(a) What is its configuration space?
(b) What is the set of all points the tip can reach?
(c) Suppose there is an infinitely long wall at $x = 1$ and $x = -1$. What is its free space?

13. Consider the slider-crank mechanism of Figure **??**. 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 and slider joint variables.

14. (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 task space.
(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 task space. Which of these two chains has a larger task space?
(c) A not so clever designer claims that he can make the task space of any planar open chain larger simply by increasing the length of the last link. Explain the fallacy behind this claim.

15. Determine if the following differential constraints are holonomic or non-holonomic:
(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)

$$
\begin{aligned}
-\dot{q}_1 \cos q_2 + \dot{q}_3 \sin(q_1 + q_2) - \dot{q}_4 \cos(q_1 + q_2) &= 0 \\
\dot{q}_3 \sin q_1 - \dot{q}_4 \cos q_1 &= 0.
\end{aligned}
$$

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

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. We established this by selecting three points on a rigid body, and arguing that the distances between any pair of these three points must always be preserved regardless of where the rigid body is located. This led to three constraints, which when imposed on the nine Cartesian coordinates—(x, y, z) coordinates for each of the three points—led us to conclude that only six of these nine coordinates could be independently chosen.

In this chapter we develop a more systematic way to describe the position and orientation of a rigid body. Rather than choosing three points on a body, we instead attach a reference frame to the body, and represent the configuration of this frame with respect to a fixed reference frame as a 4×4 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 4×4 real matrices.

Such a matrix not only represents the configuration of a frame, but it also can be used to (1) translate and rotate a vector or a frame or (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 4×4 matrix.

The non-Euclidean (non-"flat") nature of the C-space of position and orientations leads us to use the matrix representation. A 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*. To see that the feasible velocities at any configuration form a Euclidean space, even though the C-space is non-Euclidean, consider motions on 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, you can think of the space of velocities 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 spatial velocity 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*.

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 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 column 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 column vector 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
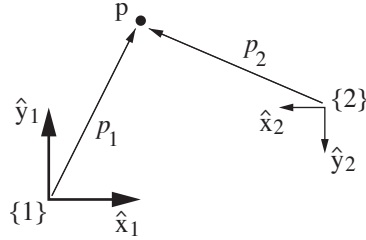
Figure 3.1: The point p exists in physical space, and it does not care how we represent it. If we fix a reference frame $\{1\}$, with unit coordinate axes $\hat{x}_1$ and $\hat{y}_1$, we can represent p as $p_1 = (1, 2)$. If we fix a reference frame $\{2\}$ at a different location, a different orientation, and a different length scale, we can represent p as $p_2 = (4, -2)$.

in it. However, we always assume that exactly one stationary **fixed frame**, or **space frame**, denoted $\{s\}$ (for "stationary" or "space"), has been defined. This might be attached to a corner of a room, for example. Similarly, we assume that at least one **moving frame**, or **body frame**, denoted $\{b\}$, has been attached to some moving rigid body, such as the body of a quadrotor flying in the room. 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 A Motivating Example

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 moving frame, or 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.
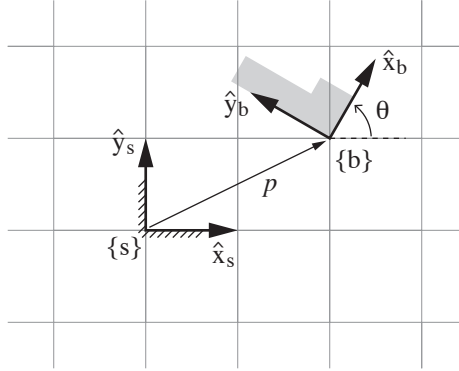
Figure 3.2: The body frame {b} in fixed-frame coordinates {s} is represented by the vector p and the direction of the unit axes $\hat{x}_b$ and $\hat{y}_b$ expressed in {s}. In this example, $p = (2,1)^T$ and $\theta = 60°$, so $\hat{x}_b = (\cos\theta, \sin\theta)^T = (0.5, 1/\sqrt{2})^T$ and $\hat{y}_b = (-\sin\theta, \cos\theta)^T = (-1/\sqrt{2}, 0.5)^T$.

The simplest way to describe the orientation of the body frame {b} relative to the fixed frame {s} is by specifying the angle θ 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 θ 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 point p can be represented as a column vector $p \in \mathbb{R}^2$ of the form

$$p = \left[\begin{array}{c} p_x \\ p_y \end{array} \right], \tag{3.4}$$

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

$$P = [\hat{x}_b \ \ \hat{y}_b] = \left[\begin{array}{cc} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{array} \right]. \tag{3.5}$$

The matrix P is an example of a **rotation matrix**. Although P is constructed of four numbers, they are subject to three constraints, and the one remaining
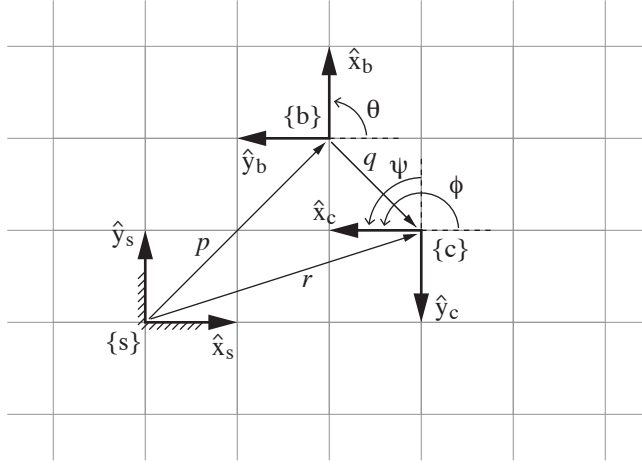
Figure 3.3: The frame {b} in {s} is given by (P, p), and the frame {c} in {b} is given by (Q, q). From these we can derive the frame {c} in {s}, described by (R, r). The numerical values of the vectors p, q, and r, and the coordinate axis directions of the three frames, are evident from the grid of unit squares.

degree of freedom is parametrized by θ. 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 above approach, and expressing {c} in {s} as the pair (R, r), we can write

$$r = \begin{bmatrix} r_x \\ r_y \end{bmatrix}, \quad R = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}. \tag{3.6}$$

We could also describe the frame {c} relative to {b} (that is, pretend {b} is now the fixed frame). 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

$$q = \begin{bmatrix} q_x \\ q_y \end{bmatrix}, \quad Q = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}. \tag{3.7}$$

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:

$$R = PQ \quad \text{(convert } Q \text{ to the {s} frame)} \tag{3.8}$$
$$r = Pq + p \quad \text{(convert } q \text{ to the {s} frame and vector sum with } p\text{).} \tag{3.9}$$

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.

(a)   (b)
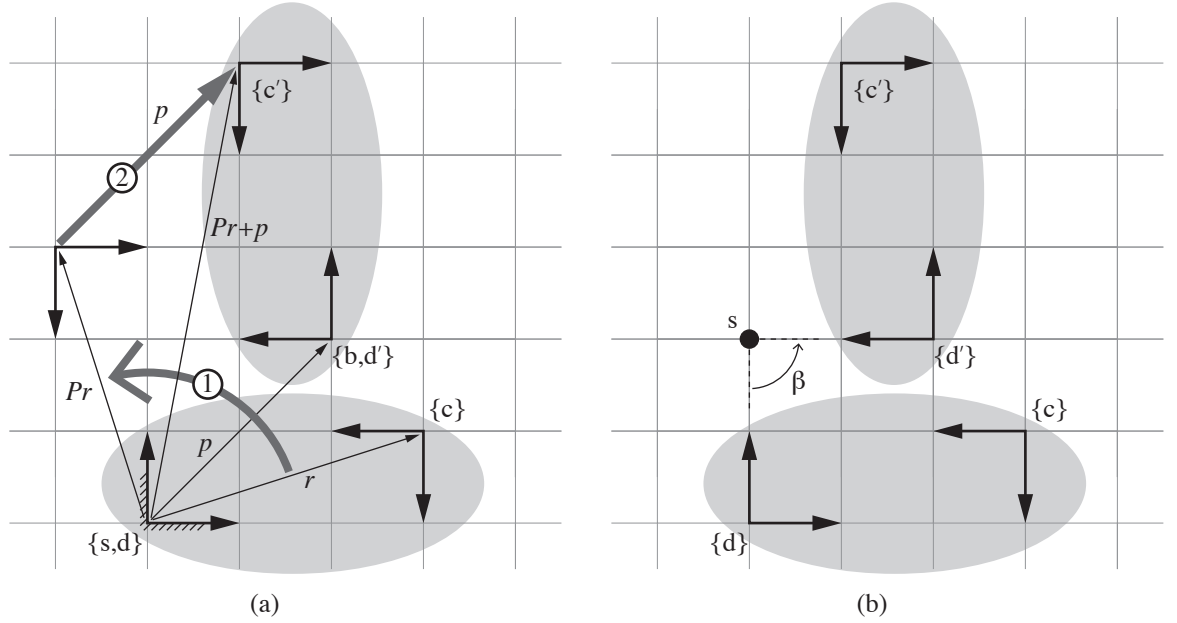
Figure 3.4: (a) The frame {d}, fixed to an elliptical rigid body and initially coincident with {s}, is displaced to {d'} (coincident with the stationary frame {b}), by first rotating according to P then translating according to p, where (P, p) is the representation of {b} in {s}. The same transformation takes the frame {c}, also attached to the rigid body, to {c'}. The transformation marked ① rigidly rotates {c} about the origin of {s}, and then transformation 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 $\beta = 90°$ about a fixed point s.

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 the stationary 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

$$R' = PR \tag{3.10}$$
$$r' = Pr + p, \tag{3.11}$$

very similar to Equations (3.8) and (3.9). The difference is that (P, p) is expressed in the same frame as (R, r) (unlike (Q, q)), 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 (1) rotates {c} according to P and (2) translates it by p in {s}. See Figure 3.4(a).
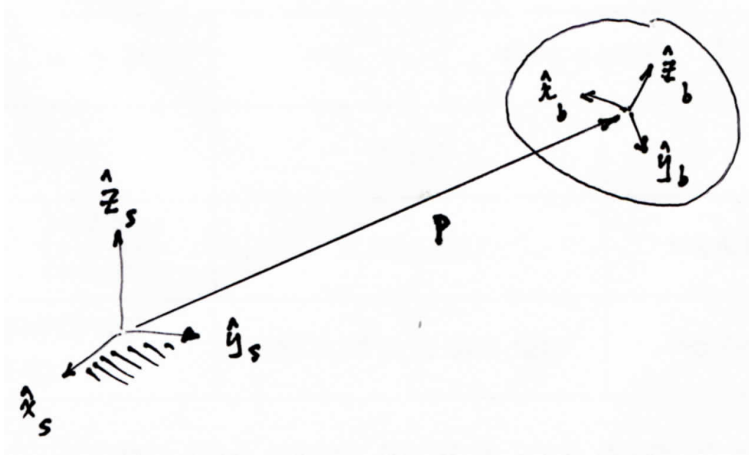
Figure 3.5: Mathematical description of position and orientation.

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)).

We make one final observation. 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 β. This is a planar example of a **screw motion**.[1] The displacement can therefore be parametrized by the three coordinates (β, 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 $\{s\}$.

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

$$p = p_1\hat{x}_s + p_2\hat{y}_s + p_3\hat{z}_s. \tag{3.12}$$

---

[1] If the displacement is a pure translation without rotation, then s lies at infinity.

The axes of the body frame can also be expressed as

$$\hat{x}_b = r_{11}\hat{x}_s + r_{21}\hat{y}_s + r_{31}\hat{z}_s \tag{3.13}$$

$$\hat{y}_b = r_{12}\hat{x}_s + r_{22}\hat{y}_s + r_{32}\hat{z}_s \tag{3.14}$$

$$\hat{z}_b = r_{13}\hat{x}_s + r_{23}\hat{y}_s + r_{33}\hat{z}_s. \tag{3.15}$$

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

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

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 a minimum of six parameters are required to describe the configuration of a rigid body, if we agree to keep the three parameters in p as they are, then of the nine parameters in R, only three can be chosen independently. As with the exponential coordinates of planar rigid-body motion, the three exponential coordinates of spatial rotation can describe any orientation by an axis of rotation and the distance rotated about that axis from the home orientation. We leave other popular representations of orientations (three-parameter **Euler angles** and **roll-pitch-yaw angles**, and the **unit quaternions** that use four variables subject to one constraint) to an appendix.

We then examine six-parameter exponential coordinates for the configuration of a rigid body that arise from integrating a six-dimensional **spatial velocity** 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 **spatial force**. The spatial velocity and spatial force, 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 deriving 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

$$
\begin{aligned}
r_{11}^2 + r_{21}^2 + r_{31}^2 &= 1 \\
r_{12}^2 + r_{22}^2 + r_{32}^2 &= 1 \\
r_{13}^2 + r_{23}^2 + r_{33}^2 &= 1.
\end{aligned}
\tag{3.17}
$$

(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

$$
\begin{aligned}
r_{11}r_{12} + r_{21}r_{22} + r_{31}r_{32} &= 0 \\
r_{12}r_{13} + r_{22}r_{23} + r_{32}r_{33} &= 0 \\
r_{11}r_{13} + r_{21}r_{23} + r_{31}r_{33} &= 0.
\end{aligned}
\tag{3.18}
$$

These six constraints can be expressed more compactly as a single constraint on the matrix R:

$$
R^T R = I,
\tag{3.19}
$$

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 3×3 matrix M: denoting the three columns of M by a, b, and c, respectively, its determinant is given by

$$
\det M = a^T(b \times c) = c^T(a \times b) = b^T(c \times a).
\tag{3.20}
$$

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 3×3 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 3×3 real matrices R that satisfy (i) $R^T R = I$ and (ii) $\det R = 1$.

The set of 2×2 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 2×2 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 viewed as the top-left 2×2 submatrix of an element of $SO(3)$ and 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 $\theta \in [0, 2\pi)$.

3.2.1.1 Properties of Rotation Matrices

We now list some basic properties of rotation matrices. First, the identity I is a trivial example of a rotation matrix. The inverse of a rotation matrix is also 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. □

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. □

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. □

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 $\|y\|^2 = y^T y = (Rx)^T Rx = x^T R^T Rx = x^T x = \|x\|^2$. □

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.
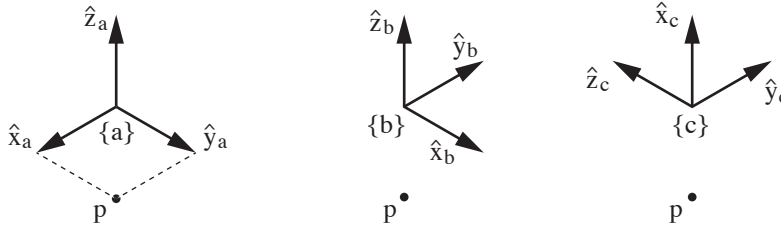
Figure 3.6: The same space and the same point p represented in three different frames with different orientations.

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, we refer to Figure 3.6, which shows three different coordinate frames—{a}, {b}, and {c}—representing the same space.[2] These frames have the same origin, since we are only representing orientations, but 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, which is aligned with {a}. The orientations of the three frames 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 $-90°$.

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_{sc}: 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.

---

[2]In the rest of the book, all coordinate frames will use the same length scale; only their position and orientation may be different.

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 $\{e\}$,

$$R_{de} = R_{ed}^{-1} = R_{ed}^{T}.$$

You can prove this for two arbitrary frames, or you can test 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

$$R_{ac} = R_{ab}R_{bc}.$$

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:

$$R_{ab}R_{bc} = R_{a\cancel{b}}R_{\cancel{b}c} = R_{ac}.$$

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:

$$R_{ab}p_{b} = R_{a\cancel{b}}p_{\cancel{b}} = p_{a}.$$

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 an $\{\hat{x}, \hat{y}, \hat{z}\}$ frame that is rotated about a unit axis $\hat{\omega}$ (expressed in the $\{\hat{x}, \hat{y}, \hat{z}\}$ coordinates) by an amount θ. (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.) The resulting frame $\{\hat{x}', \hat{y}', \hat{z}'\}$ is
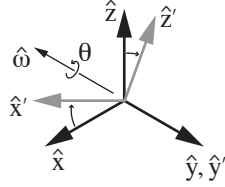
Figure 3.7: (a) A coordinate frame with axes $\{\hat{x}, \hat{y}, \hat{z}\}$ is rotated an amount θ about a unit axis $\hat{\omega}$ (which is aligned with $-\hat{y}$ in this figure). The orientation of the final frame, with axes $\{\hat{x}', \hat{y}', \hat{z}'\}$, is written as R relative to the original frame.

expressed as R relative to $\{\hat{x}, \hat{y}, \hat{z}\}$. Emphasizing our view of R as a rotation operator, instead of as an orientation, we can write

$$R = \text{Rot}(\hat{\omega}, \theta).$$

As we will see in Section 3.2.3, any $R \in SO(3)$ can be obtained by rotating by some θ about some $\hat{\omega}$.

Now say R_{sb} represents $\{b\}$ relative to $\{s\}$, and we want to rotate it by a given $R = \text{Rot}(\hat{\omega}, \theta)$. 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 θ 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 θ 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

$$R_{sb'} = \text{rotate\_by\_}R\text{\_in\_}\{s\}\text{\_frame} \ (R_{sb}) = RR_{sb} \tag{3.22}$$
$$R_{sb''} = \text{rotate\_by\_}R\text{\_in\_}\{b\}\text{\_frame} \ (R_{sb}) = R_{sb}R. \tag{3.23}$$

In other words, premultiplying by R yields a rotation about ω in the fixed frame and postmultiplying by R yields a rotation about ω 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 Rotations

Other popular representations of rotations include Euler angles, roll-pitch-yaw angles, and unit quaternions. These representations, and their relation to rotation matrices, are discussed in Appendix B.
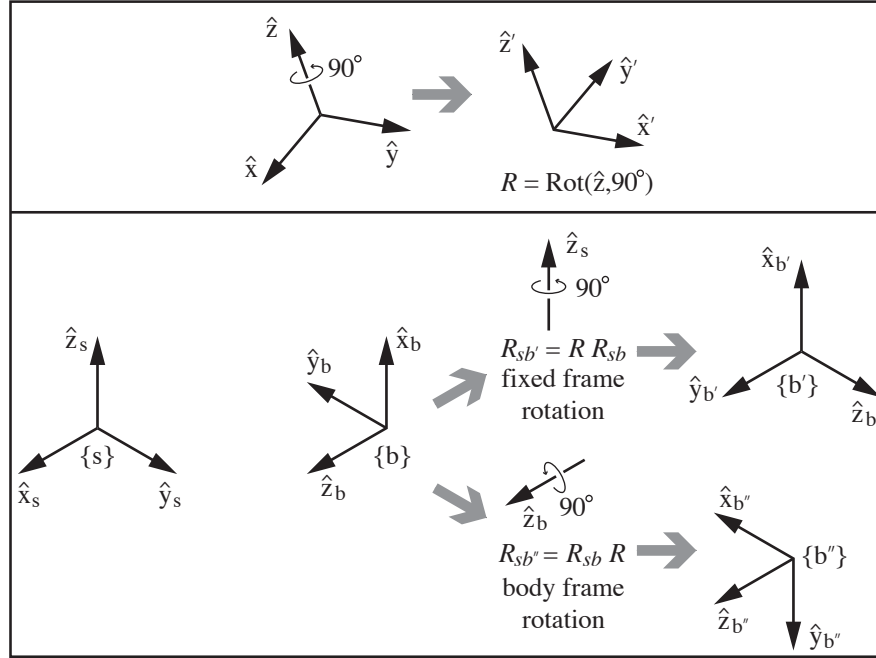
Figure 3.8: (Top) A rotation operator R defined as $R = \mathrm{Rot}(\hat{z}, 90°)$, the orientation of the right frame in the left frame. (Bottom) On the left are shown a fixed frame {s} and a body frame {b}, expressed as R_{sb}. The quantity RR_{sb} rotates {b} to {b'} by rotating by 90° about the fixed frame axis $\hat{z}_{\mathrm{s}}$. The quantity $R_{sb}R$ rotates {b} to {b''} by rotating by 90° about the body frame axis $\hat{z}_{\mathrm{b}}$.

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 $\dot{\hat{x}}$, 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 $t + \Delta t$—since what's relevant for us is the orientation of the body frame, for better visualization we draw the frame at the two instants with coinciding origins—the change in frame orientation from t to $t + \Delta t$ 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 Δt 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** w as follows:

$$\mathrm{w} = \hat{w}\dot{\theta}. \tag{3.24}$$
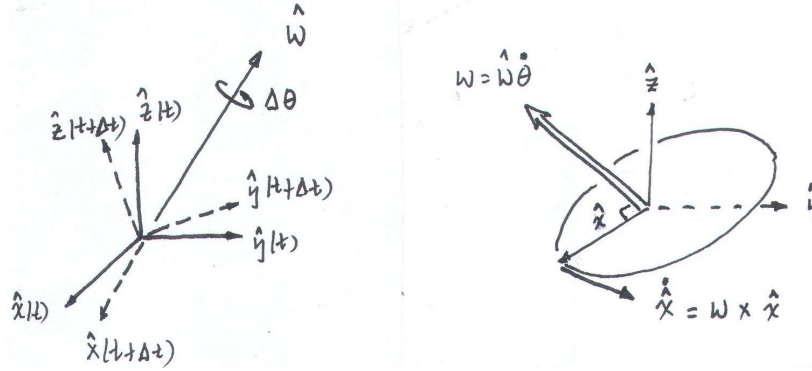
Figure 3.9: (Left) The instantaneous angular velocity vector. (Right) Calculating $\dot{\hat{x}}$.

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

$$\dot{\hat{x}} = w \times \hat{x} \qquad (3.25)$$
$$\dot{\hat{y}} = w \times \hat{y} \qquad (3.26)$$
$$\dot{\hat{z}} = w \times \hat{z}. \qquad (3.27)$$

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; $\dot{R}(t)$ 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 (3.25)–(3.27) can be expressed in fixed frame coordinates as

$$\dot{r}_i = \omega_s \times r_i, \quad i = 1, 2, 3.$$

These three equations can be rearranged into the following single 3×3 matrix equation:

$$\dot{R} = [\omega_s \times r_1 \mid \omega_s \times r_2 \mid \omega_s \times r_3] = \omega_s \times R. \qquad (3.28)$$

To eliminate the cross product in Equation (3.28), we introduce some new notation and rewrite $\omega_s \times R$ as $[\omega_s]R$, where $[\omega_s]$ is a 3×3 *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

$$[x] = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}. \qquad (3.29)$$

The matrix $[x]$ is a 3×3 **skew-symmetric** matrix representation of x; that is,

$$[x] = -[x]^T.$$

The set of all 3×3 real skew-symmetric matrices is called $so(3)$.

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:*

$$R[\omega]R^T = [R\omega]. \tag{3.30}$$

Proof. Letting r_i^T be the ith row of R,

$$
\begin{aligned}
R[\omega]R^T &= \begin{bmatrix} r_1^T(\omega \times r_1) & r_1^T(\omega \times r_2) & r_1^T(\omega \times r_3) \\ r_2^T(\omega \times r_1) & r_2^T(\omega \times r_2) & r_2^T(\omega \times r_3) \\ r_3^T(\omega \times r_1) & r_3^T(\omega \times r_2) & r_3^T(\omega \times r_3) \end{bmatrix} \\
&= \begin{bmatrix} 0 & -r_3^T\omega & r_2^T\omega \\ r_3^T\omega & 0 & -r_1^T\omega \\ -r_2^T\omega & r_1^T\omega & 0 \end{bmatrix} \\
&= [R\omega],
\end{aligned}
\tag{3.31}
$$

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

With the skew-symmetric notation, we can rewrite Equation (3.28) as

$$[\omega_s]R = \dot{R}. \tag{3.32}$$

We can post-multiply both sides of Equation (3.32) by R^{-1} to get

$$[\omega_s] = \dot{R}R^{-1}. \tag{3.33}$$

Now let ω_b be w expressed in body frame coordinates. To see how to obtain ω_b from ω_s and vice versa, we explicitly write R as R_{sb}. Then ω_s and ω_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

$$\omega_b = R_{sb}^{-1}\omega_s = R^{-1}\omega_s = R^T\omega_s. \tag{3.34}$$

Let us now express this relation in skew-symmetric matrix form:

$$
\begin{aligned}
[\omega_b] &= [R^T\omega_s] \\
&= R^T[\omega_s]R \quad \text{(by Proposition 3.5)} \\
&= R^T(\dot{R}R^T)R \\
&= R^T\dot{R} = R^{-1}\dot{R}.
\end{aligned}
\tag{3.35}
$$

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*

$$\dot{R}R^{-1} = [\omega_s] \tag{3.36}$$
$$R^{-1}\dot{R} = [\omega_b], \tag{3.37}$$

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

It is important to note that ω_b is *not* the angular velocity of the moving frame relative to the moving frame; that must be zero, since the moving frame is always coincident with itself! Instead, ω_b is the angular velocity relative to a *stationary* frame that is coincident with {b}.

It is also important to note that the fixed-frame angular velocity ω_s *does not depend on the choice of the body frame.* Similarly, the body-frame angular velocity ω_b *does not depend on the choice of the fixed frame.* While Equations (3.36) and (3.37) may appear to depend on both frames (since R and $\dot{R}$ individually depend on both {s} and {b}), the product $\dot{R}R^{-1}$ 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}:

$$\omega_c = R_{cd}\omega_d.$$

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 vector $\hat{\omega}$ of unit length), together with an angle of rotation θ 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 viewed as a rotation axis $\hat{\omega}$ rotated about by an angle θ, or they can be viewed equivalently as an angular velocity $\hat{\omega}\theta$ (in rad/s) that is integrated for one second. This latter view, that a rotation is the integral of an angular velocity for one second, suggests that we view 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

$$\dot{x}(t) = ax(t), \tag{3.38}$$

where $x(t) \in \mathbb{R}$, $a \in \mathbb{R}$ is constant, and the initial condition $x(0) = x_0$ is assumed given. Equation (3.38) has solution

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

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

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

Now consider the vector linear differential equation

$$\dot{x}(t) = Ax(t) \tag{3.39}$$

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

$$x(t) = e^{At} x_0 \tag{3.40}$$

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

$$e^{At} = I + At + \frac{(At)^2}{2!} + \frac{(At)^3}{3!} + \dots \tag{3.41}$$

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 (3.40), using Equation (3.41), is indeed a solution to Equation (3.39). Taking the time derivative of $x(t) = e^{At} x_0$,

$$
\begin{aligned}
\dot{x}(t) &= \left(\frac{d}{dt} e^{At} \right) x_0 \\
&= \frac{d}{dt} \left(I + At + \frac{A^2 t^2}{2!} + \frac{A^3 t^3}{3!} + \dots \right) x_0 \\
&= \left(A + A^2 t + \frac{A^3 t^2}{2!} + \dots \right) x_0 \\
&= A e^{At} x_0 \\
&= Ax(t),
\end{aligned} \tag{3.42}
$$

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

$$Ae^{At} = e^{At} A \tag{3.43}$$

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 (3.42), A could also have been factored to the right, i.e.,

$$\dot{x}(t) = e^{At} A x_0.$$

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 $A = PDP^{-1}$ for some $D \in \mathbb{R}^{n \times n}$ and invertible $P \in \mathbb{R}^{n \times n}$, then

$$
\begin{aligned}
e^{At} &= I + At + \frac{(At)^2}{2!} + \dots \\
&= I + (PDP^{-1})t + (PDP^{-1})(PDP^{-1})\frac{t^2}{2!} + \dots \\
&= P(I + Dt + \frac{(Dt)^2}{2!} + \dots)P^{-1} \\
&= Pe^{Dt}P^{-1}.
\end{aligned}
\tag{3.44}
$$

If moreover D is diagonal, i.e., $D = \operatorname{diag}\{d_1, d_2, \dots, d_n\}$, then its matrix exponential is particularly simple to evaluate:

$$
e^{Dt} = \begin{bmatrix}
e^{d_1 t} & 0 & \cdots & 0 \\
0 & e^{d_2 t} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & e^{d_n t}
\end{bmatrix}.
\tag{3.45}
$$

We summarize the above results 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*

$$x(t) = e^{At} x_0 \tag{3.46}$$

where

$$e^{At} = I + tA + \frac{t^2}{2!}A^2 + \frac{t^3}{3!}A^3 + \dots. \tag{3.47}$$

The matrix exponential e^{At} further satisifies the following properties:

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

(ii) *If $A = PDP^{-1}$ for some $D \in \mathbb{R}^{n \times n}$ and invertible $P \in \mathbb{R}^{n \times n}$, then* $e^{At} = Pe^{Dt}P^{-1}$.

(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*

$$p(s) = \det(sI - A) = s^n + c_{n-1}s^{n-1} + \ldots + c_1 s + c_0,$$

and define $p(A)$ as

$$p(A) = A^n + c_{n-1}A^{n-1} + \ldots + c_1 A + c_0 I.$$

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 $\hat{\omega}$ ($\omega \in \mathbb{R}^3, \|\omega\| = 1$) 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 joint axis from the motion θ about the axis.

Referring to Figure 3.10, suppose a three-dimensional vector $p(0)$ is rotated by θ 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 $p(0)$ is rotating 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

$$\dot{p} = \hat{\omega} \times p. \tag{3.48}$$

To see why this is true, let ϕ 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 (3.48).

The differential equation (3.48) can be expressed as

$$\dot{p} = [\hat{\omega}]p \tag{3.49}$$

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

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

Since t and θ are interchangeable as a result of assuming that $p(t)$ rotates at a constant rate of 1 rad/s (after t seconds, $p(t)$ will have rotated t radians), the equation above can also be written

$$p(\theta) = e^{[\hat{\omega}]\theta}p(0).$$
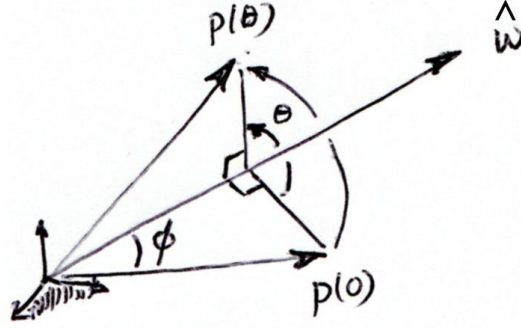
Figure 3.10: The vector $p(0)$ is rotated by an angle θ about the axis $\hat{\omega}$, to $p(\theta)$.

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 3×3 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

$$
\begin{aligned}
e^{[\hat{\omega}]\theta} &= I + [\hat{\omega}]\theta + [\hat{\omega}]^2 \frac{\theta^2}{2!} + [\hat{\omega}]^3 \frac{\theta^3}{3!} + \dots \\
&= I + \left(\theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \dots \right) [\hat{\omega}] + \left(\frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \dots \right) [\hat{\omega}]^2.
\end{aligned}
$$

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

$$
\begin{aligned}
\sin\theta &= \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \dots \\
\cos\theta &= 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \dots
\end{aligned}
$$

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

Proposition 3.9. *Given a vector $\hat{\omega}\theta \in \mathbb{R}^3$, such that θ is any scalar and $\hat{\omega} \in \mathbb{R}^3$ is a unit vector,*

$$e^{[\hat{\omega}]\theta} = I + \sin\theta\, [\hat{\omega}] + (1 - \cos\theta)[\hat{\omega}]^2 \in SO(3). \qquad (3.50)$$

This formula provides the matrix exponential of $[\hat{\omega}]\theta \in so(3)$.

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 θ. 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 θ. Similarly, considering that a rotation matrix R consists of three column vectors, the rotation matrix $R' = e^{[\hat{\omega}]\theta}R$ is the orientation achieved by rotating R by θ about the axis $\hat{\omega}$ in the fixed frame. Reversing the order of matrix multiplication, $R'' = Re^{[\hat{\omega}]\theta}$ is the orientation achieved by rotating R by θ 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 $\theta = 30° = 0.524$ rad. The rotation matrix representation of {b} can be calculated as

$$
\begin{aligned}
R &= e^{[\hat{\omega}]\theta} \\
&= I + \sin\theta[\hat{\omega}] + (1 - \cos\theta)[\hat{\omega}]^2 \\
&= I + 0.5 \begin{bmatrix} 0 & -0.5 & 0.866 \\ 0.5 & 0 & 0 \\ -0.866 & 0 & 0 \end{bmatrix} + 0.134 \begin{bmatrix} 0 & -0.5 & 0.866 \\ 0.5 & 0 & 0 \\ -0.866 & 0 & 0 \end{bmatrix}^2 \\
&= \begin{bmatrix} 0.866 & -0.250 & 0.433 \\ 0.250 & 0.967 & 0.058 \\ -0.433 & 0.058 & 0.900 \end{bmatrix}.
\end{aligned}
$$

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 θ 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
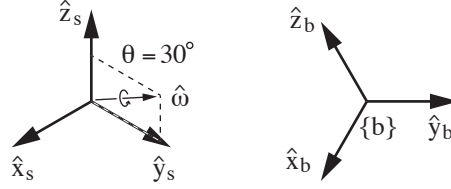
Figure 3.11: The frame {b} is obtained by rotating from {s} by $\theta = 30°$ about $\hat{\omega} = (0, 0.866, 0.5)^T$.

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,

$$\begin{aligned} \exp: \quad & [\hat{\omega}]\theta \in so(3) && \rightarrow && R \in SO(3) \\ \log: \quad & R \in SO(3) && \rightarrow && [\hat{\omega}]\theta \in so(3) \end{aligned}$$

To derive the matrix logarithm, let us expand each of the entries for $e^{[\hat{\omega}]\theta}$ in Equation (3.50):

$$\begin{bmatrix} c_\theta + \hat{\omega}_1^2(1 - c_\theta) & \hat{\omega}_1\hat{\omega}_2(1 - c_\theta) - \hat{\omega}_3 s_\theta & \hat{\omega}_1\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_2 s_\theta \\ \hat{\omega}_1\hat{\omega}_2(1 - c_\theta) + \hat{\omega}_3 s_\theta & c_\theta + \hat{\omega}_2^2(1 - c_\theta) & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_1 s_\theta \\ \hat{\omega}_1\hat{\omega}_3(1 - c_\theta) - \hat{\omega}_2 s_\theta & \hat{\omega}_2\hat{\omega}_3(1 - c_\theta) + \hat{\omega}_1 s_\theta & c_\theta + \hat{\omega}_3^2(1 - c_\theta) \end{bmatrix}, \quad (3.51)$$

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:

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

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

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

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

$$[\hat{\omega}] = \begin{bmatrix} 0 & -\hat{\omega}_3 & \hat{\omega}_2 \\ \hat{\omega}_3 & 0 & -\hat{\omega}_1 \\ -\hat{\omega}_2 & \hat{\omega}_1 & 0 \end{bmatrix} = \frac{1}{2\sin\theta}\left(R - R^T\right). \quad (3.52)$$

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 θ is an integer multiple of π.[3] We address this situation next, but for now let us assume this is not the case and find an expression for θ. Setting R equal to (3.51) and taking the trace of both sides (recall that the trace of a matrix is the sum of its diagonal entries),

$$\operatorname{tr} R = r_{11} + r_{22} + r_{33} = 1 + 2\cos\theta. \tag{3.53}$$

The above follows since $\hat{\omega}_1^2 + \hat{\omega}_2^2 + \hat{\omega}_3^2 = 1$. For any θ satisfying $1 + 2\cos\theta = \operatorname{tr} R$ such that θ is not an integer multiple of π, R can be expressed as the exponential $e^{[\hat{\omega}]\theta}$ with $[\hat{\omega}]$ as given in Equation (3.52).

Let us now return to the case $\theta = k\pi$, where k is some integer. When k is an even integer (corresponding to $\theta = 0, \pm 2\pi, \pm 4\pi, \ldots$) we have $\operatorname{tr} R = 3$, or equivalently $R = I$, and it follows straightforwardly that $\theta = 0$ is the only possible solution. When k is an odd integer (corresponding to $\theta = \pm\pi, \pm 3\pi, \ldots$, which in turn implies $\operatorname{tr} R = -1$), the exponential formula (3.50) simplifies to

$$R = e^{[\hat{\omega}]\pi} = I + 2[\hat{\omega}]^2. \tag{3.54}$$

The three diagonal terms of Equation (3.54) can be manipulated to

$$\hat{\omega}_i = \pm\sqrt{\frac{r_{ii} + 1}{2}}, \quad i = 1, 2, 3. \tag{3.55}$$

These may not always lead to a feasible unit-norm solution $\hat{\omega}$. The off-diagonal terms lead to the following three equations:

$$\begin{aligned}
2\hat{\omega}_1\hat{\omega}_2 &= r_{12} \\
2\hat{\omega}_2\hat{\omega}_3 &= r_{23} \\
2\hat{\omega}_1\hat{\omega}_3 &= r_{13},
\end{aligned} \tag{3.56}$$

From Equation (3.54) we also know that R must be symmetric: $r_{12} = r_{21}$, $r_{23} = r_{32}$, $r_{13} = r_{31}$. Both Equations (3.55) and (3.56) may be necessary to obtain a feasible solution. Once a solution $\hat{\omega}$ has been found, then $R = e^{[\hat{\omega}]k\pi}$, $k = \pm\pi, \pm 3\pi, \ldots$.

From the above it can be seen that solutions for θ exist at 2π intervals. If we restrict θ 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.

---

[3] A singularity such as this is unavoidable for any three-parameter representation of rotation. Euler angles and roll-pitch-yaw angles suffer similar singularities.
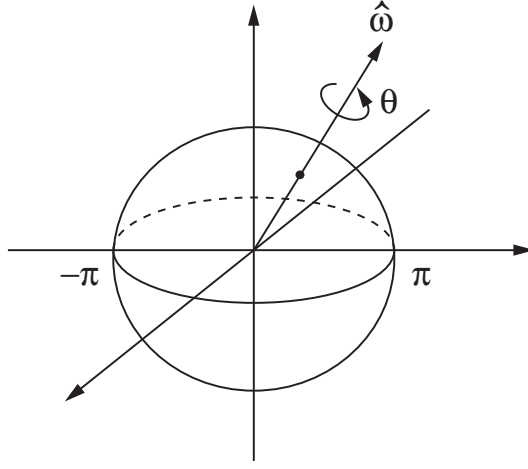
Figure 3.12: *SO(3)* as a solid ball of radius π.

(ii) If tr $R = -1$, then $\theta = \pi$, and set $\hat{\omega}$ to any of the following three vectors that is a feasible solution:

$$\hat{\omega} = \frac{1}{\sqrt{2(1 + r_{33})}} \begin{bmatrix} r_{13} \\ r_{23} \\ 1 + r_{33} \end{bmatrix} \tag{3.57}$$

or

$$\hat{\omega} = \frac{1}{\sqrt{2(1 + r_{22})}} \begin{bmatrix} r_{12} \\ 1 + r_{22} \\ r_{32} \end{bmatrix} \tag{3.58}$$

or

$$\hat{\omega} = \frac{1}{\sqrt{2(1 + r_{11})}} \begin{bmatrix} 1 + r_{11} \\ r_{21} \\ r_{31} \end{bmatrix}. \tag{3.59}$$

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

$$[\hat{\omega}] = \frac{1}{2 \sin \theta}(R - R^T). \tag{3.60}$$

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 π (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 θ. For any $R \in SO(3)$ such that 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 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 $R = e^{[r]}$, then $\|r\| = \pi$, and $R = e^{[-r]}$ also holds; both r and $-r$ correspond to the same rotation R.

3.3 Rigid-Body Motions and Spatial Velocities

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 R; a screw axis $\mathcal{S}$ is analogous to a rotation axis $\hat{\omega}$; a spatial velocity $\mathcal{V}$ can be expressed as $\mathcal{S}\dot{\theta}$ and is analogous to an angular velocity $\omega = \hat{\omega}\dot{\theta}$; and exponential coordinates $\mathcal{S}\theta \in \mathbb{R}^6$ 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 4×4 real matrices T of the form

$$T = \left[\begin{array}{cc} R & p \\ 0 & 1 \end{array} \right] = \left[\begin{array}{cccc} r_{11} & r_{12} & r_{13} & p_1 \\ r_{21} & r_{22} & r_{23} & p_2 \\ r_{31} & r_{32} & r_{33} & p_3 \\ 0 & 0 & 0 & 1 \end{array} \right], \tag{3.61}$$

where $R \in SO(3)$, $p \in \mathbb{R}^3$ is a column vector, and 0 denotes a row vector of three zeros.

An element $T \in SE(3)$ will sometimes be denoted as (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.

From the definition it should be apparent that six coordinates are needed to parametrize $SE(3)$. The most obvious choice might be to use the three exponential coordinates for the rotation R and the usual three Cartesian coordinates in $\mathbb{R}^3$ to parametrize the position p. Instead, we derive a six-dimensional version of exponential coordinates on $SE(3)$ that turns out to have several advantages over other parametrizations.

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 3×3 real matrices T of the form

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}, \tag{3.62}$$

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 inverse of a transformation matrix is also a transformation matrix:

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

$$T^{-1} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix}. \tag{3.63}$$

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

Proposition 3.12. *Multiplication of transformation matrices is associative, $(T_1 T_2)T_3 = T_1(T_2 T_3)$, but generally not commutative, $T_1 T_2 \neq T_2 T_1$.*

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:

$$T \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} Rx + p \\ 1 \end{bmatrix}. \tag{3.64}$$

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:*

(i) $\|Tx - Ty\| = \|x - y\|$, *where* $\| \cdot \|$ *denotes the standard Euclidean norm in* $\mathbb{R}^3$, *i.e.,* $\|x\| = \sqrt{x^T x}$.

(ii) $\langle Tx - Tz, Ty - Tz \rangle = \langle x - z, y - z \rangle$ *for all* $z \in \mathbb{R}^3$, *where* $\langle \cdot, \cdot \rangle$ *denotes the standard Euclidean inner product in* $\mathbb{R}^3$, *i.e.,* $\langle x, y \rangle = x^T y$.

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 x, 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

$$
p_{sa} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad p_{sb} = \begin{bmatrix} 0 \\ -2 \\ 0 \end{bmatrix}, \quad p_{sc} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}.
$$
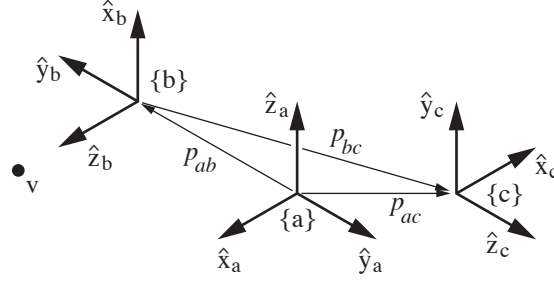
Figure 3.13: Three reference frames in space, and a point v that can be represented in {b} as $v_b = (0, 0, 1.5)^T$.

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}:

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

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,

$$
T_{ab}T_{bc} = T_{a\not{b}}T_{\not{b}c} = T_{ac}
$$
$$
T_{ab}v_b = T_{a\not{b}}v_{\not{b}} = v_a.
$$

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 θ 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}:

$$
T_{sb'} = TT_{sb} = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} RR_{sb} & Rp_{sb} + p \\ 0 & 1 \end{bmatrix} \quad \text{(fixed frame)}
$$
$$
\tag{3.65}
$$

$$
T_{sb''} = T_{sb}T = \begin{bmatrix} R_{sb} & p_{sb} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{sb}R & R_{sb}p + p_{sb} \\ 0 & 1 \end{bmatrix} \quad \text{(body frame)}.
$$
$$
\tag{3.66}
$$

The fixed-frame transformation (pre-multiplication by T) can be interpreted as first rotating the {b} frame about $\hat{\omega}$ in the {s} frame, then translating it by p in the {s} frame. The body-frame transformation (post-multiplication by T) can be interpreted as first translating {b} by p in the {b} frame, then rotating about $\hat{\omega}$ in the {b} frame. Fixed-frame and body-frame transformations are illustrated in Figure 3.14 for a transformation T with $\hat{\omega} = (0,0,1)^T$, $\theta = 90°$, and $p = (0,2,0)^T$, for

$$
T = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
$$

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:

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

$$
T_{de} = \begin{bmatrix} 0 & 0 & -1 & 300 \\ 0 & -1 & 0 & 100 \\ -1 & 0 & 0 & 120 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$
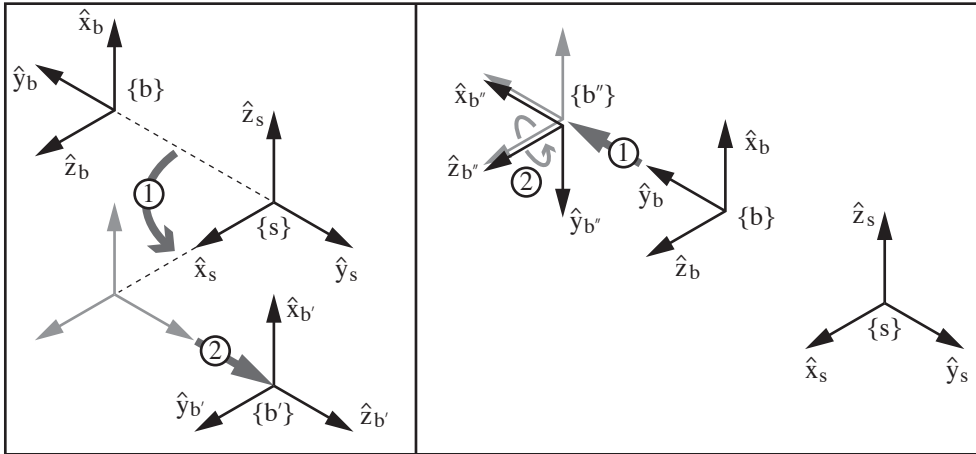
Figure 3.14: Fixed-frame and body-frame transformations corresponding to $\hat{\omega} = (0,0,1)^T$, $\theta = 90°$, and $p = (0,2,0)^T$. (Left) The frame {b} rotated by 90° about $\hat{z}_s$ and then translated by two units in $\hat{y}_s$, resulting in the new frame {b$'$}. (Right) The frame {b} translated by two units in $\hat{y}_b$ and then rotated by 90° about $\hat{z}_b$, resulting in the new frame {b$''$}.
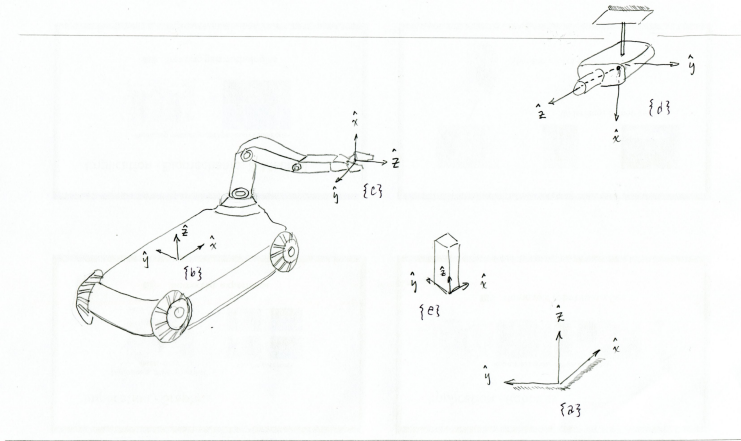
Figure 3.15: Assignment of reference frames.

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

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

In order to calculate how to move the robot arm to pick up the object, 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_{ab}. However, since $T_{ab} = T_{ad}T_{db}$, we can determine T_{ce} as follows:

$$T_{ce} = (T_{ad}T_{db}T_{bc})^{-1} T_{ad}T_{de}.$$

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 Spatial Velocity

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

$$T_{sb}(t) = T(t) = \begin{bmatrix} R(t) & p(t) \\ 0 & 1 \end{bmatrix} \tag{3.67}$$

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_{sb}).

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 $\dot{T}T^{-1}$ carry similar physical interpretations.

Let us first see what happens when we pre-multiply $\dot{T}$ by T^{-1}:

$$T^{-1}\dot{T} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{R} & \dot{p} \\ 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} R^T \dot{R} & R^T \dot{p} \\ 0 & 0 \end{bmatrix} \tag{3.68}$$

$$= \begin{bmatrix} [\omega_b] & v_b \\ 0 & 0 \end{bmatrix}. \tag{3.69}$$

Recall that $R^T \dot{R} = [\omega_b]$ is just the skew-symmetric matrix representation of the angular velocity expressed in moving frame coordinates. Also, $\dot{p}$ is the linear velocity of the moving frame origin expressed in fixed frame coordinates, and $R^T \dot{p} = v_b$ is this linear velocity expressed in moving frame coordinates. 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 a stationary frame currently aligned with the moving frame.

The previous calculation of $T^{-1}\dot{T}$ suggests that it is reasonable to merge ω_b and v_b into a single six-dimensional velocity vector. We define the **spatial velocity in the body frame** to be

$$\mathcal{V}_b = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} \in \mathbb{R}^6. \tag{3.70}$$

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 spatial velocity vector, as shown in Equation (3.69). We overload the $[\cdot]$ notation, writing

$$T^{-1}\dot{T} = [\mathcal{V}_b] = \begin{bmatrix} [\omega_b] & v_b \\ 0 & 0 \end{bmatrix} \in se(3), \tag{3.71}$$

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

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

$$\dot{T}T^{-1} = \begin{bmatrix} \dot{R} & \dot{p} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \dot{R}R^T & \dot{p} - \dot{R}R^T p \\ 0 & 0 \end{bmatrix} \tag{3.72}$$

$$= \begin{bmatrix} [\omega_s] & v_s \\ 0 & 0 \end{bmatrix}.$$
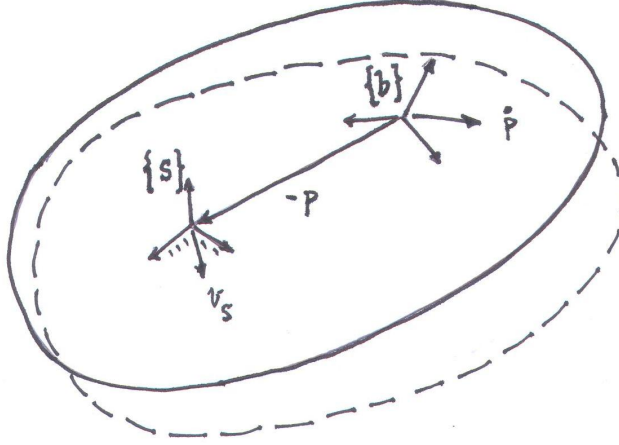
Figure 3.16: Physical interpretation of v_s. The initial (solid line) and displaced (dotted line) configurations of a rigid body.

Observe that the skew-symmetric matrix $[\omega_s] = \dot{R}R^T$ is the angular velocity expressed in fixed frame coordinates, but that $v_s = \dot{p} - \dot{R}R^T p$ is **not** the linear velocity of the moving frame origin expressed in the fixed frame (that quantity would simply be $\dot{p}$). On the other hand, if we write v_s as

$$v_s = \dot{p} - \omega_s \times p = \dot{p} + \omega_s \times (-p), \qquad (3.73)$$

the physical meaning of v_s can now be inferred: imagining the moving frame is attached to an infinitely large rigid body, v_s is the instantaneous velocity of the point on this body corresponding to the fixed frame origin (see Figure 3.16).

As we did for ω_b and v_b, we merge ω_s and v_s into a six-dimensional spatial velocity:

$$\mathcal{V}_s = \left[\begin{array}{c} \omega_s \\ v_s \end{array} \right] \in \mathbb{R}^6, \;\; [\mathcal{V}_s] = \left[\begin{array}{cc} [\omega_s] & v_s \\ 0 & 0 \end{array} \right] = \dot{T}T^{-1} \in se(3), \qquad (3.74)$$

where $[\mathcal{V}_s]$ is the 4×4 matrix representation of $\mathcal{V}_s$. We call $\mathcal{V}_s$ the **spatial velocity in the fixed frame**.

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

(i) ω_b is the angular velocity relative to a stationary frame aligned with the **moving frame**;

(ii) ω_s is the angular velocity in **fixed frame** coordinates;

(iii) v_b is the linear velocity of the **moving frame origin**, in a stationary coordinate frame coincident with the **moving frame**;

(iv) v_s is the linear velocity of a point on the infinitely large moving body currently at the **fixed frame origin**, in **fixed frame** coordinates.

$\mathcal{V}_b$ can be obtained from $\mathcal{V}_s$ as follows:

$$\begin{aligned}
[\mathcal{V}_b] &= T^{-1}\dot{T} \\
&= T^{-1}[\mathcal{V}_s]T.
\end{aligned} \tag{3.75}$$

Going the other way,

$$[\mathcal{V}_s] = T[\mathcal{V}_b]T^{-1}. \tag{3.76}$$

Writing out the terms of Equation (3.76), we get

$$\mathcal{V}_s = \begin{bmatrix} R[\omega_b]R^T & -R[\omega_b]R^Tp + Rv_b \\ 0 & 0 \end{bmatrix}$$

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 $\mathcal{V}_b$ and $\mathcal{V}_s$:

$$\begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = \begin{bmatrix} R & 0 \\ [p]R & R \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}.$$

Because the 6×6 matrix pre-multiplying $\mathcal{V}_b$ is useful for changing the frame of reference for spatial velocities and forces, as we will see shortly, we give it its own name.

Definition 3.6. Given $T = (R, p) \in SE(3)$, its **adjoint representation** $[\mathrm{Ad}_T]$ is

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

For any $\mathcal{V} \in \mathbb{R}^6$, the **adjoint map** associated with T is

$$\mathcal{V}' = [\mathrm{Ad}_T]\mathcal{V},$$

also sometimes written as

$$\mathcal{V}' = \mathrm{Ad}_T(\mathcal{V}).$$

In terms of the matrix form $[\mathcal{V}] \in se(3)$ of $\mathcal{V} \in \mathbb{R}^6$,

$$[\mathcal{V}'] = T[\mathcal{V}]T^{-1}.$$

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

Proposition 3.14. *Let $T_1, T_2 \in SE(3)$, and $\mathcal{V} = (\omega, v)$. Then*

$$\mathrm{Ad}_{T_1}(\mathrm{Ad}_{T_2}(\mathcal{V})) = [\mathrm{Ad}_{T_1}][\mathrm{Ad}_{T_2}]\mathcal{V} = [\mathrm{Ad}_{T_1 T_2}]\mathcal{V}. \tag{3.77}$$

Also, for any $T \in SE(3)$ the following holds:

$$[\mathrm{Ad}_T^{-1}] = [\mathrm{Ad}_{T^{-1}}], \tag{3.78}$$

The second property follows from the first by choosing $T_1 = T^{-1}$ and $T_2 = T$, so that

$$\mathrm{Ad}_{T^{-1}}(\mathrm{Ad}_T(\mathcal{V})) = \mathrm{Ad}_{T^{-1}T}(\mathcal{V}) = \mathrm{Ad}_I(\mathcal{V}) = \mathcal{V}. \tag{3.79}$$

3.3.2.1 Summary of Results on Spatial Velocities

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

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

$$T_{sb}(t) = \begin{bmatrix} R(t) & p(t) \\ 0 & 1 \end{bmatrix}. \tag{3.80}$$

Then

$$T_{sb}^{-1}\dot{T}_{sb} = [\mathcal{V}_b] = \begin{bmatrix} [\omega_b] & v_b \\ 0 & 0 \end{bmatrix} \in se(3) \tag{3.81}$$

is the matrix representation of **spatial velocity in body coordinates**, *and*

$$\dot{T}_{sb}T_{sb}^{-1} = [\mathcal{V}_s] = \begin{bmatrix} [\omega_s] & v_s \\ 0 & 0 \end{bmatrix} \in se(3) \tag{3.82}$$

is the matrix representation of the **spatial velocity in fixed (space) coordinates**. *The spatial velocity vectors $\mathcal{V}_s$ and $\mathcal{V}_b$ are related by*

$$\mathcal{V}_s \;=\; \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = \begin{bmatrix} R & 0 \\ [p]R & R \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = [\mathrm{Ad}_{T_{sb}}]\mathcal{V}_b \tag{3.83}$$

$$\mathcal{V}_b \;=\; \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = \begin{bmatrix} R^T & 0 \\ -R^T[p] & R^T \end{bmatrix} \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = [\mathrm{Ad}_{T_{bs}}]\mathcal{V}_s. \tag{3.84}$$

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

$$\mathcal{V}_a = [\mathrm{Ad}_{T_{ab}}]\mathcal{V}_b, \;\; \mathcal{V}_b = [\mathrm{Ad}_{T_{ba}}]\mathcal{V}_a.$$

Again analogous to angular velocities, it is important to realize that for a given spatial velocity, 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* $\{s\}$. It is also important to realize that $\mathcal{V}_b$ is the spatial velocity relative to a stationary frame instantaneously coincident with $\{b\}$; it is not the velocity of the moving frame relative to the moving frame, which would always be zero.

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 w = 2 rad/s about an axis out of the page, at the point r in the plane. Inspecting the figure,
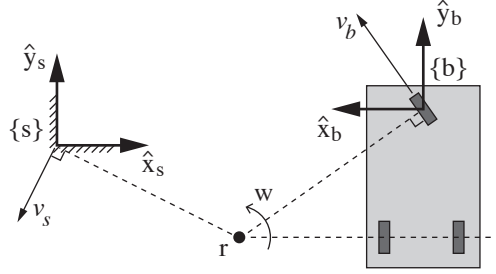
Figure 3.17: The spatial velocity corresponding to the instantaneous motion of the chassis of a three-wheel vehicle can be visualized as an angular velocity w about the point r.

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

$$T_{sb} = \left[\begin{array}{cc} R_{sb} & p_{sb} \\ 0 & 1 \end{array} \right] = \left[\begin{array}{cccc} -1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 0.4 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right].$$

From the figure and simple geometry, we get

$$v_s = \omega_s \times (-r_s) = r_s \times \omega_s = (-2, -4, 0)^T$$
$$v_b = \omega_b \times (-r_b) = r_b \times \omega_b = (2.8, 4, 0)^T$$

to get the spatial velocity $\mathcal{V}_s$ and $\mathcal{V}_b$:

$$\mathcal{V}_s = \left[\begin{array}{c} \omega_s \\ v_s \end{array} \right] = \left[\begin{array}{c} 0 \\ 0 \\ 2 \\ -2 \\ -4 \\ 0 \end{array} \right], \quad \mathcal{V}_b = \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right] = \left[\begin{array}{c} 0 \\ 0 \\ -2 \\ 2.8 \\ 4 \\ 0 \end{array} \right].$$

To confirm these results, try calculating $\mathcal{V}_s = [\mathrm{Ad}_{T_{sb}}]\mathcal{V}_b$.

3.3.2.2 The Screw Interpretation of Spatial Velocity

Just as an angular velocity ω can be viewed as $\hat{\omega}\dot{\theta}$, where $\hat{\omega}$ is the unit rotation axis and $\dot{\theta}$ is the rate of rotation about that axis, a spatial velocity $\mathcal{V}$ can be interpreted as a **screw axis** $\mathcal{S}$ and a velocity $\dot{\theta}$ 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 $\mathcal{S}$ 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
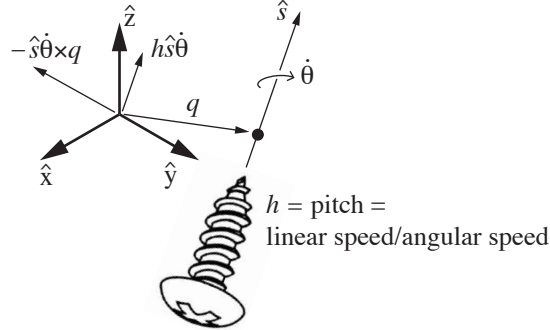
Figure 3.18: A screw axis $\mathcal{S}$ represented by a point q, a unit direction $\hat{s}$, and a pitch h.

ratio of the linear velocity along the screw axis to the angular velocity $\dot{\theta}$ about the screw axis (Figure 3.18).

Using Figure 3.18 and geometry, we can write the spatial velocity $\mathcal{V} = (\omega, v)$ corresponding to an angular velocity $\dot{\theta}$ about $\mathcal{S}$ (represented by $\{q, \hat{s}, h\}$) as

$$\mathcal{V} = \left[\begin{array}{c} \omega \\ v \end{array} \right] = \left[\begin{array}{c} \hat{s}\dot{\theta} \\ -\hat{s}\dot{\theta} \times q + h\hat{s}\dot{\theta} \end{array} \right].$$

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 $\mathcal{V} = (\omega, v)$ where $\omega \neq 0$, there exists an equivalent screw axis $\{q, \hat{s}, h\}$ and velocity $\dot{\theta}$, where $\hat{s} = \omega/\|\omega\|$, $\dot{\theta} = \|\omega\|$, $h = \hat{\omega}^T v/\dot{\theta}$, 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 $\mathcal{S}$ 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 $\mathcal{S}$ using a normalized version of any spatial velocity $\mathcal{V} = (\omega, v)$ along/about the axis:

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

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

This leads to the following definition of a "unit" screw axis:

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

$$\mathcal{S} = \left[\begin{array}{c} \omega \\ v \end{array} \right] \in \mathbb{R}^6,$$

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 $\|v\| = 1$, the pitch of the screw is $h = \infty$ and the spatial velocity is a translation along the axis defined by v.

The 4×4 matrix representation $[\mathcal{S}]$ of $\mathcal{S}$ is

$$[\mathcal{S}] = \left[\begin{array}{cc} [\omega] & v \\ 0 & 0 \end{array} \right] \in se(3), \quad [\omega] = \left[\begin{array}{ccc} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{array} \right] \in so(3), \quad (3.85)$$

where the bottom row of $[\mathcal{S}]$ consists of all zeros.

   Important: Although we use the pair (ω, v) for both screw axes (where one of $\|\omega\|$ or $\|v\|$ must be unit) and general spatial velocities, their meaning should be clear from context.

   Since a screw axis $\mathcal{S}$ is just a normalized spatial velocity, a screw axis represented as $\mathcal{S}_a$ in a frame {a} is related to the representation $\mathcal{S}_b$ in a frame {b} by

$$\mathcal{S}_a = [\mathrm{Ad}_{T_{ab}}]\mathcal{S}_b, \quad \mathcal{S}_b = [\mathrm{Ad}_{T_{ba}}]\mathcal{S}_a.$$

3.3.3 Exponential Coordinate Representation of Rigid-Body Motions

3.3.3.1 Exponential Coordinates of Rigid-Body Motions

In the planar example in Section 3.1, we saw that any planar rigid-body displacement can be achieved by rotating the rigid body about some fixed point in the plane (for a pure translation, this point lies at infinity). A similar result also exists for spatial rigid-body displacements: called the **Chasles-Mozzi Theorem**, it states that every rigid-body displacement can be expressed as a **twist** about a fixed screw axis $\mathcal{S}$ in space.

   By analogy to the exponential coordinates $\hat{\omega}\theta$ for rotations, we define the six-dimensional **exponential coordinates of a homogeneous transformation** T as $\mathcal{S}\theta \in \mathbb{R}^6$, where $\mathcal{S}$ is the screw axis and θ is the distance that must be traveled along/about the screw axis to take a frame from the origin I to T. If the pitch of the screw axis $\mathcal{S} = (\omega, v)$ is finite, then $\|\omega\| = 1$ and θ corresponds to the angle of rotation about the screw axis. If the pitch of the screw is infinite, then $\omega = 0$ and $\|v\| = 1$, and θ corresponds to the linear distance traveled along the screw axis.

   Also by analogy to the rotations, we define a matrix exponential and matrix logarithm:

$$\begin{array}{rcl} \exp: & [\mathcal{S}]\theta \in se(3) & \rightarrow & T \in SE(3) \\ \log: & T \in SE(3) & \rightarrow & [\mathcal{S}]\theta \in se(3) \end{array}$$

We begin by deriving a closed-form expression for the matrix exponential $e^{[\mathcal{S}]\theta}$. Expanding the matrix exponential in series form leads to

$$
\begin{aligned}
e^{[\mathcal{S}]\theta} &= I + [\mathcal{S}]\theta + [\mathcal{S}]^2\frac{\theta^2}{2!} + [\mathcal{S}]^3\frac{\theta^3}{3!} + \ldots \\
&= \begin{bmatrix} e^{[\omega]\theta} & G(\theta)v \\ 0 & 1 \end{bmatrix}, \quad G(\theta) = I\theta + [\omega]\frac{\theta^2}{2!} + [\omega]^2\frac{\theta^3}{3!} + \ldots \quad (3.86)
\end{aligned}
$$

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

$$
\begin{aligned}
G(\theta) &= I\theta + [\omega]\frac{\theta^2}{2!} + [\omega]^2\frac{\theta^3}{3!} + \ldots \\
&= I\theta + \left(\frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \ldots\right)[\omega] + \left(\frac{\theta^3}{3!} - \frac{\theta^5}{5!} + \frac{\theta^7}{7!} - \ldots\right)[\omega]^2 \\
&= I\theta + (1 - \cos\theta)[\omega] + (\theta - \sin\theta)[\omega]^2. \quad (3.87)
\end{aligned}
$$

Putting everything together,

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

$$
e^{[\mathcal{S}]\theta} = \begin{bmatrix} e^{[\omega]\theta} & \left(I\theta + (1 - \cos\theta)[\omega] + (\theta - \sin\theta)[\omega]^2\right)v \\ 0 & 1 \end{bmatrix}. \quad (3.88)
$$

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

$$
e^{[\mathcal{S}]\theta} = \begin{bmatrix} I & v\theta \\ 0 & 1 \end{bmatrix}. \quad (3.89)
$$

The latter result of the proposition can be verified directly from the series expansion of $e^{[\mathcal{S}]\theta}$ with ω 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 $\mathcal{S} = (\omega, v)$ and a scalar θ such that

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

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 tr $R \neq -1$, one solution is given by

$$[\omega] = \frac{1}{2 \sin \theta}(R - R^T) \tag{3.91}$$

$$v = G^{-1}(\theta)p, \tag{3.92}$$

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

$$G^{-1}(\theta) = \frac{1}{\theta}I - \frac{1}{2}[\omega] + \left(\frac{1}{\theta} - \frac{1}{2}\cot\frac{\theta}{2}\right)[\omega]^2. \tag{3.93}$$

Finally, if tr $R = -1$, we choose $\theta = \pi$, and $[\omega]$ can be obtained via the matrix logarithm formula on $SO(3)$. Once $[\omega]$ and θ 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 $\mathcal{S} = (\omega, v) \in \mathbb{R}^6$ such that $e^{[\mathcal{S}]\theta} = T$. The vector $\mathcal{S}\theta \in \mathbb{R}^6$ comprises the exponential coordinates for T and the matrix $[\mathcal{S}]\theta \in se(3)$ is a matrix logarithm of T.

(i) If $R = I$, then set $\omega = 0$, $v = p/\|p\|$, and $\theta = \|p\|$.

(ii) If tr $R = -1$, then set $\theta = \pi$, and $[\omega] = \log R$ as determined by the matrix logarithm formula on $SO(3)$ for the case tr $R = -1$. v is then given by $v = G^{-1}(\theta)p$.

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

$$[\omega] = \frac{1}{2\sin\theta}(R - R^T) \tag{3.94}$$

$$v = G^{-1}(\theta)p, \tag{3.95}$$

where $G^{-1}(\theta)$ is given by Equation (3.93).

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 the $SE(2)$ matrices in Figure 3.19:

$$T_{sb} = \begin{bmatrix} \cos 30° & -\sin 30° & 1 \\ \sin 30° & \cos 30° & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_{sc} = \begin{bmatrix} \cos 60° & -\sin 60° & 2 \\ \sin 60° & \cos 60° & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

For this example, the rigid-body displacement occurs in the x-y plane. The corresponding screw motion therefore has its screw axis in the direction of the
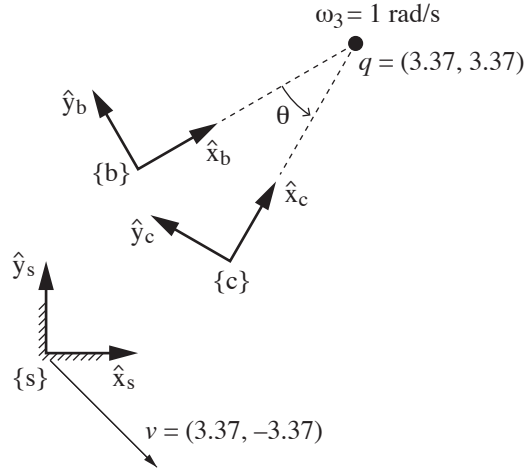
Figure 3.19: Two frames in a plane.

z-axis, and is of zero pitch. The screw axis $\mathcal{S} = (\omega, v)$, expressed in {s}, is of the form

$$\omega = (0, 0, \omega_3)^T$$
$$v = (v_1, v_2, 0)^T.$$

Using this reduced form, we seek the screw motion that displaces the frame at T_{sb} to T_{sc}, i.e., $T_{sc} = e^{[\mathcal{S}]\theta} T_{sb}$, or

$$T_{sc} T_{sb}^{-1} = e^{[\mathcal{S}]\theta},$$

where

$$[\mathcal{S}] = \begin{bmatrix} 0 & -\omega_3 & v_1 \\ \omega_3 & 0 & v_2 \\ 0 & 0 & 0 \end{bmatrix}.$$

We can apply the matrix logarithm algorithm directly to $T_{sc} T_{sb}^{-1}$ to obtain $[\mathcal{S}]$ (and therefore $\mathcal{S}$) and θ as follows:

$$[\mathcal{S}] = \begin{bmatrix} 0 & -1 & 3.37 \\ 1 & 0 & -3.37 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathcal{S} = \begin{bmatrix} \omega_3 \\ v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 3.37 \\ -3.37 \end{bmatrix}, \quad \theta = \pi/6 \text{ rad (or } 30°).$$

The value of $\mathcal{S}$ means that the constant screw axis, expressed in the fixed frame {s}, is represented by an angular velocity of 1 rad/s about $\hat{z}_s$ and a linear velocity of a point attached to the moving frame, but currently at the origin of {s}, of $(3.37, -3.37)$ expressed in the {s} frame.

Alternatively, we can observe that the displacement is not a pure translation—T_{sb} and T_{sc} have rotation components that differ by an angle of $30°$—and quickly
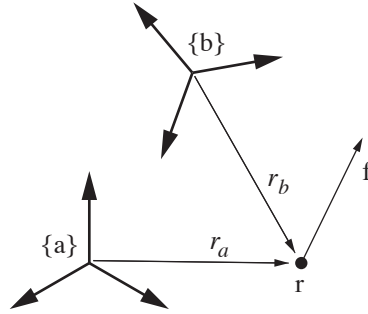
Figure 3.20: Relation between a spatial force represented as $\mathcal{F}_a$ and $\mathcal{F}_b$.

determine that $\theta = 30°$ and $\omega_3 = 1$. We can also graphically determine the point $q = (q_x, q_y)$ in the $\hat{x}_s$-$\hat{y}_s$ plane that the screw axis must pass through; for our example this point is given by $q = (3.37, 3.37)$.

3.4 Spatial Forces

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

$$m_a = r_a \times f_a.$$

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

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

$$\mathcal{F}_a = \begin{bmatrix} m_a \\ f_a \end{bmatrix} \in \mathbb{R}^6. \tag{3.96}$$

A spatial force is also known as a **wrench**. If more than one spatial force acts on a rigid body, the total spatial force on the body is simply the vector sum of the individual spatial forces. A spatial force with zero linear component is called a **pure moment**.

A spatial force in the {a} frame can be represented in another frame {b} if T_{ba} is known (Figure 3.20). One way to derive the relationship between $\mathcal{F}_a$ and $\mathcal{F}_b$ 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 $\mathcal{F}_a$ and $\mathcal{F}_b$ is to (1) use the results we have already derived relating the representations $\mathcal{V}_a$ and $\mathcal{V}_b$ of the same spatial velocity, and (2) use the fact that the power generated (or dissipated) by an $(\mathcal{F}, \mathcal{V})$ 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 frame-independent quantity. Because of this, we know

$$\mathcal{V}_b^T \mathcal{F}_b = \mathcal{V}_a^T \mathcal{F}_a. \tag{3.97}$$

From Proposition 3.15 we know that $\mathcal{V}_a = [\mathrm{Ad}_{T_{ab}}]\mathcal{V}_b$, and therefore Equation (3.97) can be rewritten as

$$\begin{aligned} \mathcal{V}_b^T \mathcal{F}_b &= ([\mathrm{Ad}_{T_{ab}}]\mathcal{V}_b)^T \mathcal{F}_a \\ &= \mathcal{V}_b^T [\mathrm{Ad}_{T_{ab}}]^T \mathcal{F}_a. \end{aligned}$$

Since this must hold for all $\mathcal{V}_b$, this simplifies to

$$\mathcal{F}_b = [\mathrm{Ad}_{T_{ab}}]^T \mathcal{F}_a. \tag{3.98}$$

Similarly,

$$\mathcal{F}_a = [\mathrm{Ad}_{T_{ba}}]^T \mathcal{F}_b. \tag{3.99}$$

Proposition 3.17. *Given a reference frame {a}, let $f_a \in \mathbb{R}^3$ and $m_a = r_a \times f_a \in \mathbb{R}^3$ represent a spatial force in frame {a} coordinates, where $r_a \in \mathbb{R}^3$ is the vector from the {a} frame origin to* r, *also expressed in {a} frame coordinates. Similarly, given another reference frame {b}, let $f_b \in \mathbb{R}^3$ and $m_b = r_b \times f_b \in \mathbb{R}^3$ be representations of the same spatial force, where $r_b \in \mathbb{R}^3$ is the vector from the {b} frame origin to* r, *also expressed in {b} frame coordinates. Defining the spatial forces $\mathcal{F}_a = (m_a, f_a)$ and $\mathcal{F}_b = (m_b, f_b)$, $\mathcal{F}_a$ and $\mathcal{F}_b$ are related by*

$$\begin{aligned} \mathcal{F}_b &= \mathrm{Ad}_{T_{ab}}^T(\mathcal{F}_a) = [\mathrm{Ad}_{T_{ab}}]^T \mathcal{F}_a \tag{3.100} \\ \mathcal{F}_a &= \mathrm{Ad}_{T_{ba}}^T(\mathcal{F}_b) = [\mathrm{Ad}_{T_{ba}}]^T \mathcal{F}_b. \tag{3.101} \end{aligned}$$

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	Rigid-Body Motions
$R \in SO(3) : 3 \times 3$ matrices satisfying $R^T R = I, \det R = 1$	$T \in SE(3) : 4 \times 4$ matrices $T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix},$ where $R \in SO(3), p \in \mathbb{R}^3$
$R^{-1} = R^T$	$T^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix}$
$R_{ab}R_{bc} = R_{ac}, \quad R_{ab}v_b = v_a$	$T_{ab}T_{bc} = T_{ac}, \quad T_{ab}v_b = v_a$
$R = \mathrm{Rot}(\hat{\omega}, \theta)$ $R_{sb'} = RR_{sb}$: rotate θ about $\hat{\omega}_s = \hat{\omega}$ $R_{sb''} = R_{sb}R$: rotate θ about $\hat{\omega}_b = \hat{\omega}$	$T = \begin{bmatrix} \mathrm{Rot}(\hat{\omega}, \theta) & p \\ 0 & 1 \end{bmatrix}$ $T_{sb'} = TT_{sb}$: rotate θ about $\hat{\omega}_s = \hat{\omega}$, translate by p in {s} $T_{sb''} = T_{sb}T$: translate by p in {b}, rotate θ about $\hat{\omega}_b = \hat{\omega}$
unit rotation axis is $\hat{\omega} \in \mathbb{R}^3$, where $\|\hat{\omega}\| = 1$	"unit" screw axis is $\mathcal{S} = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6$, where either (i) $\|\omega\| = 1$ or (ii) $\omega = 0$ and $\|v\| = 1$
	for a screw axis $\{q, \hat{s}, h\}$ with finite h, $\mathcal{S} = \begin{bmatrix} \omega \\ v \end{bmatrix} = \begin{bmatrix} \hat{s} \\ -\hat{s} \times q + h\hat{s} \end{bmatrix}$
angular velocity can be written $\omega = \hat{\omega}\dot{\theta}$	spatial velocity can be written $\mathcal{V} = \mathcal{S}\dot{\theta}$
for $\omega \in \mathbb{R}^3$, $[\omega] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in so(3)$ (the same holds for any three-vector, angular velocity or not)	for $\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6$, $[\mathcal{V}] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} \in se(3)$ (the pair (ω, v) can be a spatial velocity $\mathcal{V}$ or a "unit" screw axis $\mathcal{S}$, depending on the context)
$\dot{R}R^{-1} = [\omega_s], \quad R^{-1}\dot{R} = [\omega_b]$	$\dot{T}T^{-1} = [\mathcal{V}_s], \quad T^{-1}\dot{T} = [\mathcal{V}_b]$
	$[\mathrm{Ad}_T] = \begin{bmatrix} R & 0 \\ [p]R & R \end{bmatrix} \in \mathbb{R}^{6 \times 6}$
$\hat{\omega}_a = R_{ab}\hat{\omega}_b, \quad \omega_a = R_{ab}\omega_b$	$\mathcal{S}_a = [\mathrm{Ad}_{T_{ab}}]\mathcal{S}_b, \quad \mathcal{V}_a = [\mathrm{Ad}_{T_{ab}}]\mathcal{V}_b$
$\hat{\omega}\theta \in \mathbb{R}^3$ are exp coords for $R \in SO(3)$	$\mathcal{S}\theta \in \mathbb{R}^6$ are exp coords for $T \in SE(3)$
$\exp : [\hat{\omega}]\theta \in so(3) \to R \in SO(3)$ $R = e^{[\hat{\omega}]\theta} = I + \sin\theta[\hat{\omega}] + (1 - \cos\theta)[\hat{\omega}]^2$	$\exp : [\mathcal{S}]\theta \in se(3) \to T \in SE(3)$ $T = e^{[\mathcal{S}]\theta} = \begin{bmatrix} e^{[\hat{\omega}]\theta} & * \\ 0 & 1 \end{bmatrix}$ where $* =$ $(I\theta + (1 - \cos\theta)[\omega] + (\theta - \sin\theta)[\omega]^2)v$
$\log : R \in SO(3) \to [\hat{\omega}]\theta \in so(3)$ algorithm in Section 3.2.3.3	$\log : T \in SE(3) \to [\mathcal{S}]\theta \in se(3)$ algorithm in Section 3.3.3.2
$m_a = R_{ab}m_b$	$\mathcal{F}_a = [\mathrm{Ad}_{T_{ba}}]^T \mathcal{F}_b$

3.6   Notes and References

More detailed coverage of the various parametrizations of $SO(3)$ can be found in, e.g., [32] and the references cited there. The treatment of the matrix exponential representation for screw motions is based on the work of Brockett [4], and a more mathematically detailed discussion can be found in [25]. Classical screw theory is presented in its original form in R. Ball's treatise [2]. More modern (algebraic and geometric) treatments can be found in, e.g., Bottema and Roth [3], Angeles [1], and McCarthy [20].

3.7   Exercises

1. Given a fixed frame $\{\hat{X}, \hat{Y}, \hat{Z}\}$ for physical space, let p be a point whose co-ordinates are $(\frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{6}}, \frac{1}{\sqrt{2}})$. 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.
(a) What are the coordinates of the point p following these three rotations?
(b) Find the rotation matrix R such that Rp are the coordinates you obtained in (a).

2. (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:

$$\begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{2} & -\frac{1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{\sqrt{2}} \end{bmatrix}.$$

3. 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 θ and ϕ as follows:

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta\cos\phi & \cos\theta\cos\phi & -\sin\phi \\ \sin\theta\sin\phi & \cos\theta\sin\phi & \cos\phi \end{bmatrix}.$$

What should the range of values be for θ and ϕ?

4. (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 $\{\mu+i\nu, \mu-i\nu, 1\}$, where $\mu^2 + \nu^2 = 1$.
(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 $\mu^2 + \nu^2 = 1$. (*Hint*: Denote the eigenvector associated with the eigenvalue $\mu + i\nu$ 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.)

5. (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 $t \to \infty$?

6. 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}.

7. 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 $\dot{z}(t)$.)

8. (a) Prove the matrix identity $Me^A M^{-1} = e^{MAM^{-1}}$.

(b) Under what conditions on $A, B \in \mathbb{R}^{n \times n}$ is the following matrix identity true?

$$e^A e^B = e^{A+B}.$$

9. Consider a wrist mechanism with two revolute joints θ_1 and θ_2, in which the end-effector frame orientation $R \in SO(3)$ is given by

$$R = e^{[w_1]\theta_1} e^{[w_2]\theta_2},$$

with $\omega_1 = (0, 0, 1)$ and $\omega_2 = (0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$. Determine whether the following orientation is reachable (that is, find, if it exists, a solution (θ_1, θ_2) for the

following R):

$$R = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\ 0 & 1 & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}$$
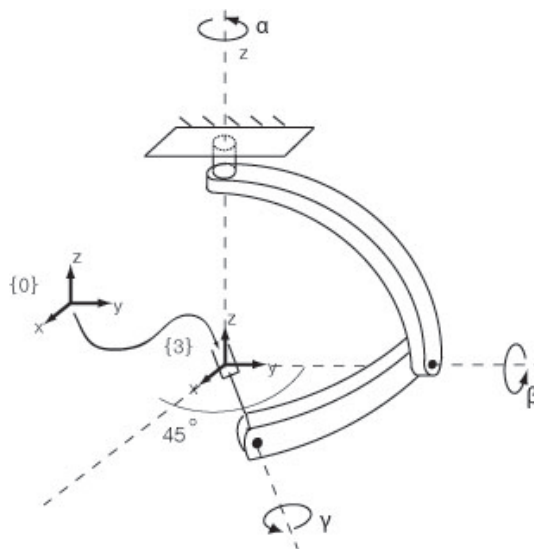
Figure 3.21: A three degree of freedom wrist mechanism.

10. Figure 3.21 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 (α, β, γ) 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 π.

(i) $R_{03} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$.

(ii) $R_{03} = e^{[w]\frac{\pi}{2}}$, where $w = (0, \frac{1}{\sqrt{5}}, \frac{2}{\sqrt{5}})$.

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

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

and recalling that tr $R = -1$ implies $\theta = \pi$, the above equation simplifies to

$$R = I + 2[\omega]^2 = \begin{bmatrix} 1 - 2(\omega_2^2 + \omega_3^2) & 2\omega_1\omega_2 & 2\omega_1\omega_3 \\ 2\omega_1\omega_2 & 1 - 2(\omega_1^2 + \omega_3^2) & 2\omega_2\omega_3 \\ 2\omega_1\omega_2 & 2\omega_2\omega_3 & 1 - 2(\omega_1^2 + \omega_2^2) \end{bmatrix}$$

Using the fact that $\omega_1^2 + \omega_2^2 + \omega_3^2 = 1$, is it correct to conclude that

$$\omega_1 = \sqrt{\frac{r_{11} + 1}{2}}, \quad \omega_2 = \sqrt{\frac{r_{22} + 1}{2}}, \quad \omega_3 = \sqrt{\frac{r_{33} + 1}{2}}.$$

is also a solution?

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

$$\begin{aligned} R - I &= 2[\omega]^2 \\ [\omega](R - I) &= 2[\omega]^3 = -2[\omega] \\ [\omega](R + I) &= 0. \end{aligned}$$

The resulting equation is a system of three linear equations in $(\omega_1, \omega_2, \omega_3)$. What is the relation between the solution to this linear system and the logarithm of R?

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

(b) Given rotation matrices $A = \mathrm{Rot}(\hat{z}, \alpha)$ and $B = \mathrm{Rot}(\hat{z}, \beta)$, 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) 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.

(b) 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

$$\|A - R\|^2 = \mathrm{tr}\,(A - R)(A - R)^T.$$

14. (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 $RQ \in SO(3)$.

(d) Compare the number of arithmetic operations for multiplying two rotation matrices versus two unit quaternions. Which requires fewer arithmetic operations?
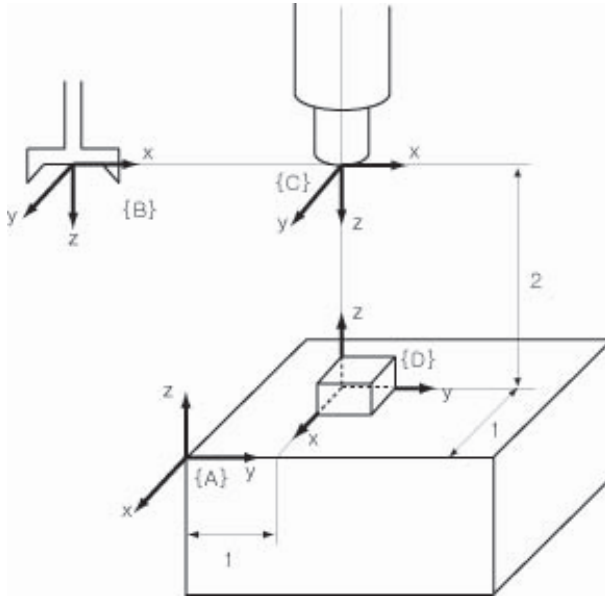
Figure 3.22: Four reference frames defined in a robot's work environment.

15. Consider the robot of Figure 3.22, in which four reference frames are depicted: 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

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

16. Consider a robot arm mounted on a spacecraft as shown in Figure 3.23, 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}.
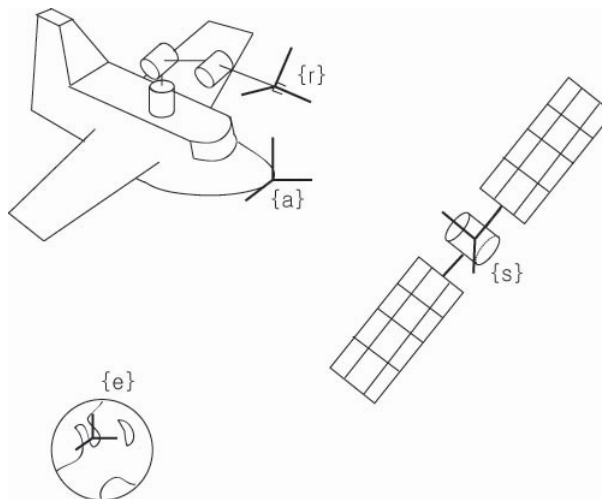
Figure 3.23: A robot arm mounted on a spacecraft.

(b) Suppose the frame $\{s\}$ origin as seen from $\{e\}$ is $(1, 1, 1)$. Suppose further-more that

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

Write down the coordinates of the frame $\{s\}$ origin as seen from frame $\{r\}$.
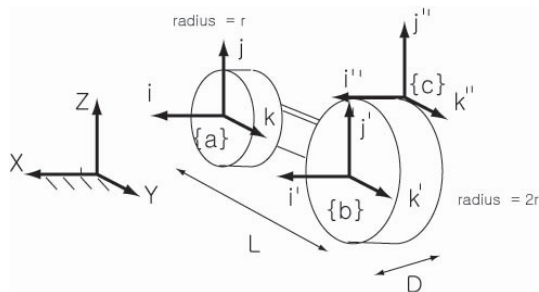
Figure 3.24: A classical bicycle with a larger front wheel.

17. Consider the classical 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 θ (assume $\theta = 0$ at the instant shown in the figure).
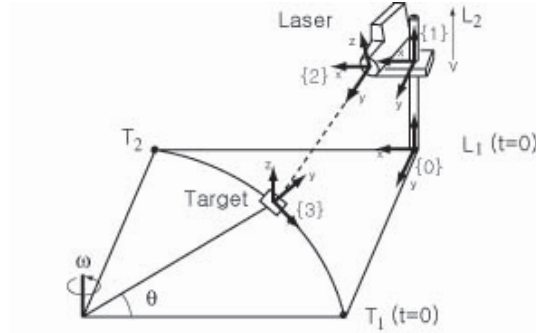
Figure 3.25: A laser tracking a moving target.

18. A target moves along a circular path at constant angular velocity ω rad/sec as shown in Figure 3.25. 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.
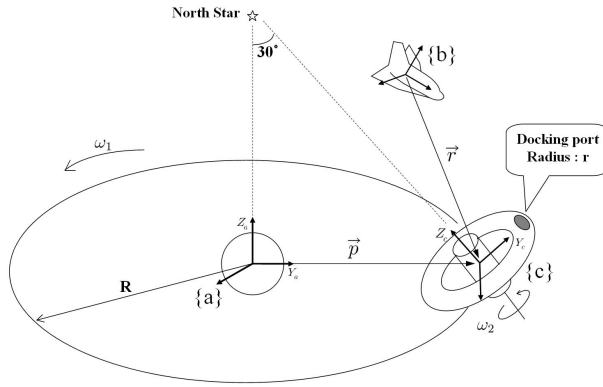
Figure 3.26: Spacecraft and space station.

19. Suppose the space station of Figure 18 is in circular orbit around the earth, and at the same time rotates about an axis always pointing toward the north star. A spacecraft heading toward the space station is unable to locate the docking port due to an instrument malfunction. An earth-based ground station

sends the following information to the spacecraft:

$$T_{ab} = \begin{pmatrix} 0 & -1 & 0 & -100 \\ 1 & 0 & 0 & 300 \\ 0 & 0 & 1 & 500 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad p_a = \begin{pmatrix} 0 \\ 800 \\ 0 \end{pmatrix},$$

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.
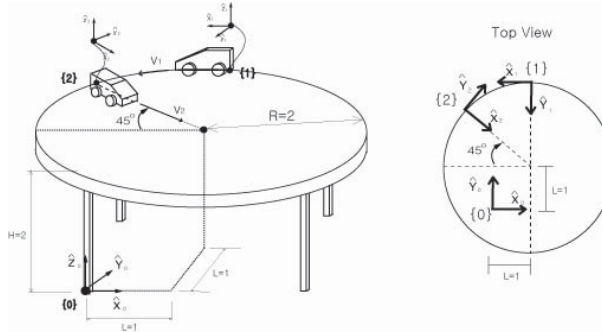
Figure 3.27: Two toy cars on a round table.

20. 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.

21. 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, with values $\omega_1 = \frac{\pi}{4}$, $\omega_2 = \frac{\pi}{8}$, $w_3 = -\frac{\pi}{4}$ rad/sec. 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 $t = 4$.

22. A cube undergoes two different screw motions from frame $\{1\}$ to frame $\{2\}$ as shown in Figure 3.29. In both cases (a) and (b), the initial configuration of the cube is

$$T_{01} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$
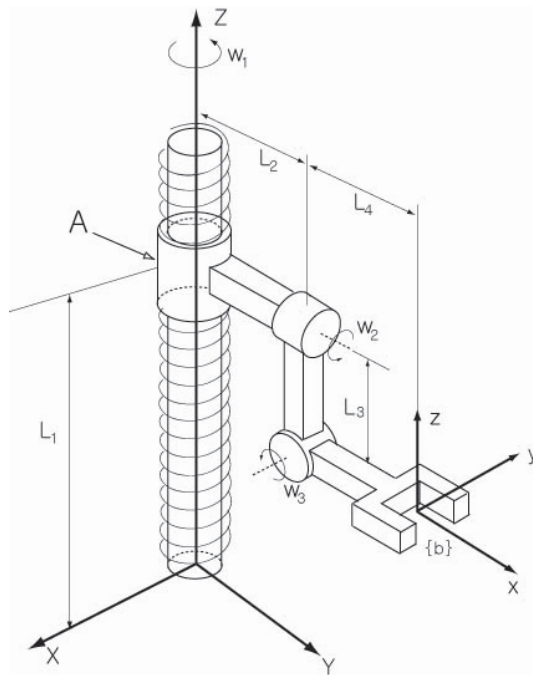
Figure 3.28: A robot arm with a screw joint.

(a) a   (b) b
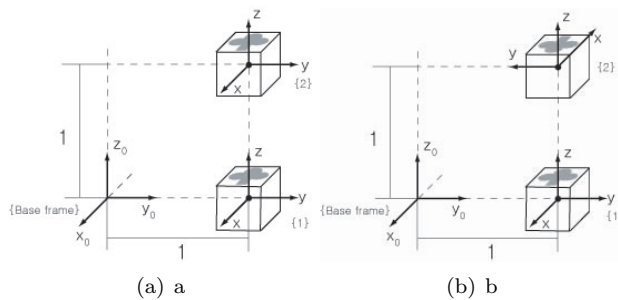
Figure 3.29: A cube undergoing two different screw motions.

(a) For each case (a) and (b), find the screw parameter $S = (\omega, v)$ such that $T_{02} = e^{[S]}T_{01}$, where no constraints are placed on ω or v.

(b) Repeat (a), this time with the constraint that $\|\omega\| \in [-\pi, \pi]$.

23. A particle starts from the origin, and undergoes a radially increasing circular spiral motion in the $\hat{y}$ direction as illustrated in Figure 3.30. For every translation of 10 units in the $\hat{y}$ direction, the particle completes one revolution. Attaching a moving frame to the particle, the motion of this moving frame can
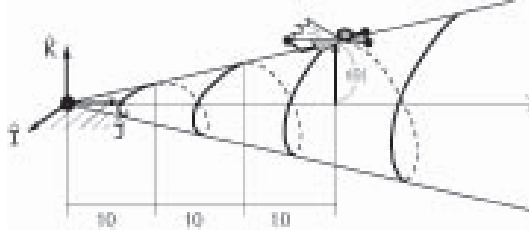
Figure 3.30: A particle undergoing a screw motion.

be expressed in the form
$$T(\theta) = e^{[\mathcal{S}_a]\theta} e^{[\mathcal{S}_b]\theta},$$
for some $\mathcal{S}_a = (\omega_a, v_a)$, $\mathcal{S}_b = (\omega_a, v_b)$. Find $\mathcal{S}_a$ and $\mathcal{S}_b$.

24. Given $\omega \in \mathbb{R}^3$, $\|\omega\| = 1$, and $\theta \in \mathbb{R}$, show that

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

Under what conditions, if any, will the inverse fail to exist? (*Hint:* Express the inverse as a quadratic matrix polynomial in $[\omega]$, and determine the coefficients. The quadratic polynomial assumption can be justified via the identity $[\omega]^3 = -[\omega]$.)

25. 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 $R_{ab} = e^{[\omega]\theta}$. Suppose the fixed frame is displaced to another frame $\{o'\}$, and that $T_{o'a} = ST_{oa}$, $T_{o'b} = ST_{o'b}$ for some constant $S = (R_s, p_s) \in SE(3)$.
(a) Evaluate $\text{dist}(T_{o'a}, T_{o'b})$ using the above distance formula.
(b) Under what conditions on S does $\text{dist}(T_{oa}, T_{ob}) = \text{dist}(T_{o'a}, T_{o'b})$?

26. Two frames $\{a\}$ and $\{b\}$ are attached to a moving rigid body. Show that the spatial velocity of $\{a\}$ in space frame coordinates is the same as the spatial velocity of $\{b\}$ in space frame coordinates.

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 ϕ of the end-effector frame as a function of the joint angles $(\theta_1, \theta_2, \theta_3)$ are then given by

$$
\begin{aligned}
x &= L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) + L_3 \cos(\theta_1 + \theta_2 + \theta_3) & (4.1) \\
y &= L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) + L_3 \sin(\theta_1 + \theta_2 + \theta_3) & (4.2) \\
\phi &= \theta_1 + \theta_2 + \theta_3. & (4.3)
\end{aligned}
$$

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 would be to first attach 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,

$$
T_{04} = T_{01} T_{12} T_{23} T_{34}, \tag{4.4}
$$
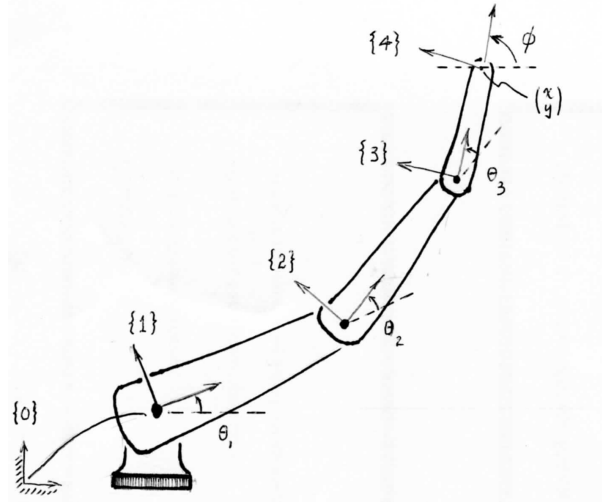
Figure 4.1: Forward kinematics of a 3R planar open chain. For each frame, the $\hat{x}$ and $\hat{y}$ axes are shown, and the $\hat{z}$ axes are parallel and out of the page.

where

$$
T_{01} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad
T_{12} = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & L_1 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
T_{23} = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & L_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad
T_{34} = \begin{bmatrix} 1 & 0 & 0 & L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.5}
$$

Observe that T_{34} is constant, and that each remaining $T_{i-1,i}$ depends only on the joint variable θ_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

$$
M = \begin{bmatrix} 1 & 0 & 0 & L_1 + L_2 + L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.6}
$$

Now consider each of the revolute joint axes to be a zero-pitch screw axis. If θ_1 and θ_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

$$\mathcal{S}_3 = \begin{bmatrix} \omega \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ -(L_1 + L_2) \\ 0 \end{bmatrix}$$

or

$$[\mathcal{S}_3] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -(L_1 + L_2) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Therefore, for any θ_3, the matrix exponential representation for screw motions from the previous chapter allows us to write

$$T_{04} = e^{[\mathcal{S}_3]\theta_3} M \quad (\text{for } \theta_1 = \theta_2 = 0). \tag{4.7}$$

Now, for $\theta_1 = 0$ and any fixed (but arbitrary) θ_3, rotation about joint two can be viewed as applying a screw motion to the rigid (link two)/(link three) pair, i.e.,

$$T_{04} = e^{[\mathcal{S}_2]\theta_2} e^{[\mathcal{S}_3]\theta_3} M \quad (\text{for } \theta_1 = 0), \tag{4.8}$$

where $[\mathcal{S}_3]$ and M are as defined previously, and

$$[\mathcal{S}_2] = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -L_1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.9}$$

Finally, keeping θ_2 and θ_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 $(\theta_1, \theta_2, \theta_3)$,

$$T_{04} = e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} e^{[\mathcal{S}_3]\theta_3} M, \tag{4.10}$$

where

$$[\mathcal{S}_1] = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.11}$$

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
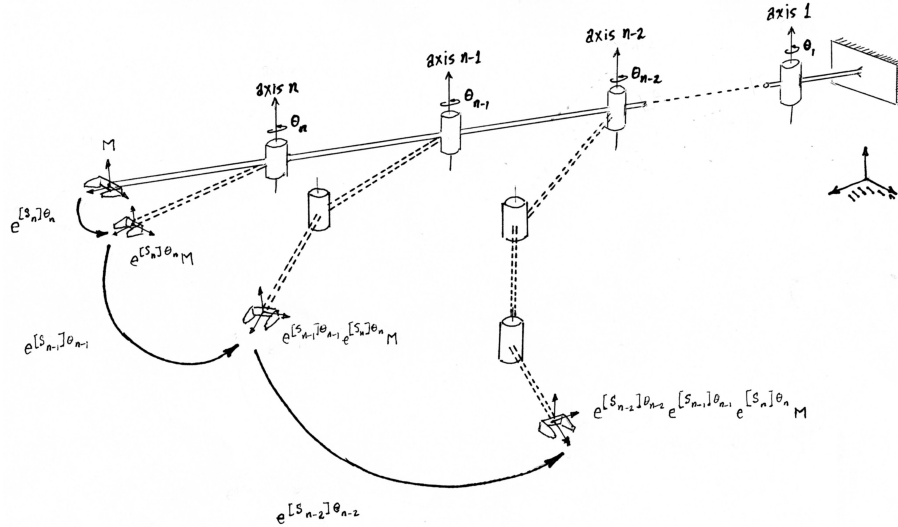
Figure 4.2: Illustration of the PoE formula for an n-link spatial open chain.

on the **Denavit-Hartenberg parameters** (D-H parameters), which corresponds to 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 is a minimal representation, in that it requires the smallest number of parameters to describe the the robot's kinematic structure. The PoE representation is not minimal, in that the number of parameters needed to describe the screw axes of a robot is larger than the number of parameters needed in the D-H representation, but it has other 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 given in Appendix C.

4.1   Product of Exponentials Formula

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 displacement (rotation for revolute joints, translation for prismatic joints) for each joint specified. Let $M \in SE(3)$ denote the configuration of the end-effector frame relative to the fixed base frame when the robot is in its zero position.

Now suppose joint n is displaced to some joint value θ_n. The end-effector frame M then undergoes a displacement of the form

$$T = e^{[\mathcal{S}_n]\theta_n} M, \tag{4.12}$$

where $T \in SE(3)$ is the new configuration of the end-effector frame, and $\mathcal{S}_n = (\omega_n, v_n)$ 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 θ_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 θ_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

$$T = e^{[\mathcal{S}_{n-1}]\theta_{n-1}} \left(e^{[\mathcal{S}_n]\theta_n} M \right). \tag{4.13}$$

Continuing with this reasoning and now allowing all the joints $(\theta_1, \ldots, \theta_n)$ to vary, it follows that

$$T = e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_{n-1}]\theta_{n-1}} e^{[\mathcal{S}_n]\theta_n} M. \tag{4.14}$$

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 $\mathcal{S}_1 \ldots \mathcal{S}_n$, 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.
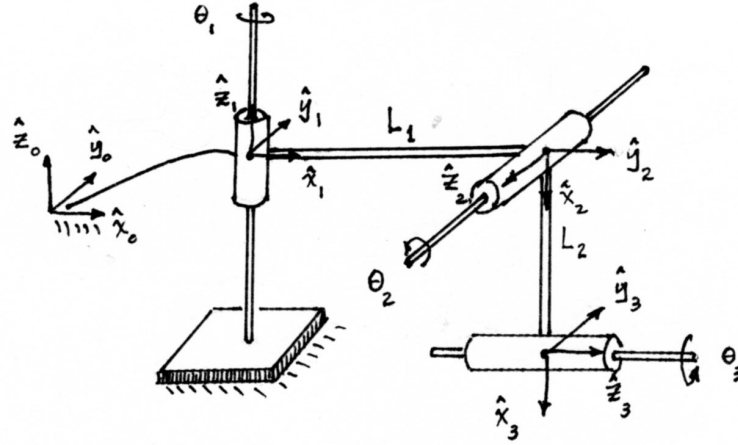
Figure 4.3: A 3R spatial open chain.

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 {3} as indicated in the figure, and express all vectors and homogeneous transformations in terms of the fixed frame. The forward kinematics will be of the form

$$T = e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} e^{[\mathcal{S}_3]\theta_3} M,$$

where $M \in SE(3)$ is the end-effector frame configuration when the robot is in its zero position. By inspection M can be obtained as

$$M = \begin{bmatrix} 0 & 0 & 1 & L_1 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & -L_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The screw axis $\mathcal{S}_1 = (\omega_1, v_1)$ 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 $\mathcal{S}_2$ 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 $\mathcal{S}_3$ 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
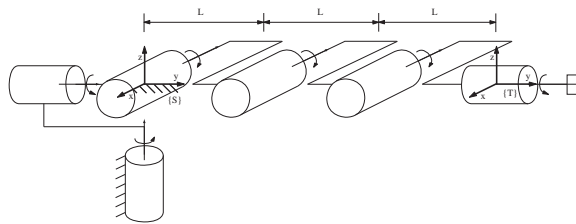
Figure 4.4: PoE forward kinematics for the 6R open chain.

joint screw axes $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$:

$$[\mathcal{S}_1] = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$[\mathcal{S}_2] = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$[\mathcal{S}_3] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

It will be more convenient to list the screw axes in the following tabular form:

i	ω_i	v_i
1	$(0,0,1)$	$(0,0,0)$
2	$(0,-1,0)$	$(0,0,-L_1)$
3	$(1,0,0)$	$(0,L_2,0)$

Example: 6R Spatial Open Chain

We now derive the forward kinematics of the 6R open chain of Figure 4.4. 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
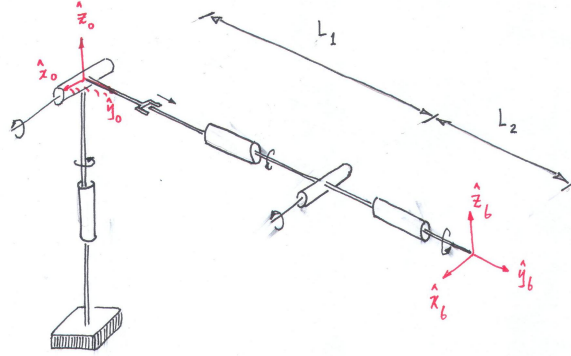
Figure 4.5: The RRPRRR spatial open chain.

assigned as shown. The end-effector frame M in the zero position is then

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 3L \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

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 $\mathcal{S}_i = (\omega_i, v_i)$, $i = 1, \ldots 6$ are as follows:

i	ω_i	v_i
1	$(0, 0, 1)$	$(0, 0, 0)$
2	$(0, 1, 0)$	$(0, 0, 0)$
3	$(-1, 0, 0)$	$(0, 0, 0)$
4	$(-1, 0, 0)$	$(0, 0, L)$
5	$(-1, 0, 0)$	$(0, 0, 2L)$
6	$(0, 1, 0)$	$(0, 0, 0)$

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 $\mathcal{S}_i = (\omega_i, v_i)$ are listed in the following table:

i	ω_i	v_i
1	(0, 0, 1)	(0, 0, 0)
2	(1, 0, 0)	(0, 0, 0)
3	(0, 0, 0)	(0, 1, 0)
4	(0, 1, 0)	(0, 0, 0)
5	(1, 0, 0)	(0, 0, $-L_1$)
6	(0, 1, 0)	(0, 0, 0)

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.

4.1.3 Second Formulation: Screw Axes Expressed in End-Effector Frame

The matrix identity $e^{M^{-1}PM} = M^{-1}e^P M$ (Proposition 3.7) can also be expressed as $Me^{M^{-1}PM} = e^P M$. Beginning with the rightmost term of the previously derived product of exponentials formula, if we repeatedly apply this identity, after n iterations we obtain

$$\begin{aligned} T &= e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_n]\theta_n} M \\ &= e^{[\mathcal{S}_1]\theta_1} \cdots M e^{M^{-1}[\mathcal{S}_n]M\theta_n} \\ &= e^{[\mathcal{S}_1]\theta_1} \cdots M e^{M^{-1}[\mathcal{S}_{n-1}]M\theta_{n-1}} e^{M^{-1}[\mathcal{S}_n]M\theta_n} \\ &= M e^{M^{-1}[\mathcal{S}_1]M\theta_1} \cdots e^{M^{-1}[\mathcal{S}_{n-1}]M\theta_{n-1}} e^{M^{-1}[\mathcal{S}_n]M\theta_n} \\ &= M e^{[\mathcal{B}_1]\theta_1} \cdots e^{[\mathcal{B}_{n-1}]\theta_{n-1}} e^{[\mathcal{B}_n]\theta_n}, \end{aligned} \tag{4.16}$$

where each $[\mathcal{B}_i] = M^{-1}[\mathcal{S}_i]M = [\text{Ad}_{M^{-1}}]\mathcal{S}_i$, $i = 1, \ldots, n$. Equation (4.16) is an alternative form of the product of exponentials formula, representing the joint axes as screw axes $\mathcal{B}_i$ 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.

Before concluding this chapter, it is worth considering 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 $\mathcal{S}_i$ is unaffected by the more distal transformations that came earlier, and any $\mathcal{B}_i$ is unaffected by the more proximal transformations that came earlier.

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

$$T = Me^{[\mathcal{B}_1]\theta_1}e^{[\mathcal{B}_2]\theta_2}\cdots e^{[\mathcal{B}_6]\theta_6}.$$

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:

i	ω_i	v_i
1	$(0,0,1)$	$(-3L,0,0)$
2	$(0,1,0)$	$(0,0,0)$
3	$(-1,0,0)$	$(0,0,-3L)$
4	$(-1,0,0)$	$(0,0,-2L)$
5	$(-1,0,0)$	$(0,0,-L)$
6	$(0,1,0)$	$(0,0,0)$

4.2   Summary

- Given an open chain with a reference frame attached to some point on its last link—this frame is denoted the **end-effector frame**—the forward kinematics is the mapping from the joint values to the position and orientation of the end-effector frame.

- 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 $\{0\},\ldots,\{n\}$, where $\{0\}$ is the fixed frame and $\{n\}$ is the end-effector frame, then the forward kinematics is expressed as

$$T_{0n} = T_{01}(\theta_1)\cdots T_{n-1,n}(\theta_n)$$

  where θ_i denotes the joint i variable.

- 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 a minimum of four parameters, the **Denavit-Hartenberg parameters**.

- The forward kinematics can also be expressed as the following **product of exponentials** (the **space form**),

$$T_{0n} = e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_n]\theta_n} M,$$

where $\mathcal{S}_i = (\omega_i, v_i)$ denotes the screw axis associated with joint i expressed in fixed frame coordinates, θ_i is the joint i variable, and $M \in SE(3)$ denotes the position and orientation of the end-effector frame when the robot is in its zero position. A choice of fixed frame and end-effector frame, together with a specification of the robot's zero position and direction of positive rotation or translation of each of the robot's joints, then completely specifies the product of exponentials formula.

- The product of exponentials formula can also be written in the equivalent **body form**,
$$T_{0n} = Me^{[\mathcal{B}_1]\theta_1} \cdots e^{[\mathcal{B}_n]\theta_n},$$

where $[\mathcal{B}]_i = [\mathrm{Ad}_{M^{-1}}]\mathcal{S}_i$, $i = 1, \ldots, n$. $\mathcal{B}_i = (\omega_i, v_i)$ is the screw axis corresponding to joint axis i, expressed in terms of the end-effector frame with the robot in its zero position.

Notes and References

The product of exponentials formula is first presented by Brockett in [4]. Computational aspects of the product of exponentials formula are also discussed in [27].

4.3   Exercises

1. (a) For each given $T \in SE(3)(3)$ below, find, if it exists, (α, a, d, ϕ) that satisfies

$$T = \mathrm{Rot}(\hat{x}, \alpha)\mathrm{Trans}(\hat{x}, a)\mathrm{Trans}(\hat{z}, d)\mathrm{Rot}(\hat{z}, \phi).$$

(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) $T = \begin{bmatrix} 0 & -1 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$

2. Let $T_1, T_2 \in SE(3)$, and suppose $T_1 = e^{[\mathcal{A}_1]}$, $T_2 = e^{[\mathcal{A}_2]}$ for some $\mathcal{A}_1 = (\omega_1, v_1)$, $\mathcal{A}_2 = (\omega_2, v_2)$. Under what conditions on $\mathcal{A}_1$ and $\mathcal{A}_2$ does $T_1 T_2 = T_2 T_1$?
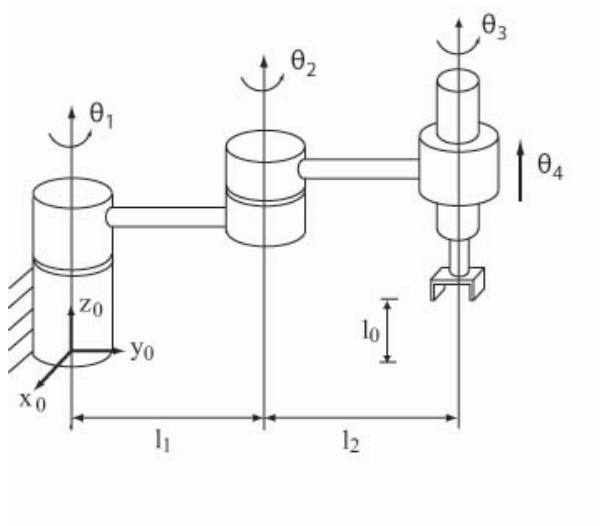
Figure 4.6: A *RRRP* robot for performing pick-and-place operations.

3. The *RRRP* spatial open chain of Figure 4.6 is shown in its zero position.
(a) Choose appropriate link reference frames, and derive the corresponding

Denvait-Hartenberg parameters (use the given fixed frame.)
(b) Find T_{04}.
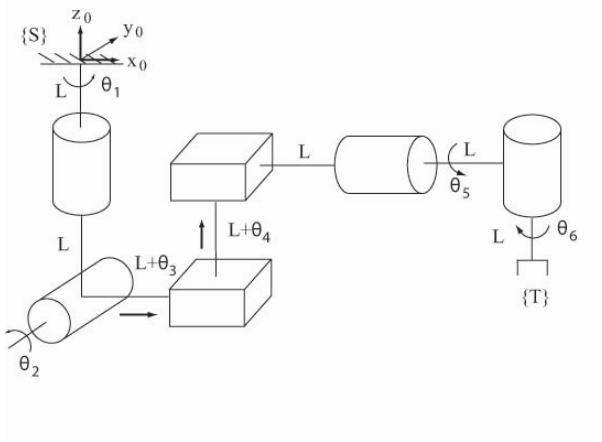
Figure 4.7: A $RRPPRR$ open chain robot.

4. The $RRPPRR$ spatial open chain of Figure 4.7 is shown in its zero position. Using the fixed reference frame given in the figure, choose appropriate link reference frames and derive the Denavit-Hartenberg parameters.

5. The robot with a screw joint in Figure 4.8 is shown in its zero position. Using the given fixed frame, choose appropriate link reference frames and derive the Denavit-Hartenberg parameters.

6. The spatial $RPRRR$ open chain of Figure 4.9 is shown in its zero position. Using the given fixed reference frame, choose appropriate link reference frames and derive the Denavit-Hartenberg parameters.

7. The $RRPRRR$ spatial open chain of Figure 4.10 is shown in its zero position (all joints lie on the same plane).
(a) Using the given fixed frame, choose appropriate link reference frames and derive the Denavit-Hartenberg parameters.
(b) Set $\theta_5 = \pi$ and all the other joint variables to zero, and find T_{60}.

8. The $PRRRRR$ spatial open chain of Figure 4.11 is shown in its home position. Using the given fixed and end-effector frames, and the direction of positive rotation or translation for each joint, derive its forward kinematics in the following product of exponentials form:

$$e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} \cdots e^{[\mathcal{S}_6]\theta_6} M$$
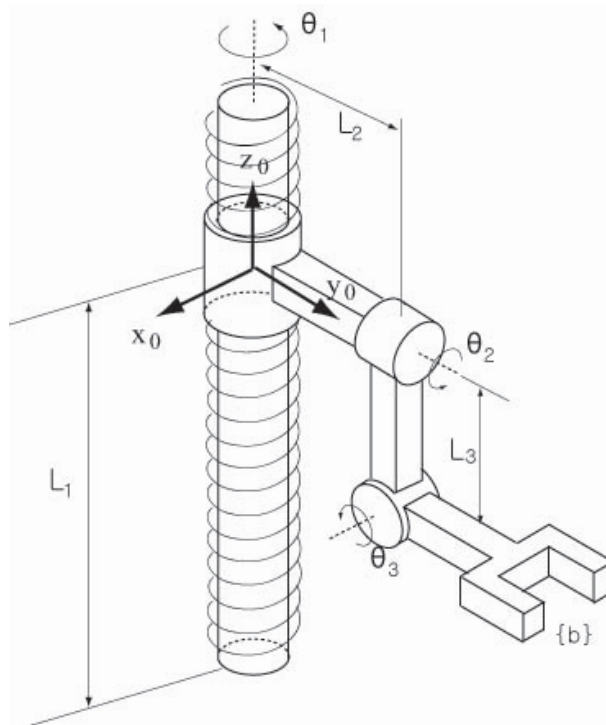
Figure 4.8: A robot with a screw joint.
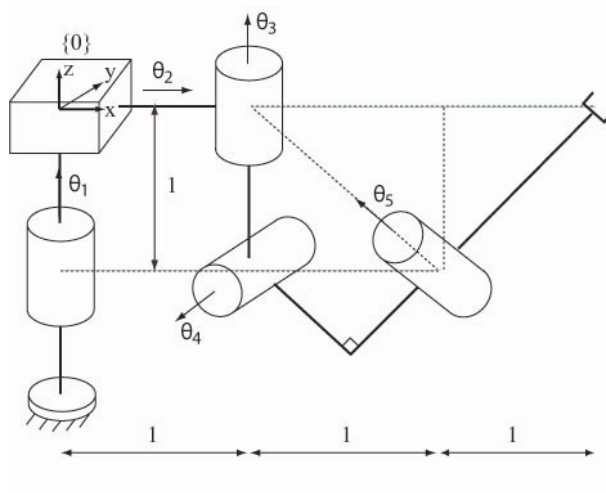
Figure 4.9: An $RPRRR$ spatial open chain.
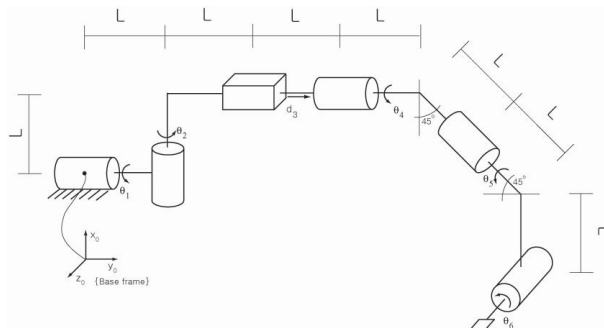
Figure 4.10: An $RRPRRR$ spatial open chain.
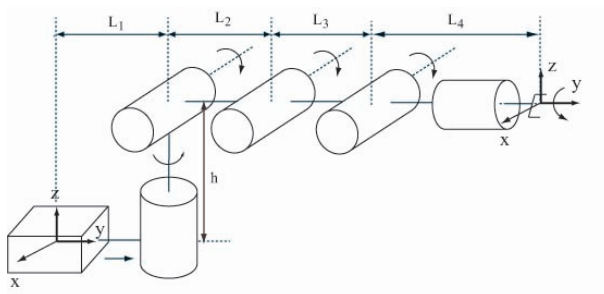
Figure 4.11: A $PRRRRR$ spatial open chain.

9. For the same open chain of Figure 4.11, express the forward kinematics in the following product of exponentials form:

$$Me^{[\mathcal{B}_\infty]\theta_1}e^{[\mathcal{B}_2]\theta_2}\cdots e^{[\mathcal{B}_6]\theta_6}.$$

10. The spatial $RRPPRR$ open chain of Figure 4.12 is shown in its zero position.
(a) Derive its forward kinematics in the form $e^{[\mathcal{S}_1]\theta_1}e^{[\mathcal{S}_2]\theta_2}\cdots e^{[\mathcal{S}_6]\theta_6}M$.
(b) Derive its forward kinematics in the form $Me^{[\mathcal{B}_\infty]\theta_1}e^{[\mathcal{B}_2]\theta_2}\cdots e^{[\mathcal{B}_6]\theta_6}$.

11. The spatial $RRRPRR$ open chain of Figure 4.13 is shown in its zero position.
(a) Choose appropriate link frames and derive the corresponding Denavit-Hartenberg parameters.
(b) Express its forward kinematics in the form $e^{[\mathcal{S}_1]\theta_1}e^{[\mathcal{S}_2]\theta_2}\cdots e^{[\mathcal{S}_6]\theta_6}M$.
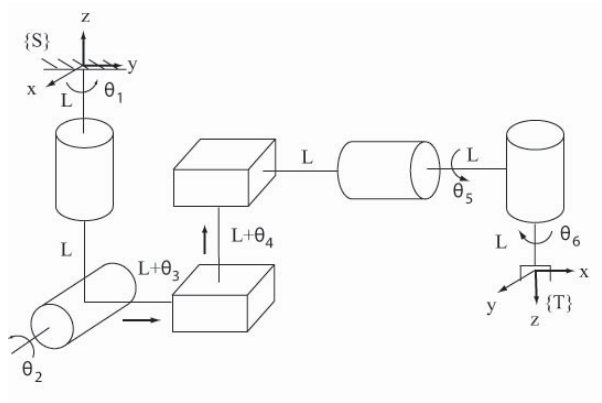
Figure 4.12: A spatial $RRPPRR$ open chain with prescribed fixed and end-effector frames.
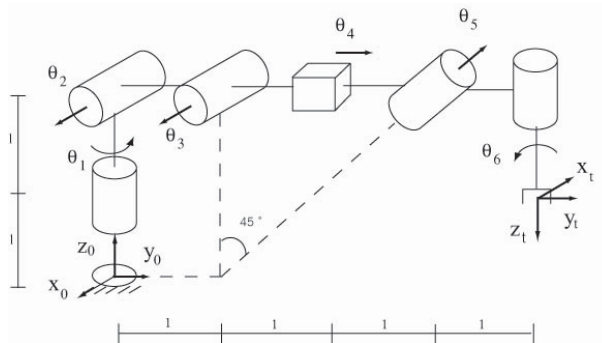
Figure 4.13: A spatial $RRPRRR$ open chain with prescribed fixed and end-effector frames.

12. The spatial $RRRRPR$ open chain of Figure 4.14 is shown in its zero position, with fixed and end-effector frames chosen as shown.
(a) Choose appropriate link frames and derive the corresponding Denavit-Hartenberg parameters.
(b) Express its forward kinematics in the form $e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} \cdots e^{[\mathcal{S}_6]\theta_6} M$.
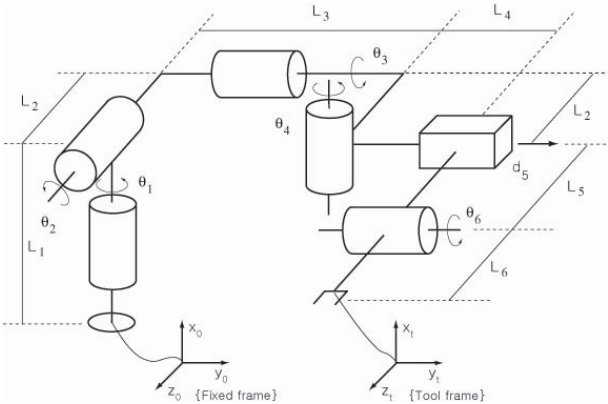
Figure 4.14: A spatial $RRRRPR$ open chain.

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 frame's spatial velocity from a given set of joint positions and velocities.

Before we get to a representation of the end-effector spatial velocity as $\mathcal{V} \in \mathbb{R}^6$ starting in Section 5.1, let us consider the case where the end-effector configuration is represented by a minimal set of coordinates $x \in \mathbb{R}^n$ and the velocity is given by $\dot{x} \in \mathbb{R}^n$. In this case, the forward kinematics can be written as

$$x(t) = f(\theta(t)),$$

where $\theta \in \mathbb{R}^m$ is a set of joint variables. By the chain rule, the time derivative at time t is

$$\dot{x} = \frac{\partial f}{\partial \theta}(\theta)\dot{\theta}$$
$$= J(\theta)\dot{\theta},$$

where $J(\theta) \in \mathbb{R}^{n \times m}$ 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 θ.

To provide a concrete example, consider a 2R planar open chain (right side of Figure 5.1) with forward kinematics given by

$$\begin{aligned}
x_1 &= L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) \\
x_2 &= L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2).
\end{aligned}$$

Differentiating both sides with respect to time t yields

$$\begin{aligned}
\dot{x}_1 &= -L_1\dot{\theta}_1 \sin\theta_1 - L_2(\dot{\theta}_1 + \dot{\theta}_2) \sin(\theta_1 + \theta_2) \\
\dot{x}_2 &= L_1\dot{\theta}_1 \cos\theta_1 + L_2(\dot{\theta}_1 + \dot{\theta}_2) \cos(\theta_1 + \theta_2),
\end{aligned}$$
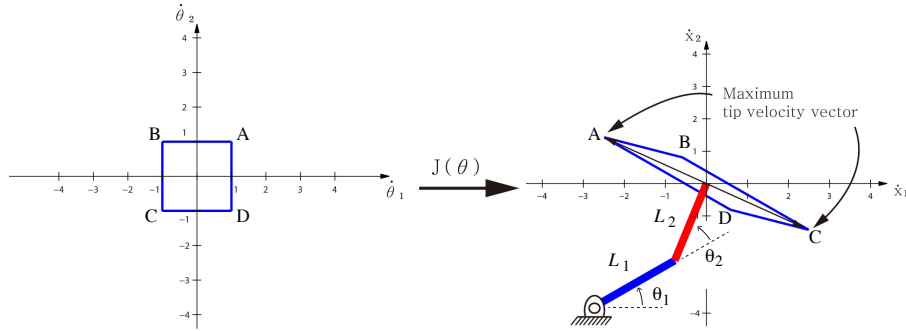
Figure 5.1: Mapping the set of possible joint velocities, represented as a square in the $\dot{\theta}_1$-$\dot{\theta}_2$ space, through the Jacobian to find the parallelogram of possible end-effector velocities. The extreme points A, B, C, and D in the joint velocity space map to the extreme points A, B, C, and D in the end-effector velocity space.

which can be rearranged into a linear equation of the form $\dot{x} = J(\theta)\dot{\theta}$:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -L_1\sin\theta_1 - L_2\sin(\theta_1 + \theta_2) & -L_2\sin(\theta_1 + \theta_2) \\ L_1\cos\theta_1 + L_2\cos(\theta_1 + \theta_2) & L_2\cos(\theta_1 + \theta_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}. \quad (5.1)$$

Writing the two columns of $J(\theta)$ as $J_1(\theta)$ and $J_2(\theta)$, and the tip velocity $\dot{x}$ as v_{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_{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 θ_1 and θ_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 θ_2 is $0°$ or $180°$, then regardless of the value of θ_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.

The Jacobian can be used to map bounds on the rotational speed of the joints to bounds on v_{tip}, as illustrated in Figure 5.1. Instead of mapping a polygon of joint velocities through the Jacobian, we could instead map a unit circle of joint velocities in the θ_1-θ_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**.[1] Figure 5.2 shows examples of this mapping for two

---

[1] A two-dimensional ellipsoid, as in our example, is commonly referred to as an ellipse.
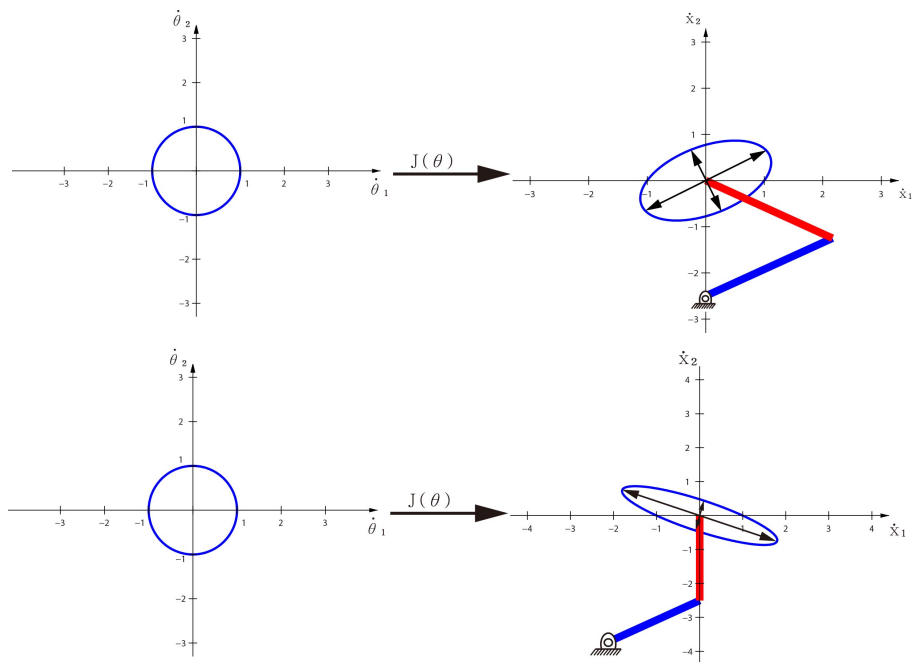
Figure 5.2: Manipulability ellipsoids for two different postures of the 2R planar open chain.

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 ℓ_{max} and ℓ_{min}. The closer the ellipsoid is to a circle—meaning the ratio ℓ_{max}/ℓ_{min} is close to 1—the more easily the tip can move in arbitrary directions, and thus the more removed it is from a singularity. The ratio ℓ_{max}/ℓ_{min} is known as the **condition number**, and it offers a quantitative measure of how close a robot configuration is to 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 tip force and maintain static equilibrium (that is, to keep all links of the robot stationary)?

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 f_{tip} and the joint torque vector by τ,
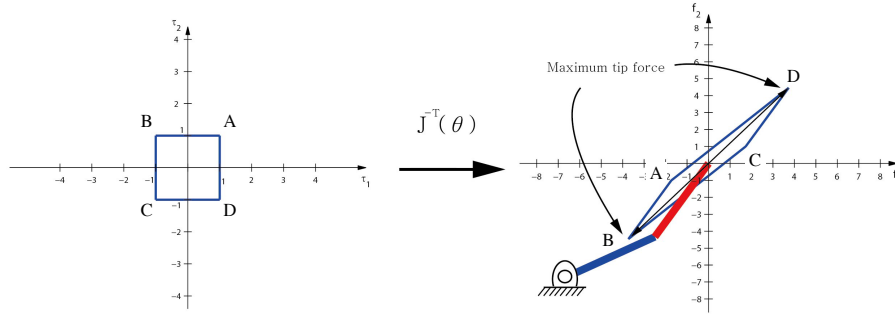
Figure 5.3: Mapping joint torque bounds to tip force bounds.

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

$$\tau = J^T(\theta)f_{\text{tip}}. \tag{5.2}$$

The joint torque τ needed to create the tip force f_{tip} is calculated from the equation above.

For our two-link planar chain example, $J(\theta)$ is a square matrix dependent on θ. If the configuration θ is not a singularity, then both $J(\theta)$ and its tranpose are invertible, and Equation (5.2) can be written

$$f_{\text{tip}} = ((J(\theta))^T)^{-1}\tau = J^{-T}(\theta)\tau. \tag{5.3}$$

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 θ of the robot at equilibrium, and given a set of joint torque limits such as

$$-1\,\text{Nm} \le \tau_1 \le 1\,\text{Nm}$$
$$-1\,\text{Nm} \le \tau_2 \le 1\,\text{Nm},$$

Equation (5.3) can be used to generate the set of all possible tip forces as shown in Figure 5.3.

---

[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.
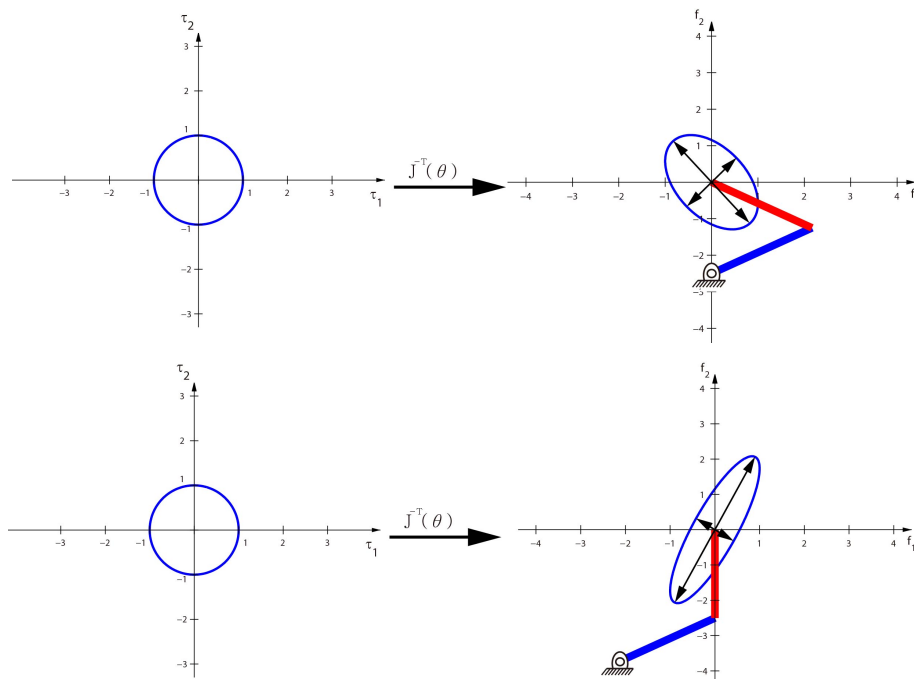
Figure 5.4: Force ellipsoids for two different postures of the 2R planar open chain.

Similar to the manipulability ellipsoid, a **force ellipsoid** can be drawn by mapping a unit circle "iso-effort" contour in the τ_1-τ_2 plane to an ellipsoid in the f_1-f_2 tip force plane via the Jacobian transpose inverse $J^{-T}(\theta)$ (see Figure 5.4). 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 5.5). This is because the power must be equivalent whether the forces and velocities are represented in joint space or tip space; the power for any joint torque/velocity pair is bounded by one; and therefore the power for any tip force/velocity pair must also be bounded by one.

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 mass 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. The joint structure itself must bear the load, not muscles to generate torques. Unlike the manipulability ellipsoid, which loses dimension at a singularity (from two to one in our example) and therefore its area
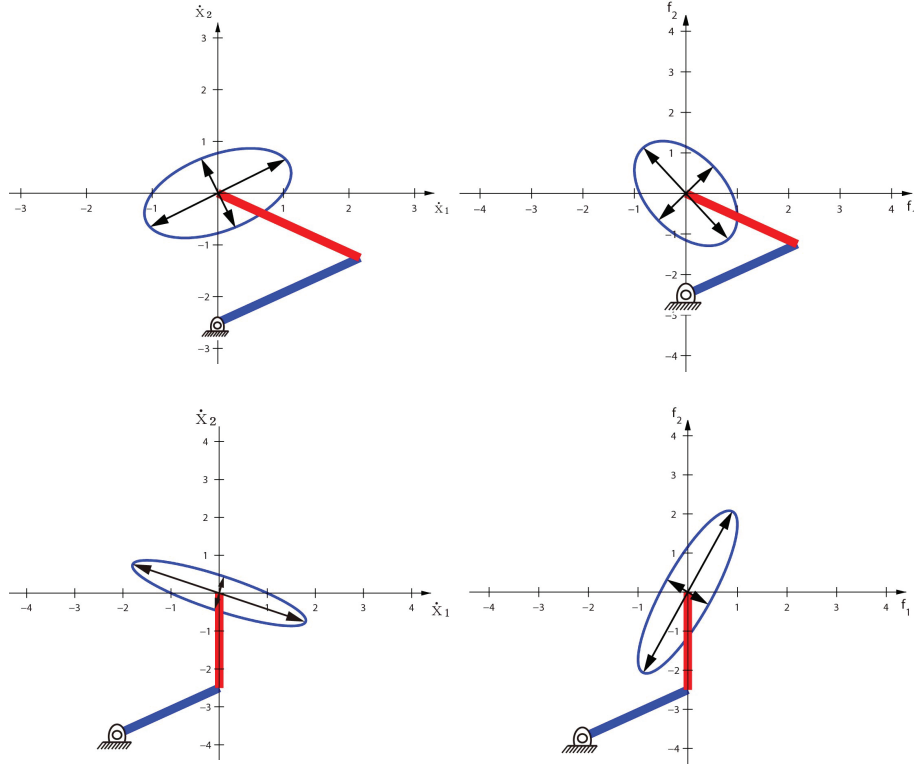
Figure 5.5: Left column: Manipulability ellipsoids at two different arm configurations. Right column: Force ellipsoids at the the same two arm configurations.

drops to zero, the manipulability 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 spatial velocity $\mathcal{V}$ 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 the identification of 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 θ, the tip velocity vector v_{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_{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_{tip} can be taken to be the six-dimensional spatial velocity of the end-effector frame, while for pure orienting devices like a wrist, v_{tip} is usually taken to be the angular velocity of the end-effector frame. Other choices for v_{tip} lead to different formulations for the Jacobian. We begin with the general case where v_{tip} is taken to be the six-dimensional end-effector spatial velocity 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 velocity $\mathcal{V}_s$. We first recall a few basic properties from linear algebra and linear differential equations: (i) if $A, B \in \mathbb{R}^{n \times n}$ are both invertible, then $(AB)^{-1} = B^{-1}A^{-1}$; (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:

$$T(\theta_1, \ldots, \theta_n) = e^{[\mathcal{S}_1]\theta_1}e^{[\mathcal{S}_2]\theta_2}\cdots e^{[\mathcal{S}_n]\theta_n}M. \tag{5.4}$$

The spatial velocity of the end-effector frame with respect to the fixed frame, $\mathcal{V}_s$, is given by $[\mathcal{V}_s] = \dot{T}T^{-1}$, where

$$
\begin{aligned}
\dot{T} &= \left(\frac{d}{dt}e^{[\mathcal{S}_1]\theta_1}\right)\cdots e^{[\mathcal{S}_n]\theta_n}M + e^{[\mathcal{S}_1]\theta_1}\left(\frac{d}{dt}e^{[\mathcal{S}_2]\theta_2}\right)\cdots e^{[\mathcal{S}_n]\theta_n}M + \ldots \\
&= [\mathcal{S}_1]\dot{\theta}_1 e^{[\mathcal{S}_1]\theta_1}\cdots e^{[\mathcal{S}_n]\theta_n}M + e^{[\mathcal{S}_1]\theta_1}[\mathcal{S}_2]\dot{\theta}_2 e^{[\mathcal{S}_2]\theta_2}\cdots e^{[\mathcal{S}_n]\theta_n}M + \ldots
\end{aligned}
$$

Also,

$$T^{-1} = M^{-1}e^{-[\mathcal{S}_n]\theta_n}\cdots e^{-[\mathcal{S}_1]\theta_1}.$$

Calculating $\dot{T}T^{-1}$,

$$[\mathcal{V}_s] = [\mathcal{S}_1]\dot{\theta}_1 + e^{[\mathcal{S}_1]\theta_1}[\mathcal{S}_2]e^{-[\mathcal{S}_1]\theta_1}\dot{\theta}_2 + e^{[\mathcal{S}_1]\theta_1}e^{[\mathcal{S}_2]\theta_2}[\mathcal{S}_3]e^{-[\mathcal{S}_2]\theta_2}e^{-[\mathcal{S}_1]\theta_1}\dot{\theta}_3 + \ldots.$$

The above can also be expressed in vector form by means of the adjoint mapping:

$$\mathcal{V}_s = \underbrace{\mathcal{S}_1}_{\mathcal{V}_{s1}}\dot{\theta}_1 + \underbrace{\mathrm{Ad}_{e^{[\mathcal{S}_1]\theta_1}}(\mathcal{S}_2)}_{\mathcal{V}_{s2}}\dot{\theta}_2 + \underbrace{\mathrm{Ad}_{e^{[\mathcal{S}_1]\theta_1}e^{[\mathcal{S}_2]\theta_2}}(\mathcal{S}_3)}_{\mathcal{V}_{s3}}\dot{\theta}_3 + \ldots \tag{5.5}$$

Observe that $\mathcal{V}_s$ is a sum of n spatial velocities of the form

$$\mathcal{V}_s = \mathcal{V}_{s1}(\theta)\dot{\theta}_1 + \ldots + \mathcal{V}_{sn}(\theta)\dot{\theta}_n, \tag{5.6}$$

where each $\mathcal{V}_{si}(\theta) = (\omega_{si}(\theta), v_{si}(\theta))$ depends explictly on the joint values $\theta \in \mathbb{R}^n$ for $i = 2, \ldots, n$. In matrix form,

$$
\begin{aligned}
\mathcal{V}_s &= \begin{bmatrix} \mathcal{V}_{s1}(\theta) & \mathcal{V}_{s2}(\theta) & \cdots & \mathcal{V}_{sn}(\theta) \end{bmatrix}\begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} \\
&= J_s(\theta)\dot{\theta}.
\end{aligned} \tag{5.7}
$$

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:

$$T = e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_n]\theta_n} M. \tag{5.8}$$

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 velocity $\mathcal{V}_s$ via

$$\mathcal{V}_s = J_s(\theta)\dot{\theta}. \tag{5.9}$$

The ith column of $J_s(\theta)$ is

$$\mathcal{V}_{si}(\theta) = \mathrm{Ad}_{e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_{i-1}]\theta_{i-1}}}(\mathcal{S}_i), \tag{5.10}$$

for $i = 2, \ldots, n$, with the first column $\mathcal{V}_{s1}(\theta) = \mathcal{S}_1$. $\square$

To understand the physical meaning behind the columns of $J_s(\theta)$, observe that the ith column is of the form $\mathrm{Ad}_{T_{i-1}}(\mathcal{S}_i)$, where $T_{i-1} = e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_{i-1}]\theta_{i-1}}$; recall that $\mathcal{S}_i$ is the screw axis describing the ith joint axis in terms of the fixed frame with the robot in its zero position. $\mathrm{Ad}_{T_{i-1}}(\mathcal{S}_i)$ is therefore the screw axis describing the ith 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 ith column $\mathcal{V}_{si}(\theta)$ 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 of $J_s(\theta)$ is similar to that for deriving the $\mathcal{S}_i$ in the product of exponentials formula $e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_n]\theta_n} M$: each column $\mathcal{V}_{si}(\theta)$ is the screw vector describing joint axis i, expressed in fixed frame coordinates, but for arbitrary θ 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 5.6. Denote the ith column of $J_s(\theta)$ by $\mathcal{V}_i = (\omega_i, v_i)$. The $[\mathrm{Ad}_{T_{i-1}}]$ matrices are implicit in our calculations of the screw axes of the displaced joint axes.

- Observe that ω_1 is constant and in the $\hat{z}_s$-direction: $\omega_1 = (0, 0, 1)$. Picking q_1 to be the origin, $v_1 = (0, 0, 0)$.

- ω_2 is also constant in the $\hat{z}_s$-direction, so $\omega_2 = (0, 0, 1)$. 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 $v_2 = -\omega_2 \times q_2 = (L_1 s_1, -L_1 c_1, 0)$.

- The direction of ω_3 is always fixed in the $\hat{z}_s$-direction regardless of the values of θ_1 and θ_2, so $\omega_3 = (0, 0, 1)$. 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 $v_3 = (L_1 s_1 + L_2 s_{12}, -L_1 c_1 - L_2 c_{12}, 0)$.
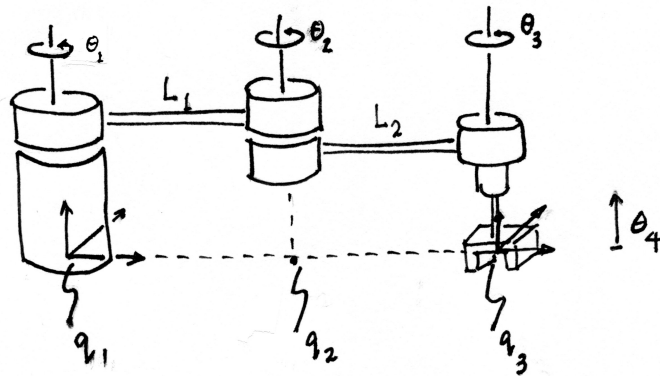
Figure 5.6: Space Jacobian for a spatial $RRRP$ chain.
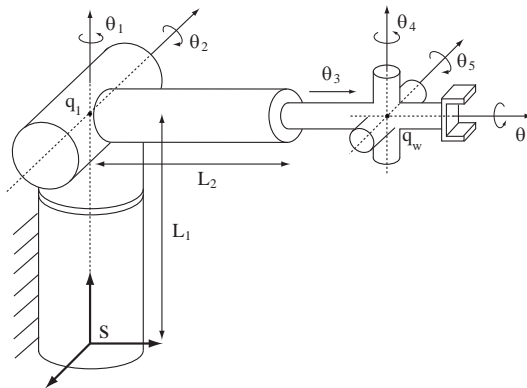
Figure 5.7: Space Jacobian for the spatial $RRPRRR$ chain.

- Since the final joint is prismatic, $\omega_4 = (0, 0, 0)$, and the joint axis direction is given by $v_4 = (0, 0, 1)$.

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 5.7. The base frame is chosen as shown in the figure.

- The first joint axis is in the direction $\omega_1 = (0, 0, 1)$. Picking $q_1 = (0, 0, L_1)$, we get $v_1 = -\omega_1 \times q_1 = (0, 0, 0)$.

- The second joint axis is in the direction $\omega_2 = (-c_1, -s_1, 0)$. Picking $q_2 = (0, 0, L_1)$, we get $v_2 = -\omega_2 \times q_2 = (L_1 s_1, -L_1 c_1, 0)$.

- The third joint is prismatic, so $\omega_3 = (0, 0, 0)$. The direction of the prismatic joint axis is given by

$$
v_3 = \mathrm{Rot}(\hat{z}, \theta_1)\mathrm{Rot}(\hat{x}, -\theta_2) \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -s_1 c_2 \\ c_1 c_2 \\ -s_2 \end{bmatrix}.
$$

- Now consider the wrist portion of the chain. The wrist center is located at the point

$$
q_w = \begin{bmatrix} 0 \\ 0 \\ L_1 \end{bmatrix} + \mathrm{Rot}(\hat{z}, \theta_1)\mathrm{Rot}(\hat{x}, -\theta_2) \begin{bmatrix} 0 \\ L_1 + \theta_3 \\ 0 \end{bmatrix} = \begin{bmatrix} -(L_2 + \theta_3)s_1 c_2 \\ (L_2 + \theta_3)c_1 c_2 \\ L_1 - (L_2 + \theta_3)s_2 \end{bmatrix}.
$$

Observe that the directions of the wrist axes depend on θ_1, θ_2, and the preceding wrist axes. These are

$$
\begin{aligned}
\omega_4 &= \mathrm{Rot}(\hat{z}, \theta_1)\mathrm{Rot}(\hat{x}, -\theta_2) \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -s_1 s_2 \\ c_1 s_2 \\ c_2 \end{bmatrix} \\
\omega_5 &= \mathrm{Rot}(\hat{z}, \theta_1)\mathrm{Rot}(\hat{x}, -\theta_2)\mathrm{Rot}(\hat{z}, \theta_4) \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -c_1 c_4 + s_1 c_2 s_4 \\ -s_1 c_4 - c_1 c_2 s_4 \\ s_2 s_4 \end{bmatrix} \\
\omega_6 &= \mathrm{Rot}(\hat{z}, \theta_1)\mathrm{Rot}(\hat{x}, -\theta_2)\mathrm{Rot}(\hat{z}, \theta_4)\mathrm{Rot}(\hat{x}, -\theta_5) \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\
&= \begin{bmatrix} -c_5(s_1 c_2 c_4 + c_1 s_4) + s_1 s_2 s_5 \\ c_5(c_1 c_2 c_4 - s_1 s_4) - c_1 s_2 s_5 \\ -s_2 c_4 c_5 - c_2 s_5 \end{bmatrix}.
\end{aligned}
$$

The space Jacobian can now be computed and written in matrix form as follows:

$$
J_s(\theta) = \begin{bmatrix} \omega_1 & \omega_2 & 0 & \omega_4 & \omega_5 & \omega_6 \\ 0 & -\omega_2 \times q_2 & v_3 & -\omega_4 \times q_w & -\omega_5 \times q_w & -\omega_6 \times q_w \end{bmatrix}.
$$

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 spatial velocity 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 spatial velocity 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:

$$T(\theta) = Me^{[\mathcal{B}_1]\theta_1}e^{[\mathcal{B}_2]\theta_2}\cdots e^{[\mathcal{B}_n]\theta_n}. \tag{5.11}$$

Computing $\dot{T}$,

$$
\begin{aligned}
\dot{T} &= Me^{[\mathcal{B}_1]\theta_1}\cdots e^{[\mathcal{B}_{n-1}]\theta_{n-1}}(\frac{d}{dt}e^{[\mathcal{B}_n]\theta_n}) + Me^{[\mathcal{B}_1]\theta_1}\cdots(\frac{d}{dt}e^{[\mathcal{B}_{n-1}]\theta_{n-1}})e^{[\mathcal{B}_n]\theta_n} + \ldots \\
&= Me^{[\mathcal{B}_1]\theta_1}\cdots e^{[\mathcal{B}_n]\theta_n}[\mathcal{B}_n]\dot{\theta}_n + Me^{[\mathcal{B}_1]\theta_1}\cdots e^{[\mathcal{B}_{n-1}]\theta_{n-1}}[\mathcal{B}_{n-1}]e^{[\mathcal{B}_n]\theta_n}\dot{\theta}_{n-1} + \ldots \\
&\quad + Me^{[\mathcal{B}_1]\theta_1}[\mathcal{B}_1]e^{[\mathcal{B}_2]\theta_2}\cdots e^{[\mathcal{B}_n]\theta_n}\dot{\theta}_1.
\end{aligned}
$$

Also,

$$T^{-1} = e^{-[\mathcal{B}_n]\theta_n}\cdots e^{-[\mathcal{B}_1]\theta_1}M^{-1}.$$

Evaluating $T^{-1}\dot{T}$,

$$
\begin{aligned}
[\mathcal{V}_b] &= [\mathcal{B}_n]\dot{\theta}_n + e^{-[\mathcal{B}_n]\theta_n}[\mathcal{B}_{n-1}]e^{[\mathcal{B}_n]\theta_n}\dot{\theta}_{n-1} + \ldots \\
&\quad + e^{-[\mathcal{B}_n]\theta_n}\cdots e^{-[\mathcal{B}_2]\theta_2}[\mathcal{B}_1]e^{[\mathcal{B}_2]\theta_2}\cdots e^{[\mathcal{B}_n]\theta_n}\dot{\theta}_1,
\end{aligned}
$$

or in vector form,

$$\mathcal{V}_b = \underbrace{\mathcal{B}_n}_{\mathcal{V}_{bn}}\dot{\theta}_n + \underbrace{\mathrm{Ad}_{e^{-[\mathcal{B}_n]\theta_n}}(\mathcal{B}_{n-1})}_{\mathcal{V}_{b,n-1}}\dot{\theta}_{n-1} + \ldots + \underbrace{\mathrm{Ad}_{e^{-[\mathcal{B}_n]\theta_n}\cdots e^{-[\mathcal{B}_2]\theta_2}}(\mathcal{B}_1)}_{\mathcal{V}_{b1}}\dot{\theta}_1. \tag{5.12}$$

$\mathcal{V}_b$ can therefore be expressed as a sum of n spatial velocities, i.e.,

$$\mathcal{V}_b = \mathcal{V}_{b1}(\theta)\dot{\theta}_1 + \ldots + \mathcal{V}_{bn}(\theta)\dot{\theta}_n, \tag{5.13}$$

where each $\mathcal{V}_{bi}(\theta) = (\omega_{bi}(\theta), v_{bi}(\theta))$ depends explictly on the joint values θ for $i = 1, \ldots, n-1$. In matrix form,

$$
\begin{aligned}
\mathcal{V}_b &= \begin{bmatrix} \mathcal{V}_{b1}(\theta) & \mathcal{V}_{b2}(\theta) & \cdots & \mathcal{V}_{bn}(\theta) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} \\
&= J_b(\theta)\dot{\theta}.
\end{aligned}
\tag{5.14}
$$

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:

$$T = Me^{[\mathcal{B}_1]\theta_1}\cdots e^{[\mathcal{B}_n]\theta_n}. \tag{5.15}$$

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 spatial velocity $\mathcal{V}_b = (\omega_b, v_b)$ via

$$\mathcal{V}_b = J_b(\theta)\dot{\theta}. \tag{5.16}$$

The ith column of $J_b(\theta)$ is

$$\mathcal{V}_{bi}(\theta) = \text{Ad}_{e^{-[\mathcal{B}_n]\theta_n}\cdots e^{-[\mathcal{B}_{i+1}]\theta_{i+1}}}(\mathcal{B}_i), \tag{5.17}$$

for $i = n-1, \ldots, 1$, with $\mathcal{V}_{bn}(\theta) = \mathcal{B}_n$. $\square$

Analogous to the columns of the space Jacobian, a similar physical interpretation can be given to the columns of $J_b(\theta)$: each column $\mathcal{V}_{bi}(\theta) = (\omega_{bi}(\theta), v_{bi}(\theta))$ 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 $Me^{[\mathcal{B}_1]\theta_1}\cdots e^{[\mathcal{B}_n]\theta_n}$, the only difference being that each of the joint screws are derived for arbitrary θ 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 spatial velocity of the end-effector frame can be written in terms of the fixed and end-effector frame coordinates as

$$
\begin{aligned}
[\mathcal{V}_s] &= \dot{T}_{sb}T_{sb}^{-1} \\
[\mathcal{V}_b] &= T_{sb}^{-1}\dot{T}_{sb},
\end{aligned}
$$

with $\mathcal{V}_s$ and $\mathcal{V}_b$ related by $\mathcal{V}_s = \text{Ad}_{T_{sb}}(\mathcal{V}_b)$ and $\mathcal{V}_b = \text{Ad}_{T_{bs}}(\mathcal{V}_s)$. $\mathcal{V}_s$ and $\mathcal{V}_b$ are also related to their respective Jacobians via

$$
\begin{aligned}
\mathcal{V}_s &= J_s(\theta)\dot{\theta} & (5.18) \\
\mathcal{V}_b &= J_b(\theta)\dot{\theta}. & (5.19)
\end{aligned}
$$

Equation (5.18) can therefore be written

$$\text{Ad}_{T_{sb}}(\mathcal{V}_b) = J_s(\theta)\dot{\theta}. \tag{5.20}$$

Applying $[\text{Ad}_{T_{bs}}]$ 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

$$\text{Ad}_{T_{bs}}(\text{Ad}_{T_{sb}}(\mathcal{V}_b)) = \text{Ad}_{T_{bs}T_{sb}}(\mathcal{V}_b) = \mathcal{V}_b = \text{Ad}_{T_{bs}}(J_s(q)\dot{\theta}).$$

Since we also have $\mathcal{V}_b = J_b(\theta)\dot{\theta}$ for all $\dot{\theta}$, it follows that $J_s(\theta)$ and $J_b(\theta)$ are related by

$$J_b(\theta) = \text{Ad}_{T_{bs}}(J_s(\theta)) = [\text{Ad}_{T_{bs}}]J_s(\theta). \tag{5.21}$$

The space Jacobian can in turn be obtained from the body Jacobian via

$$J_s(\theta) = \text{Ad}_{T_{sb}}(J_b(\theta)) = [\text{Ad}_{T_{sb}}]J_b(\theta). \tag{5.22}$$

The fact that the space and body Jacobians, and space and body velocities, are similarly related by the adjoint map should not be surprising, since each column of the space and body Jacobian corresponds to a spatial velocity.

One of the important implications of Equation (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 spatial velocity 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 spatial velocity $\mathcal{V}$ 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

$$\mathcal{V}_b = J_b(\theta)\dot{\theta},$$

where $J_b(\theta) \in \mathbb{R}^{6 \times n}$. The angular and linear velocity components of $\mathcal{V}_b = (\omega_b, v_b)$ can be written explicitly as

$$\mathcal{V}_b = \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right] = J_b(\theta)\dot{\theta} = \left[\begin{array}{c} J_\omega(\theta) \\ J_v(\theta) \end{array} \right] \dot{\theta},$$

where J_ω 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,

$$\dot{x} = R_{sb}v_b = R_{sb}J_v(\theta)\dot{\theta},$$

where $R_{sb} = e^{[r]} = e^{[\hat{\omega}]\theta}$.

The time-derivative $\dot{r}$ is related to the body angular velocity ω_b by

$$\begin{aligned} \omega_b &= A(r)\dot{r} \\ A(r) &= I - \frac{1 - \cos\|r\|}{\|r\|^2}[r] + \frac{\|r\| - \sin\|r\|}{\|r\|^3}[r]^2. \end{aligned}$$

(The derivation of this formula is explored in Exercise 16.) Provided the matrix $A(r)$ is invertible, $\dot{r}$ can be obtained from ω_b as

$$\dot{r} = A^{-1}(r)\omega_b = A^{-1}(r)J_\omega(\theta)\dot{\theta}.$$

Putting these together,

$$\dot{q} = \begin{bmatrix} \dot{r} \\ \dot{x} \end{bmatrix} = \begin{bmatrix} A^{-1}(r) & 0 \\ 0 & R_{sb} \end{bmatrix} \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}, \tag{5.23}$$

i.e., the analytic Jacobian J_a is related to the body Jacobian J_b by

$$J_a(\theta) = \begin{bmatrix} A^{-1}(r) & 0 \\ 0 & R_{sb} \end{bmatrix} \begin{bmatrix} J_\omega(\theta) \\ J_v(\theta) \end{bmatrix} = \begin{bmatrix} A^{-1}(r) & 0 \\ 0 & R_{sb} \end{bmatrix} J_b(\theta). \tag{5.24}$$

5.1.5 Inverse Velocity Kinematics

The sections above answer the question "What spatial velocity results from a given set of joint velocities?" The answer, written independently of the frame in which spatial velocities are represented, is given by

$$\mathcal{V} = J(\theta)\dot{\theta}.$$

Often we are interested in the inverse question: given a desired spatial velocity $\mathcal{V}$, what joint velocities $\dot{\theta}$ are needed? This is a question of inverse velocity kinematics, which is discussed in more detail in Chapter 6. In brief, if $J(\theta)$ is square (the number of joints n is equal to six, the number of elements of the spatial velocity) and full rank, then $\dot{\theta} = J^{-1}(\theta)\mathcal{V}$. If $n \neq 6$, however, then $J(\theta)$ is not invertible. In the case $n < 6$, arbitrary spatial velocities $\mathcal{V}$ cannot be achieved—the robot does not have enough joints. If $n > 6$, then we call the robot **redundant**. In this case, a desired spatial velocity $\mathcal{V}$ 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,

power at the joints = (power to move the robot) + (power at end-effector),

and considering the robot to be at static equilibrium (no power is used to move the robot, because external forces at the end-effector immobilize the robot), we

can equate the power at the joints to the power at the end-effector[3],

$$\tau^T \dot{\theta} = \mathcal{F}_b^T \mathcal{V}_b,$$

where τ is the set of joint torques. Using the identity $\mathcal{V}_b = J_b(\theta)\dot{\theta}$, we get

$$\tau = J_b^T(\theta)\mathcal{F}_b$$

relating the joint torques to the end-effector spatial force written in the end-effector frame. Similarly,

$$\tau = J_s^T(\theta)\mathcal{F}_s$$

in the fixed space frame. Leaving out the choice of the frame, we can simply write

$$\tau = J^T(\theta)\mathcal{F}. \tag{5.25}$$

Thus if we are given a desired end-effector spatial force $\mathcal{F}$ and the joint angles θ, we can calculate the joint torques τ needed to generate $\mathcal{F}$. This is important in force control of a robot, for example.

One could also ask the opposite question, namely, what is the spatial force at the tip generated by a given joint torque? If J^T is a square invertible matrix, then clearly $\mathcal{F} = J^{-T}(\theta)\tau$. However, if the number of joints n is not equal to six, then J^T is not invertible. In particular, if the robot is redundant ($n > 6$), then even if the end-effector is held stationary, the joint torques may cause internal motions of the links, so that the static equilibrium condition is no longer satisfied.

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 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

$$
\begin{aligned}
\mathcal{V}_b &= \begin{bmatrix} \mathcal{V}_{b1}(\theta) & \mathcal{V}_{b2}(\theta) & \cdots & \mathcal{V}_{bn}(\theta) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} \\
&= \mathcal{V}_{b1}(\theta)\dot{\theta}_1 + \ldots + \mathcal{V}_{bn}(\theta)\dot{\theta}_n.
\end{aligned}
$$

Thus, the set of all possible instantaneous spatial velocities of the tip frame is given by a linear combination 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

---

[3]We are considering the limiting case as $\dot{\theta}$ goes to zero, consistent with our assumption that the robot is at equilibrium.
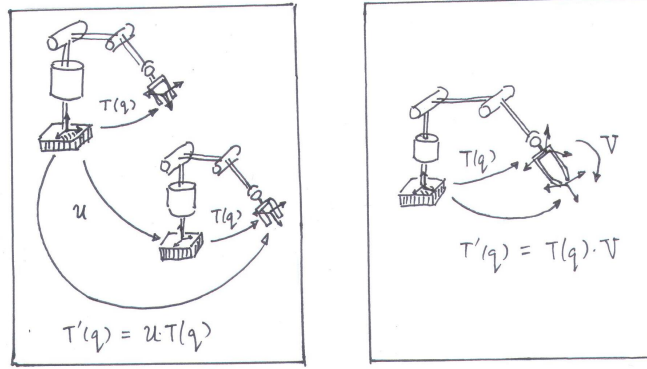
Figure 5.8: Kinematic singularities are invariant with respect to choice of fixed and end-effector frames. (a) Choosing a different fixed frame, which is equivalent to relocating the base of the robot arm; (b) Choosing a different end-effector frame.

of θ at which the rank of $J_b(\theta)$ drops below the maximum possible value; at such postures the tip frame loses to the ability to generate instantaneous spatial velocities in in one or more dimensions.

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 $J_b(\theta)$: $J_s(\theta) = \mathrm{Ad}_{T_{sb}}(J_b(\theta)) = [\mathrm{Ad}_{T_{sb}}]J_b(\theta)$, or more explicitly,

$$J_s(\theta) = \begin{bmatrix} R_{sb} & 0 \\ [p_{sb}]\,R_{sb} & R_{sb} \end{bmatrix} J_b(\theta).$$

We now claim that the matrix $[\mathrm{Ad}_{T_{sb}}]$ is always invertible. This can be established by examining the linear equation

$$\begin{bmatrix} R_{sb} & 0 \\ [p_{sb}]\,R_{sb} & R_{sb} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = 0.$$

Its unique solution is $x = y = 0$, implying that the matrix $[\mathrm{Ad}_{T_{sb}}]$ is invertible. Since multiplying any matrix by an invertible matrix does not change its rank, it follows that

$$\mathrm{rank}\ J_s(\theta) = \mathrm{rank}\ J_b(\theta),$$

as claimed; singularities of the space and body Jacobian are one and the same.

Kinematic singularities are also independent of the choice of fixed frame. In some sense this is rather obvious—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 5.8(a). The forward kinematics with respect to the original fixed frame is denoted $T(\theta)$, while the forward kinematics with respect to the relocated fixed frame is denoted $T'(\theta) = PT(\theta)$, where $P \in SE(3)$ is constant.
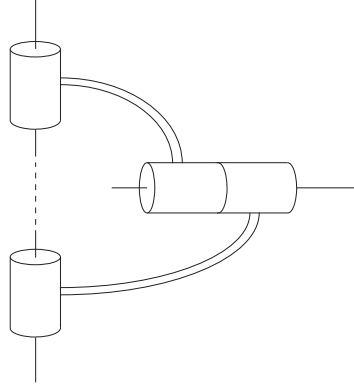
Figure 5.9: A kinematic singularity in which two joint axes are collinear.

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.

Somewhat less obvious is the fact that kinematic singularities are also independent of the choice of end-effector frame. Referring to Figure 5.8(b), 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 5.9). Without loss of generality these joint axes can be labeled 1 and 2. The corresponding columns of the Jacobian are

$$\mathcal{V}_{s1}(\theta) = \begin{bmatrix} \omega_1 \\ -\omega_1 \times q_1 \end{bmatrix}, \quad \mathcal{V}_{s2}(\theta) = \begin{bmatrix} \omega_2 \\ -\omega_2 \times q_2 \end{bmatrix}$$
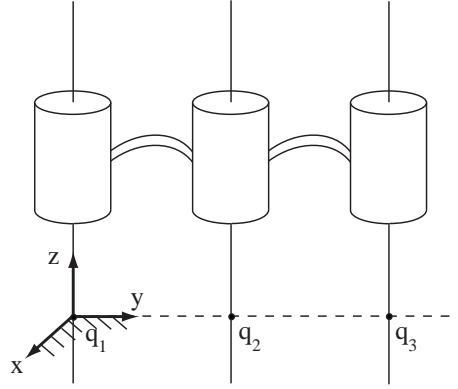
Figure 5.10: A kinematic singularity in which three revolute joint axes are parallel and coplanar.

Since the two joint axes are collinear, we must have $\omega_1 = \pm\omega_2$; let us assume the positive sign. Also, $\omega_i \times (q_1 - q_2) = 0$ for $i = 1, 2$. Then $\mathcal{V}_{s1} = \mathcal{V}_{s2}$, which implies that $\mathcal{V}_{s1}$ and $\mathcal{V}_{s2}$ lie on the same line in six-dimensional space. Therefore, the set $\{\mathcal{V}_{s1}, \mathcal{V}_{s2}, \ldots, \mathcal{V}_{s6}\}$ 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). 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

$$J_s(\theta) = \left[\begin{array}{cccc} \omega_1 & \omega_1 & \omega_1 & \cdots \\ 0 & -\omega_1 \times q_2 & -\omega_1 \times q_3 & \cdots \end{array} \right]$$

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

$$J_s(\theta) = \left[\begin{array}{ccccc} \omega_1 & \omega_2 & \omega_3 & \omega_4 & \cdots \\ 0 & 0 & 0 & 0 & \cdots \end{array} \right].$$

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
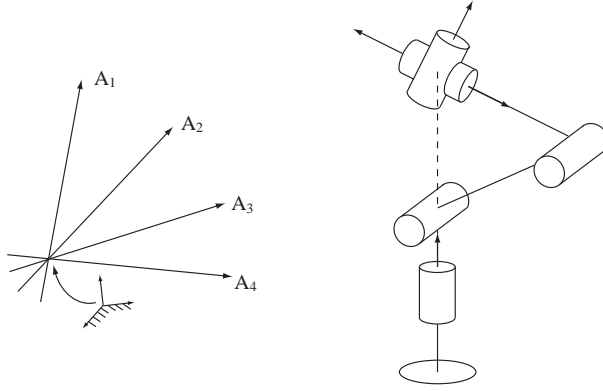
Figure 5.11: A kinematic singularity in which four revolute joint axes intersect at a common point.

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 $\omega_i \in \mathbb{R}^3$ in the direction of joint axis i is of the form

$$\omega_i = \begin{bmatrix} \omega_{ix} \\ \omega_{iy} \\ 0 \end{bmatrix}.$$

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

$$v_i = -\omega_i \times q_i = \begin{bmatrix} 0 \\ 0 \\ \omega_{iy}q_{ix} - \omega_{ix}q_{iy} \end{bmatrix}.$$

The first four columns of the space Jacobian $J_s(\theta)$ are

$$\begin{bmatrix} \omega_{1x} & \omega_{2x} & \omega_{3x} & \omega_{4x} \\ \omega_{1y} & \omega_{2y} & \omega_{3y} & \omega_{4y} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \omega_{1y}q_{1x} - \omega_{1x}q_{1y} & \omega_{2y}q_{2x} - \omega_{2x}q_{2y} & \omega_{3y}q_{3x} - \omega_{3x}q_{3y} & \omega_{4y}q_{4x} - \omega_{4x}q_{4y} \end{bmatrix}.$$

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

$$v_i = -\omega_i \times q_i = (\omega_{iy}q_{iz}, -\omega_{ix}q_{iz}, 0)$$

$i = 1, \ldots, 6$. The space Jacobian $J_s(\theta)$ thus becomes

$$J_s(\theta) = \begin{bmatrix} \omega_{1x} & \omega_{2x} & \omega_{3x} & \omega_{4x} & \omega_{5x} & \omega_{6x} \\ \omega_{1y} & \omega_{2y} & \omega_{3y} & \omega_{4y} & \omega_{5y} & \omega_{6y} \\ \omega_{1z} & \omega_{2z} & \omega_{3z} & \omega_{4z} & \omega_{5z} & \omega_{6z} \\ \omega_{1y}q_{1z} & \omega_{2y}q_{2z} & \omega_{3y}q_{3z} & \omega_{4y}q_{4z} & \omega_{5y}q_{5z} & \omega_{6y}q_{6z} \\ -\omega_{1x}q_{1z} & -\omega_{2x}q_{2z} & -\omega_{3x}q_{3z} & -\omega_{4x}q_{4z} & -\omega_{5x}q_{5z} & -\omega_{6x}q_{6z} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

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 can move with the least "effort"; directions that are orthogonal to these directions in contrast require the greatest effort.

Manipulability ellipsoids are illustrated for a 2R planar arm in Figure 5.2. The Jacobian is given by Equation (5.1).

For a general n-joint open chain and a task space with coordinates $q \in \mathbb{R}^m$, where $m \leq n$, the manipulability ellipsoid corresponds to the end-effector velocities for joint rates $\dot{\theta}$ satisfying $\|\dot{\theta}\| = 1$, a unit hypersphere in the n-dimensional joint velocity space. Assuming J is invertible, the unit joint velocity
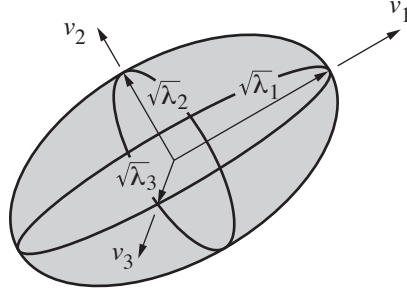
Figure 5.12: An ellipsoid visualization of $\dot{q}^T A^{-1} \dot{q} = 1$ in the $\dot{q}$ space $\mathbb{R}^3$, where the principal semi-axis lengths are the square roots of the eigenvalues λ_i of A and the directions of the principal semi-axes are the eigenvectors v_i.

condition can be written

$$
\begin{aligned}
1 = \dot{\theta}^T \dot{\theta} \\
= (J^{-1}\dot{q})^T (J^{-1}\dot{q}) \\
= \dot{q}^T J^{-T} J^{-1} \dot{q} \\
= \dot{q}^T (JJ^T)^{-1} \dot{q} \\
= \dot{q}^T A^{-1} \dot{q}.
\end{aligned}
$$

If J is full rank (rank m), the matrix $A = JJ^T \in \mathbb{R}^{m \times m}$ is square, symmetric, and positive definite. Let $v_1, \ldots, v_m$ be the eigenvectors of A and $\lambda_1, \ldots, \lambda_m$ be the corresponding eigenvalues. Then the manipulability ellipsoid corresponding to unit joint velocities has principal semi-axes aligned with the eigenvectors v_i of lengths $\sqrt{\lambda_i}$ (Figure 5.12).[4]

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

$$
J(\theta) = \left[\begin{array}{c} J_\omega(\theta) \\ J_v(\theta) \end{array} \right],
$$

where J_ω 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,

---

[4]The ellipsoid interpretation of a positive-definite quadratic form can be found in linear algebra textbooks.

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.

A manipulability measure is given by the ratio of the longest semi-axis of the manipulability ellipsoid to the shortest semi-axis,

$$\kappa = \sqrt{\frac{\lambda_{\max}}{\lambda_{\min}}}.$$

When the condition number κ is 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. When κ becomes large, then it is easy to move in one direction but difficult to move in another. As the robot approaches a singularity, κ approaches infinity.

A force ellipsoid can also be drawn for joint torques τ satisfying $\|\tau\| = 1$. Beginning from $\tau = J^T(\theta)\mathcal{F}$, we arrive at a similar result as above, except now it is the eigenvectors and the square roots of eigenvalues of $A^{-1} = (JJ^T)^{-1}$ that define the force ellipsoid.

5.5 Summary

- Let the forward kinematics of an n-link open chain be expressed in the following product of exponentials form:

$$T = e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_n]\theta_n} M.$$

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 velocity $\mathcal{V}_s$ via $\mathcal{V}_s = J_s(\theta)\dot{\theta}$. The ith column of $J_s(\theta)$ is

$$\mathcal{V}_{si}(\theta) = Ad_{e^{[\mathcal{S}_1]\theta_1} \dots e^{[\mathcal{S}_{i-1}]\theta_{i-1}}}(\mathcal{S}_i), \tag{5.26}$$

for $i = 2, \dots, n$, with the first column $\mathcal{V}_{s1}(\theta) = \mathcal{S}_1$. $\mathcal{V}_{si}$ is the screw vector for joint i expressed in space frame coordinates, with the joint values θ 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:

$$T = Me^{[\mathcal{B}_1]\theta_1} \cdots e^{[\mathcal{B}_n]\theta_n}. \tag{5.27}$$

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 spatial velocity $\mathcal{V}_b = (\omega_b, v_b)$ via

$$\mathcal{V}_b = J_b(\theta)\dot{\theta}. \tag{5.28}$$

The ith column of $J_b(\theta)$ is given by

$$\mathcal{V}_{b,i}(\theta) = \mathrm{Ad}_{e^{-[\mathcal{B}_n]\theta_n} \dots e^{-[\mathcal{B}_{i+1}]\theta_{i+1}}}(\mathcal{B}_i), \tag{5.29}$$

for $i = n - 1, \ldots, 1$, with $\mathcal{V}_{bn}(\theta) = \mathcal{B}_n$. $\mathcal{V}_{bi}$ is the screw vector for joint i expressed in body frame coordinates, with the joint values θ assumed to be arbitrary rather than zero. The body Jacobian is related to the space Jacobian via the relation

$$
\begin{aligned}
J_s(\theta) &= [\mathrm{Ad}_{T_{sb}}] J_b(\theta) \\
J_b(\theta) &= [\mathrm{Ad}_{T_{bs}}] J_s(\theta)
\end{aligned}
$$

where $T_{sb} = T$.

- 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 $\mathcal{F} \in \mathbb{R}^6$ be the spatial force applied at the end-effector, in either space or body frame coordinates. Then τ and $\mathcal{F}$ are related by

$$
\tau = J_b^T(\theta) \mathcal{F}_b = J_s^T(\theta) \mathcal{F}_s.
$$

- 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 (either $J_s(\theta)$ or $J_b(\theta)$) is not maximal. For spatial open chains of mobility six 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.

5.6 Notes and References

The terms spatial velocity and spatial force were first coined by Roy Featherstone [10], 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., [3], [2]. 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 [30].
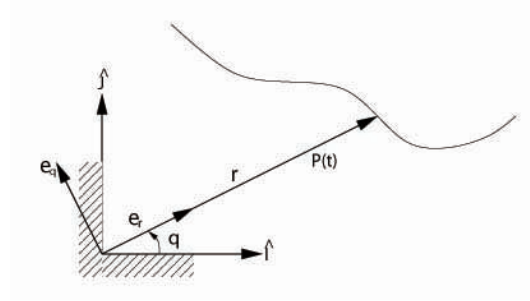
Figure 5.13: Polar coordinates.

5.7   Exercises

1. Given a particle moving in the plane, define a moving reference frame $\{\hat{e}_r, \hat{e}_\theta\}$ such that its origin is fixed to the origin of the fixed frame, and its $\hat{e}_r$ axis always points toward the particle (Figure 5.13). Let (r, θ) be the polar coordinates for the particle position, i.e., r is the distance of the particle from the origin, and θ is the angle from the horizontal line to the $\hat{e}_r$ axis. The particle position $\vec{p}$ can then be written

$$\vec{p} = r\hat{e}_r,$$

and its velocity $\vec{v}$ is given by

$$\vec{v} = \dot{r}\hat{e}_r + r\dot{\hat{e}}_r.$$

The acceleration $\vec{v}$ is the time derivative of $\vec{v}$.
(a) Express $\dot{\hat{e}}_r$ in terms of $\hat{e}_r$ and $\hat{e}_\theta$.
(b) Show that $\vec{v}$ and $\vec{a}$ are given by

$$\begin{aligned} \vec{v} &= \dot{r}\hat{e}_r + r\dot{\theta}\hat{e}_\theta \\ \vec{a} &= (\ddot{r} - r\dot{\theta}^2)\hat{e}_r + (r\ddot{\theta} + 2\dot{r}\dot{\theta})\hat{e}_\theta. \end{aligned}$$

2. Let $\{\hat{I}, \hat{J}\}$ denote the unit axes of the fixed frame, and let

$$\vec{p} = X(t)\hat{I} + Y(t)\hat{J}$$

denote the position of a particle moving in the plane (see Figure 5.14). Suppose the path traced by the particle has nonzero curvature everywhere, so that for every point on the path there exists some circle tangent to the path; the center of this circle is called the **center of curvature**, while its radius is the **radius of curvature**. Clearly both the center and radius of curvature vary along the path, and are well-defined only at points of nonzero curvature (or, at points where the curvature is zero, the center of curvature can be regarded to lie at infinity).
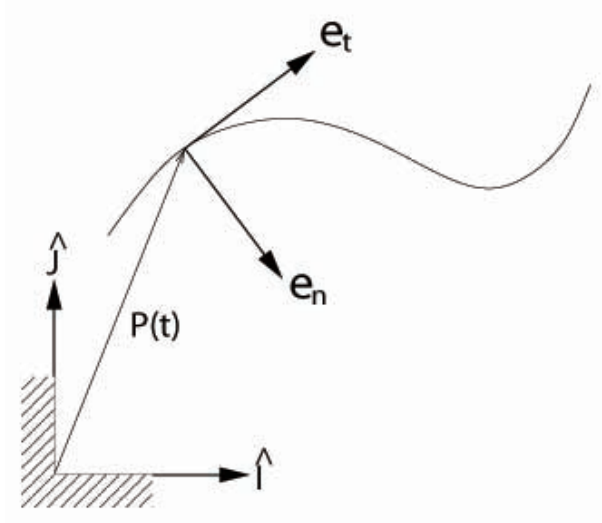
Figure 5.14: Tangential-normal coordinates.

Now attach a moving reference frame $\{\hat{e}_t, \hat{e}_n\}$ to the particle, in such a way that that $\hat{e}_t$ always points in the same direction as the velocity vector; $\hat{e}_n$ then points toward the center of curvature. Since the speed of the particle is given by

$$v = \sqrt{\dot{X}^2 + \dot{Y}^2}.$$

and $\hat{e}_t$ always points in the direction of the velocity vector, it follows that the velocity vector $\vec{v}$ of the particle can be written

$$\vec{v} = v\hat{e}_t,$$

while its acceleration is given by

$$\vec{a} = \dot{v}\hat{e}_t + v\dot{\hat{e}}_t.$$

(a) Show that $\dot{\hat{e}}_t = \|\dot{\hat{e}}_t\|\hat{e}_n$, or

$$\hat{e}_n = \frac{\dot{\hat{e}}_t}{\|\dot{\hat{e}}_t\|},$$

and that consequently the acceleration $\vec{a}$ is

$$\vec{a} = \dot{v}\hat{e}_t + v\|\dot{\hat{e}}_t\|\hat{e}_n.$$

(b) The radius of curvature ρ can be found from the following formula:

$$
\begin{aligned}
\rho &= \frac{v^3}{\ddot{X}\dot{Y} - \ddot{Y}\dot{X}} \\
&= \frac{(\dot{X}^2 + \dot{Y}^2)^{3/2}}{\dot{X}\ddot{Y} - \dot{Y}\ddot{X}}.
\end{aligned}
$$
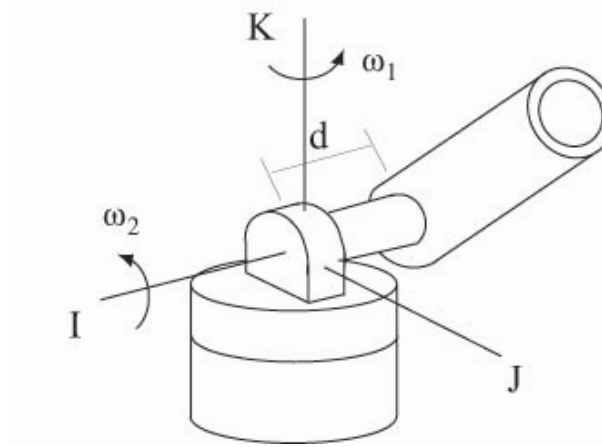
Figure 5.15: A cannon mounted on a $2R$ rotating platform.

Using the formula, show that the acceleration $\vec{a}$ can be written

$$\vec{a} = \dot{v}\hat{e}_t + \frac{v^2}{\rho}\hat{e}_n.$$

3. In standard treatments of particle kinematics using moving frames, two moving particles, A and B, are assumed, with a moving frame $\{\hat{x}, \hat{y}, \hat{z}\}$ attached to particle A. Writing the position of particle B as

$$\vec{p}_B = \vec{p}_A + \vec{p}_{B|A},$$

where $\vec{p}_{B|A}$ denotes the vector from A to B, the following formulas for the velocity and acceleration of B are usually provided:

$$
\begin{aligned}
\vec{v}_B &= \vec{v}_A + \left(\dot{\vec{p}}_{B|A}\right)_{xyz} + \vec{\omega} \times \vec{p}_{B|A} \\
\vec{a}_B &= \vec{a}_A + \left(\ddot{\vec{p}}_{B|A}\right)_{xyz} + 2\vec{\omega} \times \left(\dot{\vec{p}}_{B|A}\right)_{xyz} + \vec{\alpha} \times \vec{p}_{B|A} + \vec{\omega} \times (\vec{\omega} \times \vec{p}_{B|A}),
\end{aligned}
$$

where $\vec{\omega}$ and $\vec{\alpha}$ respectively denote the angular velocity and angular acceleration vector of the moving frame, and $\left(\dot{\vec{p}}_{B|A}\right)_{xyz}$ $\left(\ddot{\vec{p}}_{B|A}\right)_{xyz}$ are certain time derivatives of $\vec{p}_{B|A}$. Writing $\vec{p}_A$ and $\vec{p}_{B|A}$ in terms of fixed and moving frame coordinates, i.e.,

$$
\begin{aligned}
\vec{p}_A &= X\hat{X} + Y\hat{Y} + Z\hat{Z} \\
\vec{p}_{B|A} &= x\hat{x} + y\hat{y} + z\hat{z},
\end{aligned}
$$

derive the above formulas for $\vec{v}_B$ and $\vec{a}_B$. Be sure to explicitly identify all terms, in particular $\left(\dot{\vec{p}}_{B|A}\right)_{xyz}$ and $\left(\ddot{\vec{p}}_{B|A}\right)_{xyz}$.
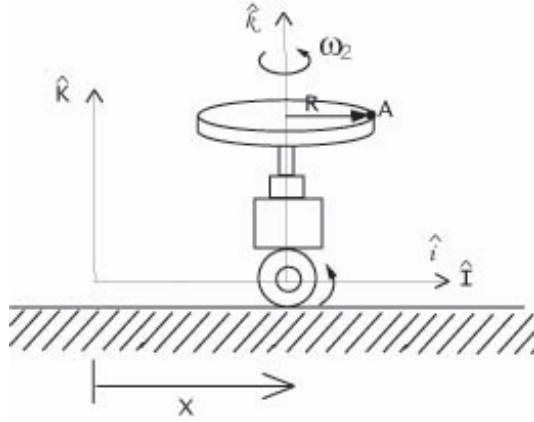
Figure 5.16: A circular plate of radius R.

4. Figure 5.15 depicts a cannon mounted on a $2R$ rotating platform at time $t = 0$. The platform rotates at constant angular velocities ω_1 and ω_2 radians/sec as shown. The axis of the cannon barrel is displaced at a distance d from the origin of the fixed frame. Assume that a cannonball is fired at $t = 0$ from the same height as the $\hat{I}$ axis, at a constant speed v_0 along the axis of the barrel.
(a) Choose a moving frame and describe how the frame moves.
(b) Determine the velocity of the cannonball at $t = 0$ in terms of the moving frame chosen in part (a).

5. As shown in Figure 5.16, a revolving circular plate of radius R, rotating at a constant angular velocity of ω_2 radians/sec, is mounted on a wheeled mobile base that moves periodically back and forth along the $\hat{I}$ axis according to

$$x(t) = a \sin \omega_1 t.$$

(a) Assuming $t = 0$ at the instant shown in the figure, determine the velocity of point A as a function of t in fixed frame coordinates.
(b) Determine the acceleration of point A as a function of t in fixed frame coordinates.

6. The circular pipe of Figure 5.17 is rotating about the $\hat{X}$ axis at a constant rate ω_1 radians/sec, while a marble D is circling the pipe at a constant speed u.
(a) Find the angle θ at which the magnitude of the velocity of D is maximal. What is the maximal velocity magnitude at this angle?
(b) Find the angle θ at which the magnitude of the acceleration of D is maximal. What is the maximal acceleration magnitude at this angle?
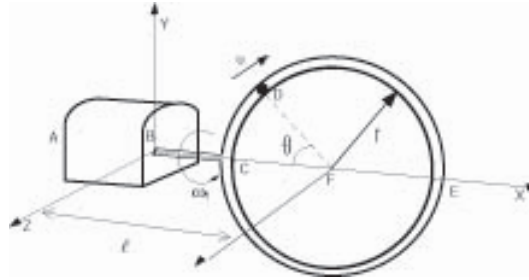
Figure 5.17: A marble traversing a rotating circular pipe.
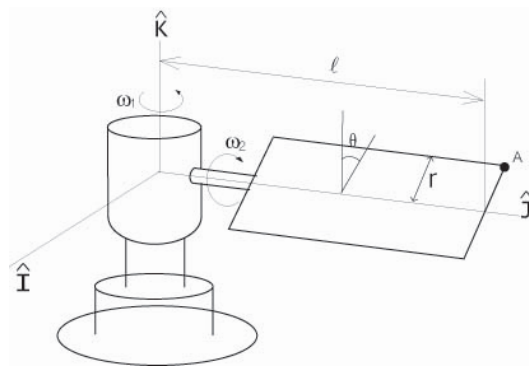
Figure 5.18: A satellite with a rotating panel.

7. The satellite of Figure 5.18 is rotating about its own vertical $\hat{K}$ axis at a constant rate ω_1 radians/sec, while at the same time its solar panel rotates at a constant rate ω_2 radians/sec as shown.
(a) Determine the velocity of point A when $\omega_1 = 0.5$, $\omega_2 = 0.25$, $l = 2m$, $r = 0.5m$, and $\theta = 30°$.
(b) Determine the acceleration of point A under the same conditions as part (a).

8. The two revolute joints in the spherical $2R$ open chain of Figure 5.19 rotating at constant angular velocities ω_1 radians/sec and ω_2 radians/sec as shown. Denote by r the length of link AB, while θ is the angle between link AB and the x-y plane.
(a) Choose a moving frame and explain how the frame moves.
(b) Determine the angular velocity and angular of link AB in terms of your moving frame coordinates chosen in part (a).
(c) Determine the velocity of point B in terms of the chosen moving frame coordinates.
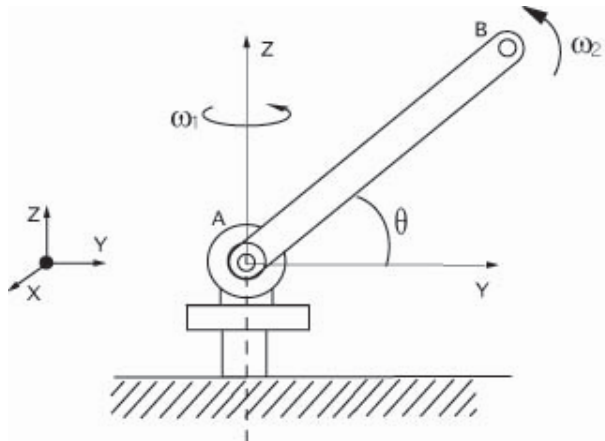
Figure 5.19: A spherical $2R$ open chain.
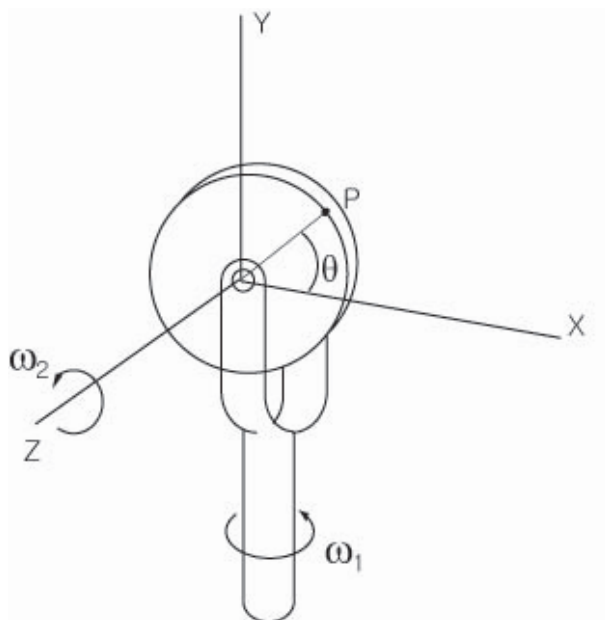
Figure 5.20: A rotating disk.

(d) Determine the acceleration of point B in terms of the chosen moving frame coordinates.

(e) Setting $\omega_1 = 0.1$, $\omega_2 = 0.2$, and $r = 100mm$, determine the velocity and acceleration of point B in terms of the fixed frame coordinates when $\theta = \pi/6$.

9. As shown in Figure 5.20, a disk of radius r spins at a constant angular
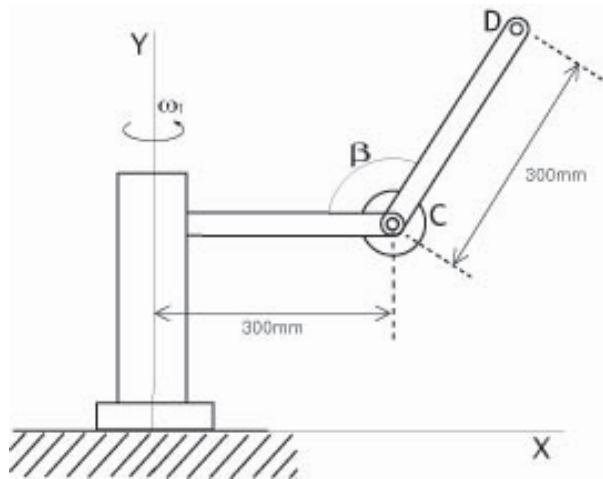
Figure 5.21: A toroidal $2R$ open chain.

velocity of ω_2 radians/sec about its horizontal axis, while at the same time the disk assembly rotates about the vertical axis at a constant angular velocity of ω_1 radians/sec.
(a) Determine the angular velocity and the angular acceleration of the disk in terms of fixed frame coordinates.
(b) Determine the velocity and the acceleration of point P as a function of the angle θ.

10. As shown in Figure 5.21, the two revolute joints of the toroidal $2R$ open chain are rotating at a constant angular velocity $\omega_1 = 0.6$ radians/sec about the $\hat{Y}$ axis, and $\omega_2 = 0.45$ radians/sec about the horizontal axis through C. When $\beta = 120°$, determine the following in terms of fixed frame coordinates:
(a) the angular acceleration of link CD.
(b) the velocity of point D.
(c) the acceleration of point D.

11. Figure 5.22 shows an RRP open chain at $t = 0$. The revolute joints rotate at constant angular velocities ω_1 and ω_2 radians/sec. Suppose the vertical position of point B is given by $x(t) = \sin t$. Determine the following quantities in terms of fixed frame coordinates.
(a) The velocity of point B at $t = 0$.
(b) The acceleration of point B at $t = 0$.

12. The square plate of Figure 5.23 rotates about axis $\hat{I}$ with angular velocity $\omega_2 = 0.5$ radians/sec and angular acceleration $\alpha_2 = 0.01$ radians/sec^2, while the circular disk attached to the square plate rotates about the axis normal
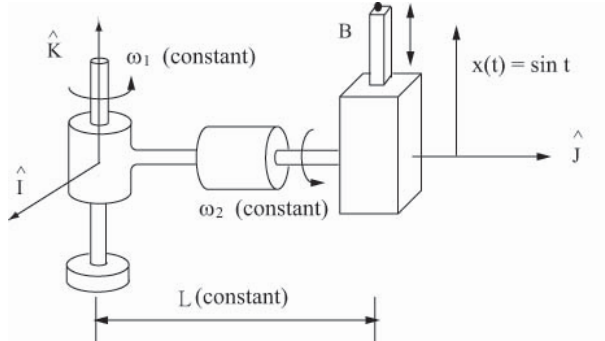
Figure 5.22: An RRP open chain.
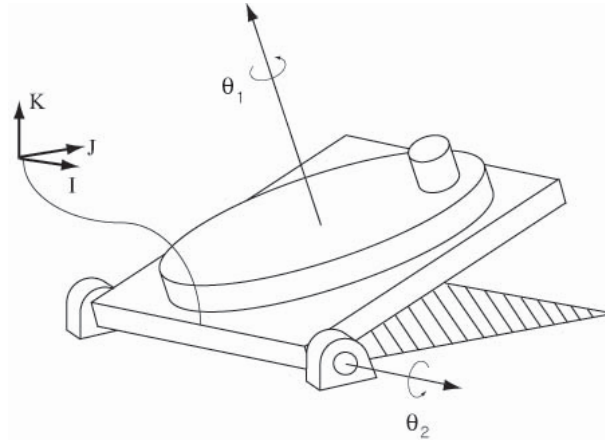
Figure 5.23: A rotating square plate.

to the plate with angular velocity $\omega_1 = 1$ radians/sec and angular acceleration $\alpha_1 = 0.5$ radians/sec^2. The radius of the circular disk is $R = 5m$, while the length of each side of the square plate is $2R = 10m$. The distance from the center of the circular disk to the small circular knob is $d = 3m$. Assume that both the disk and the square plate have zero thickness. Setting $\theta_1 = 0°$ and $\theta_2 = 45°$, find the following in terms of fixed frame coordinates:
(a) The velocity of the circular knob.
(b) The acceleration of the circular knob.

13. A person is riding the $2R$ gyro swing of Figure 5.24. Joint θ oscillates periodically according to $\theta(t) = \cos t$, and the circular plate connected to the axis of the gyro swing rotates with constant angular velocity ω_2 radians/sec. At $t = 0$, the person on the circular plate is at the maximal height as shown in the figure. Setting $l = 1$, $r = 1$, and $\omega_2 = 1$ radian/sec, determine the velocity of the person in terms of the given fixed frame coordinates when $t = \frac{\pi}{2}$.
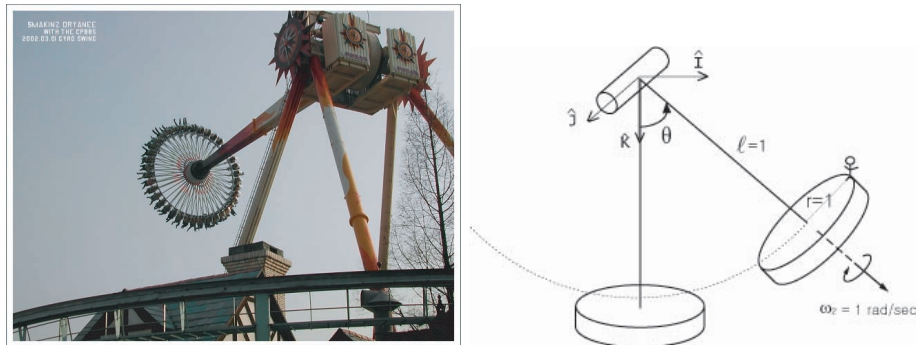
Figure 5.24: A $2R$ gyro swing.
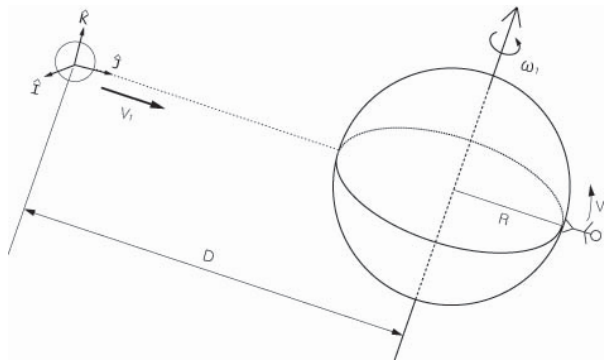
Figure 5.25: A meteorite approaching the earth.

14. As shown in Figure 5.25, a meteorite is approaching a rotating asteroid along the meteorite's $\hat{J}$ axis with velocity $v_1 = 100$ m/sec. Suppose the radius of the asteroid is $R = 1000m$, and the distance of the meteorite from the asteroid is initially $D = 10^7 m$. The asteroid takes 6 hours to complete a full revolution. An astronaut stands at the point antipodal to the expected point of collision, and unwittingly starts walking north along a longitudinal arc at a velocity of $v_2 = 1$ m/sec. After three hours, determine the velocity of the astronaut in terms of the moving frame coordinates attached to the meteorite. of the moving frame at the meteorite.

15. As shown in Figure 5.26, a clock of radius r is mounted on a gimbal assembly as shown. The angles θ_1, θ_2, and θ_3 are adjustable to arbitrary values; in the figure the angles are all assumed to be set to zero. A moving frame $\{T\}$ is attached to the tip of the clock's second hand, with its $\hat{x}$-axis aligned along the tip of the hand as shown. Setting $r = 1m$, $a = 3m$, $b = 7m$, answer the
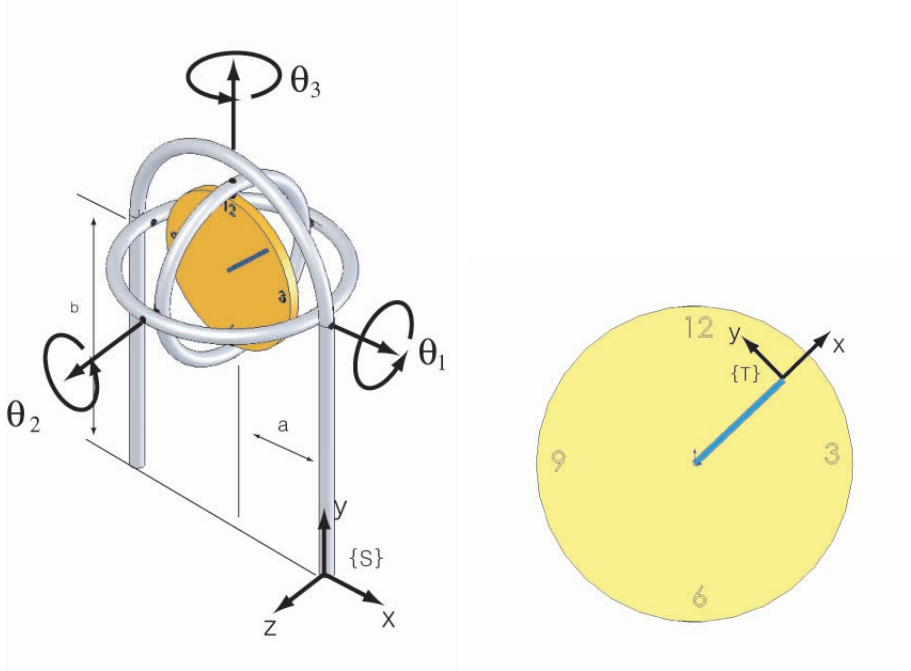
Figure 5.26: A clock mounted on a gimbal assembly.

following:

(a) Assuming the second hand starts at 12 at $t = 0$, when the second hand reaches 10, find $T_{ST} \in SE(3)$ as a function of the angles $(\theta_1, \theta_2, \theta_3)$.

(b) Setting $\theta_1 = 90°$, $\theta_2 = 0$, $\theta_3 = 90°$, find the the velocity of the tip of the second hand at the moment it passes 10.

16. 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) Using the above result to show that for $r(t) \in \mathbb{R}^3$ and $R(t) = e^{[r(t)]}$, the
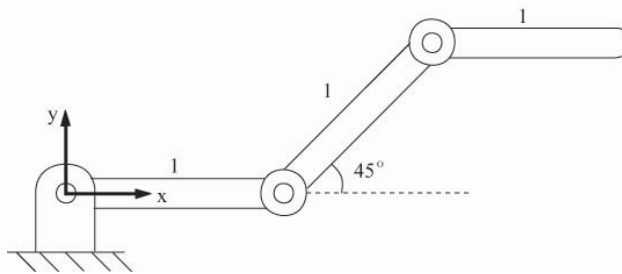
Figure 5.27: A $3R$ planar open chain.
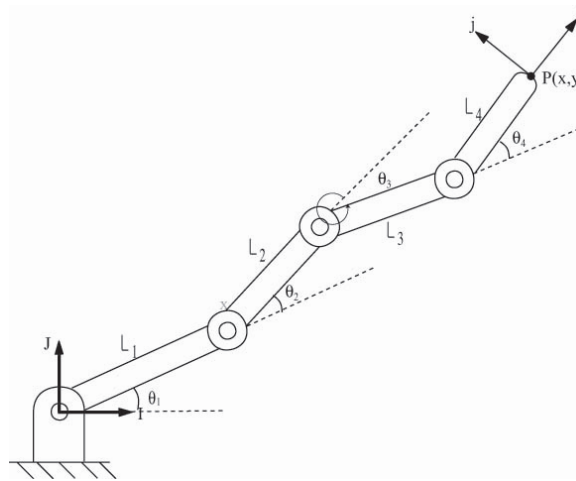
Figure 5.28: A planar $4R$ open chain.

angular velocity in the body frame, $[\omega_b] \, R^T \dot{R}$ is related to $\dot{r}$ by

$$
\begin{aligned}
\omega_b &= A(r)\dot{r} \\
A(r) &= I - \frac{1 - \cos \|r\|}{\|r\|^2}[r] + \frac{\|r\| - \sin \|r\|}{\|r\|^3}[r]^2.
\end{aligned}
$$

(c) Derive the corresponding formula relating the angular velocity in the space frame, $[\omega_s] \, \dot{R}R^T$, with $\dot{r}$.

17. The $3R$ planar open chain of Figure 5.27 is shown in its zero position.
(a) Suppose the tip must apply a force of $5N$ 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 $5N$ in the $\hat{y}$-direction. What torques should be applied at each of the joints?
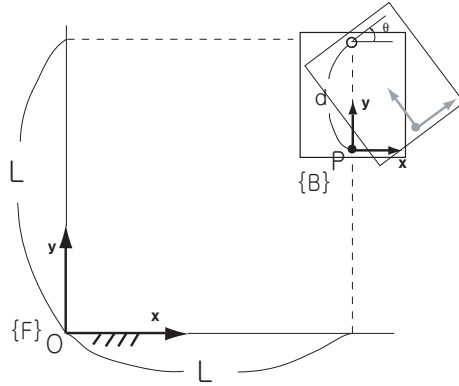
Figure 5.29: A rigid body rotating in the plane.

18. Answer the following questions for the $4R$ planar open chain of Figure 5.28.
(a) Derive the forward kinematics in the form

$$T(\theta) = e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} e^{[\mathcal{S}_3]\theta_3} e^{[\mathcal{S}_4]\theta_4} M.$$

where each $\mathcal{S}_\rangle \in \mathbb{R}^3$ 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.

19. Referring to Figure 5.29, 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 θ.
(b) Find the velocity of point P in terms of the fixed frame.
(c) What is T_{fb}, the displacement of frame $\{b\}$ as seen from the fixed frame $\{f\}$?
(d) Find the spatial velocity of T_{fb} in body coordinates.
(e) Find the spatial velocity of T_{fb} in space coordinates.
(f) What is the relation between the spatial velocities obtained in (d) and (e)?
(g) What is the relation between the spatial velocity obtained in (d) and $\dot{P}$ obtained in (b)?
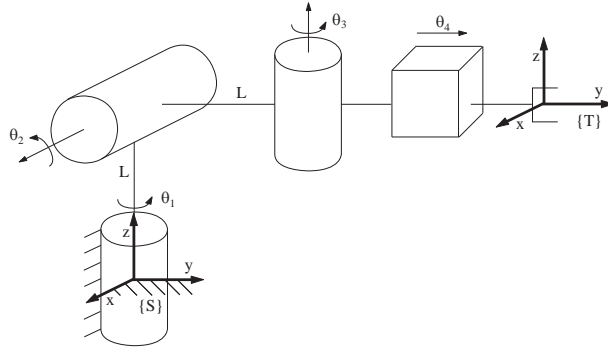
Figure 5.30: An $RRRP$ spatial open chain.
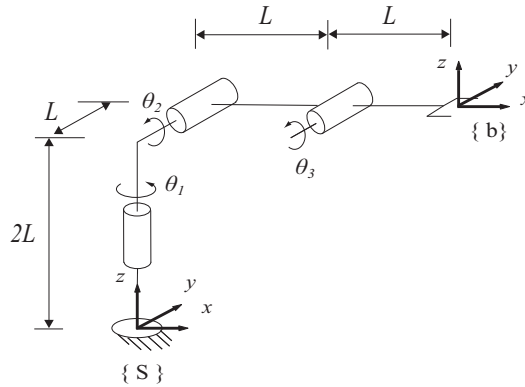
Figure 5.31: A spatial $3R$ open chain.

(h) What is the relation between the spatial velocity obtained in (e) and $\dot{P}$ obtained in (b)?

20. The $RRRP$ chain of Figure 5.30 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 $\dot{\theta}_1 = \dot{\theta}_2 = \dot{\theta}_3 = \dot{\theta}_4 = 1$.

21. The spatial $3R$ open chain of Figure 5.31 is shown in its zero position.
(a) In its zero position, suppose we wish to make the end-effector move with linear velocity $v_{tip} = (10, 0, 0)$, where v_{tip} is expressed with respect to the space frame $\{s\}$. What are the necessary input joint velocities $\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3$?
(b) Suppose the robot is in the configuration $\theta_1 = 0, \theta_2 = 45°, \theta_3 = -45°$. 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 necessary input joint torques τ_1, τ_2, τ_3?
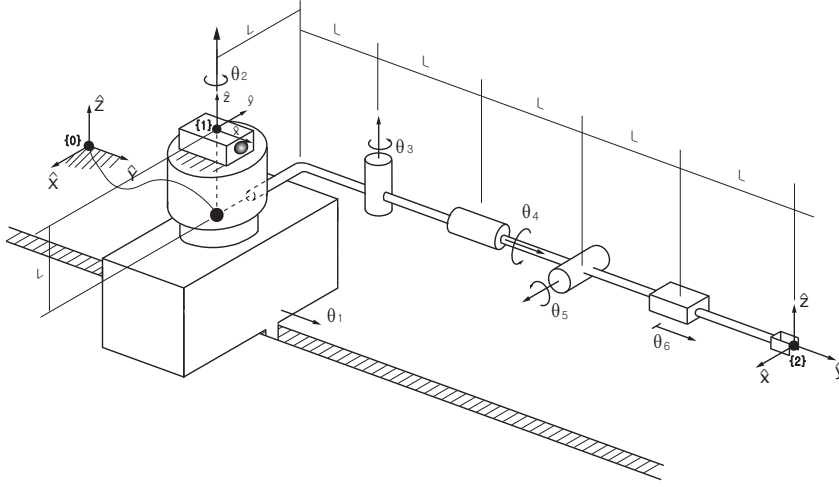
Figure 5.32: A spatial $PRRRRP$ open chain.

(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 necessary input joint torques τ_1, τ_2, τ_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 home position, what is the maximum force that can be applied by the tip in the end-effector frame x-direction?

22.   The spatial $PRRRRP$ open chain of Figure 5.32 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 $\theta_1 = \theta_2 = \theta_3 = \theta_5 = \theta_6 = 0, \theta_4 = 90^o$. 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 τ_1, τ_2, τ_3 experienced at the first three joints.

23. Consider the $PRRRRR$ spatial open chain of Figure 5.33 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?
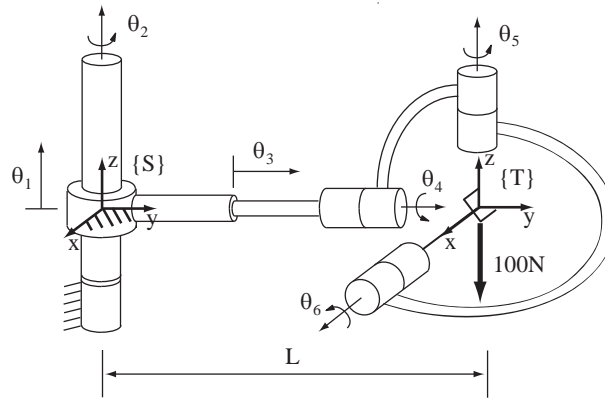
Figure 5.33: A *PRRRRR* spatial open chain.
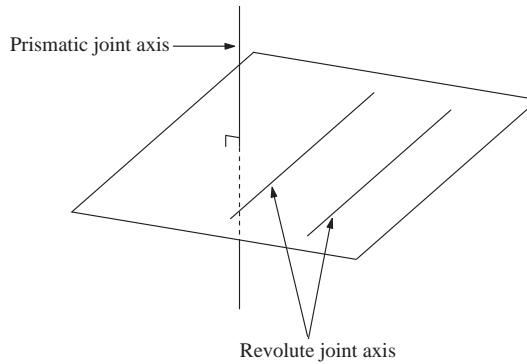
Figure 5.34: A kinematic singularity involving prismatic and revolute joints.

(d) In the zero position, what joint torques must be applied in order to generate a pure end-effector force of $100N$ in the $-\hat{z}$ direction?

24. Find all kinematic singularities of the $3R$ wrist with the following forward kinematics:
$$R = e^{[\omega_1]\theta_1} e^{[\omega_2]\theta_2} e^{[\omega_3]\theta_3}$$
where $\omega_1 = (0, 0, 1)$, $\omega_2 = (1/\sqrt{2}, 0, 1/\sqrt{2})$, and $\omega_3 = (1, 0, 0)$.

25. Show that a six degree of freedom 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 5.34).

26. (a) Determine the space Jacobian $J_s(\theta)$ of the $6R$ spatial open chain of Figure 5.35.
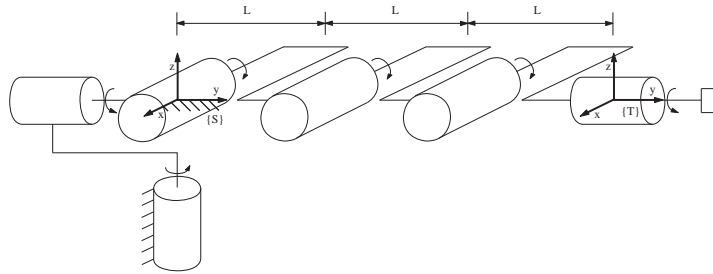
Figure 5.35: Singularities of an elbow-type $6R$ open chain.
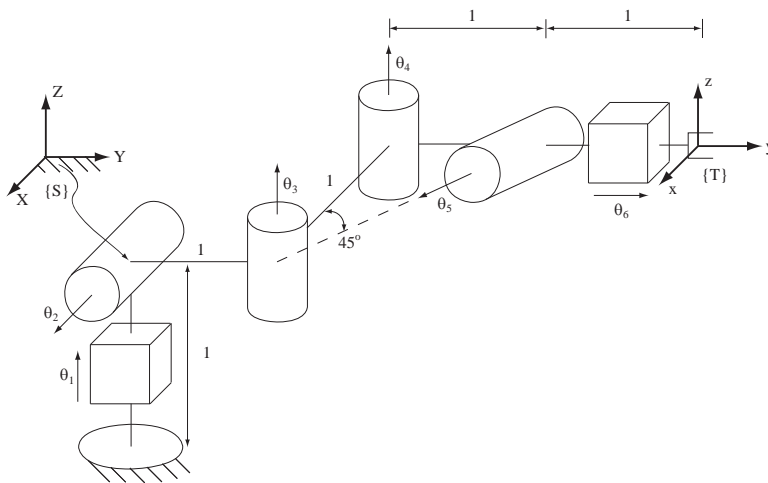
Figure 5.36: A spatial $PRRRRP$ open chain with a skewed joint axis.

(b) Find the kinematic singularities of the given chain. Explain each singularity in terms of the alignment of the joint screws, and the directions in which the end-effector loses one or more degrees of freedom of motion.

27. The spatial $PRRRRP$ open chain of Figure 5.36 is shown in its zero position.
(a) Determine the first 4 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 spatial forces:

(i) $F_s = (0, 1, -1, 1, 0, 0)^T$

(ii) $F_s = (1, -1, 0, 1, 0, -1)^T$.

28. The spatial $RRPRRR$ open chain of Figure 5.37 is shown in its zero
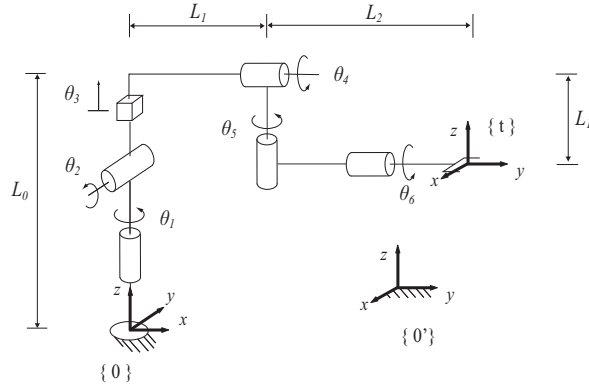
Figure 5.37: A spatial $RRPRRR$ open chain.
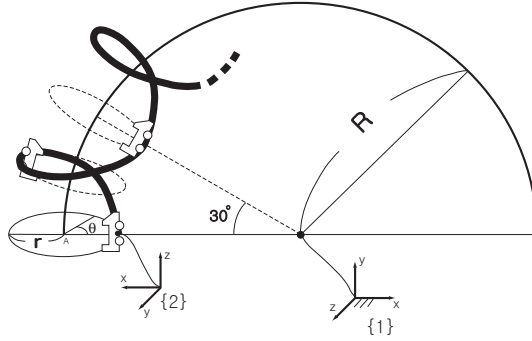
Figure 5.38: A rollercoaster undergoing a screw motion.

position.

(a) For the fixed frame $\{0\}$ and tool frame $\{t\}$ as shown, express the forward kinematics in the following product of exponentials form:

$$T(\theta) = e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} e^{[\mathcal{S}_3]\theta_3} e^{[\mathcal{S}_4]\theta_4} e^{[\mathcal{S}_5]\theta_5} e^{[\mathcal{S}_6]\theta_6} M.$$

(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.

29. The rollercoaster of Figure 5.38 undergoes a screw motion as shown: point A traces a circle of radius R, and the rollercoaster moves in screw-like fashion at a distance r from this larger circle. The roller coaster completes one revolution about this larger circle when point A traverses $30°$ along the larger circle.

(a) Find T_{12}, the relative displacement of the rollercoaster frame $\{2\}$ as seen from the fixed frame $\{1\}$, in terms of the angle θ as indicated in the figure.
(b) Derive the space Jacobian for $T_{12}(\theta)$.

30. Two frames $\{a\}$ and $\{b\}$ are attached to a moving rigid body. Show that the spatial velocity of $\{a\}$ in space frame coordinates is the same as the spatial velocity of $\{b\}$ in space frame coordinates.

31. Consider an n-link open chain, with reference frames attached to each link. Let

$$T_{0k} = e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_k]\theta_k} M_k, \quad k = 1, \ldots, n$$

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

$$J_s(\theta) = \begin{bmatrix} \mathcal{V}_{s1}(\theta) & \cdots & \mathcal{V}_{sn}(\theta) \end{bmatrix}.$$

Let $[\mathcal{V}_k] = \dot{T}_{0k} T_{0k}^{-1}$ be the spatial velocity of link frame $\{k\}$ in frame $\{k\}$ coordinates.
(a) Derive explicit expressions for $\mathcal{V}_2$ and $\mathcal{V}_3$.
(b) Based on your results from (a), derive a recursive formula for $\mathcal{V}_{k+1}$ in terms of $\mathcal{V}_k$, $\mathcal{V}_{s1}, \ldots, \mathcal{V}_{s,k+1}$, and $\dot{\theta}$.

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 θ 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

$$\left[\begin{array}{c} x \\ y \end{array} \right] = \left[\begin{array}{c} L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) \\ L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) \end{array} \right]. \tag{6.1}$$

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. The case of two solutions is given by the familiar elbow-up and elbow-down configurations.

Finding an explicit solution (θ_1, θ_2) for a given (x, y) is also not difficult. Referring this time to Figure 6.1(b), assume that (x, y) lies in the first quadrant, i.e., both x and y are positive (solutions for the other quadrants follow straightforwardly). Angle β, restricted to lie in the interval $[0, \pi]$, can be determined from the law of cosines:

$$L_1^2 + L_2^2 - 2L_1 L_2 \cos\beta = x^2 + y^2,$$

from which it follows that

$$\begin{aligned} \beta &= \cos^{-1}\left(\frac{L_1^2 + L_2^2 - x^2 - y^2}{2L_1 L_2} \right) \\ \theta_2 &= \pi - \beta. \end{aligned}$$

Also from the law of cosines,

$$\alpha = \cos^{-1}\left(\frac{x^2 + y^2 + L_1^2 - L_2^2}{2L_1 \sqrt{x^2 + y^2}} \right).$$
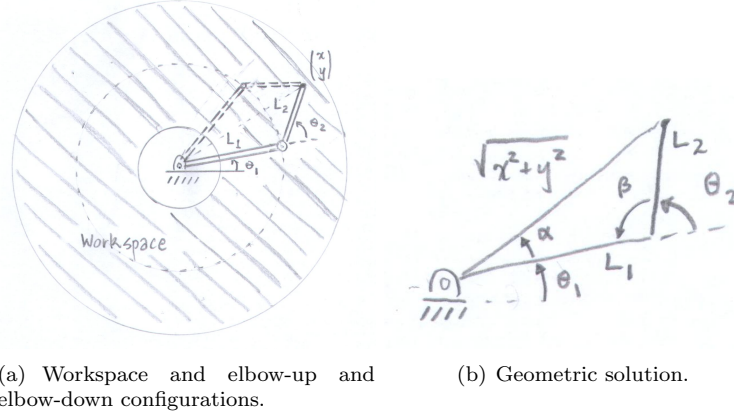
(a) Workspace and elbow-up and elbow-down configurations.

(b) Geometric solution.

Figure 6.1: $2R$ planar open chain inverse kinematics.

and since $\tan(\theta_1 + \alpha) = y/x$, it follows that

$$\theta_1 = \tan^{-1} \frac{y}{x} - \alpha.$$

The above values for θ_1 and θ_2 correspond to the elbow-down solution. The elbow-up solution is given by

$$\theta_1 = \tan^{-1} \frac{y}{x} + \alpha, \ \ \theta_2 = \pi + \beta.$$

If $x^2 + y^2$ lies outside the range $[L_1 - L_2, L_1 + L_2]$, then no solution exists. Again, the case when (x, y) lies in other quadrants follows straightforwardly from the above solution for the first quadrant.

   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 joint values θ. In fact, three-link planar open chains have an infinite number of solutions for points (x, y) lying in the interior of the workspace; in this case the chain possesses an extra degree of freedom, and is said to be kinematically redundant.

   This chapter begins by considering the inverse kinematics of spatial open chains with six degrees of freedom. A finite number of solutions exists in this case, and we first make some simplifying assumptions about the kinematic structure that lead to analytic solutions. As we shall see, these assumptions are not overly restrictive, as they accommodate the most commonly used six-degree-of-freedom open chains. For more general open chains, we specialize the Newton-Raphson method for nonlinear root finding to the inverse kinematics problem. The result is an iterative numerical algorithm that, provided an initial guess of the joint variables is sufficiently close to the true solution, eventually converges to the solution. The chapter concludes with a discussion of pseudoinverse-based inverse kinematics solutions for redundant open chains.
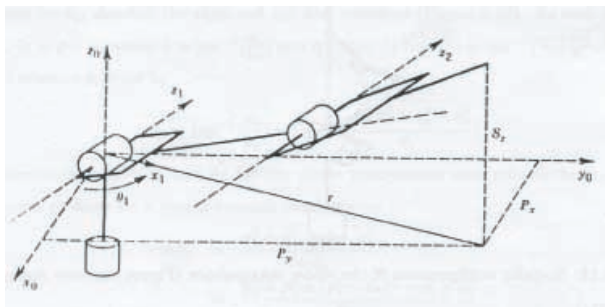
Figure 6.2: Inverse position kinematics of a 6R PUMA-type arm.
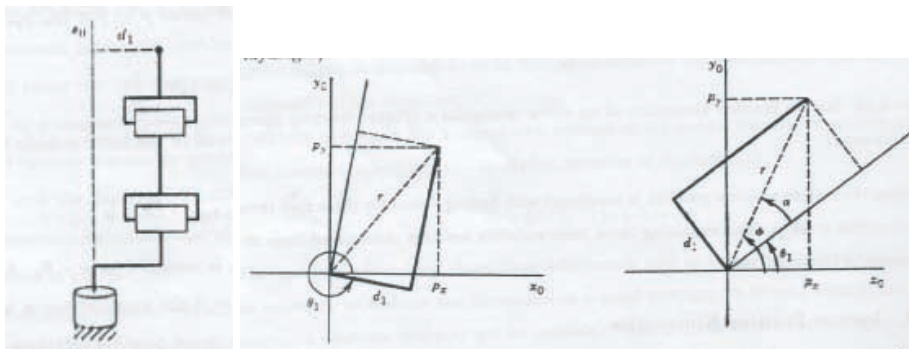
(a) Elbow arm with offset.

(b) Kinematic diagram.

Figure 6.3: A 6R PUMA-type arm with a shoulder offset.

6.1 Analytic Inverse Kinematics

We begin by writing the forward kinematics of a spatial six-degree-of-freedom 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 $X \in SE(3)$, the inverse kinematics problem is concerned with finding all solutions $\theta \in \mathbb{R}^6$ that satisfy $T(\theta) = X$. In the following subsections we shall make some simplifying assumptions about the kinematic structure of the open chain that will lead to analytic solutions for the inverse kinematics.

6.1.1 6R PUMA-Type Arm

We first consider a $6R$ arm of the PUMA type (see Figure 6.2). Such arms are characterized by (i) the last three joint axes intersecting orthogonally at a common point (the wrist center) to form an orthogonal wrist; (ii) the first two axes intersecting orthogonally to form a shoulder joint; and (iii) the elbow
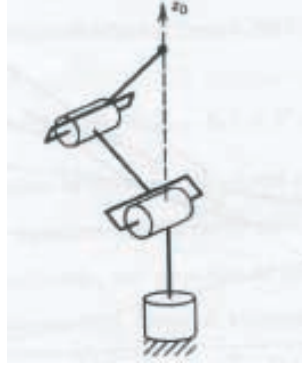
Figure 6.4: Singular configuration of the zero-offset $6R$ PUMA-type arm.

joint axis being parallel with the horizontal shoulder joint axis. Such arms 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, which we now discuss.

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

$$\theta_1 = \tan^{-1} \frac{p_y}{p_x},$$

where the atan2() function can be used instead of $\tan^{-1}$. Note that a second valid solution for θ_1 is given by

$$\theta_1 = \tan^{-1} \frac{p_y}{p_x} + \pi,$$

provided that the original solution for θ_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 θ_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 θ_1, denoted the **right** and **left arm** solutions (Figure 6.3). As seen from the figure, $\theta_1 = \phi - \alpha$, where $\phi = \tan^{-1}(\frac{p_y}{p_x})$ and $\alpha = \tan^{-1}(\frac{\sqrt{r^2 - d_1^2}}{d_1}) = \tan^{-1}(\frac{\sqrt{p_x^2 + p_y^2 - d_1^2}}{d_1})$. The second solution is given by

$$\theta_1 = \tan^{-1}(\frac{p_y}{p_x}) - \tan^{-1}(\frac{-\sqrt{p_x^2 + p_y^2 - d_1^2}}{d_1})$$

Determining angles θ_2 and θ_3 for the PUMA-type arm now reduces to the inverse
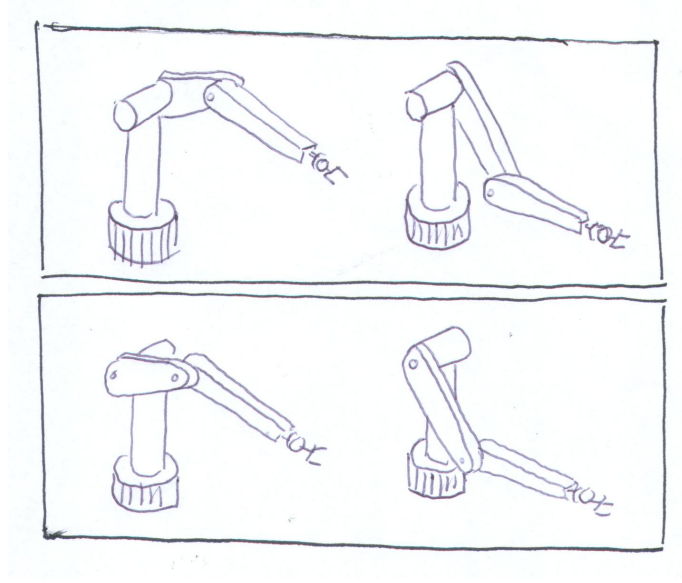
Figure 6.5: Four possible inverse kinematics solutions for the 6R PUMA type arm with shoulder offset.

kinematics problem for a planar two-link chain:

$$
\begin{aligned}
\cos\theta_3 &= \frac{r^2 + s^2 - a_2^2 - a_3^2}{2a_2a_3} \\
&= \frac{p_x^2 + p_y^2 + (p_z^2 - d_1)^2 - a_2^2 - a_3^2}{2a_2a_3}
\end{aligned}
$$

If we let $\cos\theta_3 = D$, then θ_3 is given by

$$
\theta_3 = \tan^{-1}(\pm\frac{\sqrt{1 - D^2}}{D})
$$

θ_2 can be obtained in a similar fashion as

$$
\begin{aligned}
\theta_2 &= \tan^{-1}(\frac{s}{r}) - \tan^{-1}(\frac{a_3s_3}{a_2 + a_3c_3}) \\
&= \tan^{-1}(\frac{p_z - d_1}{\sqrt{p_x^2 + p_y^2}}) - \tan^{-1}(\frac{a_3s_3}{a_2 + a_3c_3})
\end{aligned}
$$

The two solutions for θ_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).
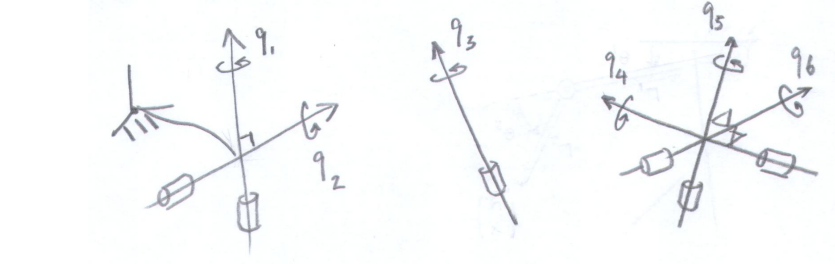
Figure 6.6: A $6R$ spatial open chain of the generalized PUMA type.

We now solve the inverse orientation problem, i.e., finding values for $(\theta_4, \theta_5, \theta_6)$ given the end-effector frame orientation. This is completely straightforward: since the final three joints form a 3R wrist with orthogonal axes, the joint values can be determined via an appropriate set of Euler angles as discussed in Appendix B (e.g., ZYX, ZYZ, depending on how the final three joint axes are aligned when in the zero position).

6.1.2 Generalized 6R PUMA-Type Arms

We now relax some of the assumptions made for the 6R PUMA-type arm: a **generalized 6R PUMA-type arm** is characterized by (i) the first two joint axes intersecting orthogonally, and (ii) the last three joint axes intersecting orthogonally at a common point. Referring to Figure 6.6, place the fixed frame origin at the intersection of joint axes 1 and 2, and let $r_w \in \mathbb{R}^3$ be the fixed frame representation of the point of intersection of the final three axes. The assumptions about the joint axes then lead to the following relations among the joint screws $S_i = (\omega_i, -\omega_i \times r_i)$, $i = 1, \ldots, 6$, where r_i denotes a reference point on axis i:

- $\omega_1^T \omega_2 = 0$;

- $\omega_4^T \omega_5 = 0$ and $\omega_5^T \omega_6 = 0$.

The inverse kinematics problem can now be stated as finding solutions θ to

$$e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} e^{[S_4]\theta_4} e^{[S_5]\theta_5} e^{[S_6]\theta_6} = XM^{-1}, \tag{6.2}$$

where

$$
\begin{aligned}
S_1 &= (\omega_1, 0) \\
S_2 &= (\omega_2, 0) \\
S_3 &= (\omega_3, -\omega_3 \times r_3) \\
S_4 &= (\omega_4, -\omega_4 \times r_w) \\
S_5 &= (\omega_5, -\omega_5 \times r_w) \\
S_6 &= (\omega_6, -\omega_6 \times r_w),
\end{aligned}
$$
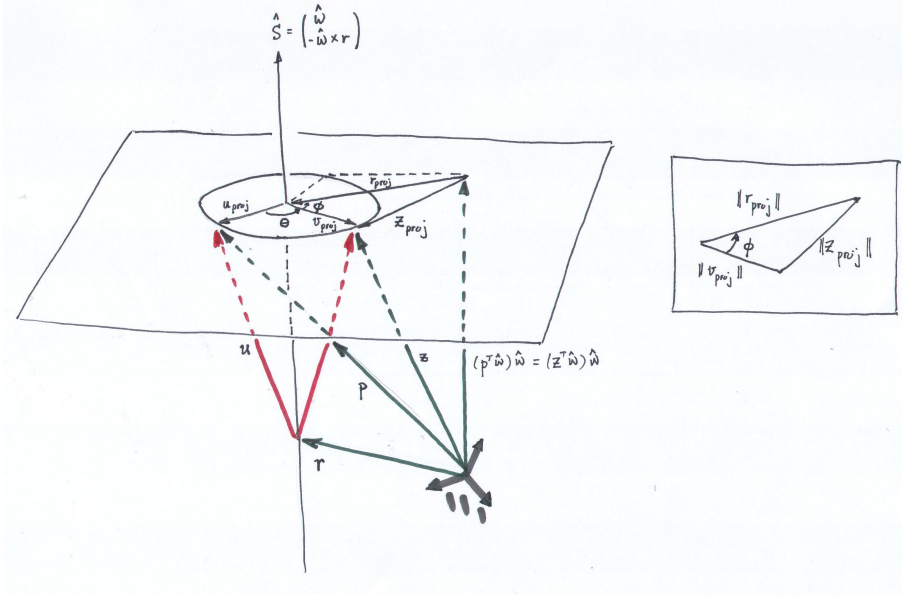
Figure 6.7: Solving $\|e^{[S]\theta}p\| = c$ for θ.

and the right-hand side XM^{-1} is given; denote this known quantity by $X_1 = XM^{-1}$. Solving the inverse kinematics then proceeds in three steps:

Step 1: Solve for θ_3

We first multiply both sides of Equation (6.2) by r_w (here multiplication of a vector by a homogeneous transformation is understood in the usual sense, i.e.,

$$Tr_w = Rr_w + p, \quad T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}.$$

The net effect is to consecutively apply the screw motions defined by $e^{[S_6]\theta_6}$, $e^{[S_5]\theta_5}$, and $e^{[S_4]\theta_4}$ to r_w. However, since all three of these screw motions are zero pitch (all joints are revolute), and r_w is a point lying on all three screw axes, it follows that

$$e^{[S_4]\theta_4} e^{[S_5]\theta_5} e^{[S_6]\theta_6} r_w = r_w.$$

We are thus left with

$$e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} r_w = X_1 r_w = p_1, \tag{6.3}$$

where the vector $p_1 = X_1 r_w$ is known.

Now take the norm of both sides of (6.3). Since both $e^{[S_1]\theta_1}$ and $e^{[S_2]\theta_2}$ are pure rotations, and the general identity $\|Rv\| = \|v\|$ holds for any rotation R and vector v, Equation (6.3) becomes

$$\|e^{[S_3]\theta_3} r_w\| = \|p_1\|.$$

This is a problem of the general form

$$\|e^{[S]\theta}p\| = c,$$

where $S = (\omega, -\omega \times r)$, $p \in \mathbb{R}^3$, and the scalar $c > 0$ are known, and the objective is to find all values of $\theta \in [0, 2\pi]$ satisfying this equality. solve this problem. Referring to Figure 6.7, define the vectors $w, u, v \in \mathbb{R}^3$ according to

$$
\begin{aligned}
z &= e^{[S]\theta}p \\
u &= p - r \\
v &= w - r,
\end{aligned}
$$

with $\|z\| = c$ given. We shall now project the vectors p, u, v, r, and z onto the plane normal to the screw axis and containing p: define these respectively to be

$$
\begin{aligned}
p_{proj} &= p - (p^T\omega)\omega \\
u_{proj} &= u - (u^T\omega)\omega \\
v_{proj} &= v - (v^T\omega)\omega \\
r_{proj} &= r - (r^T\omega)\omega \\
z_{proj} &= z - (z^T\omega)\omega.
\end{aligned}
$$

Note that u_{proj} and q_{proj} are known *a priori*. From the figure it can be seen that

$$\|(z^T\omega)\omega\| = \|(p^T\omega)\omega\| = |p^T\omega|.$$

Since $z_{proj} = z - (z^T\omega)\omega$ and $\|z\|^2 = c^2$, it follows that

$$\|z_{proj}\|^2 = c^2 - (p^T\omega)^2,$$

which is also known *a priori*. Let us first find the angle $\psi = \theta + \phi$, where ϕ is defined as indicated in the figure. Since

$$
\begin{aligned}
u_{proj}^T(-q_{proj}) &= \|u_{proj}\| \cdot \|q_{proj})\| \cos(\theta + \phi) & (6.4) \\
u_{proj} \times (-q_{proj}) &= \hat{\omega}\left(\|u_{proj}\| \cdot \|q_{proj}\| \sin(\theta + \phi)\right), & (6.5)
\end{aligned}
$$

from the latter equality it follows that

$$\omega^T\left(u_{proj} \times (-q_{proj})\right) = \|u_{proj}\|\|q_{proj}\| \sin(\theta + \phi). \qquad (6.6)$$

From Equations (6.4) and (6.6) we have

$$\psi = \tan^{-1}\left(\frac{\omega^T(u_{proj} \times q_{proj})}{u_{proj}^T q_{proj}}\right). \qquad (6.7)$$

We now determine ϕ from the law of cosines: referring to the inset of Figure 6.7,

$$\|r_{proj}\|^2 + \|v_{proj}\|^2 - 2\|r_{proj}\| \cdot \|v_{proj}\| \cos\phi = \|z_{proj}\|^2.$$
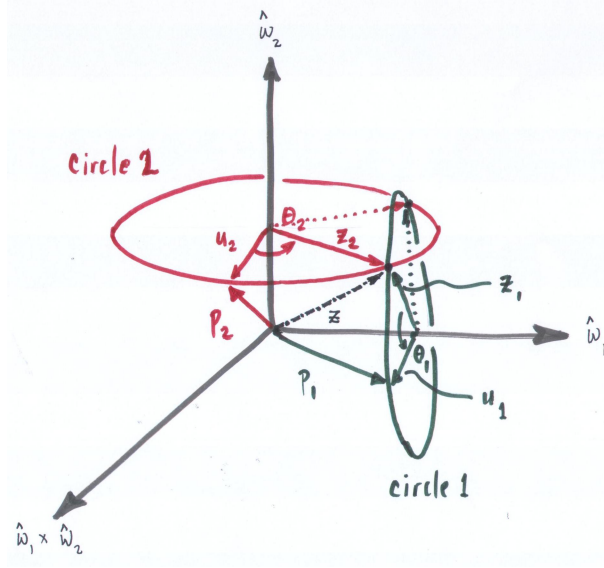
Figure 6.8: Solving $e^{[\omega_1]\theta_1}e^{[\omega_2]\theta_2}p_2 = p_1$ for θ_1 and θ_2.

Since $\|z_{proj}\|^2 = c^2 - (p^T\omega)^2$ is known, and $\|v_{proj}\| = \|u_{proj}\|$ is also known, ϕ can be determined as follows:

$$\phi = \cos^{-1}\left(\frac{\|r_{proj}\|^2 + \|v_{proj}\|^2 - \|z_{proj}\|^2}{2\|r_{proj}\| \cdot \|v_{proj}\|}\right). \tag{6.8}$$

From the figure it should be apparent that there can be up to two solutions for θ:

$$\theta = \psi \pm \phi.$$

If $\phi = 0$, the two solutions collapse to a single solution, while a solution does not exist in the event that ϕ does not exist.

Step 2: Solve for θ_1 and θ_2

Having solved for θ_3, Equation (6.3) can be written

$$e^{[S_1]\theta_1}e^{[S_2]\theta_2}p_2 = p_1, \tag{6.9}$$

where $p_2 = e^{[S_3]\theta_3}r_w$ and p_1 are both known. Observe that $S_1 = (\omega_1, 0)$ and $S_2 = (\omega_2, 0)$ are both pure rotations, and ω_1 and ω_2 are orthogonal to each other. Thus, only the rotation component of (6.9) needs to be considered:

$$e^{[\omega_1]\theta_1}e^{[\omega_2]\theta_2}p_2 = p_1. \tag{6.10}$$

Referring to Figure ??, clearly a necessary condition for a solution (θ_1, θ_2) to exist is that $\|p_1\| = \|p_2\|$. Assuming this is the case, the solutions are then

marked by the intersection of the two circles indicated in Figure 6.8. In the general case there can be up to two solutions, with one or no solution also a possibility.

Assuming a solution exists, let $z \in \mathbb{R}^3$ be the vector p_2 rotated about $\hat{\omega}_2$ by angle θ_2. z can also be obtained by rotating p_1 about ω_1 by an angle $-\theta_1$. Mathematically,

$$z = e^{[\omega_2]\theta_2} p_2 = e^{-[\omega_1]\theta_1} p_1.$$

Clearly $\{\omega_1, \omega_2, \omega_1 \times \omega_2\}$ forms an orthonormal basis for $\mathbb{R}^3$. Further observe that the ω_1 component of z is the same as the ω_1 component of p_1, and the ω_2 component of z is the same as the ω_2 component of p_2). z can therefore be expressed in terms of this orthonormal basis as

$$z = (p_1^T \omega_1)\omega_1 + (p_2^T \omega_2)\omega_2 \pm c(\omega_1 \times \omega_2)$$

for some scalar constant $c \geq 0$. The length $\|z\|$ is then, straightforwardly,

$$\|z\| = (p_1^T \omega_1)^2 + (p_2^T \omega_2)^2 + c^2.$$

Since z is also a rotated version of p_2 (and also of p_1), it follows that $\|z\| = \|p_2\| = \|p_1\|$. Solving the above for c^2, z can now be written

$$z = (p_1^T \omega_1)\omega_1 + (p_2^T \omega_2)\omega_2 \pm \sqrt{\|p_2\|^2 - (p_1^T \omega_1)^2 - (p_2^T \omega_2)}(\omega_1 \times \omega_2).$$

If $c = 0$ then there exists a unique solution (θ_1, θ_2), while if c does not exist, then no solution (θ_1, θ_2) exists.

Having found up to two possible solutions for z, what remains is to find θ_1 and θ_2 for each z. This is relatively straightforward: letting u_1 and z_1 respectively be the projections of p_1 and z onto circle 1, and u_2 and z_2 respectively be the projections of p_2 and z onto circle 2, it follows that

$$
\begin{aligned}
\theta_1 &= \cos^{-1}(u_1^T z_1) \\
\theta_2 &= \cos^{-1}(u_2^T z_2).
\end{aligned}
$$

Step 3: Solve for θ_4, θ_5, and θ_6

Having found θ_1, θ_2, and θ_3, it remains to solve for θ_4, θ_5, and θ_6. We have

$$
\begin{aligned}
e^{[S_4]\theta_4} e^{[S_5]\theta_5} e^{[S_6]\theta_6} &= e^{-[S_3]\theta_3} e^{-[S_2]\theta_2} e^{-[S_1]\theta_1} X M^{-1} \qquad (6.11) \\
&= X_2,
\end{aligned}
$$

where the right-side X_2 is now known. Recall that $\omega_4^T \omega_5 = 0$ and $\omega_5^T \omega_6 = 0$; this implies that ω_4 and ω_6 are either orthogonal or parallel. Assume for the time being that ω_6 is orthogonal to ω_4, or more precisely, $\omega_6 = \omega_4 \times \omega_5$ (the parallel case will be considered later). Define the transformation

$$T_w = \begin{bmatrix} R_w & r_w \\ 0 & 1 \end{bmatrix}, \quad R_w = \begin{bmatrix} \omega_6 & -\omega_5 & \omega_4 \end{bmatrix} \in SO(3).$$

Multiplying both sides of (6.11) by T_w^{-1} leads to

$$
\begin{aligned}
T_w^{-1}e^{[S_4]\theta_4}e^{[S_5]\theta_5}e^{[S_6]\theta_6} &= T_w^{-1}X_2 \\
e^{T_w^{-1}[S_4]T_w\theta_4}e^{T_w^{-1}[S_5]T_w\theta_5}e^{T_w^{-1}[S_6]T_w\theta_6} &= T_w^{-1}X_2T_w.
\end{aligned}
$$

Noting that $T_w^{-1}[S_i]T_w$ is the 4×4 matrix representation of the adjoint mapping $Ad_{T_w^{-1}}(S_i)$, it can be verified by calculation that

$$
\begin{aligned}
Ad_{T_w^{-1}}(S_6) &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\[6pt]
Ad_{T_w^{-1}}(S_5) &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\[6pt]
Ad_{T_w^{-1}}(S_4) &= \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.
\end{aligned}
$$

As long as the translational component of $T_w^{-1}X_1T_w$ is zero, the solution $(\theta_4, \theta_5, \theta_6)$ can now be obtained from the following:

$$
\mathrm{Rot}(\hat{z}, \theta_1) \cdot \mathrm{Rot}(\hat{y}, \theta_2) \cdot \mathrm{Rot}(\hat{x}, \theta_3) = \bar{R},
$$

where $\bar{R}$ is the rotational component of $T_w^{-1}X_1T_w$. The solution $(\theta_4, \theta_5, \theta_6)$ therefore corresponds to the ZYX Euler angles that parametrize $\bar{R}$:

$$
\begin{aligned}
\theta_5 &= \mathrm{atan2}(-\bar{r}_{31}, \pm\sqrt{\bar{r}_11^2 + \bar{r}_{21}^2}) \\
\theta_4 &= \mathrm{atan2}(\bar{r}_{21}, \bar{r}_{11}) \\
\theta_6 &= \mathrm{atan2}(\bar{r}_{32}, \bar{r}_{33}),
\end{aligned}
$$

where $\bar{r}_{ij}$ denotes the ij-th entry of $\bar{R}$. Recall that there are two solutions determined by the choice of sign for θ_5.

The above derivation can be repeated for the case when ω_6 and ω_4 are parallel, in which case the solutions $(\theta_4, \theta_5, \theta_6)$ will now correspond to the ZYZ Euler angles.

Like the earlier 6R PUMA-type arm, since there are up to two possible solutions for θ_3, and also for (θ_1, θ_2) and $(\theta_4, \theta_5, \theta_6)$, for the generalized PUMA-type robot arm one can expect up to $2 \times 2 \times 2 = 8$ possible inverse kinematic solutions.

6.1.3 Stanford-Type Arms

If the elbow joint in a 6R PUMA-type arm is replaced by a prismatic joint as shown in Figure 6.9, we then have a Stanford-type arm. Here we consider
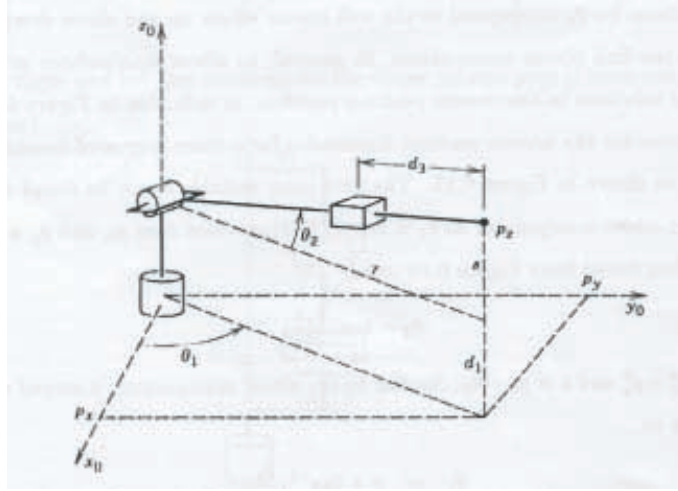
Figure 6.9: The first three joints of a Stanford-type arm.

the inverse position kinematics for the arm of Figure 6.9; the inverse orientation kinematics is identical to that for the PUMA-type arm and is not repeated here.

The first joint variable θ_1 an be found in a similar fashion to the PUMA-type arm: $\theta_1 = \tan^{-1}(\frac{p_y}{p_x})$ (provided that p_x and p_y are not both zero). θ_2 is then found from Figure 6.9 to be

$$\theta_2 = \tan^{-1}(\frac{s}{r})$$

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 θ_1, θ_2 is given by

$$\theta_1 = \pi + \tan^{-1}(\frac{p_y}{p_x})$$
$$\theta_2 = \pi - \tan^{-1}(\frac{p_y}{p_x})$$

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 generalized PUMA-type arm; the only difference occurs in the first step (obtaining θ_3). The screw vector for the third joint now becomes $\mathcal{S}_3 = (0, v_3)$, with $\|v_3\| = 1$, and θ_3 is found by solving the equation

$$\|e^{[\mathcal{S}_3]\theta_3}p\| = c,$$

for some given $p \in \mathbb{R}^3$ and nonnegative positive scalar c. The above equation reduces to solving the following quadratic in θ_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

In cases where the inverse kinematics equations do not admit analytic solutions, one must resort to numerical methods. Even in cases where an analytic solution exists, numerical methods are often used to improve the accuracy of these solutions. For example, in a generalized PUMA-type arm, the last three axes may not exactly intersect at a common point, and the shoulder joint axes may not be exactly orthogonal. In such cases, rather than throw away any analytic inverse kinematic solutions that are available, such solutions can be used as the initial point in an iterative procedure for numerically 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. 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 twice differentiable, and let x_d be the desired end-effector coordinates. The goal is to find joint coordinates θ_d such that

$$x_d - f(\theta_d) = 0.$$

Solving this equation for θ_d is a nonlinear root-finding problem.

Given an initial guess θ_0 which is "close by" a solution θ_d, the kinematics can be approximated by the Taylor expansion truncated at first-order

$$x_d = f(\theta_d) \approx f(\theta_0) + \underbrace{\left.\frac{\partial f}{\partial x}\right|_{\theta_0}}_{J(\theta_0)} \underbrace{(\theta_d - \theta_0)}_{\Delta\theta}, \tag{6.12}$$
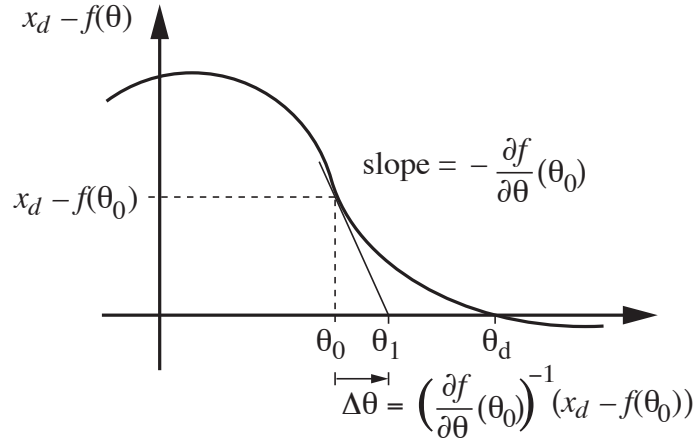
Figure 6.10: The first step of the Newton-Raphson method for nonlinear root-finding for a scalar x and θ.

where $J(\theta_0) \in \mathbb{R}^{m \times n}$ is the coordinate Jacobian evaluated at θ_0. Assuming $J(\theta_0)$ is square ($m = n$) and invertible, we can solve for $\Delta\theta$ as

$$\Delta\theta = J^{-1}(\theta_0)\,(x_d - f(\theta_0))\,. \tag{6.13}$$

If the kinematics were linear, then the new guess $\theta_1 = \theta_0 + \Delta\theta$ would exactly satisfy $x_d = f(\theta_1)$. If not, the new guess θ_1 should be closer to the root than θ_0, and the process is then repeated, with the sequence $\{\theta_0, \theta_1, \theta_2, \dots\}$ converging to θ_d (Figure 6.10).

If J is not square, then J^{-1} does not exist. The Moore-Penrose pseudoinverse $J^\dagger$ is used in place of J^{-1}, so Equation (6.13) becomes

$$\Delta\theta = J^\dagger(\theta_0)\,(x_d - f(\theta_0))\,, \tag{6.14}$$

where

$$J^\dagger = J^T(JJ^T)^{-1} \quad \text{if } n > m \text{ and } JJ^T \text{ is invertible}$$
$$J^\dagger = (J^TJ)^{-1}J^T \quad \text{if } n < m \text{ and } J^TJ \text{ is invertible.}$$

If $n < m$, then there may not exist a $\Delta\theta$ exactly solving $J(\theta)\Delta\theta = x_d - f(\theta_0)$. In this case, the solution of Equation (6.14) using the pseudoinverse is a $\Delta\theta$ that best satisfies $J(\theta)\Delta\theta = x_d - f(\theta_0)$ in the least-squares sense. If $n > m$, then the robot may be *redundant*, meaning that there may exist an $(n-m)$-dimensional set of solutions $\Delta\theta$ to $J(\theta)\Delta\theta = x_d - f(\theta_0)$. In this case, the the solution of Equation (6.14) is a $\Delta\theta$ that exactly satisfies $J(\theta)\Delta\theta = x_d - f(\theta_0)$ while also minimizing the two-norm $\sqrt{\Delta\theta^T \Delta\theta}$.

Equation (6.14) suggests the Newton-Raphson iterative algorithm for finding θ_d:

(i) **Initialization**: Given $x_d \in \mathbb{R}^m$ and an initial guess $\theta_0 \in \mathbb{R}^n$. Set $i = 0$.

(ii) While $\|x_d - f(\theta_i)\| > \epsilon$ for some small ϵ:

- Set $\theta_{i+1} = \theta_i + J^\dagger(\theta_i)(x_d - f(\theta_i))$.
- 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. Note, however, that the vector $x_d - f(\theta_i)$, representing the direction from the current guess (evaluated through the forward kinematics) to the desired end-effector configuration, cannot simply be replaced with $T_{sd} - T_{sb}(\theta_i)$, which is not even an element of $SE(3)$. Instead we should think of $x_d - f(\theta_i)$ 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 velocity $\mathcal{V}_b$ which, if followed for unit time, would cause a motion from $T_{sb}(\theta_i)$ to the desired configuration T_{sd}.

To find this $\mathcal{V}_b$, we first calculate the desired configuration represented in the body frame,

$$T_{bd}(\theta_i) = T_{sb}^{-1}(\theta_i)T_{sd} = T_{bs}(\theta_i)T_{sd}.$$

Then $\mathcal{V}_b$ is determined using the matrix logarithm,

$$[\mathcal{V}_b] = \log T_{bd}(\theta_i).$$

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 $[\mathcal{V}_b] = \log\left(T_{sb}^{-1}(\theta_i)T_{sd}\right)$. While $\|\omega_b\| > \epsilon_\omega$ or $\|v_b\| > \epsilon_v$ for small $\epsilon_\omega, \epsilon_v$:

- Set $\theta_{i+1} = \theta_i + J_b^\dagger(\theta_i)\mathcal{V}_b$.
- Increment i.

An equivalent form can be derived in the space frame, using the space Jacobian $J_s(\theta)$ and the space velocity $\mathcal{V}_s = [\mathrm{Ad}_{T_{sb}}]\mathcal{V}_b$.

For this numerical inverse kinematics method to converge, the initial guess θ_0 should be sufficiently close to a solution θ_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 θ_d at the previous timestep serves as the initial guess θ_0 for the new T_{sd} at the next timestep.

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 is to calculate the required joint velocities directly from the relationship $J\dot{\theta} = \mathcal{V}_d$, where the desired end-effector velocity $\mathcal{V}_d$ and J are expressed with respect to the same frame, without calculating the inverse kinematics:

$$\dot{\theta} = J^\dagger(\theta)\mathcal{V}_d. \tag{6.15}$$

In this case, a position feedback controller should be designed to choose $\mathcal{V}_d$ to keep the end-effector following $T_{sd}(t)$.

In the case of a redundant robot with $n > 6$ joints, of the $(n-6)$-dimensional set of solutions satisfying Equation (6.15), the use of the Moore-Penrose pseudoinverse $J^\dagger(\theta)$ returns the $\dot{\theta}$ minimizing the two-norm $\|\dot{\theta}\|_2 = \sqrt{\dot{\theta}^T\dot{\theta}}$. This is appealing; the motion is the minimum joint velocity motion that achieves the desired end-effector velocity.

Another option is 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 configuration-dependent inertia matrix of the robot. In this case, we might wish to choose the joint velocities $\dot{\theta}$ to minimize the kinetic energy of the robot (e.g., minimize the velocities of the joints moving a lot of mass), while still statisfying the desired end-effector velocity. To do this, we can choose to use the weighted Moore-Penrose pseudoniverse

$$J^\dagger = MJ^T(JMJ^T)^{-1},$$

which yields a solution minimizing the weighted norm $\sqrt{\dot{\theta}^T M\dot{\theta}}$.

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 θ 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, $n \leq 6$ 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 a large class of open chains that possess some degree of structure. One important class is the generalized $6R$ PUMA-type arm; such an arm consists of a $3R$ orthogonal axis wrist connected to a $2R$ orthogonal axis shoulder by an elbow joint. Geometric algorithms have been developed for this class of arms that exploits the product of exponentials forward kinematics representation. Further assumptions on the joint axes, e.g., joint axes 2 and 3 are always parallel, lead to simpler analytic forms for the inverse kinematics.

- 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 also been developed.

- Iterative numerical methods are used in cases where analytic inverse kinematics solutions are not available. These typically involve solving the inverse kinematics equations through an iterative procedure like the Newton-Raphson method, and 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 point, and only one inverse kinematic solution is produced per iteration. An iterative procedure based on the Jacobian of the forward kinematics has been developed for general six degree-of-freedom spatial open chains.

6.6 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 [15], and also by Raghavan and Roth [33]. Iterative numerical procedures for finding all sixteen solutions of a general $6R$ open chain are reported in [18]. A summary of inverse kinematics methods for kinematically redundant robot arms are discussed in [36]. The repeatability conditions for kinematically redundant inverse kinematics schemes are examined in [35].
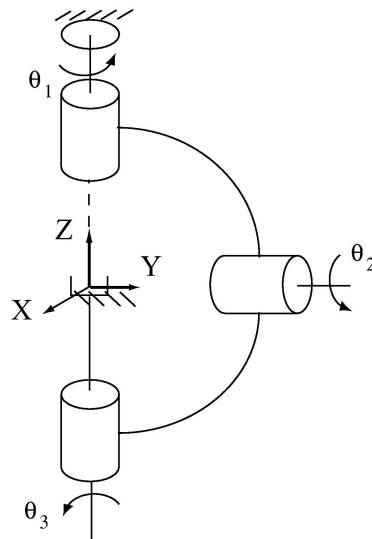
Figure 6.11: A 3R wrist.

6.7   Exercises

1. The $3R$ orthogonal axis wrist mechanism of Figure 6.11 is shown in its zero position, with joint axes 1 and 3 collinear.
(a) Ignoring position and considering only the orientation of the end-effector frame, find all kinematic singularities of the wrist mechanism.
(b) Given a desired wrist orientation $R \in SO(3)$, derive an iterative numerical procedure for solving its inverse kinematics.
(c) Assuming static equilibrium, suppose that at the zero position we wish to generate a moment $(0, 1, 0)$ (in fixed frame coordinates) at the end-effector. What joint torques should be applied?

2. The 3R nonorthogonal chain of Figure 6.12 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 $(\theta_1, \theta_2, \theta_3)$.

3. The RRP open chain of Figure 6.13 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 is its zero position.
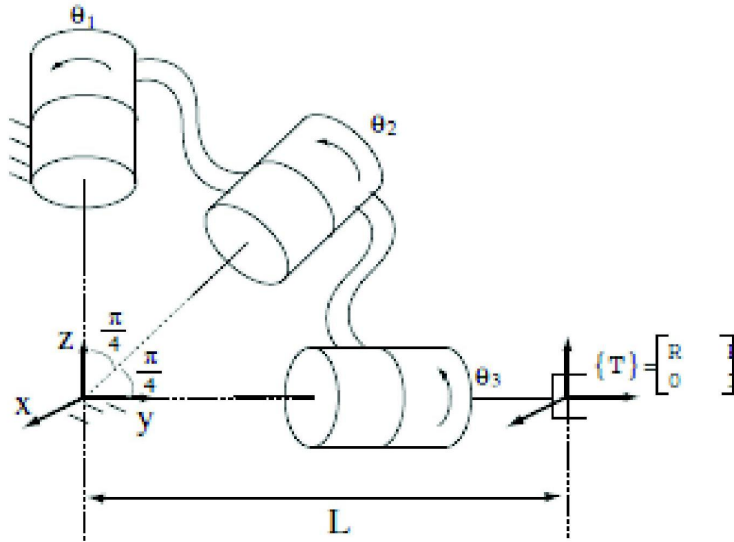
Figure 6.12: A 3R nonorthogonal chain.
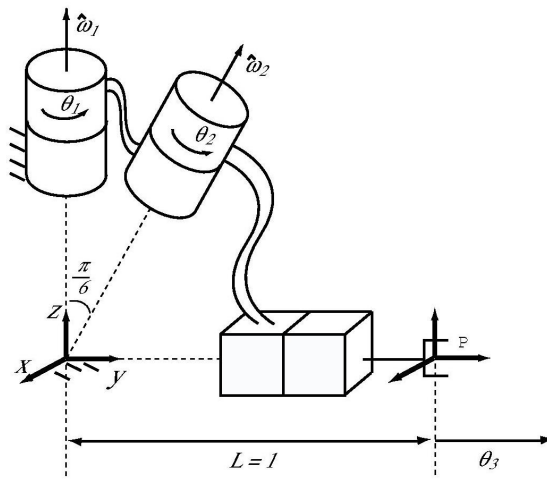
Figure 6.13: An RRP open chain.

(a) Suppose $\theta_1 = 0$. Solve for θ_2 and θ_3 when the end-effector frame origin p is 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 part (a).
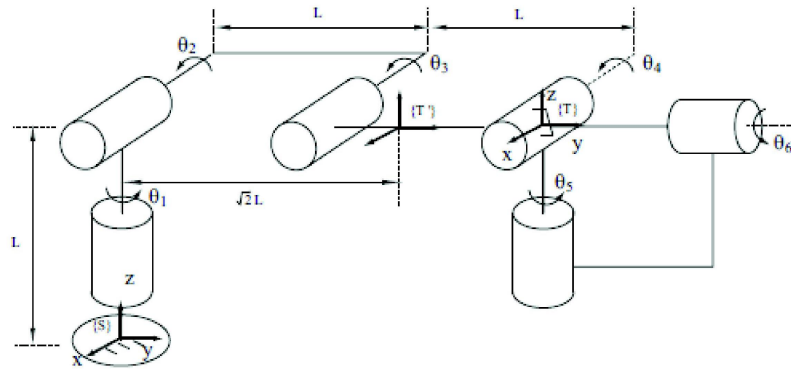
Figure 6.14: A 6R open chain of the elbow type.
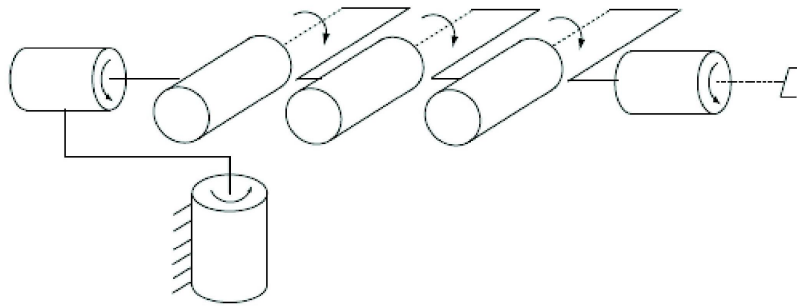
Figure 6.15: An inverse elbow type robot arm.

4. Find the inverse kinematics solutions when tool frame $\{T\}$ of the 6R open chain of the elbow type shown in Figure 6.14 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 y-axis.

5. (a) Solve the inverse position kinematics (you do not need to solve the orientation kinematics) of the inverse elbow type robot arm shown in Figure 6.15.
(b) Determine the spatial Jacobian $J_s(theta)$ of the inverse elbow robot arm.
(c) Find as many kinematic singularities of the inverse elbow arm that you can, and for each singularity, describe the directions in which the end-effector loses degrees of freedom of motion.

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. In this chapter we shall analyze the kinematics of closed chains, paying special attention to a class of closed chains that we shall refer to as **parallel mechanisms**; these are closed chains consisting of a fixed and moving platform connected by a set of "legs"; these legs are mostly open chains, but sometimes can themselves be closed chains.

Figures 7.1-7.3 depict some well-known parallel mechanisms. The Stewart-Gough Platform is a six degree of freedom mechanism, used widely as both a motion simulator and six-axis force-torque sensor. It is typically realized as either a $6 \times UPS$ or $6 \times SPS$ platform; note that the additional torsional rotations of each of the six legs in the $6 \times SPS$ platform have no effect on the moving platform. 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 degree of freedom mechanism that has the unusual feature of the moving platform always remaining parallel to the fixed platform. Because the three actuators are all attached to the three revolute joints of the fixed upper platform, the moving parts are relatively light; this allows the Delta to achieve very fast motions. The Eclipse mechanism is another six degree of freedom parallel mechanism whose moving platform is capable of $\pm 90°$ orientations with respect to ground, and also of rotating $360°$ about the vertical axis.

Closed chains admit a much greater variety of designs than open chains, and not surprisingly their kinematic analysis is considerably more complicated. This can be traced to two defining features of closed chains: (i) the configuration space is curved (e.g., a multidimensional surface embedded in a higher-
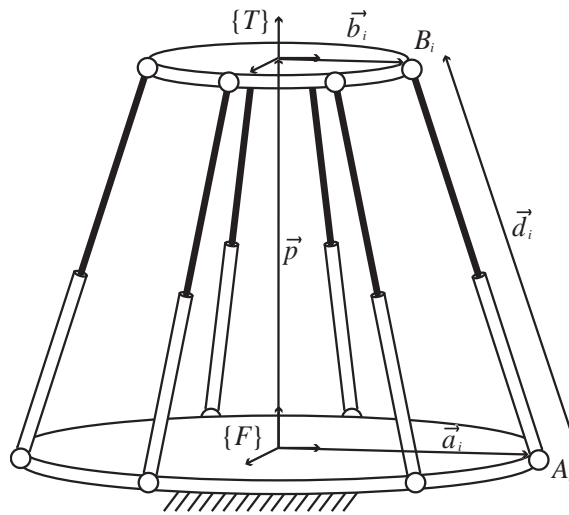
Figure 7.1: The Stewart-Gough platform.
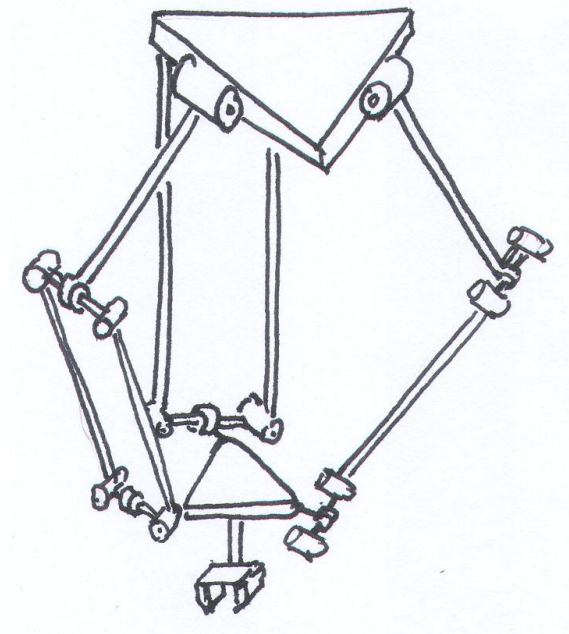
Figure 7.2: The Delta robot.

dimensional vector space), and (ii) not all of the joints are actuated. The presence of such non-actuated, or passive joints, together with the fact that the number of actuated joints may deliberately exceed the mechanism's kinematic
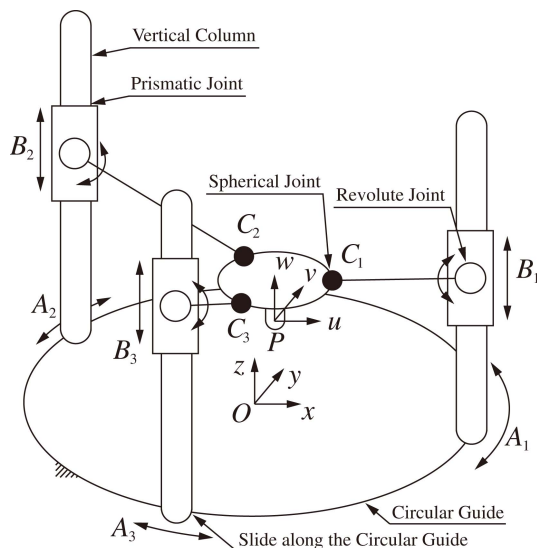
Figure 7.3: The Eclipse mechanism.

degrees of freedom—such mechanisms are said to be **redundantly actuated**—makes not only the position and differential kinematics analysis more challenging, but also introduces new types of singularities not witnessed in open chains.

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 most effective approaches for closed chain kinematic analysis are based on a collection of tools and methodologies that exploit as much as possible any kinematic symmetries and other special features of the mechanism.

For this reason we shall proceed in this chapter with a series of case studies involving some well-known parallel mechanisms, and eventually build up a repetoire of kinematic analysis tools and methodologies that can be synthesized to handle more general closed chains. We shall consider only parallel mechanisms that are exactly actuated, i.e., the number of actuated degrees of freedom is equal to the mechanism's kinematic mobility. 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 an examination of the various kinematic singularities that can arise in closed chains.
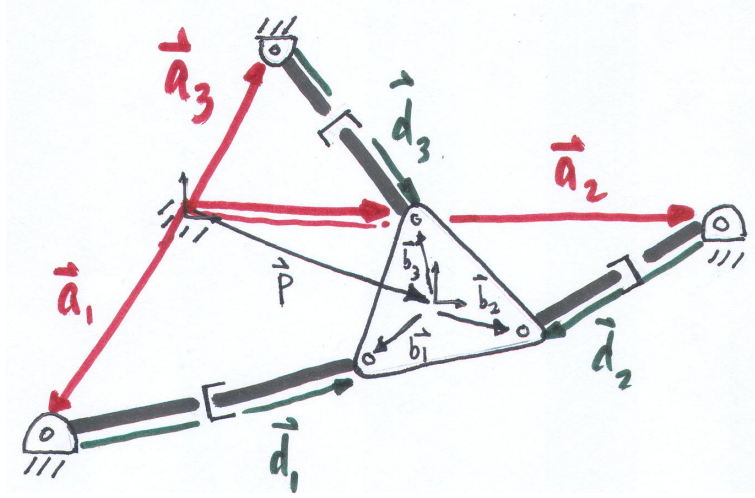
Figure 7.4: A three degree-of-freedom $3 \times RPR$ planar parallel mechanism.

7.1 Inverse and Forward Kinematics

This section examines methods for the inverse and forward kinematics of closed chains. Rather than attempt to develop a general methodology applicable to all types of closed chains, we consider two case studies, the $3 \times RPR$ planar parallel mechanism, and its spatial counterpart, the $3 \times SPS$ Stewart-Gough platform. The analysis of these two mechanisms draws upon some reduction techniques that result in a reduced form of the governing kinematic equations. We briefly describe how these methods can be generalized to the analysis of more general parallel mechanisms.

7.1.1 $3 \times$ RPR Planar Parallel Mechanism

The first example we consider is the planar $3 \times RPR$ parallel mechanism shown in Figure 7.4. It is easily verified from the planar version of Gruebler's formula that this mechanism has mobility three. Assign a fixed frame $\{s\}$ and end-effector frame $\{b\}$ as shown. Typically the three prismatic joints are actuated; 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 end-effector frame's position and orientation.

Let $\vec{p}$ be the vector from the origin of the $\{s\}$ frame to the origin of the $\{b\}$ frame. Let ϕ denote the angle measured from the $\hat{x}$ axis of the $\{s\}$ frame to the $\{x\}$ axis of the $\{b\}$ frame. Further define the vectors $\vec{a}_i$, $\vec{b}_i$, $\vec{d}_i$, $i = 1, 2, 3$ as shown in the figure. From these definitions, clearly

$$\vec{d}_i = \vec{p} + \vec{b}_i - \vec{a}_i, \tag{7.1}$$

for $i = 1, 2, 3$. Let

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \vec{p} \text{ in } \{s\} \text{ frame coordinates}$$

$$\begin{bmatrix} a_{ix} \\ a_{iy} \end{bmatrix} = \vec{a}_i \text{ in } \{s\} \text{ frame coordinates}$$

$$\begin{bmatrix} d_{ix} \\ d_{iy} \end{bmatrix} = \vec{d}_i \text{ in } \{s\} \text{ frame coordinates}$$

$$\begin{bmatrix} b_{ix} \\ b_{iy} \end{bmatrix} = \vec{b}_i \text{ in } \{b\} \text{ frame coordinates.}$$

Note that the vectors (a_{ix}, a_{iy}), (b_{ix}, b_{iy}), $i = 1, 2, 3$ are constant, and that with the exception of (b_{ix}, b_{iy}), all other vectors are expressed in $\{s\}$ frame coordinates. To express Equation (7.1) in terms of $\{s\}$ frame coordinates, it is first necessary to find the $\{s\}$ frame representation of the vector $\vec{b}_i$. This is straightforward: defining

$$R_{sb} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix},$$

it now follows that

$$\begin{bmatrix} d_{ix} \\ d_{iy} \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} b_{ix} \\ b_{iy} \end{bmatrix} - \begin{bmatrix} a_{ix} \\ a_{iy} \end{bmatrix},$$

for $i = 1, 2, 3$. Also, since $s_i^2 = d_{ix}^2 + d_{iy}^2$, we have

$$\begin{aligned} s_i^2 &= (p_x + b_{ix}\cos\phi - b_{iy}\sin\phi - a_{ix})^2 \\ &\quad + (p_y + b_{ix}\sin\phi + b_{iy}\cos\phi - a_{iy})^2, \end{aligned}$$

for $i = 1, 2, 3$.

Formulated as above, the inverse kinematics is trivial to compute: given values for (p_x, p_y, ϕ), the leg lengths (s_1, s_2, s_3) can be directly calculated from the above equations (negative values of s_i in most cases will not be physically realizable, and can be ignored). The forward kinematics problem, in contrast, is not trivial: here the objective is to determine, for given values of (s_1, s_2, s_3), the end-effector frame's position and orientation (p_x, p_y, ϕ). The following tangent half-angle substitution, widely used in kinematic analysis, transforms the above three equations 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 considerable algebraic manipulation, this system of polynomials can eventually be reduced to a single sixth-order polynomial in t, which effectively shows that the $3 \times RPR$ mechanism may have up to six forward kinematics solutions (showing that six real solutions are possible requires further verification, which we do not pursue here).

7.1.2 Stewart-Gough Platform

We now examine the inverse and forward kinematics of the $6 \times SPS$ Stewart-Gough platform of Figure 7.1. 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 our earlier planar $3 \times RPR$ mechanism. Let $\{s\}$ and $\{b\}$ denote the fixed and end-effector frames, respectively, and let $\vec{d_i}$ be the vector directed from joint A_i to joint B_i. Referring to Figure 7.1, we make the following definitions:

- $p \in \mathbb{R}^3 = \vec{p}$ in $\{s\}$ frame coordinates;

- $a_i \in \mathbb{R}^3 = \vec{a_i}$ in $\{s\}$ frame coordinates;

- $b_i \in \mathbb{R}^3 = \vec{b_i}$ in $\{b\}$ frame coordinates;

- $d_i \in \mathbb{R}^3 = \vec{d_i}$ in $\{s\}$ frame coordinates.

- $R \in SO(3) =$ orientation of $\{b\}$ as seen from $\{s\}$.

In order to derive the kinematic constraint equations, note that vectorially,

$$\vec{d_i} = \vec{p} + \vec{b_i} - \vec{a_i}, \ \ i = 1, \ldots, 6.$$

Writing the above equations explicitly in $\{s\}$ frame coordinates,

$$d_i = p + Rb_i - a_i, \ \ i = 1, \ldots, 6.$$

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 $i = 1, \ldots, 6$. Observe that a_i and b_i as defined above are all known constant vectors. Having written the constraint equations in this form, the inverse kinematics now 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. Here we are given each of the leg lengths s_i, $i = 1, \ldots, 6$, and must solve for $p \in \mathbb{R}^3$ and $R \in SO(3)$. The six constraint equations, together with the rotation matrix constraint $R^T R = I$, constitute a set of twelve equations in twelve unknowns. Several methods exist for finding all solutions to such a set of polynomial equations, e.g., methods
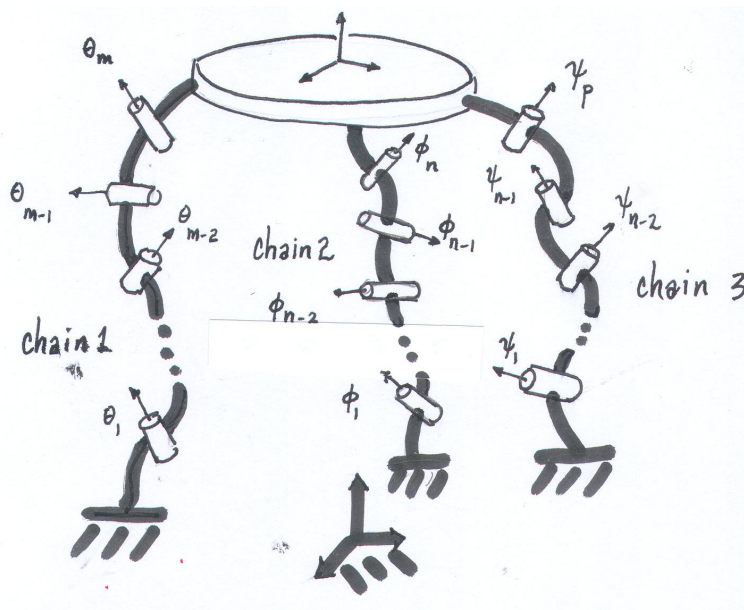
Figure 7.5: A general spatial parallel mechanism.

based on dialytic elimination, Grobner bases, etc. Of particular note is the work of Raghavan and Roth [**?**], who show that there can be at most forty solutions to the forward kinematics, and Husty [**?**], who develops a computational algorithm for finding all forty solutions analytically.

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, in the case of the Stewart-Gough Platform, the fact that each of the "legs" can be modelled as straight lines considerably simplified the analysis. In this brief section we consider the more general case where the legs have the structure of an arbitrary open chain.

Consider such a parallel mechanism as shown in Figure 7.5; here the fixed and moving platforms are connected by three open chains. 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

$$T_1(\theta) \quad = \quad T_2(\phi) \tag{7.2}$$
$$T_2(\phi) \quad = \quad T_3(\psi). \tag{7.3}$$

Equation 7.2 and 7.3 each consists of 12 equations (9 for the rotation component and 3 for the position component), 6 of which are independent (recall that the nine equations for the rotation component can be reduced to a set of three

independent equations from the rotation matrix constraint, i.e., $R^T R = I$); there are thus 24 equations, 12 of which are independent, with $n + m + p$ unknown variables, and the mobility of the mechanism is $d = 12 - (n + m + p)$.

In the forward kinematics problem, given values for d of the joint variables (θ, ϕ, ψ), Equations 7.2 and 7.2 can be solved for the remaining joint variables; note that multiple solutions will be 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, we are given the end-effector frame displacement $T \in SE(3)$. Setting $T = T_1 = T_2 = T_3$, the objective is to solve Equations 7.2 and 7.2 for all the joint variables (θ, ϕ, ψ). As hinted 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 to simplify them into a reduced form.

7.2 Differential Kinematics

We now consider the differential kinematics of parallel mechanisms. Unlike differential kinematics for open chains, in which the objective was to relate the input joint velocities to the spatial velocity 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 spatial velocity of the closed chain's end-effector frame.

For open chains, the Jacobian of the forward kinematics played a defining role in both velocity and static analysis. For closed chains, in addition to the forward kinematics Jacobian, the Jacobian defined by the kinematic constraint equations—for this reason we refer to this latter Jacobian as the **constraint Jacobian**—also plays a central role in velocity and static analysis. Much like the case for the inverse and forward kinematic analysis of parallel mechanisms, often there are features of the mechanism that can be exploited to simplify and reduce the procedure for obtaining the Jacobians. We therefore begin 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 end-effector 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 this equation can be differentiated and manipulated to eventually produce a differential version, e.g.,

$$\dot{s} = G(R, p)V_s, \tag{7.4}$$

where $\dot{s} \in \mathbb{R}^6$ denotes the leg velocities, $V_s \in \mathbb{R}^6$ is the end-effector's spatial velocity in fixed frame coordinates, and $G(R, p) \in \mathbb{R}^{6\times6}$ is the Jacobian of the inverse kinematics. This derivation, however, will likely involve considerable algebraic manipulation.

Here we take a different approach based on static analysis. Based on the same virtual work considerations that were used to determine the static relationship for open chains, the static relationship for closed chains (expressed in the fixed frame) is also given by $\tau = J_s^T F_s$, where τ is the vector of input joint torques, F_s is (the fixed frame representation of) the external spatial force applied at the end-effector frame, and J_s denotes the space Jacobian of the forward kinematics.

For the Stewart-Gough platform, note that the only forces being applied to the moving platform occur at the spherical joints. Let

$$f_i = \omega_i \tau_i$$

be the three-dimensional linear force applied by leg i, where $\omega_i \in \mathbb{R}^3$ is a unit vector indicating the direction of the applied force, and $\tau_i \in \mathbb{R}$ is the magnitude of the linear force; we emphasize that f_i is expressed in terms of the fixed frame coordinates. The moment generated by f_i, denoted m_i, is then given by

$$m_i = r_i \times f_i,$$

where $r_i \in \mathbb{R}^3$ denotes the vector from the fixed frame origin to the point of application of the force (spherical joint i in this case); again, both r_i and m_i are expressed in fixed frame coordinates. It is not too difficult to see that this same moment can also be expressed as

$$m_i = q_i \times f_i,$$

where $q_i \in \mathbb{R}^3$ denotes the vector from the fixed frame origin to the base of leg i, i.e., the joint connecting leg i to the fixed base. Expressing the moment as $q_i \times f_i$ is preferred, since q_i as defined is constant.

Combining f_i and m_i into a six-dimensional spatial force $F_i = (m_i, f_i)$, the resultant spatial force F_s on the moving platform is then given by

$$
\begin{aligned}
F_s &= \sum_{i=1}^{6} F_i = \sum_{i=1}^{6} \begin{bmatrix} r_i \times f_i \\ \omega_i \end{bmatrix} \tau_i \\
&= \begin{bmatrix} -\omega_1 \times q_1 & \cdots & -\omega_6 \times q_6 \\ \omega_1 & \cdots & \omega_6 \end{bmatrix} \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_6 \end{bmatrix}.
\end{aligned}
$$

Since earlier we asserted that the static relationship for the Stewart-Gough platform is also of the form $\tau = J_s^T F_s$, based on the previous derivation we can conclude that the inverse Jacobian J_s^{-1} (or equivalently, the Jacobian of the inverse kinematics) is given by

$$
J_s^{-1} = \begin{bmatrix} -\omega_1 \times q_1 & \cdots & -\omega_6 \times q_6 \\ \omega_1 & \cdots & \omega_6 \end{bmatrix}^T.
$$

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 a static analysis is not as straightforward. Using the previous three-legged, three degree-of-freedom spatial parallel mechanism of Figure 7.5 as an example, we illustrate a general procedure for determining the forward kinematics Jacobian; generalizing this method to arbitrary parallel mechanisms should be completely straightforward.

   The mechanism of Figure 7.5 consists of two platforms connected by three legs, with each leg a five degree of freedom open chain. For the given fixed and end-effector frames as indicated in the figure, we first write the forward kinematics for the three chains as follows:

$$
\begin{aligned}
T_1(\theta_1, \theta_2, \ldots, \theta_5) &= e^{[S_1]\theta_1} e^{[S_2]\theta_2} \cdots e^{[S_5]\theta_5} M_1 \\
T_2(\phi_1, \phi_2, \ldots, \phi_5) &= e^{[P_1]\phi_1} e^{[P_2]\phi_2} \cdots e^{[P_5]\phi_5} M_2 \\
T_3(\psi_1, \psi_2, \ldots, \psi_5) &= e^{[Q_1]\psi_1} e^{[Q_2]\psi_2} \cdots e^{[Q_5]\psi_5} M_3.
\end{aligned}
$$

The kinematic loop constraints can be expressed as

$$
\begin{aligned}
T_1(\theta) &= T_2(\phi) & (7.5) \\
T_2(\phi) &= T_3(\psi). & (7.6)
\end{aligned}
$$

Taking right differentials of both sides of the above two equations, we have

$$
\begin{aligned}
\dot{T}_1 T_1^{-1} &= \dot{T}_2 T_2^{-1} & (7.7) \\
\dot{T}_2 T_2^{-1} &= \dot{T}_3 T_3^{-1}. & (7.8)
\end{aligned}
$$

Since $\dot{T}_i T_i^{-1} = [V_i]$, where V_i is the spatial velocity of chain i's end-effector frame, the above identities can also be expressed in terms of the forward kinematics Jacobian for each chain:

$$
\begin{aligned}
J_1(\theta)\dot{\theta} &= J_2(\phi)\dot{\phi} & (7.9) \\
J_2(\phi)\dot{\phi} &= J_3(\psi)\dot{\psi}, & (7.10)
\end{aligned}
$$

which can also be rearranged as

$$
\begin{bmatrix} J_1(\theta) & -J_2(\phi) & 0 \\ 0 & -J_2(\phi) & J_3(\psi) \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = 0. \tag{7.11}
$$

   At this point we now rearrange the fifteen joints into those that are actuated, and those that are passive. Let us 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 (7.11) can now be rearranged into the form

$$\begin{bmatrix} H_a(q) & H_p(q) \end{bmatrix} \begin{bmatrix} \dot{q}_a \\ \dot{q}_p \end{bmatrix} = 0, \tag{7.12}$$

or equivalently

$$H_a \dot{q}_a + H_p \dot{q}_p = 0, \tag{7.13}$$

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

$$\dot{q}_p = -H_p^{-1} H_a \dot{q}_a. \tag{7.14}$$

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 7.14.

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 velocity 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 (7.14) we can write

$$\begin{aligned}
\dot{\theta}_2 &= g_2^T \dot{q}_a & (7.15) \\
\dot{\theta}_3 &= g_3^T \dot{q}_a & (7.16) \\
\dot{\theta}_4 &= g_4^T \dot{q}_a & (7.17) \\
\dot{\theta}_5 &= g_5^T \dot{q}_a & (7.18)
\end{aligned}$$

where each $g_i(q) \in \mathbb{R}^3$, $i = 2, \dots, 5$, can be obtained from Equation (7.14). Defining the row vector $e_1^T = (1, 0, 0)$, the differential forward kinematics for chain 1 can now be written

$$V_s = J_1(\theta) \begin{bmatrix} e_1^T \\ g_2^T \\ g_3^T \\ g_4^T \\ g_5^T \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\phi}_1 \\ \dot{\psi}_1 \end{bmatrix}. \tag{7.19}$$

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

$$J_a(q) = J_1(q_1, \dots, q_5) \begin{bmatrix} e_1^T \\ g_2(q)^T \\ g_3(q)^T \\ g_4(q)^T \\ g_5(q)^T \end{bmatrix}. \tag{7.20}$$
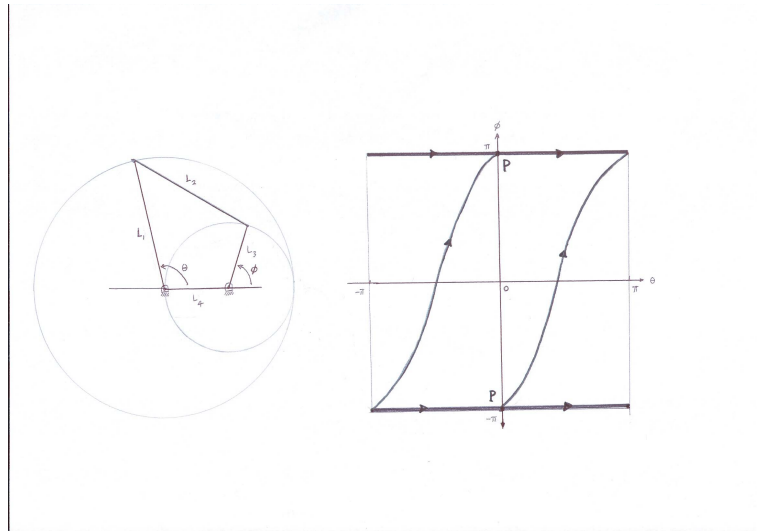
Figure 7.6: A planar four-bar linkage and its joint configuration space.
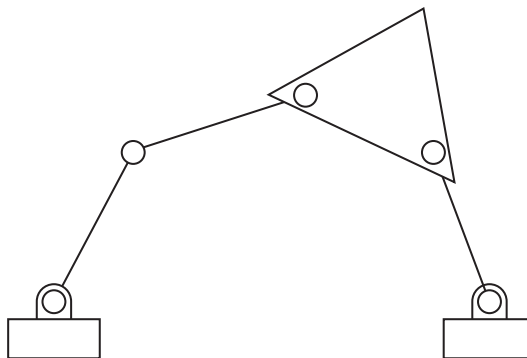
Figure 7.7: A planar five-bar linkage.

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 *a priori* will obviously simplify the task. 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.
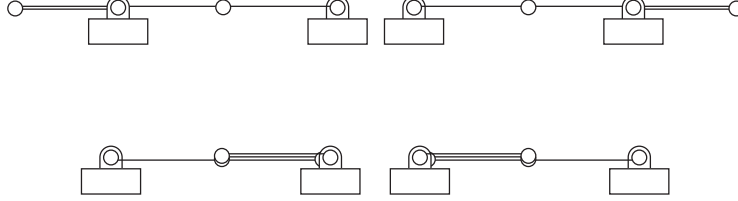
Figure 7.8: Configuration space singularities of the planar five-bar linkage.

7.3 Singularities

In this final section we shall examine the fundamental properties of closed chain singularities. Characterizing the singularities of closed chains involves many more subtleties than for open chains. Rather than attempt any such comprehensive classification for general closed chains, we instead choose to highlight the essential features of closed chain singularities via two planar examples: a four-bar linkage (see Figure 7.6), and a five-bar linkage (see Figure 7.7). The examples should also make clear how our approach to singularity analysis can be generalized to more complex closed chains.

We begin with the four-bar linkage. Recall that its configuration space is a curve embedded in a four-dimensional ambient space; even without appealing to equations, one can see that the allowable joint values for (θ, ϕ) of the four-bar form a curve of the type shown in Figure 7.6. In terms of the input and output angles θ and ϕ, the kinematic loop constraint equations can be expressed as

$$\phi = \tan^{-1} \frac{\beta}{\alpha} \pm \cos^{-1} \left(\frac{\gamma}{\sqrt{\alpha^2 + \beta^2}} \right), \tag{7.21}$$

where

$$\alpha = 2L_3 L_4 - 2L_1 L_3 \cos \theta \tag{7.22}$$

$$\beta = -2L_1 L_3 \sin \theta \tag{7.23}$$

$$\gamma = L_2^2 - L_4^2 - L_3^2 - L_1^2 + 2L_1 L_4 \cos \theta. \tag{7.24}$$

Obviously the existence and uniqueness of solutions depends on the link lengths $L_1, \ldots, L_4$; in particular, a solution fails to exist if $\gamma^2 \leq \alpha^2 + \beta^2$. The figure depicts the input-output graph for the choice of link lengths $L_1 = 4$, $L_2 = 4$, $L_3 = 3$, $L_4 = 2$; in this case both θ and ϕ can range from 0 to 2π.

One of the striking 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 joint configuration space does such a phenomenon occur.
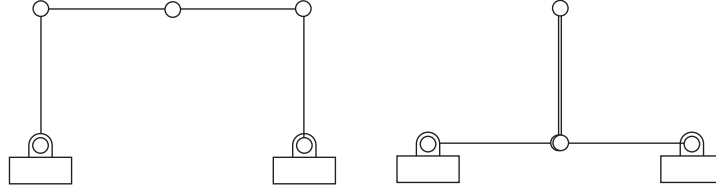
Figure 7.9: Actuator singularities of the planar five-bar linkage: the left is nondegenerate, while the right is degenerate.

.

We now turn to the five-bar linkage. The kinematic loop constraint equations can be written

$$L_1 \cos\theta_1 + \ldots + L_4 \cos(\theta_1 + \theta_2 + \theta_3 + \theta_4) = L_5 \qquad (7.25)$$
$$L_1 \sin\theta_1 + \ldots + L_4 \sin(\theta_1 + \theta_2 + \theta_3 + \theta_4) = 0 \qquad (7.26)$$

where we have eliminated joint variable θ_5 *a priori* 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 in 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 θ at which

$$\text{rank}\left(\frac{\partial f}{\partial \theta}(\theta)\right) < 2 \qquad (7.27)$$

corresponds to what we call a **configuration space singularity**. Figure 7.8 illustrates the possible configuration space singularities of the five-bar. Notice that thus far we have made no mention of which joints of the five-bar are actuated, or where the end-effector frame is placed; it is worth emphasizing that the notion of configuration space singularity is completely independent of the choice of actuated joints, or the end-effector frame.

We now consider the case when two joints of the five-bar are actuated. Referring to Figure 7.9, 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, rotating the two actuated joints in opposite directions will clearly have catastrophic consequences of the mechanism. For the **degenerate actuator singularity**

shown on the right, we have the opposite case: 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.

Intuitively visualizing the actuator singularities of the planar five-bar is straightforward enough, but for more complex spaatial 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:

$$\begin{bmatrix} H_a(q) & H_p(q) \end{bmatrix} \begin{bmatrix} \dot{q}_a \\ \dot{q}_p \end{bmatrix} = 0, \tag{7.28}$$

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 the matrix

$$H(q) = \begin{bmatrix} H_a(q) & H_p(q) \end{bmatrix} \in \mathbb{R}^{p \times (a+p)}, \tag{7.29}$$

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 shall 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 thus be regarded as the intersection of all possible actuator singularities obtained over all possible combinations of actuated joints.

The final class of singularities involves the choice of an end-effector frame. For the five-bar, we ignore the orientation of the end-effector frame, and focus exclusively on its x-y location. Figure 7.10 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. Note that end-effector singularities are entirely independent of the choice of actuated joints (note that it was not necessary to specify which, or even how many, of the joints are actuated).

End-effector singularities 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

$$f(q_a) = T \tag{7.30}$$

where T denotes the end-effector frame. 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.
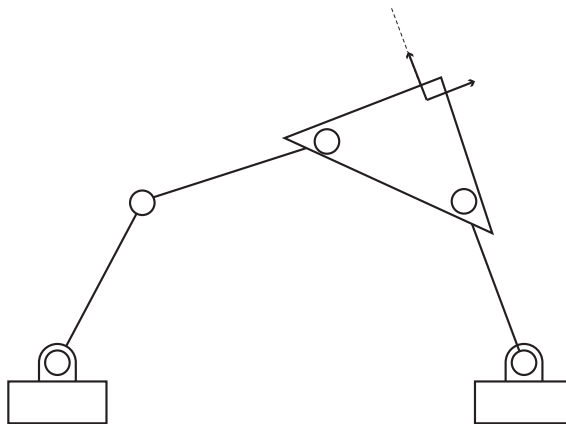
Figure 7.10: End-effector singularities of the planar five-bar linkage.

.

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 the configuration space is often curved, and only a subset of the joints are actuated.

- For a parallel mechanism whose actuated degrees of freedom equals its mobility, the inverse kinematics problem involves finding, from the given position and orientation of the moving platform, the values 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 whose actuated degrees of freedom equals its mobility, the forward kinematics problem involves finding, given values for all the actuated joints, the position and orientation of the moving platform. 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 degree of freedom joints, whose actuated degrees of freedom also equals its mobility m, 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 $g : \mathbb{R}^n \to \mathbb{R}^{n-m}$. The forward kinematics can be expressed in the form $f(\theta_a) = T$, where $f : \mathbb{R}^m \to SE(3)$. The differential kinematics then involves derivatives of both f and g with respect to θ_a and θ_p. For platforms like the Stewart-Gough platorm, the differential kinematics can also be obtained from a static analysis, by exploiting the fact that just as for closed chains, the external forces $\mathcal{F}$ at the end-effector are related to the joint torques τ by $\tau = J^T \mathcal{F}$.

- Singularities for closed chains can be classifed into three types: (i) configuration space singularities occur at, e.g., self-intersections of the configuration space surface (or bifurcation points in the event that the configuration space is a curve); (ii) nondegenerate actuator singularities when the actuated joints cannot be independently actuated, while degenerate actuator singularities are characterized by the mechanism failing to become a rigid structure even when all the actuated joints are locked in place; (iii) end-effector singularities occur when the end-effector loses one or more degrees of freedom of mobility. Configuration space singularities are independent of choice of actuated joints, while actuator singularities depend on which joints are actuated. End-effector singularities depend on where the end-effector frame is placed, but do not depend on the choice of actuated joints.
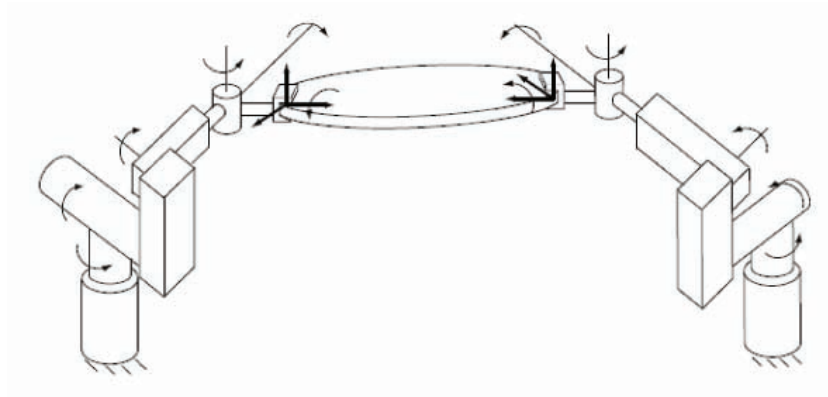
Figure 7.11: Two cooperating six degree of freedom arms grasping an object.
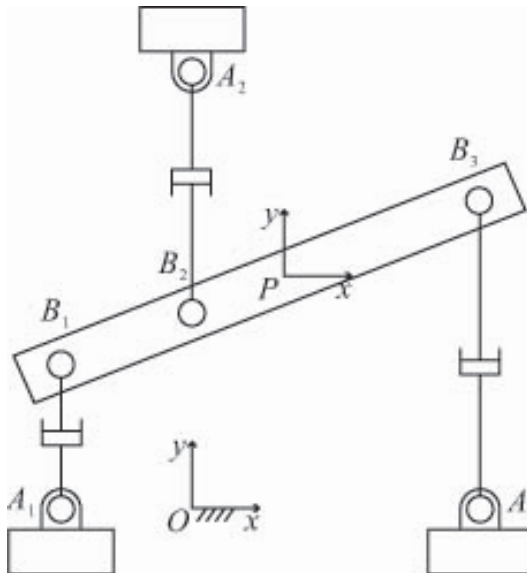
Figure 7.12: A $3 \times RPR$ planar parallel mechanism.

7.5   Exercises

1. Two six degree of freedom arms cooperate to move the disc as shown in Figure 7.11. Given the position and orientation of the disc, how many inverse kinematics solutions exist?

2. Consider the $3 \times RPR$ planar parallel mechanism of Figure 7.12, in which the prismatic joints are actuated. Define $\overrightarrow{OA_i} = a_i \in \mathbb{R}^3$ with respect to the
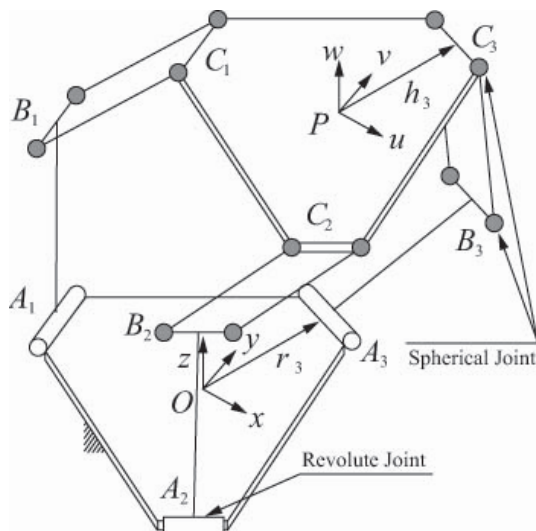
Figure 7.13: A Delta robot.

fixed frame and $\overrightarrow{PB_i} = b_i \in \mathbb{R}^3$ with respect to the moving platform frame.
(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?

3. The Delta robot of Figure 7.13 consists of a fixed base connected to a moving platform by three arms. Each arm consists of an upper arm—made up of a spatial parallelogram connected by spherical joints at the ends of each rod—and a lower arm connected to the fixed base by a revolute joint ω_i, connected orthogonally to the upper arm rod. $\overrightarrow{OA_i} = r_i$, $\hat{r}_i \times \omega_i = z$, $\overline{A_iB_i} = a_i$, $\overline{B_iC_i} = b_i$, $\omega_i \perp a_i$, $\overrightarrow{PC_i} = h_i$ are defined with respect to the moving platform frame. Derive step-by-step procedures for solving the following:
(a) Solve the forward kinematics.
(b) Solve the inverse kinematics.
(c) Find the Jacobian J_a.

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

$$\tau = M(\theta)\ddot{\theta} + b(\theta, \dot{\theta}), \tag{8.1}$$

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 and invertible matrix called the **mass matrix**, and $b(\theta, \dot{\theta}) \in \mathbb{R}^n$ are "bias" forces that lump together centrifugal and Coriolis, gravity, friction and other force terms that depend on θ 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 $b(\theta, \dot{\theta})$ 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**. From the perspective of generating and simulating entire motion trajectories, it is useful to regard the robot dynamics as an input-output system, in which the inputs are torque trajectories $\tau(t)$, and the outputs are joint trajectories $\theta(t)$. From this perspective, in the case of forward dynamics the objective is to determine, from a given input joint torque trajectory $\tau(t)$ and appropriate boundary conditions on θ and $\dot{\theta}$, the output joint trajectory $\theta(t)$; this is usually done by numerically integrating Equation (8.1). In the case of inverse dynamics, the objective is to determine the joint torque trajectory $\tau(t)$ that generates some desired joint motion trajectory $\theta(t)$.

Slight variations in these interpretations of the forward and inverse dynamics are possible. For the inverse dynamics, observe that the velocity $\dot{\theta}$ and acceleration $\ddot{\theta}$ can be obtained by taking derivatives of the desired joint trajectory $\theta(t)$. Thus, given values for $(\theta, \dot{\theta}, \ddot{\theta})$ at time t, the joint torques τ can be obtained just by algebraic evaluation of the right-hand side of (8.1). This evaluation is also commonly referred to as the inverse dynamics. In the case of forward dynamics,

since $M(\theta)$ is always invertible, Equation (8.1) can be rewritten

$$\ddot{\theta} = M^{-1}(\theta)\left(\tau - b(\theta, \dot{\theta})\right).\tag{8.2}$$

This evaluation of $\ddot{\theta}$ from given values for τ, θ, and $\dot{\theta}$ is also often referred to as the forward dynamics. While this interpretation may seem somewhat different from the previous one, in fact it is not: given an input torque trajectory $\tau(t)$ together with initial values for θ and $\dot{\theta}$ at $t = t_0$, forward integration of Equation (8.2) from $t = t_0$ then produces the complete output trajectory $\theta(t)$.

A robot's dynamic equations are typically derived in one of two ways: by a direct application of Newton's and Euler's dynamic equations for a rigid body (often called the **Newton-Euler formulation**), or from the **Lagrangian dynamics** formulation. 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 intractable 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).

In this chapter we study both the Lagrangian and Newton-Euler dynamics formulations for an open chain robot. We conclude with a formulation of the dynamics in task space coordinates, or **operational space dynamics**.

8.1 Lagrangian Formulation

8.1.1 Basic Concepts and Motivating Example

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 $f \in \mathbb{R}^n$ that are dual to q, called **generalized forces**. f and q are dual to each other in the sense that their inner product $f^T \Delta q$ corresponds to work (equivalently, $f^T \dot{q}$ corresponds to power). A Lagrangian function $\mathcal{L}(q, \dot{q})$ is then defined as the overall system's kinetic energy minus the potential energy. The equations of motion can now be expressed in terms of the Lagrangian as follows:

$$f = \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q},\tag{8.3}$$

These equations are also referred to as the **Euler-Lagrange equations with external forces**.[1]

We illustrate the Lagrangian dynamics formulation through two examples. In the first example, consider a particle of mass $\mathfrak{m}$ constrained to move on

---

[1]The external force f is zero in the standard form of the Euler-Lagrange equations.

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 $\mathfrak{m}\mathfrak{g}$ acts downward, and an external force f is applied upward. By Newton's second law, the equation of motion for the particle is

$$f - \mathfrak{m}\mathfrak{g} = \mathfrak{m}\ddot{x}. \tag{8.4}$$

We now apply the Lagrangian formalism to this particle. The kinetic energy is $\mathfrak{m}\dot{x}^2/2$, the potential energy is $\mathfrak{m}\mathfrak{g}x$, and the Lagrangian is

$$\mathcal{L}(x, \dot{x}) = \frac{1}{2}\mathfrak{m}\dot{x}^2 - \mathfrak{m}\mathfrak{g}x. \tag{8.5}$$

The equations of motion are then given by

$$f = \frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = \mathfrak{m}\ddot{x} + \mathfrak{m}\mathfrak{g}, \tag{8.6}$$

which matches Equation (8.4).

We now derive the dynamic equations for a planar 2R open chain moving in the presence of gravity. The chain moves in the x-y plane, with gravity 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 $\mathfrak{m}_1$ and $\mathfrak{m}_2$ concentrated at the ends of each link. The position and velocity of the mass of link 1 are then given by

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} L_1 \cos\theta_1 \\ L_1 \sin\theta_1 \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \end{bmatrix} = \begin{bmatrix} -L_1 \sin\theta_1 \\ L_1 \cos\theta_1 \end{bmatrix} \dot{\theta}_1,$$

while that of the link 2 mass are given by

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) \\ L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) \end{bmatrix}$$

$$\begin{bmatrix} \dot{x}_2 \\ \dot{y}_2 \end{bmatrix} = \begin{bmatrix} -L_1 \sin\theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}.$$

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 $\tau^T \Delta\theta$ must correspond to work). The Lagrangian $\mathcal{L}(\theta, \dot{\theta})$ is of the form

$$\mathcal{L}(\theta, \dot{\theta}) = \sum_{i=1}^{2} \mathcal{K}_i - \mathcal{P}_i, \tag{8.7}$$

where the link kinetic energy terms $\mathcal{K}_1$ and $\mathcal{K}_2$ are

$$
\begin{aligned}
\mathcal{K}_1 &= \frac{1}{2}\mathfrak{m}_1(\dot{x}_1^2 + \dot{y}_1^2) = \frac{1}{2}\mathfrak{m}_1 L_1^2 \dot{\theta}_1^2 \\
\mathcal{K}_2 &= \frac{1}{2}\mathfrak{m}_2(\dot{x}_2^2 + \dot{y}_2^2) \\
&= \frac{\mathfrak{m}_2}{2}\left((L_1^2 + 2L_1 L_2 \cos\theta_2 + L_2^2)\dot{\theta}_1^2 + 2(L_2^2 + L_1 L_2 \cos\theta_2)\dot{\theta}_1\dot{\theta}_2 + L_2^2\dot{\theta}_2^2 \right),
\end{aligned}
$$

and the link potential energy terms $\mathcal{P}_1$ and $\mathcal{P}_2$ are

$$
\begin{aligned}
\mathcal{P}_1 &= \mathfrak{m}_1 \mathfrak{g} L_1 \sin\theta_1 \\
\mathcal{P}_2 &= \mathfrak{m}_2 \mathfrak{g}(L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)).
\end{aligned}
$$

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):

$$
\begin{aligned}
\tau_1 &= \left((\mathfrak{m}_1 + \mathfrak{m}_2)L_1^2 + \mathfrak{m}_2(2L_1 L_2 \cos\theta_2 + L_2^2) \right)\ddot{\theta}_1 \\
&\quad + \mathfrak{m}_2(L_1 L_2 \cos\theta_2 + L_2^2)\ddot{\theta}_2 - 2\mathfrak{m}_2 L_1 L_2 \dot{\theta}_1 \dot{\theta}_2 \sin\theta_2 - \mathfrak{m}_2 L_1 L_2 \dot{\theta}_2^2 \sin\theta_2 \\
&\quad + (\mathfrak{m}_1 + \mathfrak{m}_2)L_1 g \cos\theta_1 + \mathfrak{m}_2 g L_2 \cos(\theta_1 + \theta_2) \\
\tau_2 &= \mathfrak{m}_2(L_1 L_2 \cos\theta_2 + L_2^2)\ddot{\theta}_1 + \mathfrak{m}_2 L_2^2 \ddot{\theta}_2 + \mathfrak{m}_2 L_1 L_2 \dot{\theta}_1^2 \sin\theta_2 \\
&\quad + \mathfrak{m}_2 g L_2 \cos(\theta_1 + \theta_2).
\end{aligned}
$$

In the Lagrangian formulation of dynamics, once a set of generalized coordinates has been chosen, it is conceptually straightforward to formulate the Lagrangian, and from there to arrive at the dynamic equations by taking partial derivatives of the Lagrangian. In practice, however, the calculations can very quickly become intractable, especially as the number of degrees of freedom increases.

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 θ to be the vector of joint values. The generalized forces will then be denoted $\tau \in \mathbb{R}^n$. If θ_i is a revolute joint τ_i will correspond to a torque, while if θ_i is a prismatic joint τ_i will correspond to a force.

Once θ has been chosen and the generalized forces τ identified, the next step is to formulate the Lagrangian $\mathcal{L}(\theta, \dot{\theta})$ as

$$
\mathcal{L}(\theta, \dot{\theta}) = \mathcal{K}(\theta, \dot{\theta}) - \mathcal{P}(\theta), \tag{8.9}
$$

where $\mathcal{K}(\theta, \dot{\theta})$ is the kinetic energy and $\mathcal{P}(\theta)$ the potential energy of the overall system. For rigid-link robots the kinetic energy can always be written in the form

$$\mathcal{K}(\theta) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} m_{ij}(\theta) \dot{\theta}_i \dot{\theta}_j = \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta}, \tag{8.10}$$

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 in the next section. The dynamic equations are analytically obtained by evaluating the right-hand side of

$$\tau_i = \frac{d}{dt} \frac{\partial \mathcal{L}}{\partial \dot{\theta}_i} - \frac{\partial \mathcal{L}}{\partial \theta_i}, \ i = 1, \dots, n. \tag{8.11}$$

With the kinetic energy expressed as in Equation (8.10), the dynamics can be explicitly written as

$$\tau_i = \sum_{j=1}^{n} m_{ij}(\theta) \ddot{\theta}_j + \sum_{j=1}^{n} \sum_{k=1}^{n} \Gamma_{ijk}(\theta) \dot{\theta}_j \dot{\theta}_k + \frac{\partial \mathcal{P}}{\partial \theta_i}, \ \ i = 1, \dots, n, \tag{8.12}$$

where the $\Gamma_{ijk}(\theta)$, known as the **Christoffel symbols of the first kind**, are defined as

$$\Gamma_{ijk}(\theta) = \frac{1}{2} \left(\frac{\partial m_{ij}}{\partial \theta_k} + \frac{\partial m_{ik}}{\partial \theta_j} - \frac{\partial m_{jk}}{\partial \theta_i} \right). \tag{8.13}$$

The Lagrangian formulation has traditionally been regarded as the most direct way of obtaining a set of closed-form analytical equations for the dynamics. For open chain robots not only is this no longer true, but the formula for $\Gamma_{ijk}(\theta)$ above and our earlier examples offer a strong hint of how intractable the calculations can become, especially for robots with higher degrees of freedom. The Newton-Euler formulation on the other hand allows us to bypass the evaluation of these partial derivatives. However, as we show later, the Lagrangian formulation offers important insights into the structure of the dynamics equations, especially in the development of stable robot control laws.

8.2 Dynamics of a Single Rigid Body

8.2.1 Classical Formulation

We now consider the dynamic equations for a single rigid body. In most treatments of rigid body dynamics one begins with the equations of motion for a single particle of mass $\mathfrak{m}$ (essentially, $\mathbf{f} = \mathfrak{m}\mathbf{a}$), which is then generalized to a system of particles. A rigid body is then assumed to be a system consisting of an infinite numer of particles, with the constraint that the distances between particles are always preserved (ensuring that the body is a rigid body). The equations of motion for a system of particles are then applied to this infinite collection of particles, resulting in the equations of motion for a rigid body.

In this section we shall omit the single particle dynamics formulation and proceed directly to the single rigid body case. Suppose a rigid body of mass $\mathfrak{m}$ has a reference frame {b} with axes $\{\hat{x}_b, \hat{y}_b, \hat{z}_b\}$ attached to its center of mass. As the rigid body moves, the body frame also moves with linear velocity $\mathbf{v}$ and angular velocity $\mathbf{w}$. Further assume that the rigid body is subject to an external force $\mathbf{f}$. The external moment $\mathbf{m}$ generated by $\mathbf{f}$ with respect to the center of mass is then $\mathbf{m} = \mathbf{r} \times \mathbf{f}$, where $\mathbf{r}$ is the vector from the center of mass to the point on the body at which $\mathbf{f}$ is applied. Let $\mathbf{h}$ denote the angular momentum vector about the center of mass (we'll explain shortly how to calculate $\mathbf{h}$). The dynamic equations for the rigid body are then given by

$$\mathbf{f} = \frac{d}{dt}(\mathfrak{m}\mathbf{v}) = \mathfrak{m}\frac{d}{dt}\mathbf{v} \qquad (8.14)$$

$$\mathbf{m} = \frac{d}{dt}\mathbf{h}. \qquad (8.15)$$

We now express these dynamic equations explicitly in frame {b} coordinates. First express the angular and linear velocity in {b}-frame coordinates by

$$\mathbf{w} = \omega_x\,\hat{x}_b + \omega_y\,\hat{y}_b + \omega_z\,\hat{z}_b$$
$$\mathbf{v} = v_x\,\hat{x}_b + v_y\,\hat{y}_b + v_z\,\hat{z}_b,$$

with the column vectors $\omega_b = (\omega_x,\,\omega_y,\,\omega_z)^T$, $v_b = (v_x,\,v_y,\,v_z)^T$ defined accordingly. The linear acceleration $\mathbf{a}$ is then

$$\mathbf{a} = \frac{d}{dt}\mathbf{v} = (\dot{v}_x\,\hat{x}_b + \dot{v}_y\,\hat{y}_b + \dot{v}_z\,\hat{z}_b) + v_x\,\dot{\hat{x}}_b + v_y\,\dot{\hat{y}}_b + v_z\,\dot{\hat{z}}_b. \qquad (8.16)$$

Substituting the rotating frame identities $\dot{\hat{x}}_b = \mathbf{w} \times \hat{x}_b$, $\dot{\hat{y}}_b = \mathbf{w} \times \hat{y}_b$, $\dot{\hat{z}}_b = \mathbf{w} \times \hat{z}_b$ into (8.16) leads to

$$\mathbf{a} = (\dot{v}_x\,\hat{x}_b + \dot{v}_y\,\hat{y}_b + \dot{v}_z\,\hat{z}_b) + \mathbf{w} \times \mathbf{v}.$$

The {b}-frame vector representation $a_b \in \mathbb{R}^3$ of the linear acceleration is therefore

$$a_b = \dot{v}_b + (\omega_b \times v_b),$$

where $\dot{v}_b = (\dot{v}_x, \dot{v}_y, \dot{v}_z)^T$. Equation (8.14) expressed in {b}-frame coordinates now becomes

$$f_b = \mathfrak{m}(\dot{v}_b + \omega_b \times v_b). \qquad (8.17)$$

We next express the angular momentum $\mathbf{h}$ in {b}-frame coordinates. When the {b} frame is attached to the body's center of mass as we have done here, the angular momentum assumes a particularly simple form. First, the 3×3 rotational inertia matrix of the rigid body is required; this can be obtained by imagining the rigid body as a collection of an infinite number of particles of mass $\mathfrak{m}_i$, each with coordinates $(x_i,\,y_i,\,z_i)$ with respect to the {b} frame. Denoting

the rotational inertia matrix by $\mathcal{I}_b \in \mathbb{R}^{3 \times 3}$, $\mathcal{I}_b$ is obtained via the following summation over all particles constituting the rigid body:

$$
\mathcal{I}_b \;=\; \begin{bmatrix} \sum \mathfrak{m}_i(y_i^2 + z_i^2) & -\sum \mathfrak{m}_i x_i y_i & -\sum \mathfrak{m}_i x_i z_i \\ -\sum \mathfrak{m}_i x_i y_i & \sum \mathfrak{m}_i(x_i^2 + z_i^2) & -\sum \mathfrak{m}_i y_i z_i \\ -\sum \mathfrak{m}_i x_i z_i & -\sum \mathfrak{m}_i y_i z_i & \sum \mathfrak{m}_i(x_i^2 + y_i^2) \end{bmatrix}
$$

$$
\;=\; \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix}.
$$

The matrix $\mathcal{I}_b$ defined in this way is constant, symmetric, and positive-definite. In the limit as the number of particles becomes infinite, the summations can be replaced by volume integrals over the body $\mathcal{B}$, with the particle masses m_i now replaced by a mass density function $\rho(x, y, z)$:

$$
\mathcal{I}_{xx} \;=\; \iiint_{\mathcal{B}} (y^2 + z^2)\rho(x, y, z)\, dx\, dy\, dz
$$

$$
\mathcal{I}_{yy} \;=\; \iiint_{\mathcal{B}} (x^2 + z^2)\rho(x, y, z)\, dx\, dy\, dz
$$

$$
\mathcal{I}_{zz} \;=\; \iiint_{\mathcal{B}} (x^2 + y^2)\rho(x, y, z)\, dx\, dy\, dz
$$

$$
\mathcal{I}_{xy} \;=\; \mathcal{I}_{yx} = -\iiint_{\mathcal{B}} xy\rho(x, y, z)\, dx\, dy\, dz
$$

$$
\mathcal{I}_{xz} \;=\; \mathcal{I}_{zx} = -\iiint_{\mathcal{B}} xz\rho(x, y, z)\, dx\, dy\, dz
$$

$$
\mathcal{I}_{yz} \;=\; \mathcal{I}_{zy} = -\iiint_{\mathcal{B}} yz\rho(x, y, z)\, dx\, dy\, dz.
$$

If the body has a uniform mass density throughout, $\mathcal{I}_b$ is then determined exclusively by the shape of the rigid body. The calculation of $\mathcal{I}_b$ for some standard link shapes are covered in the exercises at the end of this chapter.

Express $\mathbf{h}$ in terms of the unit axes of the {b} frame as

$$
\mathbf{h} = h_x\, \hat{\mathbf{x}}_b + h_y\, \hat{\mathbf{y}}_b + h_z\, \hat{\mathbf{z}}_b, \tag{8.18}
$$

and define $h_b = (h_x,\ h_y,\ h_z)^T \in \mathbb{R}^3$. h_b is then obtained as

$$
h_b = \mathcal{I}_b \omega_b. \tag{8.19}
$$

Since Equation (8.15) calls for the derivative of $\mathbf{h}$, differentiating (8.18) leads to

$$
\frac{d}{dt}\mathbf{h} = (\dot{h}_x\, \hat{\mathbf{x}}_b + \dot{h}_y\, \hat{\mathbf{y}}_b + \dot{h}_z\, \hat{\mathbf{z}}_b) + \mathbf{w} \times \mathbf{h}.
$$

The moment equation (8.15) expressed in {b} frame coordinates thus becomes

$$
m_b = \mathcal{I}_b \dot{\omega}_b + \omega_b \times \mathcal{I}_b \omega_b. \tag{8.20}
$$

where $m_b \in \mathbb{R}^3$ is the moment vector $\mathbf{m}$ in frame {b} coordinates. Equations (8.17) and (8.20) together constitute the dynamic equations of motion for the rigid body.

8.2.2   Twist-Wrench Formulation

Equations (8.17) and (8.20) can be written in the following combined form:

$$
\left[\begin{array}{c} m_b \\ f_b \end{array} \right] = \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \dot{\omega}_b \\ \dot{v}_b \end{array} \right] + \left[\begin{array}{cc} [\omega_b] & 0 \\ 0 & [\omega_b] \end{array} \right] \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right]. \quad (8.21)
$$

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 (8.21) in the following equivalent form:

$$
\begin{aligned}
\left[\begin{array}{c} m_b \\ f_b \end{array} \right] &= \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \dot{\omega}_b \\ \dot{v}_b \end{array} \right] + \left[\begin{array}{cc} [\omega_b] & [v_b] \\ 0 & [\omega_b] \end{array} \right] \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right] \\
&= \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \dot{\omega}_b \\ \dot{v}_b \end{array} \right] - \left[\begin{array}{cc} [\omega_b] & 0 \\ [v_b] & [\omega_b] \end{array} \right]^T \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right] \quad (8.22)
\end{aligned}
$$

Written in this way, each of the terms can now be identified with six-dimensional spatial quantities as follows:

(i) (ω_b, v_b) and (m_b, f_b) can be respectively identified with the spatial velocity (or twist) $\mathcal{V}_b$ and spatial force (or wrench) $\mathcal{F}_b$:

$$
\mathcal{V}_b = \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right], \quad \mathcal{F}_b = \left[\begin{array}{c} m_b \\ f_b \end{array} \right]. \quad (8.23)
$$

(ii) The **spatial inertia matrix** $\mathcal{G}_b \in \mathbb{R}^{6 \times 6}$ is defined as follows:

$$
\mathcal{G}_b = \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right], \quad (8.24)
$$

where I denotes the 3×3 identity matrix. Note as an aside that the kinetic energy of the rigid body can be expressed in terms of the spatial inertia matrix as

$$
\text{Kinetic Energy} = \frac{1}{2} \omega_b^T \mathcal{I}_b \omega_b + \frac{1}{2} \mathfrak{m} v_b^T v_b = \frac{1}{2} \mathcal{V}_b^T \mathcal{G}_b \mathcal{V}_b. \quad (8.25)
$$

(iii) The **spatial momentum** $\mathcal{P}_b \in \mathbb{R}^6$ is defined as

$$
\mathcal{P}_b = \left[\begin{array}{c} \mathcal{I}_b \omega_b \\ \mathfrak{m} v_b \end{array} \right] = \left[\begin{array}{cc} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{array} \right] \left[\begin{array}{c} \omega_b \\ v_b \end{array} \right] = \mathcal{G}_b \mathcal{V}_b. \quad (8.26)
$$

Observe that the $\mathcal{P}_b$ term in Equation (8.22) is left-multiplied by the matrix

$$
- \left[\begin{array}{cc} [\omega_b] & 0 \\ [v_b] & [\omega_b] \end{array} \right]^T. \quad (8.27)
$$

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:

$$
[\omega_1 \times \omega_2] = [\omega_1][\omega_2] - [\omega_2][\omega_1]. \quad (8.28)
$$

The matrix in (8.27) can be thought of as a generalization of the cross product operation to six-dimensional twists. Specifically, given two twists $\mathcal{V}_1 = (\omega_1, v_1)$ and $\mathcal{V}_2 = (\omega_2, v_2)$, we perform a calculation analogous to (8.28):

$$\left[\begin{array}{cc} [\omega_1] & v_1 \\ 0 & 0 \end{array}\right]\left[\begin{array}{cc} [\omega_2] & v_2 \\ 0 & 0 \end{array}\right] - \left[\begin{array}{cc} [\omega_2] & v_2 \\ 0 & 0 \end{array}\right]\left[\begin{array}{cc} [\omega_1] & v_1 \\ 0 & 0 \end{array}\right]$$

$$= \left[\begin{array}{cc} [\omega_1][\omega_2] - [\omega_2][\omega_1] & [\omega_1]v_2 - [\omega_2]v_1 \\ 0 & 0 \end{array}\right] = \left[\begin{array}{cc} [\omega'] & v' \\ 0 & 0 \end{array}\right],$$

which can be written more compactly in vector form as

$$\left[\begin{array}{c} \omega' \\ v' \end{array}\right] = \left[\begin{array}{cc} [\omega_1] & 0 \\ [v_1] & [\omega_1] \end{array}\right]\left[\begin{array}{c} \omega_2 \\ v_2 \end{array}\right].$$

This generalization of the cross product to two twists $\mathcal{V}_1$ and $\mathcal{V}_2$ will be called the **Lie bracket** of $\mathcal{V}_1$ and $\mathcal{V}_2$.

Definition 8.1. Given two twists $\mathcal{V}_1 = (\omega_1, v_1)$ and $\mathcal{V}_2 = (\omega_2, v_2)$, the **Lie bracket** of $\mathcal{V}_1$ and $\mathcal{V}_2$, denoted simultaneously by $[\mathcal{V}_1, \mathcal{V}_2]$ and $\mathrm{ad}_{\mathcal{V}_1}(\mathcal{V}_2)$, is defined as follows:

$$[\mathcal{V}_1, \mathcal{V}_2] = \left[\begin{array}{cc} [\omega_1] & 0 \\ [v_1] & [\omega_1] \end{array}\right]\left[\begin{array}{c} \omega_2 \\ v_2 \end{array}\right] = \mathrm{ad}_{\mathcal{V}_1}(\mathcal{V}_2). \qquad (8.29)$$

Given $\mathcal{V} = (\omega, v)$, we further define the following notation for the 6×6 matrix representation $[\mathrm{ad}_{\mathcal{V}}]$:

$$[\mathrm{ad}_{\mathcal{V}}] = \left[\begin{array}{cc} [\omega] & 0 \\ [v] & [\omega] \end{array}\right] \in \mathbb{R}^{6\times6}. \qquad (8.30)$$

With this notation the Lie bracket $[\mathcal{V}_1, \mathcal{V}_2]$ can also be expressed as

$$[\mathcal{V}_1, \mathcal{V}_2] = \mathrm{ad}_{\mathcal{V}_1}(\mathcal{V}_2) = [\mathrm{ad}_{\mathcal{V}_1}]\mathcal{V}_2. \qquad (8.31)$$

Definition 8.2. Given a twist $\mathcal{V} = (\omega, v)$ and wrench $\mathcal{F} = (m, f)$, define the mapping

$$\mathrm{ad}_{\mathcal{V}}^T(\mathcal{F}) = [\mathrm{ad}_{\mathcal{V}}]^T\mathcal{F} = \left[\begin{array}{cc} [\omega] & 0 \\ [v] & [\omega] \end{array}\right]^T\left[\begin{array}{c} m \\ f \end{array}\right] = \left[\begin{array}{c} -[\omega]m - [v]f \\ -[\omega]f \end{array}\right] \qquad (8.32)$$

Using the above notation and definitions the dynamic equations for a single rigid body can now be written as follows:

$$\mathcal{F}_b = \mathcal{G}_b\dot{\mathcal{V}}_b - \mathrm{ad}_{\mathcal{V}_b}^T(\mathcal{P}_b) \qquad (8.33)$$

$$= \mathcal{G}_b\dot{\mathcal{V}}_b - [\mathrm{ad}_{\mathcal{V}_b}]^T\mathcal{G}_b\mathcal{V}_b. \qquad (8.34)$$

Note the similarity between (8.34) and the moment equation for a rotating rigid body:

$$m_b = \mathcal{I}_b\dot{\omega}_b - [\omega_b]^T\mathcal{I}_b\omega_b. \qquad (8.35)$$

Equation (8.35) is simply the rotational component of (8.34).

8.3   Inverse Dynamics of Open Chains

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

$$\tau = M(\theta)\ddot{\theta} + b(\theta, \dot{\theta}).$$

The main result will be 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   Overview of Newton-Euler Inverse Dynamics

Before getting into the details of the calculations, it is worth summarizing the method in words.

- **Preliminaries**: 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 $\{n\}$.

- **Initialization**: Set the velocity $\mathcal{V}_0$ of the base frame (typically zero), and let $\dot{\mathcal{V}}_0$ be opposite the gravitational acceleration. (Upward acceleration of the base is equivalent to gravity acting downward.) $\mathcal{F}_{n+1}$ is a wrench that the end-effector applies to the environment, expressed in frame $\{n+1\}$.

- **Forward Iteration**: For links $i = 1$ to n do

  - Calculate $T_{i-1,i}$, the configuration of frame $\{i\}$ expressed in frame $\{i-1\}$. This is used in the next two calculations.

  - Calculate $\mathcal{V}_i$, the velocity of link i expressed in frame $\{i\}$, as the sum of the velocity of link $i-1$ (but expressed in frame $\{i\}$) plus the extra velocity due to the joint velocity $\dot{\theta}_i$.

  - Calculate $\dot{\mathcal{V}}_i$ as the sum of the acceleration of link $i-1$ (but expressed in frame $\{i\}$) plus the extra acceleration due to the joint acceleration $\ddot{\theta}_i$.

- **Backward Iteration**: For links $i = n$ to 1 do

  - The wrench $\mathcal{F}_i$ that must be applied to link i is the sum of the wrench $\mathcal{F}_{i+1}$ that must be provided to link $i+1$ (but expressed in frame $\{i\}$) plus the extra wrench $\mathcal{G}_i \dot{\mathcal{V}}_i - \mathrm{ad}_{\mathcal{V}_i}^T(\mathcal{G}_i \mathcal{V}_i)$, from the rigid-body dynamics of link i. For link $i = n$, the wrench $\mathcal{F}_{i+1}$ is just the wrench the end-effector applies to the environment. For inboard links $i < n$, the wrench $\mathcal{F}_{i+1}$ includes also the wrenches needed to support the outboard links.

– The 6-vector wrench $\mathcal{F}_i$ calculated in the previous step is the sum of the wrench provided by the motor at joint i plus the wrench provided by the structure of joint i (e.g., the bearings). The motor itself provides only a wrench that lives on a one-dimensional line in the wrench space, typically a pure torque about the axis of a revolute joint or a pure force along the axis of a prismatic joint. We get the other five dimensions of $\mathcal{F}_i$ "for free" from the structure of the joint that limits the motion to one degree-of-freedom. Thus the force or torque τ_i provided by the motor at joint i is simply the component of $\mathcal{F}_i$ along the one-dimensional line of motor wrenches. The motor effort τ_i can be calculated by taking the projection (i.e., dot product) of $\mathcal{F}_i$ onto the motor axis.

All that is needed to implement the Newton-Euler method is to introduce some notation and derive the calculations above.

8.3.2   Details of Newton-Euler Inverse Dynamics

A body-fixed reference frame $\{i\}$ is attached to the center of mass of each link i, $i = 1, \ldots, n$. The ground 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 define $M_i \in SE(3)$ to be the configuration of $\{i\}$ in $\{0\}$ and $M_{i-1,i} \in SE(3)$ to be the configuration of $\{i\}$ in $\{i-1\}$. Let $\mathcal{A}_i$ be the twist vector for joint i (assuming θ_i is set to zero) expressed in the $\{i\}$ frame. Then the displacement from frame $\{i-1\}$ to $\{i\}$, denoted $T_{i-1,i} \in SE(3)$, is expressed in the following form:

$$T_{i-1,i} = M_{i-1,i} e^{[\mathcal{A}_i]\theta_i}. \tag{8.36}$$

If the forward kinematics is expressed in the space-frame product-of-exponentials, then the forward kinematics up to each link frame $\{i\}$ can be written

$$T_{0i} \;=\; e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} \cdots e^{[\mathcal{S}_i]\theta_i} M_i \tag{8.37}$$

for $i = 1, \ldots, n$. For each $i = 1, \ldots, n$, the following can be established via direct calculation:

$$M_{i-1,i} \;=\; M_{i-1}^{-1} M_i \tag{8.38}$$
$$\mathcal{A}_i \;=\; \mathrm{Ad}_{M_i^{-1}}(\mathcal{S}_i). \tag{8.39}$$

We further define the following notation:

(i) Denote the velocity of link frame $\{i\}$ expressed in frame $\{i\}$ coordinates by $\mathcal{V}_i = (\omega_i, v_i)$. Note that $\mathcal{V}_i$ is obtained from $[\mathcal{V}_i] = T_{0i}^{-1} \dot{T}_{0i}$.

(ii) Let $\mathcal{G}_i \in \mathbb{R}^{6\times6}$ denote the 6×6 inertia matrix of link i, expressed relative to link frame $\{i\}$. Since we assume here that all link frames are situated
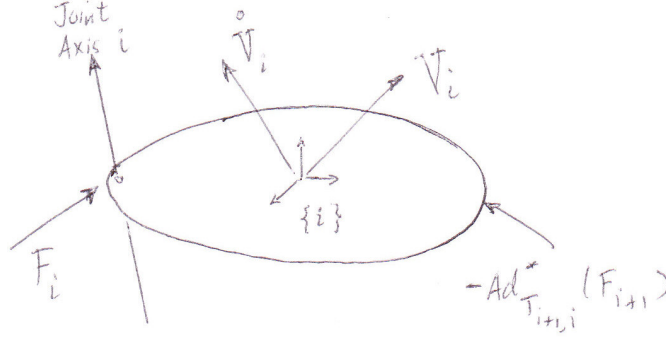
Figure 8.1: Free body diagram illustrating the moments and forces exerted on link i.

at the link center of mass, $\mathcal{G}_i$ has the block-diagonal form

$$\mathcal{G}_i = \left[\begin{array}{cc} \mathcal{I}_i & 0 \\ 0 & \mathfrak{m}_i I \end{array} \right], \tag{8.40}$$

where $\mathcal{I}_i$ denotes the 3×3 rotational inertia matrix of link i, and $\mathfrak{m}_i$ is the link mass.

(iii) Denote by $\mathcal{F}_i = (m_i, f_i)$ the spatial force transmitted from link $i-1$ to link i, expressed in frame $\{i\}$ coordinates. Note that $\mathcal{F}_i$ is transmitted entirely through joint i.

With the above notation and definitions, we now consider the free-body diagram for link i as shown in Figure 8.1. Note that $\mathcal{F}_{i+1}$ is the wrench applied by link i to link $i+1$, expressed in frame $\{i+1\}$ coordinates. What is needed is the wrench applied by link $i+1$ to link i, expressed in frame $\{i\}$ coordinates; using the wrench transformation rule under a change of reference frames, this is given by

$$\mathrm{Ad}^T_{T_{i+1,i}}(-\mathcal{F}_{i+1}) = -\mathrm{Ad}^T_{T_{i+1,i}}(\mathcal{F}_{i+1}).$$

The equations of motion for link i can therefore be written

$$\mathcal{G}_i \dot{\mathcal{V}}_i = \mathrm{ad}^T_{\mathcal{V}_i}(\mathcal{G}_i \mathcal{V}_i) + \mathcal{F}_i - \mathrm{Ad}^T_{T_{i+1,i}}(\mathcal{F}_{i+1}). \tag{8.41}$$

The joint torque $\tau_i \in \mathbb{R}$ at joint i is then the projection of the wrench $\mathcal{F}_i$ onto the joint twist $\mathcal{A}_i$:

$$\tau_i = \mathcal{F}_i^T \mathcal{A}_i. \tag{8.42}$$

We now derive the forward iteration of link velocities and accelerations from the base to the tip. First note that

$$[\mathcal{V}_1] = T_{01}^{-1} \dot{T}_{01} = [\mathcal{A}_1 \dot{\theta}_1] \tag{8.43}$$

and

$$
\begin{aligned}
[\mathcal{V}_2] &= T_{02}^{-1}\dot{T}_{02} \\
&= T_{12}^{-1}(T_{01}^{-1}\dot{T}_{01})T_{12} + T_{12}^{-1}\dot{T}_{12} \\
&= T_{12}^{-1}[\mathcal{V}_1]T_{12} + [\mathcal{A}_2\dot{\theta}_2],
\end{aligned} \tag{8.44}
$$

or equivalently, $\mathcal{V}_2 = \mathrm{Ad}_{T_{21}}(\mathcal{V}_1) + \mathcal{A}_2\dot{\theta}_2$. Repeating this procedure for the subsequent links, it can be established that

$$
\mathcal{V}_i = \mathrm{Ad}_{T_{i,i-1}}(\mathcal{V}_{i-1}) + \mathcal{A}_i\dot{\theta}_i, \quad i = 1,\ldots,n. \tag{8.45}
$$

The accelerations $\dot{\mathcal{V}}_i$ can also be found recursively. Noting that

$$
[\dot{\mathcal{V}}_i] = \frac{d}{dt}T_{i-1,i}[\mathcal{V}_i]T_{i-1,i}^{-1} + T_{i-1,i}[\dot{\mathcal{V}}_i]T_{i-1,i}^{-1} + T_{i-1,i}[\mathcal{V}_i]\frac{d}{dt}T_{i-1,i}^{-1} + [\mathcal{A}_i]\ddot{\theta}_i,
$$

and

$$
\begin{aligned}
\frac{d}{dt}T_{i-1,i} &= M_{i-1,i}[\mathcal{A}_i]e^{[\mathcal{A}_i]\theta_i}\dot{\theta}_i = M_{i-1,i}e^{[\mathcal{A}_i]\theta_i}[\mathcal{A}_i]\dot{\theta}_i \\
\frac{d}{dt}T_{i-1,i}^{-1} &= -T_{i-1,i}^{-1}\dot{T}_{i-1,i}T_{i-1,i}^{-1},
\end{aligned}
$$

it can be shown that

$$
\dot{\mathcal{V}}_i = \mathcal{A}_i\ddot{\theta}_i + \mathrm{Ad}_{T_{i,i-1}}(\dot{\mathcal{V}}_{i-1}) + [\mathrm{Ad}_{T_{i-1,i}}(\mathcal{V}_{i-1}), \mathcal{A}_i\dot{\theta}_i], \tag{8.46}
$$

where $\left[\mathrm{Ad}_{T_{i-1,i}}(\mathcal{V}_{i-1}), \mathcal{A}_i\right]$ denotes the Lie bracket of $\mathrm{Ad}_{T_{i-1,i}}(\mathcal{V}_{i-1})$ with $\mathcal{A}_i$. Note that since $[\mathcal{A}_i, \mathcal{A}_i] = 0$ and $\mathrm{Ad}_{T_{i,i-1}}(\mathcal{V}_{i-1}) = \mathcal{V}_i - \mathcal{A}_i\dot{\theta}_i$, one obtains the alternative but equivalent formula

$$
\dot{\mathcal{V}}_i = \mathcal{A}_i\ddot{\theta}_i + \mathrm{Ad}_{T_{i,i-1}}(\dot{\mathcal{V}}_{i-1}) + [\mathcal{V}_i, \mathcal{A}_i\dot{\theta}_i]. \tag{8.47}
$$

The above formulas for the velocities and accelerations, together with the dynamic equations for any given link, can now be organized into a two-stage forward-backward iterative algorithm for the inverse dynamics. Before doing so, we examine how to include gravity in the dynamics. One way to simulate the effects of gravity is to set the base frame to have an acceleration $-\mathfrak{g}$, where $\mathfrak{g} \in \mathbb{R}^3$ denotes the gravitational acceleration vector as expressed in base frame coordinates. In this case it is important to remember that the link acceleration calculated from the recursive algorithm is not its true acceleration, but rather its true acceleration minus $\mathfrak{g}$.

The algorithm is initialized by providing initial values for $\mathcal{V}_0$, $\dot{\mathcal{V}}_0$, and $\mathcal{F}_{\mathrm{tip}}$, where $\mathcal{V}_0$ and $\dot{\mathcal{V}}_0$ are respectively the spatial velocity and spatial acceleration of the base frame expressed in base frame coordinates, and $\mathcal{F}_{\mathrm{tip}}$ is the spatial force applied to to the environment by the final link, expressed in end-effector frame coordinates. The joint trajectory $\theta(t)$ and its derivatives $\dot{\theta}$, $\ddot{\theta}$ are also assumed given as input.

Newton-Euler Inverse Dynamics Algorithm Summary

- **Preliminaries**: Each link frame $\{i\}$ is assumed attached to the link's center of mass. The forward kinematics from the base frame $\{0\}$ to link frame $\{i\}$ is of the form

$$T_{0i} = e^{[\mathcal{S}_1]\theta_1} e^{[\mathcal{S}_2]\theta_2} \cdots e^{[\mathcal{S}_i]\theta_i} M_i, \ i = 1, \ldots, n. \tag{8.48}$$

Define $M_{i-1,i}$ to be the displacement from link frame $\{i-1\}$ to link frame $\{i\}$ when the manipulator is at the home position. Then $M_i = M_{01}M_{12} \cdots M_{i-1,i}$ and $M_{i-1,i} = M_{i-1}^{-1}M_i$, $i = 1, \ldots, n$. The displacement between link frames $\{i-1\}$ and $\{i\}$ is

$$T_{i-1,i} = M_{i-1,i}e^{[\mathcal{A}_i]\theta_i}, \tag{8.49}$$

where

$$\mathcal{A}_i = \mathrm{Ad}_{M_i^{-1}}(\mathcal{S}_i), \ \ i = 1, \ldots, n. \tag{8.50}$$

With respect to the link frame attached at its center of mass, the 6×6 spatial inertia $\mathcal{G}_i$ of link i is defined as

$$\mathcal{G}_i = \begin{bmatrix} \mathcal{I}_i & 0 \\ 0 & \mathrm{m}_i I \end{bmatrix}, \tag{8.51}$$

where $\mathcal{I}_i \in \mathbb{R}^{3\times3}$ is its rotational inertia matrix, and m_i is the mass of link i. Define the twist $\mathcal{V}_0 = (\omega_0, v_0)$ to be the spatial velocity of the base frame, expressed in base frame coordinates. Define $\mathfrak{g} \in \mathbb{R}^3$ to be the gravity vector expressed in base frame $\{0\}$ coordinates. Define $\mathcal{F}_{\mathrm{tip}} = (m_{\mathrm{tip}}, f_{\mathrm{tip}})$ to be the wrench applied applied to the environment by the end-effector expressed in the end-effector frame $\{n+1\}$, fixed in the frame $\{n\}$.

- **Initialization**: $\mathcal{V}_0 = $ given, $\dot{\mathcal{V}}_0 = (0, \mathfrak{g})$, $\mathcal{F}_{n+1} = \mathcal{F}_{\mathrm{tip}}$.

- **Forward Iteration**: For $i = 1$ to n do

$$
\begin{align}
T_{i-1,i} &= M_{i-1,i}e^{[\mathcal{A}_i]\theta_i} \tag{8.52}\\
\mathcal{V}_i &= \mathrm{Ad}_{T_{i,i-1}}(\mathcal{V}_{i-1}) + \mathcal{A}_i\dot{\theta}_i \tag{8.53}\\
\dot{\mathcal{V}}_i &= \mathrm{Ad}_{T_{i,i-1}}(\dot{\mathcal{V}}_{i-1}) + [\mathcal{V}_i, \mathcal{A}_i]\dot{\theta}_i + \mathcal{A}_i\ddot{\theta}_i \tag{8.54}
\end{align}
$$

- **Backward Iteration**: For $i = n$ to 1 do

$$
\begin{align}
\mathcal{F}_i &= \mathrm{Ad}_{T_{i+1,i}}^T(\mathcal{F}_{i+1}) + \mathcal{G}_i\dot{\mathcal{V}}_i - \mathrm{ad}_{\mathcal{V}_i}^T(\mathcal{G}_i\mathcal{V}_i) \tag{8.55}\\
\tau_i &= \mathcal{F}_i^T\mathcal{A}_i. \tag{8.56}
\end{align}
$$

As noted earlier, the recursion formula Equation (8.54) for the acceleration $\dot{\mathcal{V}}_i$ can also be replaced by the equivalent formula

$$\dot{\mathcal{V}}_i = \mathrm{Ad}_{T_{i,i-1}}(\dot{\mathcal{V}}_{i-1}) + [\mathrm{Ad}_{T_{i-1,i}}(\mathcal{V}_{i-1}), \mathcal{A}_i]\dot{\theta}_i + \mathcal{A}_i\ddot{\theta}_i. \tag{8.57}$$

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 $\mathcal{K}$ of the robot can be expressed as $\mathcal{K} = \frac{1}{2}\dot{\theta}^T M(\theta)\dot{\theta}$. We do so by noting that $\mathcal{K}$ can be expressed as the sum of the kinetic energies of each link:

$$\mathcal{K} = \frac{1}{2}\sum_{i=1}^{n}\mathcal{V}_i^T \mathcal{G}_i \mathcal{V}_i, \tag{8.58}$$

where $\mathcal{V}_i$ is the spatial velocity of link frame $\{i\}$, and $\mathcal{G}_i$ is the spatial inertia matrix of link i as defined by Equation (8.51) (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 $T_{0i}^{-1}\dot{T}_{0i}$. 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

$$\mathcal{V}_i = J_{ib}(\theta)\dot{\theta}, \;\; i = 1, \ldots, n.$$

The kinetic energy can then be written

$$\mathcal{K} = \frac{1}{2}\dot{\theta}^T \left(\sum_{i=1}^{n} J_{ib}(\theta)^T \mathcal{G}_i J_{ib}(\theta)\right)\dot{\theta}. \tag{8.59}$$

The term inside the parentheses is precisely the mass matrix $M(\theta)$:

$$M(\theta) = \sum_{i=1}^{n} J_{ib}(\theta)^T \mathcal{G}_i J_{ib}(\theta). \tag{8.60}$$

Some of the exercises at the end of this chapter examine ways to recursively compute the entries of $M(\theta)$.

We now return to the original task of deriving a closed-form set of dynamic equations. We start by defining the following stacked vectors:

$$\mathcal{V} = \begin{bmatrix} \mathcal{V}_1 \\ \vdots \\ \mathcal{V}_n \end{bmatrix} \in \mathbb{R}^{6n} \tag{8.61}$$

$$\mathcal{F} = \begin{bmatrix} \mathcal{F}_1 \\ \vdots \\ \mathcal{F}_n \end{bmatrix} \in \mathbb{R}^{6n}. \tag{8.62}$$

Further define the following matrices:

$$
\mathcal{A} \;=\; \begin{bmatrix} \mathcal{A}_1 & 0 & \cdots & 0 \\ 0 & \mathcal{A}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \mathcal{A}_n \end{bmatrix} \in \mathbb{R}^{6n \times n} \tag{8.63}
$$

$$
\mathcal{G} \;=\; \begin{bmatrix} \mathcal{G}_1 & 0 & \cdots & 0 \\ 0 & \mathcal{G}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \mathcal{G}_n \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.64}
$$

$$
[\mathrm{ad}_\mathcal{V}] \;=\; \begin{bmatrix} [\mathrm{ad}_{\mathcal{V}_1}] & 0 & \cdots & 0 \\ 0 & [\mathrm{ad}_{\mathcal{V}_2}] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & [\mathrm{ad}_{\mathcal{V}_n}] \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.65}
$$

$$
[\mathrm{ad}_{\mathcal{A}\dot\theta}] \;=\; \begin{bmatrix} [\mathrm{ad}_{\mathcal{A}_1\dot\theta_1}] & 0 & \cdots & 0 \\ 0 & [\mathrm{ad}_{\mathcal{A}_2\dot\theta_2}] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & [\mathrm{ad}_{\mathcal{A}_n\dot\theta_n}] \end{bmatrix} \in \mathbb{R}^{6n \times 6n} \tag{8.66}
$$

$$
\mathcal{S}(\theta) \;=\; \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ [\mathrm{Ad}_{T_{21}}] & 0 & \cdots & 0 & 0 \\ 0 & [\mathrm{Ad}_{T_{32}}] & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & [\mathrm{Ad}_{T_{n,n-1}}] & 0 \end{bmatrix} \in \mathbb{R}^{6n \times 6n}. \tag{8.67}
$$

We write $\mathcal{S}(\theta)$ to emphasize the dependence of $\mathcal{S}$ on θ. Finally, define the following stacked vectors:

$$
\mathcal{V}_{\mathrm{base}} \;=\; \begin{bmatrix} \mathrm{Ad}_{T_{10}}(\mathcal{V}_0) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{R}^{6n} \tag{8.68}
$$

$$
\dot{\mathcal{V}}_{\mathrm{base}} \;=\; \begin{bmatrix} \mathrm{Ad}_{T_{10}}(\dot{\mathcal{V}}_0) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{R}^{6n} \tag{8.69}
$$

$$
\mathcal{F}_{\mathrm{tip}} \;=\; \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \mathrm{Ad}^T_{T_{n+1,n}}(\mathcal{F}_{n+1}) \end{bmatrix} \in \mathbb{R}^{6n}. \tag{8.70}
$$

Note that $\mathcal{A} \in \mathbb{R}^{6n \times n}$ and $\mathcal{G} \in \mathbb{R}^{6n \times 6n}$ are constant block-diagonal matrices,

in which $\mathcal{A}$ contains only the kinematic parameters, while $\mathcal{G}$ 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:

$$
\begin{align}
\mathcal{V} &= \mathcal{S}(\theta)\mathcal{V} + \mathcal{A}\dot\theta + \mathcal{V}_{\text{base}} \tag{8.71}\\
\dot{\mathcal{V}} &= \mathcal{S}(\theta)\dot{\mathcal{V}} + \mathcal{A}\ddot\theta - [\text{ad}_{\mathcal{A}\dot\theta}](\mathcal{S}(\theta)\mathcal{V} + \mathcal{V}_{\text{base}}) + \dot{\mathcal{V}}_{\text{base}} \tag{8.72}\\
\mathcal{F} &= \mathcal{S}^T(\theta)\mathcal{F} + \mathcal{G}\dot{\mathcal{V}} - [\text{ad}_{\mathcal{V}}]^T \mathcal{G}\mathcal{V} + \mathcal{F}_{\text{tip}} \tag{8.73}\\
\tau &= \mathcal{A}^T\mathcal{F}. \tag{8.74}
\end{align}
$$

$\mathcal{S}(\theta)$ has the property that $\mathcal{S}^n(\theta) = 0$ (such a matrix is said to be nilpotent of order n), and one consequence verifiable through direct calculation is that $(I - \mathcal{S}(\theta))^{-1} = I + \mathcal{S}(\theta) + \ldots + \mathcal{S}^{n-1}(\theta)$. Defining $\mathcal{L}(\theta) = (I - \mathcal{S}(\theta))^{-1}$, it can further be verified via direct calculation that

$$
\mathcal{L}(\theta) = \begin{bmatrix}
I & 0 & 0 & \cdots & 0 \\
[\text{Ad}_{T_{21}}] & I & 0 & \cdots & 0 \\
[\text{Ad}_{T_{31}}] & [\text{Ad}_{T_{32}}] & I & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
[\text{Ad}_{T_{n1}}] & [\text{Ad}_{T_{n2}}] & [\text{Ad}_{T_{n3}}] & \cdots & I
\end{bmatrix} \in \mathbb{R}^{6n \times 6n}. \tag{8.75}
$$

We write $\mathcal{L}(\theta)$ to emphasize the dependence of $\mathcal{L}$ on θ. The earlier matrix equations can now be reorganized as

$$
\begin{align}
\mathcal{V} &= \mathcal{L}(\theta)\left(\mathcal{A}\dot\theta + \mathcal{V}_{\text{base}}\right) \tag{8.76}\\
\dot{\mathcal{V}} &= \mathcal{L}(\theta)\left(\mathcal{A}\ddot\theta + [\text{ad}_{\mathcal{A}\dot\theta}]\mathcal{S}(\theta)\mathcal{V} + [\text{ad}_{\mathcal{A}\dot\theta}]\mathcal{V}_{\text{base}} + \dot{\mathcal{V}}_{\text{base}}\right) \tag{8.77}\\
\mathcal{F} &= \mathcal{L}^T(\theta)\left(\mathcal{G}\dot{\mathcal{V}} - [\text{ad}_{\mathcal{V}}]^T\mathcal{G}\mathcal{V} + \mathcal{F}_{\text{tip}}\right) \tag{8.78}\\
\tau &= \mathcal{A}^T\mathcal{F}. \tag{8.79}
\end{align}
$$

If an external wrench $\mathcal{F}_{\text{tip}}$ is applied at the tip, this can be included into the following dynamics equation:

$$
\tau = M(\theta)\ddot\theta + c(\theta, \dot\theta) + g(\theta) + J^T(\theta)\mathcal{F}_{\text{tip}}, \tag{8.80}
$$

where $J(\theta)$ denotes the Jacobian of the forward kinematics expressed in the same reference frame as $\mathcal{F}_{\text{tip}}$, and

$$
\begin{align}
M(\theta) &= \mathcal{A}^T\mathcal{L}^T(\theta)\mathcal{G}\mathcal{L}(\theta)\mathcal{A} \tag{8.81}\\
c(\theta, \dot\theta) &= -\mathcal{A}^T\mathcal{L}^T(\theta)\left(\mathcal{G}\mathcal{L}(\theta)[\text{ad}_{\mathcal{A}\dot\theta}]\mathcal{S}(\theta) + [\text{ad}_{\mathcal{V}}]^T\mathcal{G}\right)\mathcal{L}(\theta)\mathcal{A}\dot\theta \tag{8.82}\\
g(\theta) &= \mathcal{A}^T\mathcal{L}^T(\theta)\mathcal{G}\mathcal{L}(\theta)\dot{\mathcal{V}}_{\text{base}}. \tag{8.83}
\end{align}
$$

The $g(\theta)$ term reflects gravitational forces, while $c(\theta, \dot\theta)$ represents the Coriolis and centrifugal forces. Comparing these equations with the Lagrangian form of

the dynamics, i.e.,

$$\tau_i = \sum_{j=1}^{n} m_{ij}(\theta)\ddot{\theta}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} \Gamma_{ijk}(\theta)\dot{\theta}_j\dot{\theta}_k + \frac{\partial \mathcal{P}}{\partial \theta_i}, \quad i = 1, \ldots, n, \qquad (8.84)$$

where the $\Gamma_{ijk}(\theta)$ are

$$\Gamma_{ijk}(\theta) = \frac{1}{2}\left(\frac{\partial m_{ij}}{\partial \theta_k} + \frac{\partial m_{ik}}{\partial \theta_j} - \frac{\partial m_{jk}}{\partial \theta_i}\right), \qquad (8.85)$$

we can see that elements of the $c(\theta, \dot{\theta})$ term can be identified with

$$\sum_{j=1}^{n}\sum_{k=1}^{n} \Gamma_{ijk}(\theta)\dot{\theta}_j\dot{\theta}_k$$

and are thus quadratic in the $\dot{\theta}_i$. Elements of the gravity term $g(\theta)$ can be identified with $\frac{\partial \mathcal{P}}{\partial \theta_i}$. With the Newton-Euler formulation, the partial derivative terms appearing in $\Gamma_{ijk}(\theta)$ can be evaluated directly from (8.82) without taking derivatives. Further, by defining the matrix $C(\theta, \dot{\theta}) \in \mathbb{R}^{n \times n}$ as

$$c_{ij}(\theta, \dot{\theta}) = \sum_{k=1}^{n} \Gamma_{ijk}(\theta)\dot{\theta}_k = \frac{1}{2}\sum_{k=1}^{n}\left(\frac{\partial m_{ij}}{\partial \theta_k} + \frac{\partial m_{ik}}{\partial \theta_j} - \frac{\partial m_{jk}}{\partial \theta_i}\right)\dot{\theta}_k, \qquad (8.86)$$

where c_{ij} denotes the (i, j) entry of $C(\theta, \dot{\theta})$, it can be seen that $c(\theta, \dot{\theta})$ can be expressed as

$$c(\theta, \dot{\theta}) = C(\theta, \dot{\theta})\dot{\theta}. \qquad (8.87)$$

The matrix $C(\theta, \dot{\theta})$ is called the **Coriolis matrix**. The following property, referred to as the **passivity property**, turns out to have important ramifications in proving the stability of certain robot control laws.

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 (8.86), is skew-symmetric.*

Proof. The (i, j) component of $\dot{M} - 2C$ is

$$\dot{m}_{ij}(\theta) - 2c_{ij}(\theta, \dot{\theta}) \quad = \quad \sum_{k=1}^{n} \frac{\partial m_{ij}}{\partial \theta_k}\dot{\theta}_k - \frac{\partial m_{ij}}{\partial \theta_k}\dot{\theta}_k - \frac{\partial m_{ik}}{\partial \theta_j}\dot{\theta}_k + \frac{\partial m_{kj}}{\partial \theta_i}\dot{\theta}_k$$

$$= \quad \sum_{k=1}^{n} \frac{\partial m_{kj}}{\partial \theta_i}\dot{\theta}_k - \frac{\partial m_{ik}}{\partial \theta_j}\dot{\theta}_k.$$

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. $\square$

The passivity property will be used later in the chapter on robot control.

8.5   Forward Dynamics of Open Chains

We now consider the forward dynamics problem, where a torque trajectory $\tau(t)$ together with a set of initial conditions on θ and $\dot{\theta}$ is assumed given, and the objective is to integrate the dynamic equations $\tau(t) = M(\theta)\ddot{\theta} + b(\theta, \dot{\theta})$ to obtain the joint trajectory $\theta(t)$. The simplest numerical scheme for integrating the general first-order differential equation $\dot{q} = f(q, t)$, $q \in \mathbb{R}^n$, is via the Euler iteration

$$q(t + h) = q(t) + hf(q(t), t),$$

where the positive scalar h denotes the timestep. The dynamic equations can be converted to a first-order differential equation by taking advantage of the fact that $M(\theta)$ is always invertible: setting $q_1 = \theta$, $q_2 = \dot{\theta}$, and $q = (q_1, q_2) \in \mathbb{R}^{2n}$, we can write

$$
\begin{aligned}
\dot{q}_1 &= q_2 \\
\dot{q}_2 &= M^{-1}(q_1)\left(\tau(t) - b(q_1, q_2)\right),
\end{aligned}
$$

which is of the form $\dot{q} = f(q, t)$. The Euler integration scheme for this equation is thus of the form

$$
\begin{aligned}
q_1(t + h) &= q_1(t) + hq_2(t) \\
q_2(t + h) &= q_2(t) + h\left(M(q_1(t))^{-1}\left(\tau(t) - b(q_1(t), q_2(t))\right)\right).
\end{aligned}
$$

Given a set of initial values for $q_1(0) = \theta(0)$ and $q_2(0) = \dot{\theta}(0)$, the above equations can then be iterated forward in time to numerically obtain the motion $\theta(t) = q_1(t)$.

   Note that the above iteration appears to require the evaluation of $M^{-1}(\theta)$, which can be computationally expensive. In fact, it is possible to integrate these equations without having to explicitly compute the inverse of $M(\theta)$. The closed-form dynamic equations can be arranged as

$$M(\theta)\ddot{\theta} = \tau(t) - b(\theta, \dot{\theta}). \tag{8.88}$$

Setting $\ddot{\theta}$ to zero in (8.88) leads to $\tau = b(\theta, \dot{\theta})$. Therefore by running the inverse dynamics algorithm with $\ddot{\theta}(t)$ set to zero and $(\theta(t), \dot{\theta}(t))$ set to their current values, $b(\theta(t), \dot{\theta}(t))$ can now be determined. Subtracting this from the given value of $\tau(t)$ then results in the right-hand side of (8.88). Therefore, with a means of evaluating $M(\theta)$, it is a straightforward matter to obtain $\ddot{\theta}(t)$ as the solution to the linear equation $Ax = c$, where $A = M(\theta(t)) \in \mathbb{R}^{n \times n}$ is assured to be nonsingular, and $c = \tau(t) - b(\theta(t), \dot{\theta}(t))$ is known.

   Once a numerical integration scheme has been chosen, solving the forward dynamics then reduces to a procedure for evaluating $\ddot{\theta}$ from given values for θ, $\dot{\theta}$, and τ. In the following algorithm we allow for the possibility of an external spatial force $\mathcal{F}_{\text{tip}}$ applied to the final link.

Algorithm for Calculating the Joint Acceleration: *GetJointAccel($\theta, \dot{\theta}, \tau, \tau_{ext}$)*

- **Prerequisites**: Algorithms for calculating the inverse dynamics, and the mass matrix, are assumed available. An algorithm for solving the linear system $Ax = c$ for $x \in \mathbb{R}^n$, with given $c \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times n}$ nonsingular, is also assumed available.

- **Inputs**: Current values for θ, $\dot{\theta}$, the input torque τ. If an external spatial tip force $\mathcal{F}_{\text{tip}}$ is also given, it is transformed via the static force-torque relation to $\tau_{\text{ext}} = J^T \mathcal{F}_{\text{tip}}$, where the Jacobian $J(\theta)$ is expressed in terms of the same reference frame as $\mathcal{F}_{\text{tip}}$.

- **Output**: The joint acceleration $\ddot{\theta}$.

- **Initialization**: Assign temporary storage variables $A \in \mathbb{R}^{n \times n}$, $\gamma \in \mathbb{R}^n$, $c \in \mathbb{R}^n$.

- **Inverse dynamics calculation**: Calculate the inverse dynamics with θ and $\dot{\theta}$ set to their given values, and $\ddot{\theta}$ set to zero; store the output joint torques in γ, and set $c = \tau - \gamma - \tau_{\text{ext}}$.

- **Evaluation of mass matrix**: Calculate the mass matrix for the given θ, and store the result in A.

- **Calculation of joint acceleration**: Solve the linear system $Ax = c$ for x; the resulting joint acceleration $\ddot{\theta}$ is then given by x.

With the above algorithm for calculating the joint acceleration, various numerical schemes for integrating the forward dynamics can be implemented; here we present an algorithm for the most basic Euler method described above:

Euler Integration Algorithm for Forward Dynamics

- **Prerequisites**: Function *GetJointAccel($\theta, \dot{\theta}, \tau, \tau_{\text{ext}}$)* required.

- **Inputs**: Initial conditions $\theta(0)$ and $\dot{\theta}(0)$, input torques $\tau(t)$ and τ_{ext} for $t \in [0, t_f]$, integration timestep $h > 0$.

- **Output**: Joint trajectory values $\theta[k] = \theta(hk)$, $k = 0, \ldots, N$.

- **Initialization**: Set $N = t_f/h$.

- **Iteration**: For $k = 1$ to N do

$$
\begin{aligned}
\ddot{\theta}[k] &= GetJointAccel(\theta[k], \dot{\theta}[k], \tau[k], \tau_{\text{ext}}[k]); \\
\theta[k+1] &= \theta[k] + h\dot{\theta}[k]; \\
[\dot{\theta}](k+1) &= \dot{\theta}[k] + h\ddot{\theta}[k];
\end{aligned}
$$

- **Joint trajectory**: $\theta[k] = \theta(hk)$, $k = 0, \ldots, N$.

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 keep things simple we consider a six degree-of-freedom open chain with joint space dynamics

$$\tau = M(\theta)\ddot{\theta} + b(\theta, \dot{\theta}), \quad \theta \in \mathbb{R}^6, \ \tau \in \mathbb{R}^6. \tag{8.89}$$

We also ignore for the time being any external spatial forces that may be applied. The spatial velocity $\mathcal{V} = (\omega, v)$ of the end-effector is related to the joint velocity $\dot{\theta}$ by

$$\mathcal{V} = J(\theta)\dot{\theta}, \tag{8.90}$$

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

$$\dot{\mathcal{V}} = \dot{J}(\theta)\dot{\theta} + J(\theta)\ddot{\theta}. \tag{8.91}$$

At configurations θ where $J(\theta)$ is invertible, we have

$$\dot{\theta} = J^{-1}\mathcal{V} \tag{8.92}$$
$$\ddot{\theta} = J^{-1}\dot{\mathcal{V}} - J^{-1}\dot{J}J^{-1}\mathcal{V}. \tag{8.93}$$

The second term in (8.93) follows from the general matrix identity $\frac{d}{dt}(A^{-1}A) = \frac{d}{dt}A^{-1} \cdot A + A^{-1} \cdot \frac{d}{dt}A$ for any invertible and differentiable matrix $A(t)$. Substituting for $\dot{\theta}$ and $\ddot{\theta}$ in Equation (8.89) leads to

$$\tau = M\left(J^{-1}\dot{\mathcal{V}} - J^{-1}\dot{J}J^{-1}\mathcal{V}\right) + b(\theta, \dot{\theta}), \tag{8.94}$$

where J^{-T} denotes $(J^{-1})^T = (J^T)^{-1}$. Pre-multiply both sides by J^{-T} to get

$$\begin{aligned} J^{-T}\tau &= J^{-T}MJ^{-1}\dot{\mathcal{V}} - J^{-T}MJ^{-1}\dot{J}J^{-1}\mathcal{V} \\ &\quad + J^{-T}b(\theta, J^{-1}\mathcal{V}). \end{aligned} \tag{8.95}$$

Expressing $J^{-T}\tau$ as the spatial force $\mathcal{F}$, the above can be written

$$\mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \eta(\theta, \mathcal{V}), \tag{8.96}$$

where

$$\Lambda(\theta) = J^{-T}M(\theta)J^{-1} \tag{8.97}$$
$$\eta(\theta, \mathcal{V}) = J^{-T}b(\theta, J^{-1}\mathcal{V}) - \Lambda\dot{J}J^{-1}\mathcal{V}. \tag{8.98}$$

These are the dynamic equations expressed in end-effector frame coordinates. If an external spatial force $\mathcal{F}$ is applied to the end-effector frame, then assuming zero joint torques, the motion of the end-effector frame is governed by these equations. Note the dependence of $\Lambda(\theta)$ and $\eta(\theta, \mathcal{V})$ on θ. If θ 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 spatial velocity $\mathcal{V}$. In practice, since X is usually obtained by measuring θ and substituting into the forward kinematics, it is preferable to leave the dependence on θ explicit.

8.7   Summary

8.8   Notes and References

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 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
Θ, $\theta : [0, 1] \to \Theta$. 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

written as

$$\dot{\theta} = \frac{d\theta}{ds}\dot{s} \tag{9.1}$$

$$\ddot{\theta} = \frac{d\theta}{ds}\ddot{s} + \frac{d^2\theta}{ds^2}\dot{s}^2. \tag{9.2}$$

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 θ_{start} to an end configuration θ_{end} could be defined in joint space or in task space. The advantage of a straight-line path from θ_{start} to θ_{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 angle θ_i, the allowable joint configurations form a convex set Θ_{free} in joint space, so the straight line between any two endpoints in Θ_{free} also lies in Θ_{free}. The straight line can be written

$$\theta(s) = (1 - s)\theta_{\text{start}} + s\theta_{\text{end}}, \quad s \in [0, 1] \tag{9.3}$$

with derivatives

$$\frac{d\theta}{ds} = \theta_{\text{end}} - \theta_{\text{start}} \tag{9.4}$$

$$\frac{d^2\theta}{ds^2} = 0. \tag{9.5}$$

Straight lines in joint space generally do not yield straight-line motion of the end-effector in Cartesian space. If Cartesian straight-line motions are desired, the start and end configurations can be specified by X_{start} and X_{end} in task space. If X_{start} and X_{end} are represented by a minimum set of coordinates, then a straight line is defined as $X(s) = (1 - s)X_{\text{start}} + sX_{\text{end}}, s \in [0, 1]$. Compared to joint coordinates, however, the following are issues that must be addressed:

- Inverse kinematics must be used to find joint coordinates θ for the robot controller.

- If the path passes near a kinematic singularity, joint velocities may become unreasonably large for almost all time scalings of the path.
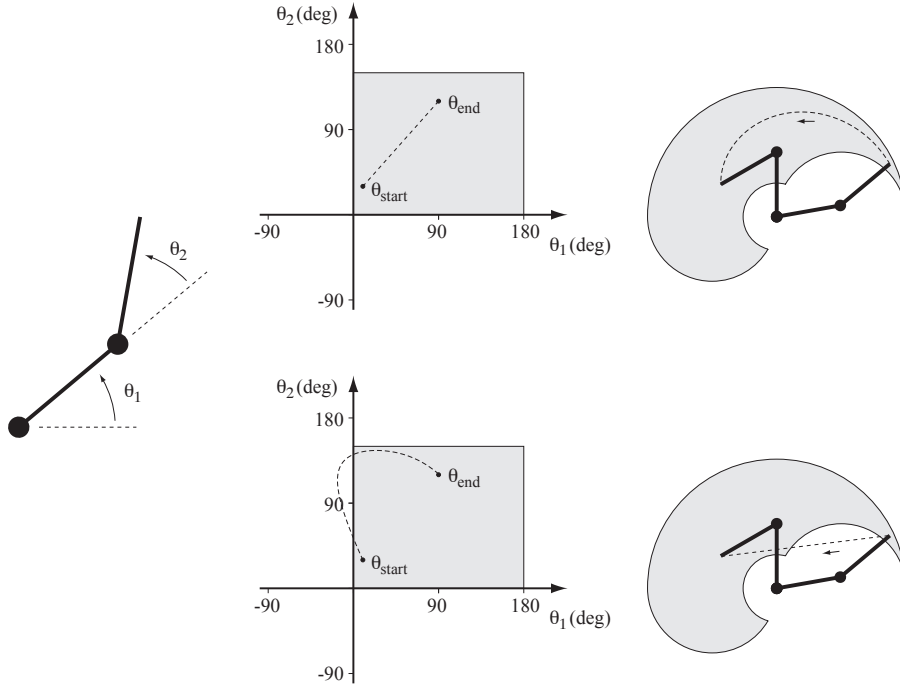
Figure 9.1: (Left) A 2R robot with joint limits $0° \leq \theta_1 \leq 180°$, $0° \leq \theta_2 \leq 150°$. (Top right) A straight-line path in joint space and the corresponding motion of the end-effector in task space. The reachable endpoint configurations, subject to joint limits, are indicated in gray. (Bottom right) A straight-line path in task space would violate the joint limits.

- 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_{start} and X_{end} are represented as elements of $SE(3)$ instead of as a minimum set of coordinates, then there is the question of how to define a "straight" line in $SE(3)$. A configuration of the form $(1 - s)X_{\text{start}} + sX_{\text{end}}$ does not generally lie in $SE(3)$.

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 spatial velocity, expressed in the {start} frame, that takes X_{start} to X_{end} in unit time. The path

can therefore be written as

$$X(s) = X_{\text{start}} \exp(\log(X_{\text{start}}^{-1} X_{\text{end}})s) \tag{9.6}$$

where X_{start} is post-multiplied by the matrix exponential since the spatial velocity is represented in the {start} frame, not the fixed world frame {s}.

This screw motion provides a "straight-line" motion in the sense that the spatial velocity (and the screw axis) is constant. The origin of the end-effector does not generally follow a straight line in Cartesian space, however, since it rotates about the axis. It may be preferable to decouple the rotational motion from the translational motion. Writing $X = (R, p)$, we can define the "straight-line" path to be

$$p(s) = (1 - s)p_{\text{start}} + sp_{\text{end}} \tag{9.7}$$

$$R(s) = R_{\text{start}} \exp(\log(R_{\text{start}}^T R_{\text{end}})s). \tag{9.8}$$

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 Equation (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 θ, $\dot{\theta}$, and $\ddot{\theta}$ by X, $\dot{X}$, and $\ddot{X}$. Inverse kinematics is then used to convert to an equivalent representation in joint space.

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,

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3. \tag{9.9}$$

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

$$\dot{s}(t) = a_1 + 2a_2 t + 3a_3 t^2 \tag{9.10}$$

at $t = 0$ and $t = T$ and solving the four constraints for $a_0, \ldots, a_3$, we find

$$a_0 = 0, \quad a_1 = 0, \quad a_2 = \frac{3}{T^2}, \quad a_3 = -\frac{2}{T^3}.$$

Plots of $s(t)$ and $\dot{s}(t)$ are shown in Figure 9.2.
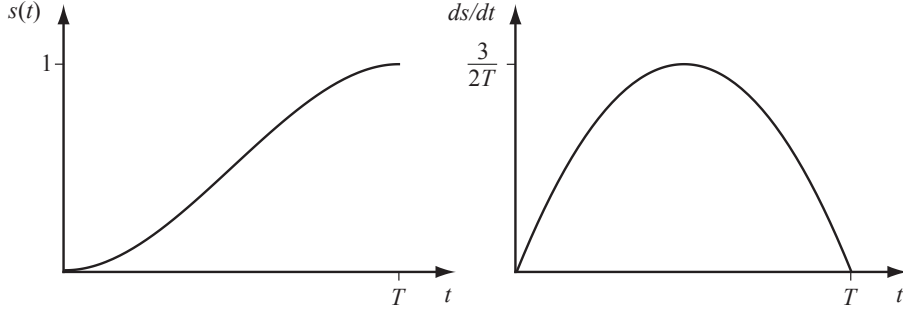
Figure 9.2: Plots of $s(t)$ and $\dot{s}(t)$ for a third-order polynomial time scaling.

Plugging $s = a_2t^2 + a_3t^3$ into Equation (9.3) yields

$$\theta(t) = \theta_{\text{start}} + \left(\frac{3t^2}{T^2} - \frac{2t^3}{T^3} \right) (\theta_{\text{end}} - \theta_{\text{start}}) \tag{9.11}$$

$$\dot{\theta}(t) = \left(\frac{6t}{T^2} - \frac{6t^2}{T^3} \right) (\theta_{\text{end}} - \theta_{\text{start}}) \tag{9.12}$$

$$\ddot{\theta}(t) = \left(\frac{6}{T^2} - \frac{12t}{T^3} \right) (\theta_{\text{end}} - \theta_{\text{start}}). \tag{9.13}$$

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 $t = T$:

$$\ddot{\theta}_{\text{max}} = \left| \frac{6}{T^2} (\theta_{\text{end}} - \theta_{\text{start}}) \right|, \quad \ddot{\theta}_{\text{min}} = - \left| \frac{6}{T^2} (\theta_{\text{end}} - \theta_{\text{start}}) \right|.$$

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.

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
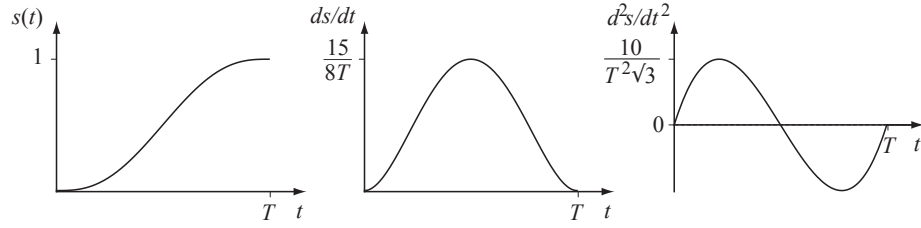
Figure 9.3: Plots of $s(t)$, $\dot{s}(t)$, and $\ddot{s}(t)$ for a fifth-order polynomial time scaling.
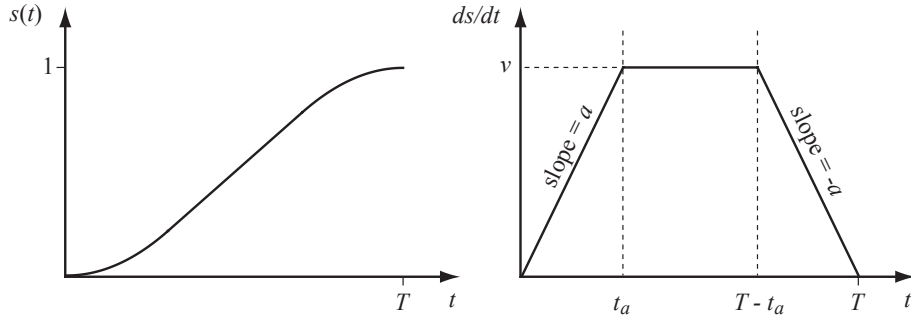
Figure 9.4: Plots of $s(t)$ and $\dot{s}(t)$ for a trapezoidal motion profile.

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 9.3.

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 9.4).

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 $\dot{\theta}_{\text{limit}}$ and joint accelerations $\ddot{\theta}_{\text{limit}}$, the trapezoidal motion using the largest v and a satisfying

$$|(\theta_{\text{end}} - \theta_{\text{start}})v| \leq \dot{\theta}_{\text{limit}} \tag{9.14}$$

$$|(\theta_{\text{end}} - \theta_{\text{start}})a| \leq \ddot{\theta}_{\text{limit}} \tag{9.15}$$

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 motion, and the trapezoidal profile $\dot{s}(t)$ in Figure 9.4 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

$$0 \leq t \leq \frac{v}{a} \; : \qquad \ddot{s}(t) = a \tag{9.16}$$

$$\dot{s}(t) = at \tag{9.17}$$

$$s(t) = \frac{1}{2}at^2 \tag{9.18}$$

$$\frac{v}{a} < t \leq T - \frac{v}{a} \; : \qquad \ddot{s}(t) = 0 \tag{9.19}$$

$$\dot{s}(t) = v \tag{9.20}$$

$$s(t) = vt - \frac{v^2}{2a} \tag{9.21}$$

$$T - \frac{v}{a} < t \leq T \; : \qquad \ddot{s}(t) = -a \tag{9.22}$$

$$\dot{s}(t) = a(T - t) \tag{9.23}$$

$$s(t) = \frac{2avT - 2v^2 - a^2(t - T)^2}{2a}. \tag{9.24}$$

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.

- 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:

$$a = \frac{v^2}{vT - 1}.$$
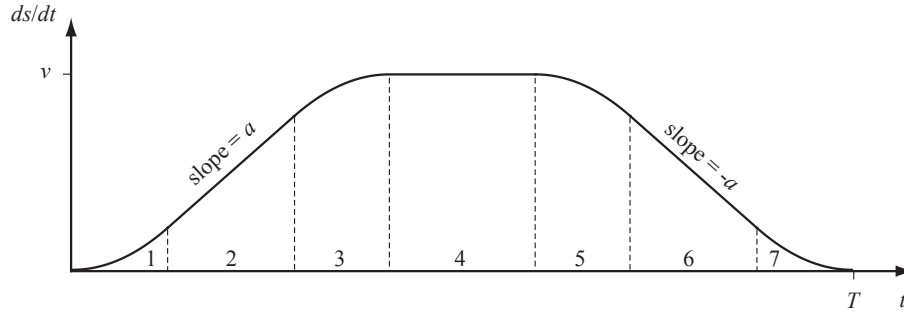
Figure 9.5: Plot of $\dot{s}(t)$ for an S-curve motion profile consisting of seven stages: (1) constant positive jerk, (2) constant acceleration, (3) constant negative jerk, (4) constant velocity, (5) constant negative jerk, (6) constant deceleration, and (7) constant positive jerk.

- Choose 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 $\dot{s} = v$ 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 9.5.

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, a simple
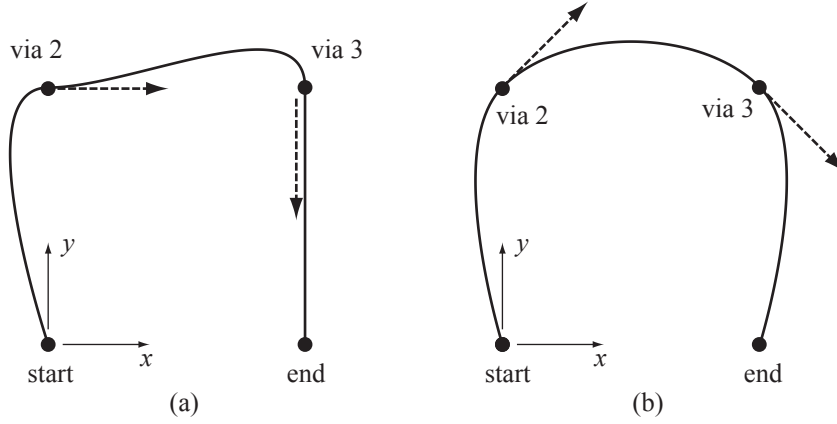
Figure 9.6: Two paths in an (x, y) space corresponding to piecewise-cubic trajectories interpolating four via points, including one start point and one end point. The velocities at the start and end are zero, and the velocities at vias 2 and 3 are indicated by dashed tangent vectors. The shape of the path depends on the velocities specified at the via points.

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 9.6).

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 angle and call it β to avoid proliferation of subscripts. At each via point $i \in \{1 \dots k\}$, the user specifies the desired position $\beta(T_i) = \beta_i$ and velocity $\dot{\beta}(T_i) = \dot{\beta}_i$. The trajectory has $j = k - 1$ segments, and the duration of segment $j \in \{1, \dots, k-1\}$ is $\Delta T_j = T_{j+1} - T_j$. The joint trajectory during segment j is expressed as the third-order polynomial

$$\beta(\Delta t) = a_{j0} + a_{j1}\Delta t + a_{j2}\Delta t^2 + a_{j3}\Delta t^3 \qquad (9.25)$$

in terms of the time Δt elapsed in segment j, where $0 \leq \Delta t \leq \Delta T_j$. Segment j is subject to the four constraints

$$\beta(0) = \beta_j \qquad \qquad \dot{\beta}(0) = \dot{\beta}_j$$
$$\beta(\Delta T_j) = \beta_{j+1} \qquad \dot{\beta}(\Delta T_j) = \dot{\beta}_{j+1}.$$

Solving these constraints for $a_{j0}, \dots, a_{j3}$ yields

$$a_{j0} = \beta_j \qquad \qquad (9.26)$$

$$a_{j1} = \dot{\beta}_j \qquad \qquad (9.27)$$

$$a_{j2} = \frac{3\beta_{j+1} - 3\beta_j - 2\dot{\beta}_j\Delta T_j - \dot{\beta}_{j+1}\Delta T_j}{\Delta T_j^2} \qquad (9.28)$$

$$a_{j3} = \frac{2\beta_j + (\dot{\beta}_j + \dot{\beta}_{j+1})\Delta T_j - 2\beta_{j+1}}{\Delta T_j^3}. \qquad (9.29)$$
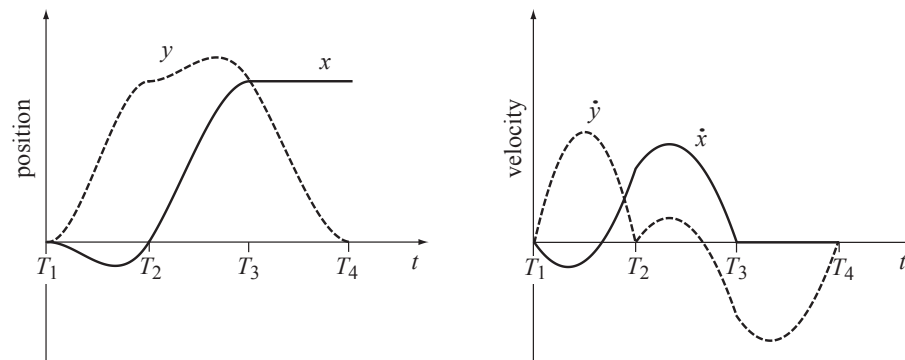
Figure 9.7: The coordinate time histories for the cubic via point interpolation of Figure 9.6(a).

Figure 9.7 shows the time histories for the interpolation of Figure 9.6(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 9.6(b) is smoother than the trajectory of Figure 9.6(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 point-to-point cubic polynomial time-scaled trajectory discussed in Section 9.2.2.1.

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 9.6. This can be important to ensure that joint limits or workspace obstacles are respected.
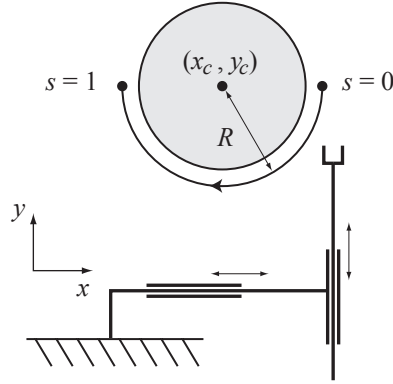
Figure 9.8: A path planner has returned a semicircular path of radius R around an obstacle in (x, y) space for a robot with two prismatic joints. The path can be represented as a function of a path parameter s as $x(s) = x_c + R\cos s\pi$ and $y(s) = y_c - R\sin s\pi$ for $s \in [0, 1]$. For a 2R robot, inverse kinematics would be used to express the path as a function of s in joint coordinates.

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 9.8), 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 that the maximum acceleration a and maximum coasting velocity v are constant. For most robots, because of state-dependent joint actuator limits and the state-dependent dynamics

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + g(\theta) = \tau, \qquad (9.30)$$

the maximum available velocities and accelerations change along the path.

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 ith actuator as

$$\tau_i^{\min}(\theta, \dot{\theta}) \leq \tau_i \leq \tau_i^{\max}(\theta, \dot{\theta}). \qquad (9.31)$$

The available actuator torque is typically a function of the state of the actuator. 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 Coriolis terms in Equation (9.30) can be written equivalently as

$$C(\theta, \dot{\theta})\dot{\theta} = \dot{\theta}^T \Gamma(\theta)\dot{\theta},$$

where $\Gamma(\theta)$ is the three-dimensional tensor of *Christoffel symbols* constructed of partial derivatives of components of the inertia matrix $M(\theta)$ with respect to θ. This form more clearly shows the Coriolis terms' 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

$$\underbrace{\left(M(\theta(s))\frac{d\theta}{ds}\right)}_{m(s)} \ddot{s} + \underbrace{\left(M(\theta(s))\frac{d^2\theta}{ds^2} + \left(\frac{d\theta}{ds}\right)^T \Gamma(\theta(s))\frac{d\theta}{ds}\right)}_{c(s)} \dot{s}^2 + \underbrace{g(\theta(s))}_{g(s)} = \tau, \quad (9.32)$$

expressed more compactly as the vector equation

$$m(s)\ddot{s} + c(s)\dot{s}^2 + g(s) = \tau, \tag{9.33}$$

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:

$$\tau_i^{\min}(s, \dot{s}) \leq \tau_i \leq \tau_i^{\max}(s, \dot{s}). \tag{9.34}$$

Plugging in the components of Equation (9.33), we get

$$\tau_i^{\min}(s, \dot{s}) \leq m_i(s)\ddot{s} + c_i(s)\dot{s}^2 + g_i(s) \leq \tau_i^{\max}(s, \dot{s}). \tag{9.35}$$

Let $L_i(s, \dot{s})$ and $U_i(s, \dot{s})$ be the minimum and maximum accelerations $\ddot{s}$ satisfying the ith component of Equation (9.35). Depending on the sign of $m_i(s)$, we have three possibilities:

$$
\begin{aligned}
\text{if } m_i(s) > 0 : \quad L_i(s) &= \frac{\tau_i^{\min}(s, \dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)} \\
U_i(s) &= \frac{\tau_i^{\max}(s, \dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)} \\
\text{if } m_i(s) < 0 : \quad L_i(s) &= \frac{\tau_i^{\max}(s, \dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)} \\
U_i(s) &= \frac{\tau_i^{\min}(s, \dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)} \\
\text{if } m_i(s) = 0 : \quad &\textit{zero-inertia point}, \text{ discussed in Section 9.4.3}
\end{aligned}
\tag{9.36}
$$

Defining

$$L(s, \dot{s}) = \max_i L_i(s, \dot{s}) \qquad \text{and} \qquad U(s, \dot{s}) = \min_i U_i(s, \dot{s}),$$
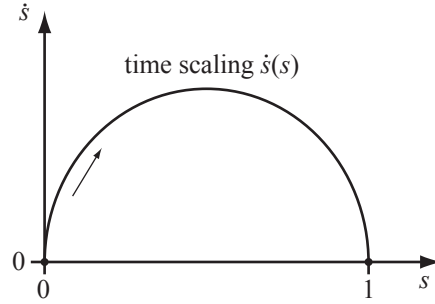
Figure 9.9: A time scaling in the $(s, \dot{s})$ phase plane is a curve with $\ddot{s} \geq 0$ at all times connecting the initial path position and velocity $(0, 0)$ to the final position and velocity $(1, 0)$.

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 9.9). 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 9.10(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 grey in Figure 9.10(a). For any s, typically there is a single limit velocity $\dot{s}_{\mathrm{lim}}(s)$ above which all velocities are inadmissible. The
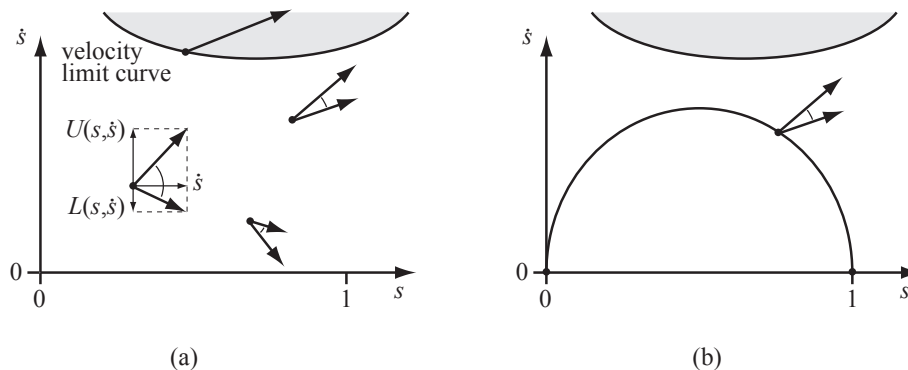
Figure 9.10: (a) Acceleration-limited motion cones at four different states. The upper ray of the cone is the sum of $U(s, \dot{s})$ plotted in the vertical direction (change in velocity) and $\dot{s}$ plotted in the horizontal direction (change in position). The lower ray of the cone is constructed from $L(s, \dot{s})$ and $\dot{s}$. Points in grey, 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 tangent vector. (b) The proposed time scaling is infeasible because the tangent to the curve is outside the motion cone at the state indicated.

function $\dot{s}_{\text{lim}}(s)$ is called the *velocity limit curve*. On the velocity limit curve, $L(s, \dot{s}) = U(s, \dot{s})$, and the cone reduces to a single vector.

For a time scaling to satisfy the acceleration constraints, the tangent of the time-scaling curve must lie inside the feasible cone at all points on the curve. This shows that the time scaling in Figure 9.10(b) is infeasible; it demands more deceleration than the actuators can provide at the state indicated.

For a minimum-time motion, the velocity $\dot{s}$ must be as high as possible at every s while still satisfying the acceleration constraints and the endpoint 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 9.11(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 9.11(b)). These cases require an algorithm to find multiple switching points.
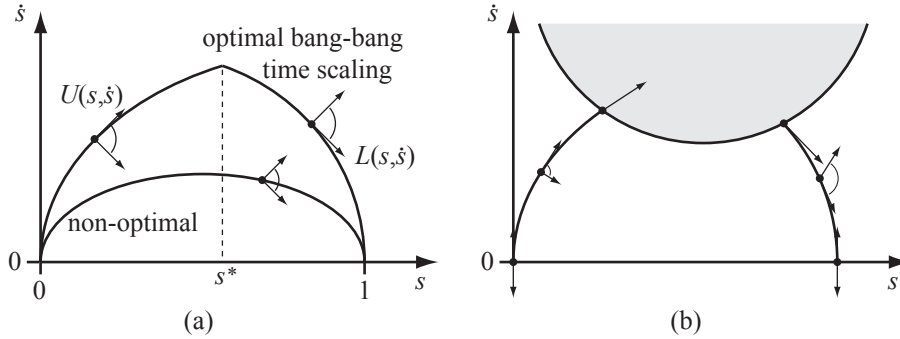
Figure 9.11: (a) A time-optimal bang-bang time scaling integrates $U(s, \dot{s})$ from $(0,0)$ and switches to $L(s, \dot{s})$ at a switching point s^*. Also shown is a non-optimal time scaling with a tangent inside a motion cone. (b) Sometimes the velocity limit curve prevents a single-switch solution.

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 $\mathcal{S} = \{\}$ 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 $\mathcal{S}$. This is a switch from maximum acceleration to maximum deceleration. The problem is solved and $\mathcal{S}$ is the set of switches expressed in the path parameter. If instead the velocity limit curve is penetrated, let $(s_{\text{lim}}, \dot{s}_{\text{lim}})$ be the point of penetration and proceed to the next step.

4. Perform a binary search on the velocity in the range $[0, \dot{s}_{\text{lim}}]$ to find the velocity $\dot{s}'$ such that the curve integrating $\ddot{s} = L(s, \dot{s})$ forward from $(s_{\text{lim}}, \dot{s}')$ touches the velocity limit curve without penetrating it. Set $\dot{s}_{\text{high}} = \dot{s}_{\text{lim}}$ and $\dot{s}_{\text{low}} = 0$.

   (a) Set the test velocity halfway between s_{low} and s_{high}, $\dot{s}_{\text{test}} = (\dot{s}_{\text{high}} + \dot{s}_{\text{low}})/2$. The test point is $(s_{\text{lim}}, \dot{s}_{\text{test}})$.

(b) If the curve from the test point penetrates the velocity limit curve, set $\dot{s}_{\text{high}}$ equal to $\dot{s}_{\text{test}}$. If instead the curve from the test point hits $\dot{s} = 0$, set $\dot{s}_{\text{low}}$ equal to $\dot{s}_{\text{test}}$. Return to step 4a.

Continue the binary search until a specified tolerance. Let $(s_{\text{tan}}, \dot{s}_{\text{tan}})$ equal the point where the resulting curve just touches the velocity limit curve tangentially. 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 $(s_{\text{tan}}, \dot{s}_{\text{tan}})$ 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 $(s_{\text{tan}}, \dot{s}_{\text{tan}})$. Append s_i to the list of switches $\mathcal{S}$. This is a switch from maximum acceleration to maximum deceleration.

6. Increment i and set $(s_i, \dot{s}_i)$ to $(s_{\text{tan}}, \dot{s}_{\text{tan}})$. Append s_i to the list of switches $\mathcal{S}$. This is a switch from maximum deceleration to maximum acceleration. Go to step 3.

The algorithm is illustrated in Figure 9.12.

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 arbirarily 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 cases where this assumption is violated.

- *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 ith 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
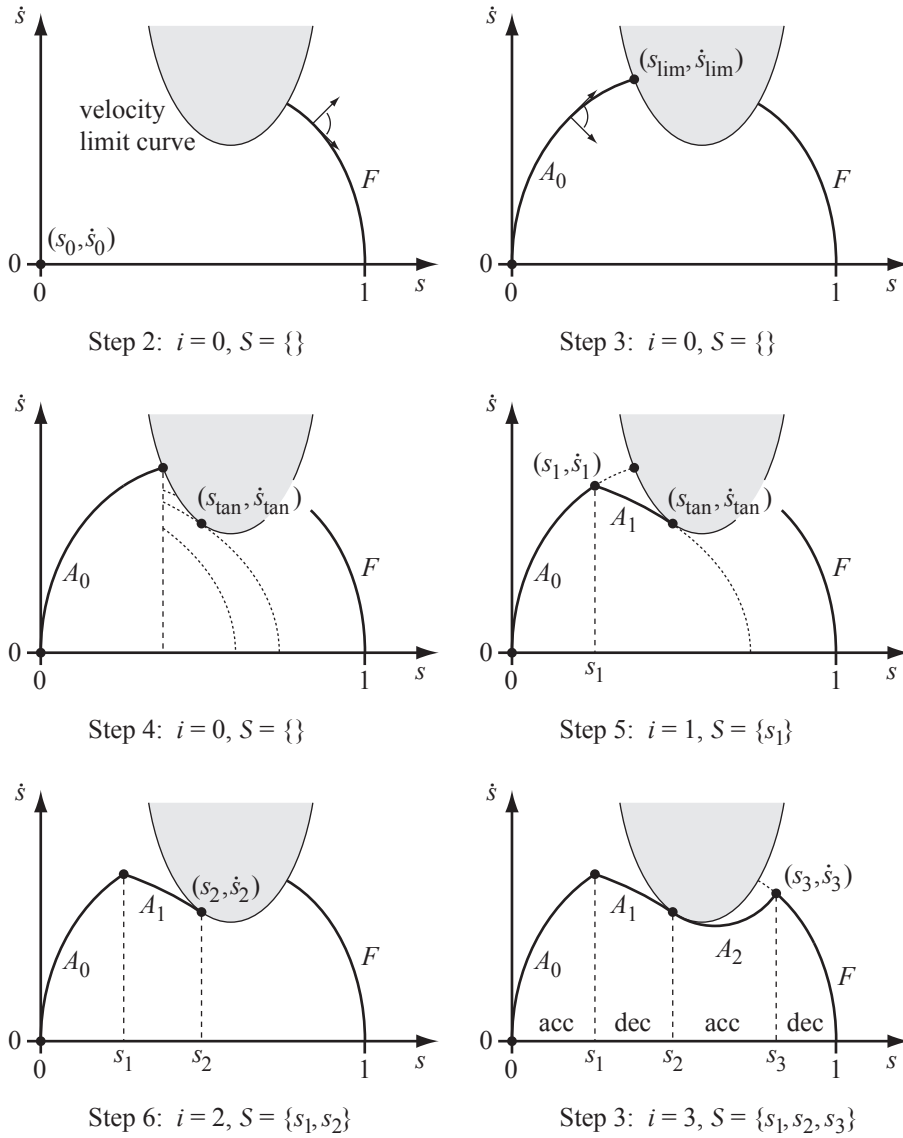
Figure 9.12: The time-scaling algorithm. (Step 2) Integrating $\ddot{s} = L(s, \dot{s})$ backward from $(1, 0)$ until the velocity limit curve is reached. (Step 3) Integrating $\ddot{s} = U(s, \dot{s})$ forward from $(0, 0)$ to the intersection $(s_{\text{lim}}, \dot{s}_{\text{lim}})$ with the velocity limit curve. (Step 4) Binary search to find $(s_{\text{lim}}, \dot{s}')$ from which $\ddot{s} = L(s, \dot{s})$, integrated forward from $(s_{\text{lim}}, \dot{s}')$, touches the velocity limit curve tangentially. (Step 5) Integrating backward along $L(s, \dot{s})$ from $(s_{\text{tan}}, \dot{s}_{\text{tan}})$ to find the first switch from acceleration to deceleration. (Step 6) The second switch, from deceleration to acceleration, is at $(s_2, \dot{s}_2) = (s_{\text{tan}}, \dot{s}_{\text{tan}})$. (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 $\mathcal{S} = \{s_1, s_2, s_3\}$.

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.

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 $\theta(t)$, $\theta : [0, T] \to \Theta$, can be written as $\theta(s(t))$, the composition of a path $\theta(s)$, $\theta : [0, 1] \to \Theta$, and a time scaling $s(t)$, $s : [0, T] \to [0, 1]$.

- A straight-line path in joint space can be written $\theta(s) = (1-s)\theta_{\text{start}} + s\theta_{\text{end}}$, $s \in [0, 1]$. 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:

$$p(s) = (1 - s)p_{\text{start}} + sp_{\text{end}} \tag{9.38}$$

$$R(s) = R_{\text{start}} \exp(\log(R_{\text{start}}^T R_{\text{end}})s). \tag{9.39}$$

- 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 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

$$L(s, \dot{s})\ddot{s} \leq \ddot{s} \leq U(s, \dot{s}).$$

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 $s(0) = \dot{s}(0) = \dot{s}(T) = 0$, $s(T) = 1$, and the actuator constraints. The optimal solution always operates at maximum acceleration $U(s, \dot{s})$ or maximum deceleration $L(s, \dot{s})$.
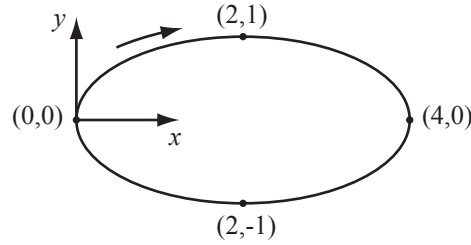
Figure 9.13: An elliptical path.

9.6   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 9.13). 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 $\dot{X}$ and $\ddot{X}$.

3. Plot the acceleration as a function of time $t \in [0, T]$ for a point-to-point cubic time scaling.

4. Consider a straight-line path $\theta(s) = (1 - s)\theta_{\text{start}} + s\theta_{\text{end}}, s \in [0, 1]$ from $\theta_{\text{start}} = (0, 0)$ to $\theta_{\text{end}} = (\pi, \pi/3)$. The feasible joint velocities are $|\dot{\theta}_1|, |\dot{\theta}_2| \leq 2$ rad/s and the feasible joint accelerations are $|\ddot{\theta}_1|, |\ddot{\theta}_2| \leq 0.5$ rad/s². 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 $s(0) = \dot{s}(0) = \ddot{s}(0) = \dot{s}(T) = \ddot{s}(T) = 0$.

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 minimum order of the polynomial?

8. Prove that the trapezoidal time scaling, using the maximum allowable ac-

celeration 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 generator program for a 2-DOF robot. 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 $t = T$,
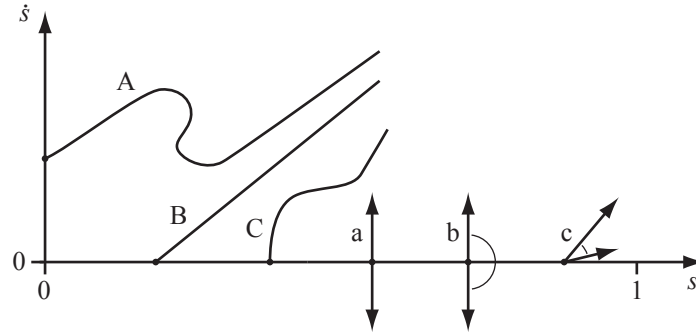
Figure 9.14: A, B, and C are candidate integral curves, originating from the dots indicated, while a, b, and c are candidate motion cones at $\dot{s} = 0$. Two of the integral curves and two of the motion cones are incorrect.

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 9.6) and (b) the position and velocity of each joint as a function of time (similar to Figure 9.7).

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 ΔT_j, with β_j, β_{j+1}, $\dot{\beta}_j$, $\dot{\beta}_{j+1}$, $\ddot{\beta}_j$, and $\ddot{\beta}_{j+1}$ 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 9.14 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, joint limits, and 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 wheeled 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 $\mathcal{C}$ corresponds to a unique configuration q of the robot, and every configuration of the robot can be represented as a point in C-space. For example, the configuration of a robot arm with n joints can be represented as a list of n joint angles, $q = (\theta_1, \ldots, \theta_n)$. The *free C-space* $\mathcal{C}_{\text{free}}$
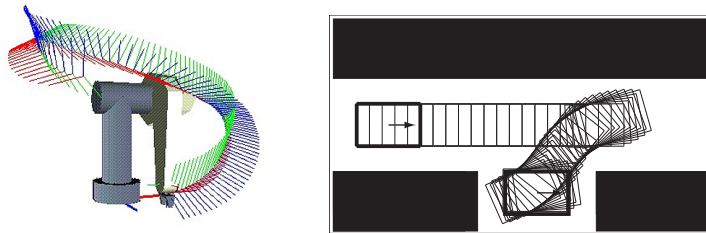
Figure 10.1: (Left) A robot arm executing a motion plan. (Right) A car-like mobile robot parallel parking.

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 $\mathcal{C} \subset \mathbb{R}^n$. With some generalization, however, the concepts of this chapter apply to non-Euclidean C-spaces like $\mathcal{C} = SE(3)$ ($n = 6$).

The control inputs available to drive the robot are written as an m-vector $u \in \mathcal{U} \subset \mathbb{R}^m$, 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 $\mathcal{X}_{\text{free}} = \{x \mid q(x) \in \mathcal{C}_{\text{free}}\}$.

The equations of motion of the robot are written

$$\dot{x} = f(x, u), \tag{10.1}$$

or, in integral form,

$$x(T) = x(0) + \int_0^T f(x(t), u(t))dt. \tag{10.2}$$

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 $x(0) = x_{start}$ and a desired final state x_{goal}, find a time T and a set of controls $u : [0, T] \to \mathcal{U}$ such that the motion (10.2) satisfies $x(T) = x_{goal}$ and $q(x(t)) \in \mathcal{C}_{free}$ for all $t \in [0, T]$.

The goal state x_{goal} can be replaced by a set of acceptable states, $\mathcal{X}_{\text{goal}}$.

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 $\mathcal{C}_{\text{free}}$ 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 sub-problem 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.

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, an 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_{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 $\mathcal{C}_{\text{free}}$. 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 not. 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 $\mathcal{C}_{\text{free}}$. 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 $\mathcal{C}_{\text{free}}$, 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 $\mathcal{C}_{\text{free}}$ into a grid and search the grid for a motion from q_{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 $\mathcal{C}_{\text{free}}$ 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 $\mathcal{X}_{\text{free}}$; 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. 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 satisficing, 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 or 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 most of them: configuration space obstacles, collision detection, and graphs.

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 $\mathcal{C} = \mathcal{C}_{\text{free}} \bigcup \mathcal{C}_{\text{obs}}$. 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_{start} and q_{goal} do not lie in the same connected component, then there is no collision-free path.
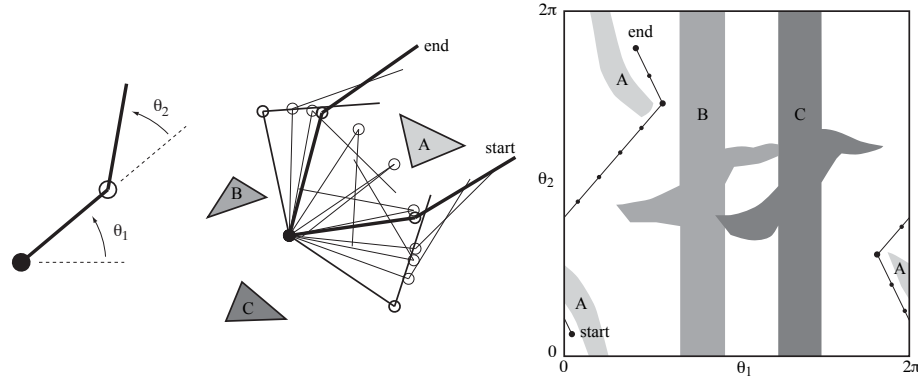
Figure 10.2: (Left) The joint angles of a 2R robot arm. (Middle) The arm navigating among obstacles. (Right) The same motion in C-space.

For technical reasons, a configuration that has the robot just barely touching an obstacle, without penetrating, is typically considered part of the C-space obstacle (or C-obstacle for short). In other words, C-obstacles are *closed* (they contain their boundaries), while $\mathcal{C}_{\text{free}}$ is *open*: from any point in $\mathcal{C}_{\text{free}}$, it is possible to move in any direction, perhaps only infinitesimally, while remaining in $\mathcal{C}_{\text{free}}$.

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.

Although the C-space is drawn as a subset of $\mathbb{R}^2$, it is correctly described as $S^1 \times S^1$, where S^k is the k-dimensional "sphere" embedded in $\mathbb{R}^{k+1}$. Thus S^1 is a circle on a plane, representing a single angle, and S^2 is a sphere in 3-space
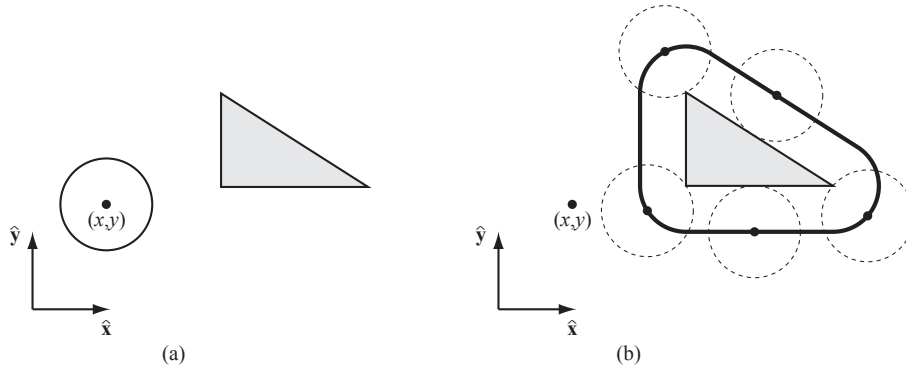
Figure 10.3: (a) A circular mobile robot (white) and a workspace obstacle (grey). The configuration of the robot is represented by (x, y), the center of the robot. (b) In the C-space, the obstacle is "grown" by the radius of the robot and the robot is treated as a point. Any (x, y) configuration outside the dark boundary is collision-free.

(e.g., the surface of the Earth). A robot with n revolute joints with no joint limits has a C-space $S^1 \times \ldots \times S^1$ (n times), which is written more compactly as T^n, the n-dimensional torus in $\mathbb{R}^{n+1}$. Note that $S^1 \times S^1 = T^2 \neq S^2$; the topology of a 2-torus and a 2-sphere are different.

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 $(x, y) \in \mathbb{R}^2$ location of its center. 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 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

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.

10.2.1.4 A Polygonal Planar Mobile Robot That 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 $\theta \in [0, 2\pi)$.
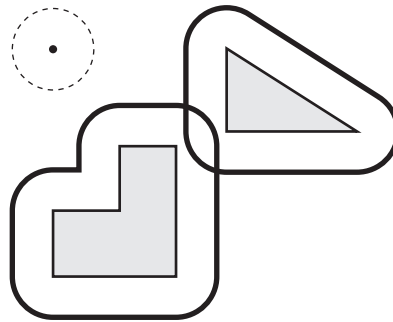
Figure 10.4: The grown C-space obstacles corresponding to two workspace obstacles and a circular mobile robot. The overlapping boundaries mean that the robot cannot move between the two obstacles.
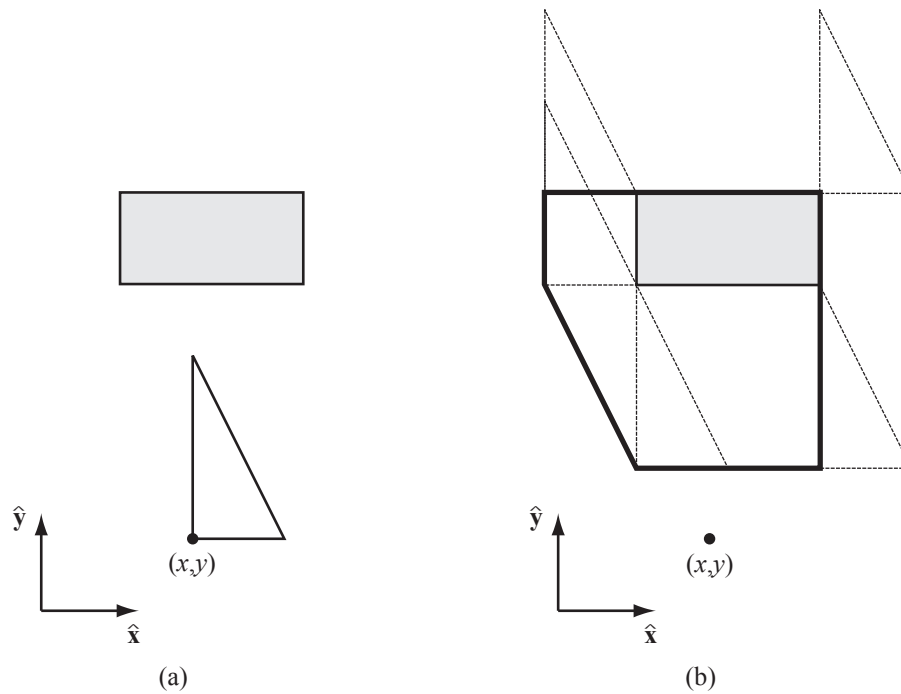
Figure 10.5: (a) The configuration of a triangular mobile robot, which can translate but not rotate, is represented by the (x, y) location of a reference point. Also shown is a workspace obstacle in grey. (b) The corresponding C-space obstacle is obtained by sliding the robot around the boundary of the obstacle and tracing the position of the reference point.

Even for this relatively low-dimensional C-space, an exact representation of the
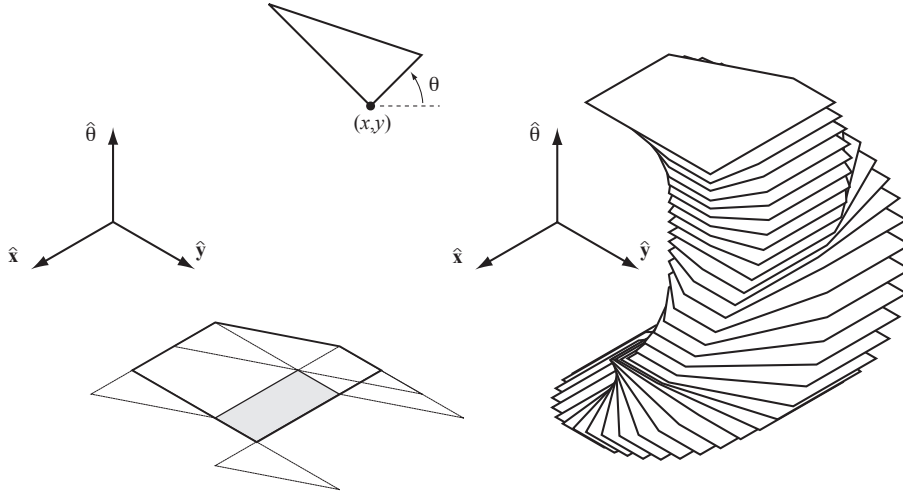
Figure 10.6: (Top) A triangular mobile robot that can rotate and translate, represented by the configuration (x, y, θ). (Left) The C-space obstacle from Figure 10.5(b) when the robot is restricted to $\theta = 0$. (Right) The full 3-D C-space obstacle shown in slices at $10°$ increments.

C-obstacle is quite complex. For this reason, C-obstacles are rarely described exactly.

10.2.2 Distance to Obstacles and Collision Detection

Given a C-obstacle $\mathcal{B}$ and a configuration q, let $d(q, \mathcal{B})$ be the distance between the robot and the obstacle, where

$$
\begin{aligned}
d(q, \mathcal{B}) > 0 &\quad \text{no contact with the obstacle} \\
d(q, \mathcal{B}) = 0 &\quad \text{contact} \\
d(q, \mathcal{B}) < 0 &\quad \text{penetration.}
\end{aligned}
$$

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 $d(q, \mathcal{B})$ for a given $\mathcal{B}$. A *collision-detection routine* determines whether $d(q, \mathcal{B}_i) \leq 0$ for any C-obstacle $\mathcal{B}_i$. 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.
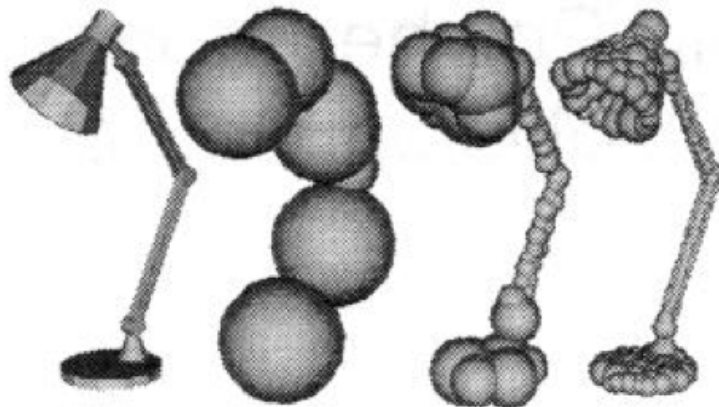
Figure 10.7: A lamp represented by spheres. The approximation improves as the number of spheres used to represent the lamp increases.

An even 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 $\mathcal{B}$ represented by ℓ 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

$$d(q, \mathcal{B}) = \min_{i,j} \|r_i(q) - b_j\| - R_i - B_j.$$

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
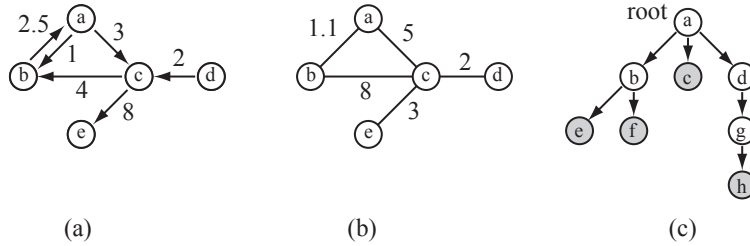
Figure 10.8: (a) A weighted digraph. (b) A weighted undirected graph. (c) A tree. Leaves are shaded grey.

n_2 indicates the ability to move from n_1 to n_2 without penetrating an obstacle or violating other constraints.

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.

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.

Given N nodes, any 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.

10.3 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 any 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.
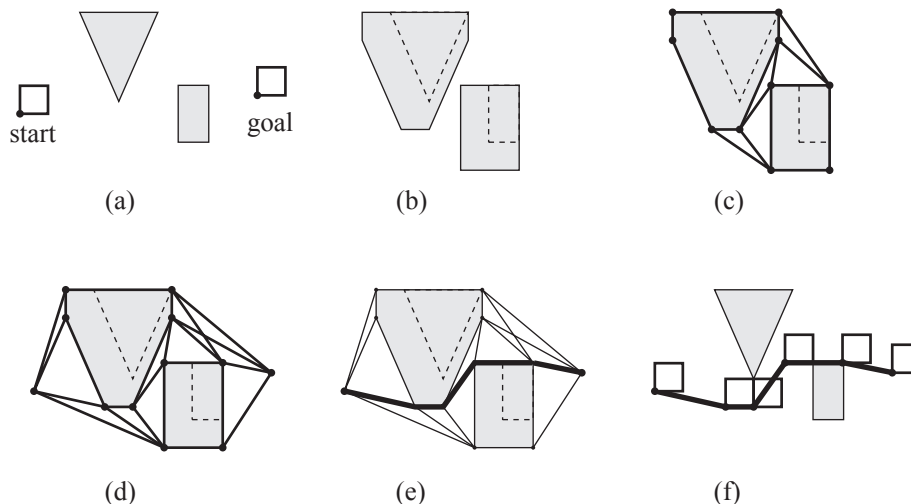
Figure 10.9: (a) The start and goal configurations for a square mobile robot (reference point shown) in an environment with a triangular and a rectangular obstacle. (b) The grown C-obstacles. (c) The visibility graph roadmap R of $\mathcal{C}_{\text{free}}$. (d) The full graph consists of R plus nodes at q_{start} and q_{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.

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_{start} to a point $q'_{\text{start}} \in R$, from a point $q'_{\text{goal}} \in R$ to q_{goal}, and from q'_{start} to q'_{goal} on the roadmap R. If a path can be found trivially between q_{start} and q_{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 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_{start} to q_{goal}, or consist of a straight line from q_{start} to a node $q'_{\text{start}} \in R$, a straight line from a node $q'_{\text{goal}} \in R$ to q_{goal}, and a path along the straight edges of R from q'_{start} to q'_{goal} (Figure 10.9). Note that the shortest path requires the robot to graze the obstacles, so we implicitly treat $\mathcal{C}_{\text{free}}$ as closed (i.e., including its boundary).

10.3.1 A^* **Search**

As with many path planners, central to the visibility graph planner is a graph search. A popular search algorithm is A^* (pronounced "A star"), which efficiently finds a minimum-cost path on a graph when the cost of the path is simply the sum of positive edge costs.

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 `OPEN` of the nodes to be explored from, and a list `CLOSED` of nodes that have already been explored from;

- an array `past_cost[node]` of the minimum cost found so far to reach node `node` from the start node; and

- a search tree defined by an array `parent[node]`, which contains a link for each `node` to the node preceding it in the shortest path found so far to `node`.

To initialize the search, the list `OPEN` is initialized to the start node 1, the cost to reach the start node (`past_cost[1]`) is initialized as 0, and `past_cost[node]` for `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 `OPEN` is removed from `OPEN` and called `current`. The node `current` is also added to `CLOSED`. The first node in `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

```
est_total_cost[node] = past_cost[node] +
                       heuristic_cost_to_go(node)
```

where `heuristic_cost_to_go(node)` ≥ 0 is an optimistic (underestimating) estimate of the actual cost-to-go to the goal from `node`. For the visibility graph example, an appropriate choice for the heuristic is the straight-line distance to the goal, ignoring any obstacles.

Because `OPEN` is a sorted list according to the estimated total cost, inserting a new node at the correct location in `OPEN` entails a small computational price.

If the node `current` is in the goal set, then the search is finished, and the path is reconstructed from the `parent` links. If not, for each neighbor `nbr` of `current` in the graph, which is not also in `CLOSED`, the `tentative_past_cost` for `nbr` is calculated as `past_cost[current] + cost[current,nbr]`. If `tentative_past_cost < past_cost[nbr]`, then `nbr` can be reached less expensively than previously thought, so `past_cost[nbr]` is set to `tentative_past_cost` and `parent[nbr]` is set to `current`. The node `nbr` is then added (or moved) in `OPEN` according to its estimated total cost.

The algorithm proceeds by returning to pop off of `OPEN` the node with the lowest estimated total cost. If `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 `current` is in the goal set, then `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 `past_cost[current]`. Therefore the path to `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.

Algorithm 1 A^* search.

1: `OPEN` $\leftarrow \{1\}$
2: `past_cost[1]` $\leftarrow 0$, `past_cost[node]` $\leftarrow$ `infinity` for `node` $\in \{2, \ldots, N\}$
3: **while** `OPEN` is not empty **do**
4:     `current` $\leftarrow$ first node in `OPEN`, remove from `OPEN`
5:     add `current` to `CLOSED`
6:     **if** `current` is in the goal set **then**
7:         **return** SUCCESS and the path to `current`
8:     **end if**
9:     **for** each `nbr` of `current` not in `CLOSED` **do**
10:         `tentative_past_cost` $\leftarrow$ `past_cost[current]` + `cost[current,nbr]`
11:         **if** `tentative_past_cost` < `past_cost[nbr]` **then**
12:           `past_cost[nbr]` $\leftarrow$ `tentative_past_cost`
13:           `parent[nbr]` $\leftarrow$ `current`
14:           put (or move) `nbr` in sorted list `OPEN` according to
                `est_total_cost[nbr]` $\leftarrow$ `past_cost[nbr]` +
                      `heuristic_cost_to_go(nbr)`
15:         **end if**
16:     **end for**
17: **end while**
18: **return** FAILURE

10.3.2 Other Search Methods

- *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 η until the time allotted for the search has expired or a solution is found with $\eta = 1$.

- *Dijkstra's method.* If the heuristic cost-to-go is always estimated as zero, then A^* always explores from the OPEN node that has been reached with minimum past cost. This variant is called *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 *breadth-first search*. All nodes one edge away from the start node are considered first, then all nodes two edges away, etc. This also results in a minimum-cost path.

- *Depth-first search.* Depth-first search primarily applies to trees. The search follows one path through the tree as far as it can go, until it reaches a goal node or hits a leaf. If it hits a non-goal leaf, the search backtracks up the tree to the most recent decision that has never been taken, then follows a path down the tree again to a new leaf. The search terminates when all leaves have been explored (failure) or a node in the goal set has been reached. Depth-first search is rarely used in motion planning due to the likelihood of finding long paths before shorter ones.

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. For example, the cost to a N/S/E/W neighbor could be 1, while the cost
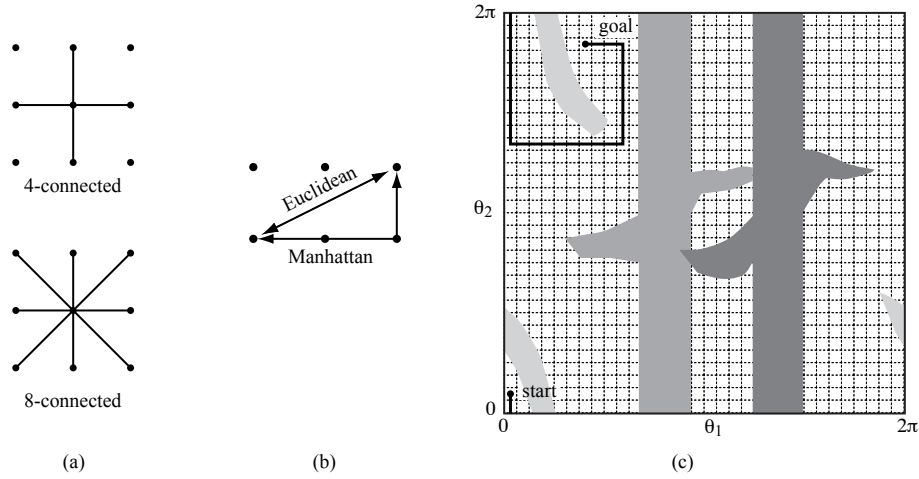
Figure 10.10: (a) A 4-connected grid point and an 8-connected grid point for a space $n = 2$. (b) Grid points spaced at unit intervals. The Euclidean distance between the two points indicated is $\sqrt{5}$ while the Manhattan distance is 3. (c) A grid representation of the C-space and a minimum-length Manhattan-distance path for the problem of Figure 10.2.

   to a diagonal neighbor could be $\sqrt{2}$. If integers are desired, for efficiency of the implementation, the approximate costs 10 and 14 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 `nbr` is only added to `OPEN` if the step from `current` to `nbr` is collision-free. (The step may be considered collision-free if a grown version of the robot at `nbr` does not intersect any obstacles.)

- Other optimizations can be achieved 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 $360°/32 = 11.25°$ 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_{goal} will be used in the same environment for multiple path planning queries, it may

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.

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 appropriate for low-dimensional C-spaces. This is because the number of grid points, and hence the computational complexity of the path planner, increases exponentially 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 to 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 $\mathcal{C}_{\text{free}}$. Conceptually, a grid point is considered an obstacle if any part of the rectangular 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
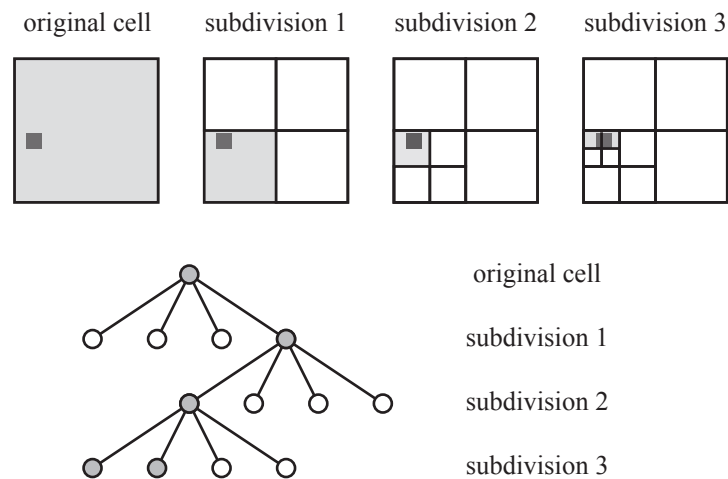
Figure 10.12: At the original C-space cell resolution, a small obstacle (indicated by a dark square) causes the whole cell to be labeled an obstacle. Subdividing the cell once shows that at least $3/4$ of the cell is actually free. Three levels of subdivision results in a representation using ten total cells: four at subdivision level 3, three at subdivision level 2, and three at subdivision level 1. The cells shaded grey are the obstacle cells in the final representation. The subdivision of the original cell is also shown as a tree, where the leaves of the tree are the final cells in the representation.

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.

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 $\mathcal{C}_{\text{free}}$ 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
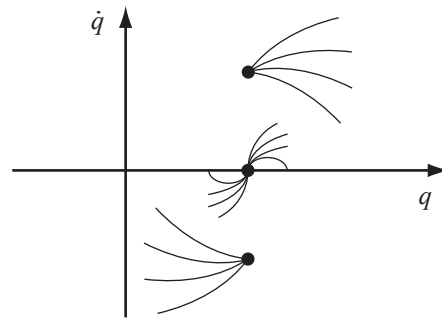
Figure 10.13: Sample trajectories emanating from three initial states in the phase space of a dynamic system with $q \in \mathbb{R}$. If the initial state has $\dot{q} > 0$, the trajectory cannot move to the left (negative motion in q) instantaneously. Similarly, if the initial state has $\dot{q} < 0$, the trajectory cannot move to the right instantaneously.
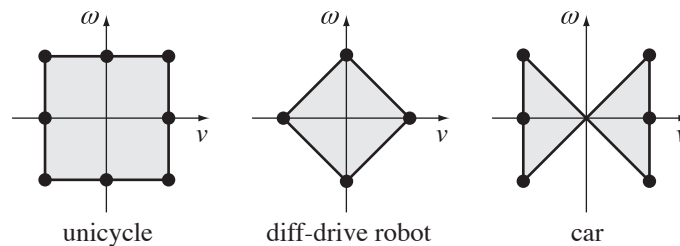
Figure 10.14: Discretizations of the control sets for unicycle, diff-drive, and car-like robots.

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, unlike the undirected graphs discussed so far. One approach is to discretize the robot *controls* while still making 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

In Chapter **??**, we saw that the controls for simplified models of unicycle, diff-drive, and car-like robots are (v, ω), 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, the A^* search can be modified slightly to a Dijkstra algorithm (Algorithm 2).

Algorithm 2 Grid-based Dijkstra planner for a wheeled mobile robot.

1: `OPEN` $\leftarrow \{q_{\text{start}}\}$
2: `past_cost`$[q_{\text{start}}] \leftarrow 0$
3: `counter` $\leftarrow 1$
4: **while** `OPEN` is not empty and `counter < MAXCOUNT` **do**
5:     `current` $\leftarrow$ 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` $\leftarrow$ `counter + 1`
12:        **for** each control in the discrete control set **do**
13:            integrate control forward a short time Δt from `current` to q_{new}
14:            **if** the path to q_{new} is collision-free **then**
15:                compute `cost` of the path to q_{new}
16:                place q_{new} in `OPEN`, sorted by `cost`
17:                `parent`$[q_{\text{new}}] \leftarrow$ `current`
18:            **end if**
19:        **end for**
20:    **end if**
21: **end while**
22: **return** FAILURE

The search expands from q_{start} by integrating forward each of the controls for a time Δt, 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, which is typically a constant (due to the constant integration time Δt) plus perhaps a penalty if the action changed from the one used to reach `current`. The constant term expresses the preference for paths of short duration, while the penalty expresses a preference for smooth paths minimizing control changes, particularly direction reversals.

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.

If there is a cost for changing the control from one step to the next, then we have to be careful about when we prune a node in an "occupied" cell, since the path cost is no longer simply the sum of edge costs. One solution is to maintain several copies of the C-space grid, one for each control in the discrete set. Then a node `current` is only pruned from the search if its grid cell had been previously occupied (explored from) by a node `previous` and if `previous` had been reached by the same control step that `current` used.
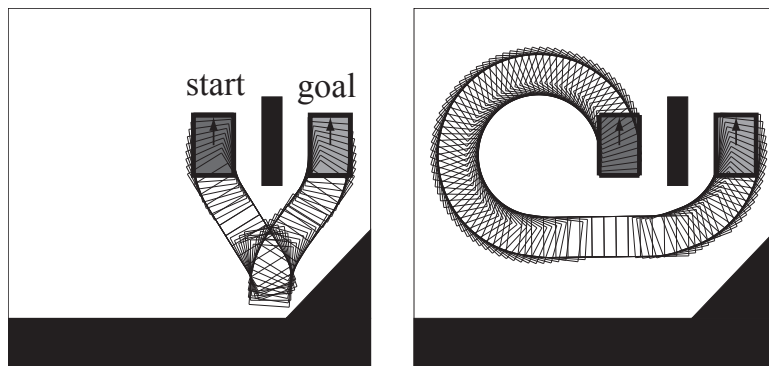
Figure 10.15: (Left) A minimum-cost path for a car-like robot where each action has identical cost, favoring a short path. (Right) A minimum-cost path where reversals are heavily penalized.

No more than `MAXCOUNT` nodes, where `MAXCOUNT` is a value chosen by the user, are considered during the search.

The time Δt 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 Δt 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.

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 $\mathcal{A}(q, \dot{q})$ represent the set of feasible accelerations based on the limited joint torques. To discretize the controls, the set $\mathcal{A}(q, \dot{q})$ is intersected
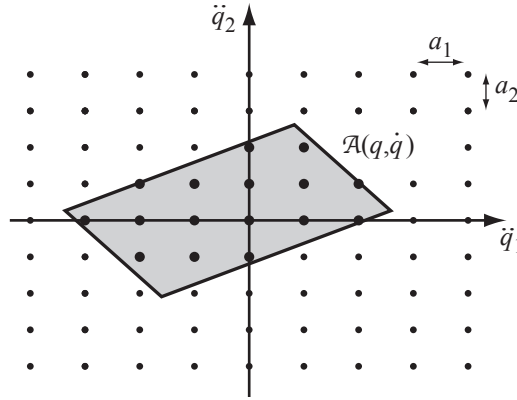
Figure 10.16: The instantaneously available acceleration set $\mathcal{A}(q,\dot{q})$ for a two-joint robot, intersected with a grid spaced at a_1 in $\ddot{q}_1$ and a_2 in $\ddot{q}_2$, gives the discretized control actions shown in bold.

with a grid of points of the form

$$\sum_{i=1}^{n} c_i a_i \mathbf{e}_i,$$

where c_i is an integer, $a_i > 0$ is the acceleration step size in the $\ddot{q}_i$ direction, and $\mathbf{e}_i$ is a unit vector in the ith direction (Figure 10.16).

As the robot moves, the acceleration set $\mathcal{A}(q,\dot{q})$ changes, but the grid remains fixed. Because of this, and assuming a fixed integration time Δt 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 $a_1 k_v \Delta t$, 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 $a_1 k_p(\Delta t)^2/2 + q_1(0)$, where k_p is an integer (Exercise **??**).

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 Δt. 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 constraints in the control grid and timestep Δt.

The control grid stepsizes 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 Δt, 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 τ, the computation time spent growing the search tree for a given robot motion duration increases by a factor of $r^{n\tau}$, 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 inertia matrix $M(q)$ changes with q and how fast the robot is moving. At low speeds $\dot{q}$ and short durations Δt, 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. This process builds 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_{start} to a goal set $\mathcal{X}_{\text{goal}}$. 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_{start} as outlined in Algorithm 3.

Algorithm 3 RRT algorithm.

1: initialize search tree T with x_{start}
2: **while** T is less than the maximum tree size **do**
3:     $x_{\text{samp}} \leftarrow$ sample from $\mathcal{X}$
4:     $x_{\text{nearest}} \leftarrow$ nearest node in T to x_{samp}
5:     employ a local planner to find a motion from x_{nearest} to x_{new} in
            the direction of x_{samp}
6:     **if** the motion is collision-free **then**
7:        add x_{new} to T with an edge from x_{nearest} to x_{new}
8:        **if** x_{new} is in $\mathcal{X}_{\text{goal}}$ **then**
9:           **return** SUCCESS and the motion to x_{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_{samp} randomly from an almost-uniform distribution over $\mathcal{X}$, with a slight bias toward states in $\mathcal{X}_{\text{goal}}$. The closest node x_{nearest} in T (line 4) is the one minimizing the Euclidean distance to x_{samp}. The state x_{new} (line 5) is chosen as the state a small distance d from x_{nearest} on the straight line to x_{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_{nearest} to x_{new}. If the motion is collision-free, the new state x_{new} is added to T.
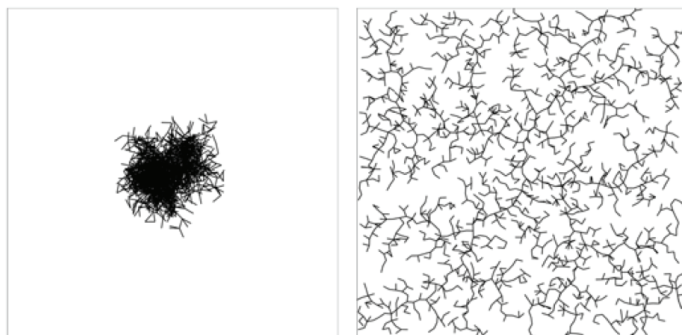
Figure 10.17: (Left) A tree generated by applying a uniformly-distributed random motion from a randomly chosen tree node results in a tree that does not explore very far. (Right) A tree generated by the RRT algorithm using samples drawn randomly from a uniform distribution. Both trees have 2000 nodes.

The net effect is that the nearly uniformly distributed samples "pull" the tree toward them, causing the tree to rapidly explore $\mathcal{X}_{\text{free}}$. 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 $\mathcal{X}$ (line 3), how to define the "nearest" node in T (line 4), and how to plan the motion to make progress toward x_{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.

10.5.1.1 Line 3: The Sampler

The most obvious sampler is one that samples randomly from a uniform distribution over $\mathcal{X}$. This is straightforward for Euclidean C-spaces $\mathbb{R}^n$; for n-joint robot C-spaces $T^n = S^1 \times ... \times S^1$ (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, notably $SO(3)$, are 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 $\mathcal{X}$ 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 even-
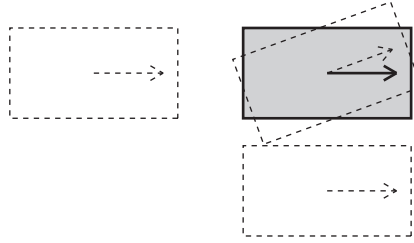
Figure 10.18: Which of the three dashed configurations of the car is "closest" to reaching the configuration in grey?

tually 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 $\mathcal{X}$).

10.5.1.2   Line 4: Defining the Nearest Node

Finding the "nearest" node depends on a definition of distance on $\mathcal{X}$. For an unconstrained kinematic robot on $\mathcal{C} = \mathbb{R}^n$, 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_{samp}: one that is rotated twenty degrees relative to x_{samp}, one that is 2 meters straight behind it, or one that is 1 meter 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_{samp}. 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_{nearest} should perhaps be defined as the one that can reach x_{samp} 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_{samp} is the weighted sum of the distances along the different components of $x_{\mathrm{samp}} - x$. 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_{nearest} to some point x_{new} which is closer to x_{samp}. The planner should be simple and it should run quickly. Two examples are:

Straight-line planner. This is for kinematic systems with no motion constraints. The plan is a straight line to x_{new}, which may be chosen at x_{samp} or at a fixed distance d from x_{nearest} on the straight line to x_{samp}.

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_{nearest} for a fixed time Δt using $\dot{x} = f(x, u)$. The resulting state that is closest to x_{samp} is chosen as x_{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 **??**.

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_{start} and one "backward" from x_{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_{samp} from the other tree. The advantage of this approach is that a single goal state x_{goal} can be reached exactly, rather than just a goal set $\mathcal{X}_{\text{goal}}$. 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 $\mathcal{X}_{\text{goal}}$ 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
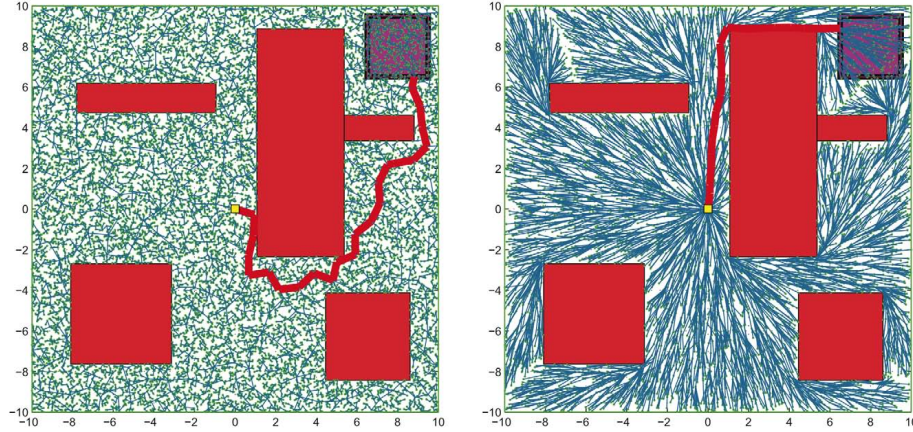
Figure 10.19: (Left) The tree generated by an RRT after 20,000 nodes. The goal region is the square at the top right corner, and the shortest path is indicated. (Right) The tree generated by RRT* after 20,000 nodes.

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 rewires the search tree to ensure that it always encodes the shortest path from x_{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_{new} in T with an edge from x_{nearest} to x_{new}, is replaced by a test of all nodes $x \in \mathcal{X}_{\text{near}}$ in T that are sufficiently near to x_{new}. An edge to x_{new} is created from the $x \in \mathcal{X}_{\text{near}}$ that (1) has a collision-free motion by the local planner and (2) minimizes the total cost of the path from x_{start} to x_{new}, not just the cost of the added edge. The total cost is the cost to reach the candidate $x \in \mathcal{X}_{\text{near}}$ plus the cost of the new edge.

The next step is to consider each $x \in \mathcal{X}_{\text{near}}$ to see if it could be reached by lower cost by a motion through x_{new}. If so, the parent of x is changed to x_{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 $\mathcal{X}_{\text{near}}$ 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. The time to produce a rewired tree by the RRT* algorithm is within a constant factor of the time to produce a tree of the same size by the RRT. 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 $\mathcal{C}_{\text{free}}$ (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_{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_{goal}. The graph is then searched for 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).

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 $\mathcal{C}_{\text{free}}$. While the default might be sampling randomly from a uniform distribution on $\mathcal{C}$ 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 $\mathcal{C}_{\text{free}}$. Another option is deterministic multi-resolution sampling.
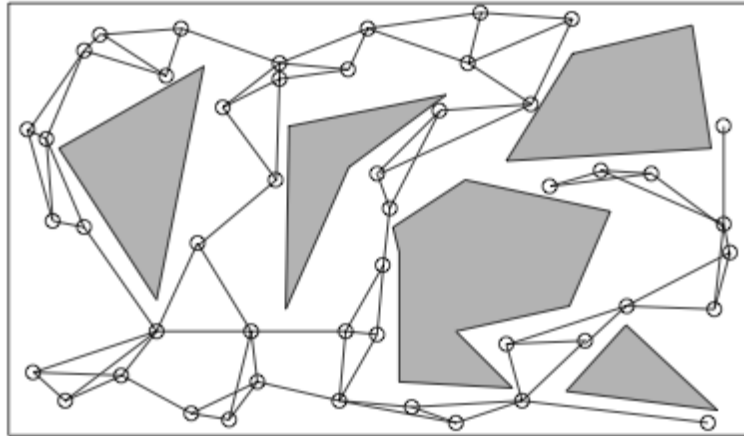
Figure 10.20: An example PRM roadmap for a point robot in $\mathcal{C} = \mathbb{R}^2$. The $k = 3$ closest neighbors are considered for connection to a sample node q. The degree of a node can be greater than three since it may be a close neighbor of many nodes.

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 $U(q)$ defined over $\mathcal{C}$ induces a force $F = -\partial U/\partial q$ 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 $U(h) = mgh$ and the force acting on it is $F = -\partial U/\partial h = -mg$. The force will cause the mass to fall to the Earth's surface.

In robot motion control, the goal configuration q_{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_{goal} is typically encoded by a quadratic potential energy "bowl" with zero energy at the goal,

$$U_{\text{goal}}(q) = \frac{1}{2}(q - q_{\text{goal}})^T K(q - q_{\text{goal}}),$$

where K is a symmetric positive-definite weighting matrix (for example, the identity matrix). The force induced by this potential is

$$F_{\text{goal}}(q) = -\frac{\partial U_{\text{goal}}}{\partial q} = K(q_{\text{goal}} - q),$$

an attractive force proportional to the distance from the goal.

The repulsive force induced by a C-obstacle $\mathcal{B}$ can be calculated from the distance $d(q, \mathcal{B})$ to the obstacle (Section 10.2.2):

$$U_{\mathcal{B}}(q) = \frac{k}{2d^2(q, \mathcal{B})},$$

where $k > 0$ is a scaling factor. The potential is only properly defined for points outside the obstacle, $d(q, \mathcal{B}) > 0$. The force induced by the obstacle potential is

$$F_{\mathcal{B}}(q) = -\frac{\partial U_{\mathcal{B}}}{\partial q} = \frac{k}{d^3(q, \mathcal{B})}\frac{\partial d}{\partial q}.$$

The total potential is obtained by summing the attractive goal potential and the repulsive obstacle potentials,

$$U(q) = U_{\text{goal}}(q) + \sum_i U_{\mathcal{B}_i}(q),$$

yielding a total force

$$F(q) = F_{\text{goal}}(q) + \sum_i F_{\mathcal{B}_i}(q).$$

A bowl-like attractive potential and obstacle repulsive potentials are illustrated for a point in $\mathbb{R}^2$ in *** **Figure** ***.

To actually control the robot using the calculated $F(q)$, we have several options, two of which are:

- Apply the calculated force plus damping,

$$u = F(q) + B\dot{q}, \tag{10.3}$$

  where B is a positive semidefinite matrix. 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 $\frac{1}{2}\dot{q}^T(0)M(q(0))\dot{q}(0)$ plus the initial virtual potential energy $U(q(0))$.

- Treat the calculated force as a commanded velocity instead:

$$\dot{q} = F(q). \tag{10.4}$$

This automatically eliminates oscillations.

Figure *** shows the motion of point robots in the plane using the velocity control law (10.4). It also shows a local minimum of the potential, which is the main drawback of potential fields.

It is important to note that obstacle potentials go to infinity at the boundaries of the obstacles, resulting in steep gradients and therefore large $F_{\mathcal{B}}(q)$ near an obstacle boundary. A simple solution is to saturate $F_{\mathcal{B}}(q)$ after calculating it. Another issue is that even distant obstacles have a small 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 $d(q, \mathcal{B}) \geq d_{\text{range}}$:

$$U_{\mathcal{B}}(q) = \begin{cases} \frac{k}{2} \left(\frac{d_{\text{range}} - d(q, \mathcal{B})}{d_{\text{range}} d(q, \mathcal{B})} \right)^2 & \text{if } d(q, \mathcal{B}) < d_{\text{range}} \\ \\ 0 & \text{otherwise.} \end{cases}$$

Another issue is that $d(q, \mathcal{B})$ and its gradient is generally difficult to calculate. An approach to dealing with this is described in Section **??**.

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 exception is the wavefront planner of Figure 10.11. The wavefront algorithm creates the equivalent of a potential function, with high potential at grid points in obstacles and zero potential at the goal. 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 gradient-following planning 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_{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).
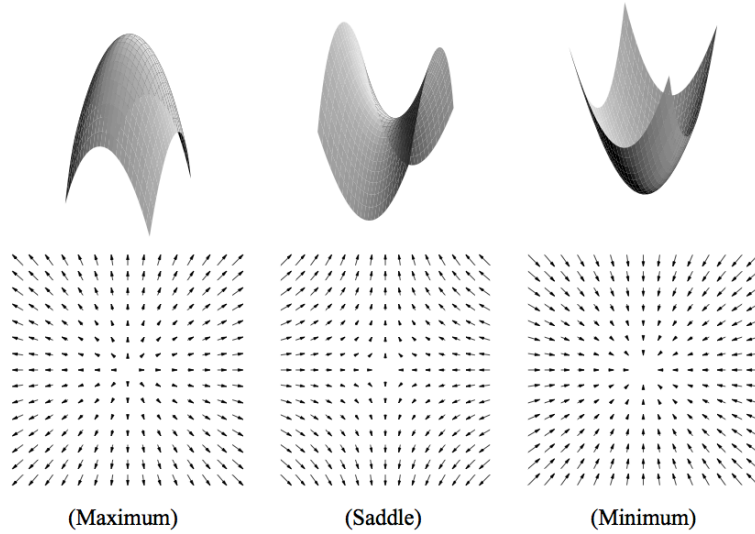
Figure 10.21: Examples of critical points in two dimensions: a maximum, a saddle (minimum in some directions and maximum in others), and a minimum. ***** From Choset book, replace**

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, as shown in Figure 10.21), there is only one minimum, at q_{goal}. This ensures that q_{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_{goal}.

While constructing navigation potential functions with only a single minimum is non-trivial, Rimon and Koditschek showed 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 showed that the boundaries of the obstacles, and the associated navigation function, can be deformed to a much broader class of *star-shaped* obstacles. 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 C-space which has star-shaped obstacles. 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 navi-
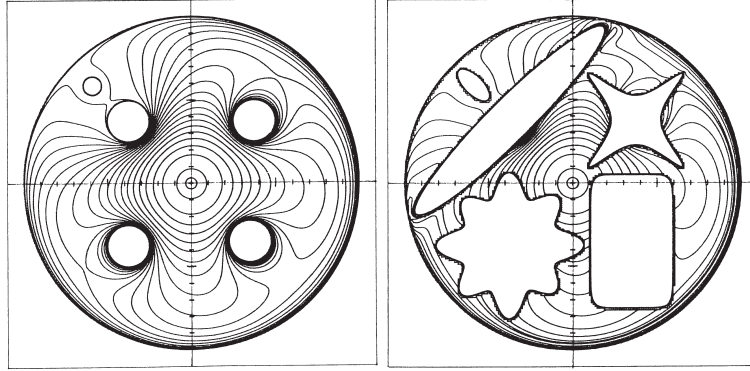
Figure 10.22: (Left) A model "sphere world" with five circular obstacles. The contour plot of a navigation function is shown. The goal is at $(0,0)$. Note that the obstacles induce saddle points near the obstacles, but no local minima. (Right) A "star world" obtained by deforming the obstacles and the potential while retaining a navigation function.

gation function to one that fits the star world. Rimon and Koditschek give a systematic procedure for accomplishing this.

Figure 10.22 shows a deformation of a navigation function on a model sphere world to a star world for the case $\mathcal{C} \subset \mathbb{R}^2$.

10.6.3 Workspace Potential

A difficulty in calculating the repulsive force from an obstacle is calculating the distance to the obstacle, $d(q, \mathcal{B})$. 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

$$U'_{ij}(q) = \frac{k}{2\|f_i(q) - c_j\|^2},$$

yielding the repulsive force at the control point

$$F'_{ij}(q) = -\frac{\partial U_{ij}}{\partial q} = \frac{k}{\|f_i(q) - c_j\|^4}(f_i(q) - c_j).$$

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 $J_i(q)$ 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 $F_{ij}(q)$ due to the repulsive linear force $F'_{ij}(q)$ 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 (10.3)) or a velocity $\dot{q} = F(q)$ (for control law (10.4)) 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

$$F_{\text{proj}}(q) = \left(\mathcal{I} - A^T(q)\big(A(q)A^T(q)\big)^{-1}A(q)\right)F(q).$$

For a dynamic robot employing the control law $u = F_{\text{proj}}(q) + B\dot{q}$, the projection is discussed in Chapter 11.4.2.

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

$$
\begin{array}{llll}
\text{find} & u(t), q(t), T & & (10.5) \\
\text{minimizing} & J(u(t), q(t), T) & & (10.6) \\
\text{subject to} & \dot{x}(t) = f(x(t), u(t)) & \forall t \in [0, T] & (10.7) \\
& u(t) \in \mathcal{U} & \forall t \in [0, T] & (10.8) \\
& q(t) \in \mathcal{C}_{\text{free}} & \forall t \in [0, T] & (10.9) \\
& x(0) = x_{\text{start}} & & (10.10) \\
& x(T) = x_{\text{goal}}. & & (10.11)
\end{array}
$$

To approximately solve this problem by nonlinear optimization, the control $u(t)$, trajectory $q(t)$, and equality and inequality constraints (10.7)–(10.11) 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 three choices of how to parameterize the position and controls:

(i) *Parameterize the trajectory $q(t)$.* In this case, we solve for the parameterized 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) *Parameterize the control $u(t)$.* We solve for $u(t)$ directly. Calculating the state $x(t)$ requires integrating the equations of motion.

(iii) *Parameterize both $q(t)$ and $u(t)$.* We have a larger number of variables, since we are parameterizing 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 parameterizations 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 parameterized 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 parameterization has implications for the efficiency of the calculation of $q(t)$ and $u(t)$ at a given time t.

The choice of parameterization 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.

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.

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, since the plan initializes the optimization with a "reasonable" solution. The initial motion must be converted to a parameterized 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,

$$J = \int_0^T \dot{u}^T(t)\dot{u}(t)dt$$

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.

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 attempt 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_{goal}, find a time T and a set of controls $u : [0, T] \to \mathcal{U}$ such that the motion (10.2) 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 $\mathcal{C}_{\text{obs}}$, $\mathcal{C} = \mathcal{C}_{\text{free}} \bigcup \mathcal{C}_{\text{obs}}$. 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 and where each node has at most one parent.

- A roadmap path planner uses a graph representation of $\mathcal{C}_{\text{free}}$, and any path planning problem can be solved using a simple path from q_{start} onto the roadmap, a path along the roadmap, and a simple path from the roadmap to q_{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.

- 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 while allowing 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_{start} to find a motion to $\mathcal{X}_{\text{goal}}$. It relies on a sampler to find a sample x_{samp} in $\mathcal{X}$; an algorithm to find the closest node x_{nearest} in the search tree; and a local planner to find a motion from x_{nearest} to a point closer to x_{samp}. The sampling is chosen to cause the tree to explore X_{free} quickly.

- The bidirectional RRT grows a search tree from both x_{start} and x_{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 $U(q)$ is the sum of these, and the virtual force applied to the robot is $F(q) = -\partial U/\partial q$. 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 tend to result in local minima where the robot gets stuck away from the goal.

- A navigation function is a potential function with no local minima. Navigation functions result in near-global convergence to q_{goal}. While navigation functions are difficult to design in general, they can be designed systematically for certain systems.

- 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.
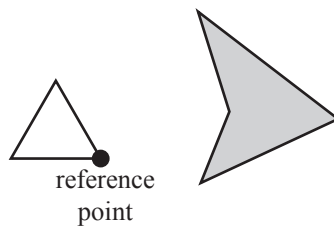
Figure 10.23: Exercise 4.

10.10   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 θ_2 joint angles in the range $[175°, 185°]$ 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 $\mathcal{C}_{\text{free}}$. 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
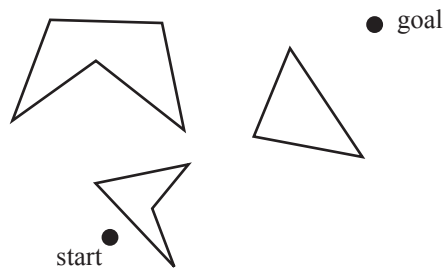
Figure 10.24: Planning problem for Exercise 8.

that minimizes the use of square roots.

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_{start} and q_{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. Are the solutions found with

the new heuristic 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 the C-space grid using 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.)

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, mimicking 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 (or position) 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 both motions and forces. If the robot imposes a motion, then the environment will determine the force, and vice-versa.

Once we have chosen a control goal consistent with the task and environment, we have a number of ways 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. *Feedforward control* uses a model of the dynamics of the robot and its environment to determine actuator signals that achieve the desired change in state. Because of modeling errors, feedforward control is rarely used by itself, but it is often used in conjunction with feedback control. Complementary control strategies include *adaptive control*, which continuously estimates properties of the dynamic system to improve performance; *robust control* to guarantee some level of performance in the face

287

of an uncertain model of the system; and *iterative learning control* for repetitive tasks, where errors from previous executions of the same task are used to generate more appropriate feedforward controls for future iterations.

In this chapter we focus on feedback and feedforward 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). The controller is often a PC or microcontroller. 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 will 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 jitter due to quantization.

Actuators could be brushed DC motors, brushless DC motors, various types of AC motors, hydraulic actuators, or pneumatic actuators, among others. Typically gears or other transmissions are used to lower the speed and increase the force or torque of the actuator. Electric motors are coupled with a power amplifier that converts signals from the controller to high currents to drive the motor. Local feedback of motor current or joint torque may be used in an inner control loop to achieve the forces or torques requested by the controller.

For each robot joint, we will lump the 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 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 are significant issues in design and control, but are beyond the scope of this chapter.
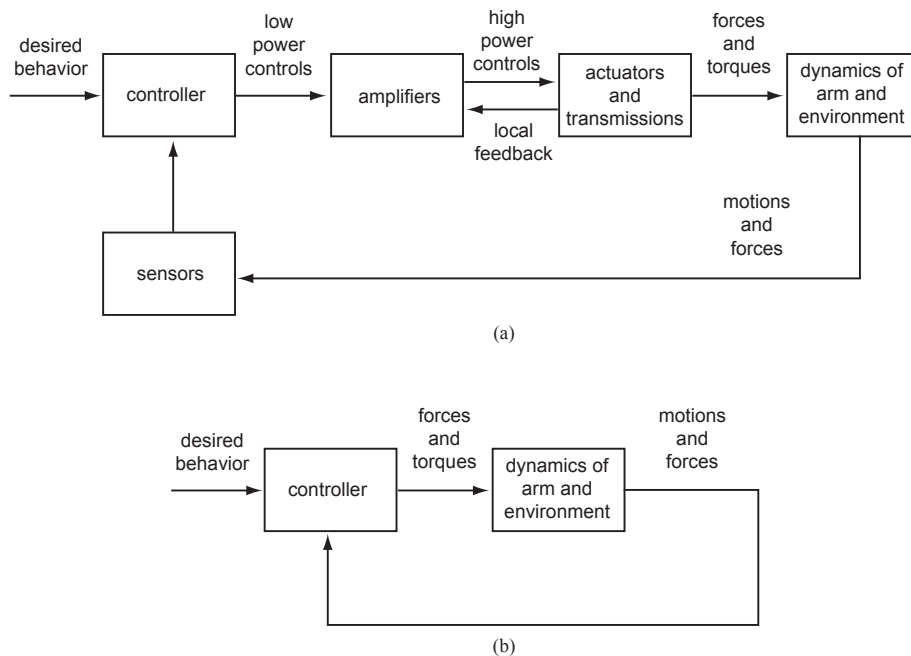
Figure 11.1: (a) A typical robot control system. An inner control loop may be used to help the amplifier and actuator achieve the desired force or torque. For example, a DC motor amplifier in torque control mode may sense the current actually flowing through the motor and implement a local controller to better match the desired current, since the current is proportional to the torque produced by the motor. (b) A simplified model with ideal sensors and a controller block that directly produces forces and torques. This assumes ideal behavior of the amplifier and actuator blocks in part (a).

11.2 Motion Control

Typically a motion control task is to have the end-effector follow a desired trajectory. If the robot is redundant, there may be more than one set of joint variable histories that yield this end-effector trajectory. We begin by assuming that inverse kinematics and its derivatives, possibly with redundancy resolution, have already been applied to yield a unique set of target joint histories as a function of time. Our goal is to construct controllers that drive the robot to track this trajectory in joint space. We consider the case of a trajectory expressed in the task space in Section 11.2.3.

The ideas are well illustrated by a robot with a single joint, so we begin there, then generalize to a multi-joint robot.
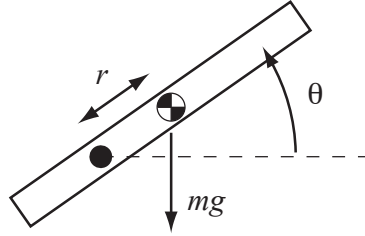
Figure 11.2: A single joint robot rotating in a gravity field.

11.2.1   Motion Control of a Single Joint

Consider a single motor attached to a single link, as shown in Figure 11.2. Let τ be the motor's torque and θ be the angle of the link. The dynamics can be written as

$$\tau = M\ddot{\theta} + mgr\cos\theta, \tag{11.1}$$

where M is the 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 (11.1), there is no dissipation: if the link is made to spin and $\tau = 0$, it would spin 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,

$$\tau_{\text{fric}} = b\dot{\theta}, \tag{11.2}$$

where $b > 0$. Adding the friction torque, our final model is

$$\tau = M\ddot{\theta} + mgr\cos\theta + b\dot{\theta}, \tag{11.3}$$

which we may write more compactly as

$$\tau = M\ddot{\theta} + h(\theta, \dot{\theta}), \tag{11.4}$$

where h contains all terms that depend only on the state, not the acceleration.

For concreteness in the following simulations, we set $M = 0.5$ kgm^2, $m = 1$ kg, $r = 0.1$ m, and $b = 0.1$ Nms/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.

11.2.1.1   Feedback Control: PID Control

The most common feedback control algorithm is linear PID (proportional-integral-derivative) control. Defining the error between the desired angle θ_d and actual angle θ as

$$\theta_e = \theta_d - \theta, \tag{11.5}$$
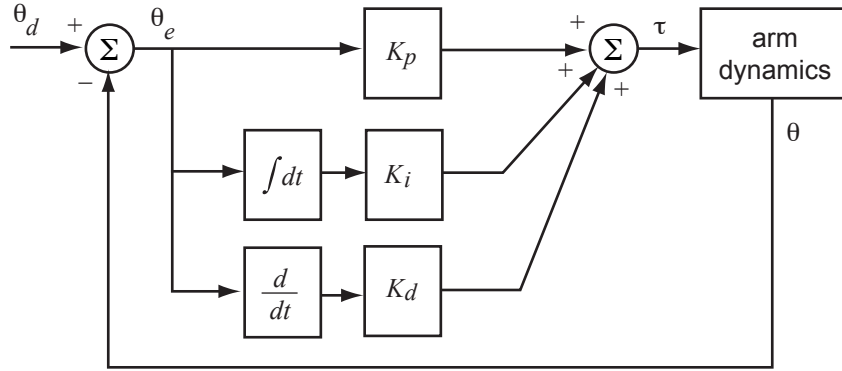
Figure 11.3: Block diagram of a PID controller.

the PID controller is simply

$$\tau = K_p\theta_e + K_i \int \theta_e(t)dt + K_d\dot{\theta}_e, \tag{11.6}$$

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 $\theta_d - \theta$, and the derivative gain K_d acts as a virtual damper that tries to reduce the velocity error $\dot{\theta}_d - \dot{\theta}$. 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 (11.6) into the dynamics (11.3), we get

$$M\ddot{\theta} + b\dot{\theta} = K_p(\theta_d - \theta) + K_d(\dot{\theta}_d - \dot{\theta}). \tag{11.7}$$

If the goal state is rest at a constant θ_d, then $\dot{\theta}_d = \ddot{\theta}_d = 0$. This is called *setpoint* control. Using $\theta_e = \theta_d - \theta$, $\dot{\theta}_e = -\dot{\theta}$, and $\ddot{\theta}_e = -\ddot{\theta}$, Equation (11.7) can be rewritten as the linear mass-spring-damper *error dynamics*

$$M\ddot{\theta}_e + (b + K_d)\dot{\theta}_e + K_p\theta_e = 0. \tag{11.8}$$

Stability   Error dynamics, such as Equation (11.8), are an important concept in the study of control systems. A minimum requirement is that the error dynamics be *stable*, i.e., initial errors tend to zero exponentially with time. A linear homogeneous ordinary differential equation of the form

$$a_n\theta_e^{(n)} + a_{n-1}\theta_e^{(n-1)} + \ldots + a_2\ddot{\theta}_e + a_1\dot{\theta}_e + a_0\theta_e = 0$$

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., $\mathrm{Re}(s_i) < 0$ 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 (11.8). For third-order dynamics, it is also required that $a_2 a_1 > a_3 a_0$.

PD Control and Second-Order Error Dynamics   To study the second-order error dynamics (11.8) more formally, we assume stability and rewrite in the standard second-order form

$$\ddot{\theta}_e + \frac{b + K_d}{M} \dot{\theta}_e + \frac{K_p}{M} \theta_e = 0 \quad \rightarrow \quad \ddot{\theta}_e + 2\zeta\omega_n \dot{\theta}_e + \omega_n^2 \theta_e = 0, \qquad (11.9)$$

where the *damping ratio* ζ and the *natural frequency* ω_n are

$$\zeta = \frac{b + K_d}{2\sqrt{K_p M}}, \qquad \omega_n = \sqrt{\frac{K_p}{M}}.$$

The characteristic equation of (11.9) is

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0, \qquad (11.10)$$

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 (11.9), 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

$$\theta_e(t) = c_1 \exp(s_1 t) + c_2 \exp(s_2 t),$$

  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

$$\theta_e(t) = \exp(-\zeta\omega_n t)(c_1 + c_2 t),$$

  i.e., a decaying exponential multiplied by a linear function of time. The time constant of the decaying exponential is $\tau = 1/(\zeta\omega_n)$.
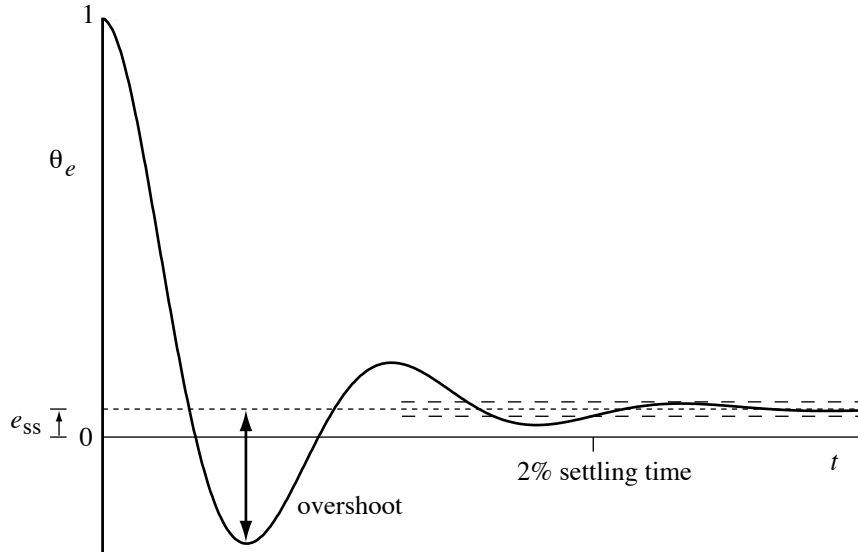
Figure 11.4: The error response to a step input for an underdamped second-order system, showing steady-state error e_{ss}, overshoot, and 2% settling time.

- **Underdamped**: $\zeta < 1$. The roots $s_{1,2}$ are complex conjugates at $s_{1,2} = -\zeta\omega_n \pm \omega_d$, where $\omega_d = \omega_n\sqrt{1 - \zeta^2}$ is the *damped natural frequency*. The solution is

$$\theta_e(t) = \exp(-\zeta\omega_n t)\left(c_1\cos(\omega_d t) + c_2\sin(\omega_d t)\right),$$

  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, $e_{\text{ss}} = 0$. The transient response is characterized by the *overshoot* and (2%) *settling time*. The 2% settling time is the first time T such that $|\theta_e(t) - e_{\text{ss}}| \leq 0.02(1 - e_{\text{ss}})$ for all $t \geq T$, and is approximately equal to 4τ, where τ is the slowest time constant in the solution. Overshoot is defined as

$$\text{overshoot} = \left|\frac{\theta_{e,\min} - e_{\text{ss}}}{1 - e_{\text{ss}}}\right| \times 100\%$$
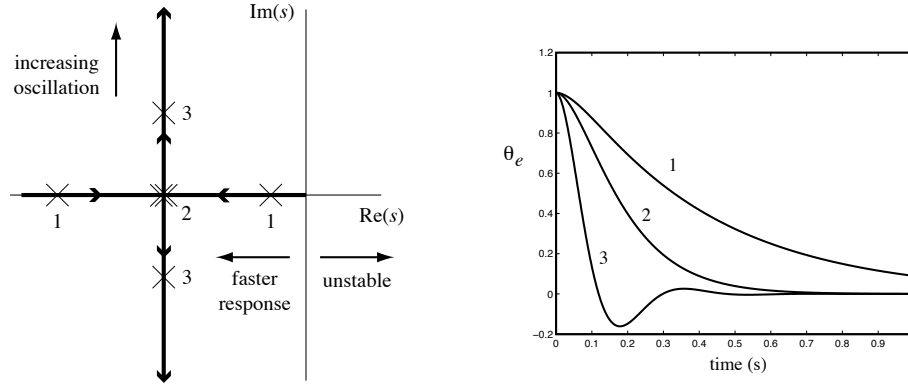
Figure 11.5: (Left) The complex roots of the characteristic equation of the PD-controlled joint for a fixed $K_d = 10$ Nms/rad as K_p increases from zero. This is known as a "root locus" plot. (Right) The response of the system to an initial error $\theta_e = 1$, $\dot{\theta}_e = 0$ is shown for overdamped ($\zeta = 1.5$, roots at "1"), critically damped ($\zeta = 1$, roots at "2"), and underdamped ($\zeta = 0.5$, roots at "3") cases.

where $\theta_{e,\min}$ is the least positive value achieved by the error. The overshoot can be calculated to be

$$\text{overshoot} = \exp(-\pi\zeta/\sqrt{1 - \zeta^2}) \times 100\%, \quad 0 \leq \zeta < 1.$$

A good transient response is characterized by a low settling time and little or no overshoot.

Figure 11.5 shows the relationship of the location of the roots of (11.10) 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, ζ 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.

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 even instability due to the finite servo rate frequency. Thus there are practical limits on the set of useful gains.
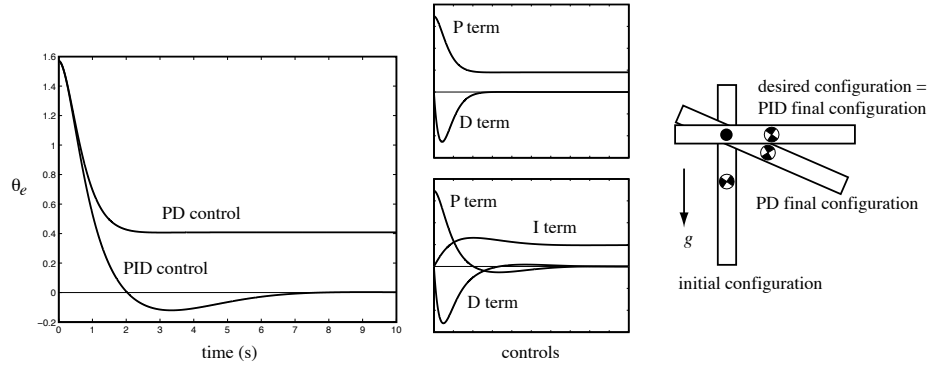
Figure 11.6: (Left) The tracking errors for a PD controller with $K_d = 2$ Nms/rad, $K_p = 2.205$ Nm/rad for critical damping, and a PID controller with the same PD gains and $K_i = 1$ Nm/(rad s). The arm starts at $\theta(0) = -\pi/2, \dot{\theta}(0) = 0$, with a goal state $\theta_d = 0, \dot{\theta}_d = 0$. (Middle) The individual contributions of the terms in the PD and PID control laws. (Right) The initial and final configurations.

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

$$M\ddot{\theta}_e + (b + K_d)\dot{\theta}_e + K_p\theta_e = mgr\cos\theta. \qquad (11.11)$$

This implies that the system comes to rest at a configuration θ satisfying $K_p\theta_e = mgr\cos\theta$, i.e., the final error θ_e is not zero when $\theta_d \neq \pm\frac{\pi}{2}$. This is because the robot must provide a nonzero torque to hold the link at rest at $\theta \neq \pm\frac{\pi}{2}$, 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

$$M\ddot{\theta}_e + (b + K_d)\dot{\theta}_e + K_p\theta_e + K_i \int \theta_e(t)dt = \tau_{\text{dist}}, \qquad (11.12)$$

where τ_{dist} is a disturbance torque substituted for the gravity term $mgr\cos\theta$. Taking derivatives of both sides, we get the third-order error dynamics

$$M\theta_e^{(3)} + (b + K_d)\ddot{\theta}_e + K_p\dot{\theta}_e + K_i\theta_e = \dot{\tau}_{\text{dist}}. \qquad (11.13)$$

If τ_{dist} is constant, then the right-hand side of (11.13) is zero. If the PID controller is stable, then (11.13) shows that θ_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 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 (11.13) when τ_{dist} is constant:

$$Ms^3 + (b + K_d)s^2 + K_ps + K_i = 0. \tag{11.14}$$

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.

11.2.1.2 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

$$\tau = \widehat{M}(\theta)\ddot{\theta} + \widehat{h}(\theta, \dot{\theta}), \tag{11.15}$$

where the model is perfect if $\widehat{M}(\theta) = M(\theta)$ and $\widehat{h}(\theta, \dot{\theta}) = h(\theta, \dot{\theta})$. Note that the inertia model $\widehat{M}(\theta)$ is written as a function of the configuration θ. While the inertia of our simple one-joint robot is not a function of configuration, writing this way allows us to re-use Equation (11.15) for multi-joint systems in Section 11.2.2.

Given θ_d, $\dot{\theta}_d$, and $\ddot{\theta}_d$ from the trajectory generator, the commanded torque is calculated as

$$\boxed{\tau = \widehat{M}(\theta_d)\ddot{\theta}_d + \widehat{h}(\theta_d, \dot{\theta}_d).} \tag{11.16}$$

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.
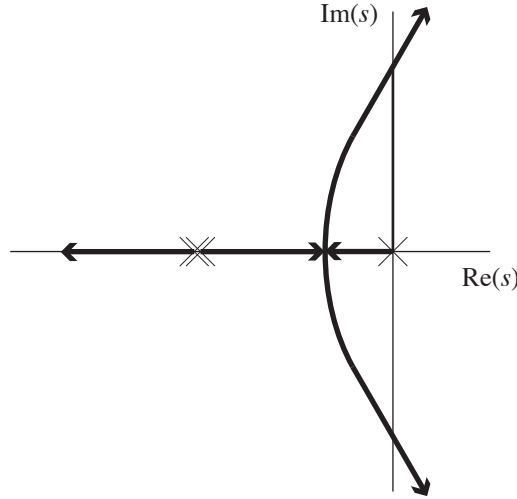
Figure 11.7: The three roots of (11.14) as K_i increases from zero. First a PD controller is chosen with K_p and K_d yielding critical damping, giving rise to two collocated roots on the negative real axis. Adding an infinitesimal gain $K_i > 0$ creates a third root at the origin. As we increase K_i, one of the two collocated roots moves to the left on the negative real axis, while the other two roots move toward each other, break away from the real axis, and move into the right-half plane when $K_i = (b + K_d)K_p/M$. The system is unstable for larger values of K_i.

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 $\hat{r} = 0.08$ 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 are always modeling errors, feedforward control is always used in conjunction with feedback, as discussed next.

11.2.1.3 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 $\{\widehat{M}, \widehat{h}\}$ to achieve the error dynamics

$$\ddot{\theta}_e + K_d\dot{\theta}_e + K_p\theta_e + K_i \int \theta_e(t)dt = 0 \qquad (11.17)$$

```
time = 0                          // dt = cycle time
eint = 0                          // error integral
qprev = senseAngle                // initial joint angle q
loop
  [qd,qdotd] = 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 = Kp*e + Kd*edot + Ki*eint
  commandTorque(tau)

  time = time + dt
end loop
```

Figure 11.8: Pseudocode for PID control.

```
time = 0                                      // dt = cycle time
loop
  [qd,qdotd,qdotdotd] = trajectory(time)   // from trajectory generator
  tau = Mhat(qd)*qdotdotd + hhat(qd,qdotd) // calculate dynamics
  commandTorque(tau)
  time = time + dt
end loop
```

Figure 11.9: Pseudocode for feedforward control.

along arbitrary trajectories, not just to a setpoint. The error dynamics (11.17) 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 (11.17), we choose the robot's commanded acceleration to be

$$\ddot{\theta}_{\text{com}} = \ddot{\theta}_d - \ddot{\theta}_e \qquad \text{then plug in Equation (11.17) to get}$$

$$= \ddot{\theta}_d + K_d\dot{\theta}_e + K_p\theta_e + K_i \int \theta_e(t)dt. \tag{11.18}$$

Plugging $\ddot{\theta}_{\text{com}}$ into a model of the robot dynamics, we get the *feedback linearizing*
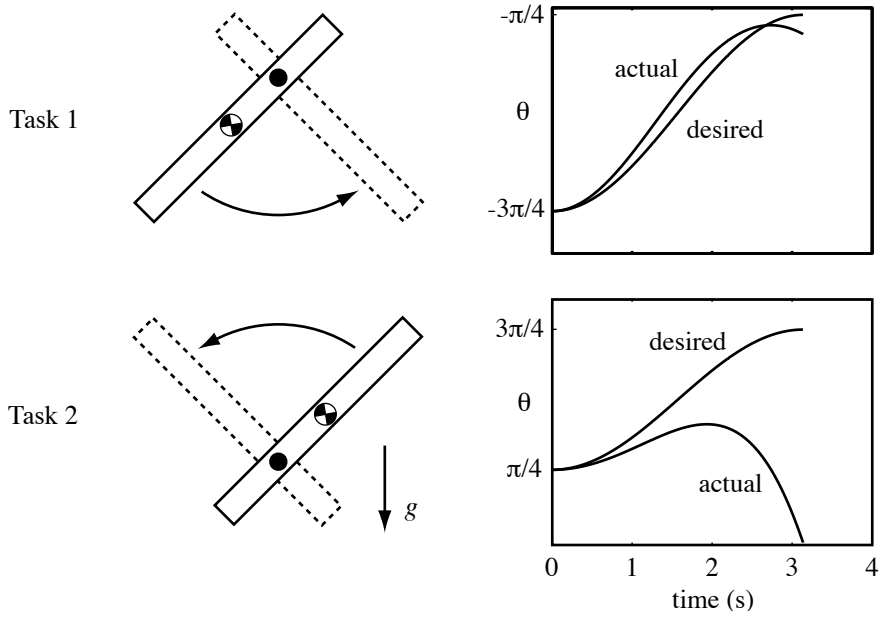
Figure 11.10: Results of feedforward control with an incorrect model: $\hat{r} = 0.08$ m, but $r = 0.1$ m. The desired trajectory in Task 1 is $\theta_d(t) = -\pi/2 - \cos(t)$ for $0 \le t \le \pi$. The desired trajectory for Task 2 is $\theta_d(t) = \pi/2 - \cos(t), 0 \le t \le \pi$.

controller

$$\tau = \widehat{M}(\theta) \left(\ddot{\theta}_d + K_p \theta_e + K_i \int \theta_e(t) dt + K_d \dot{\theta}_e \right) + \widehat{h}(\theta, \dot{\theta}). \qquad (11.19)$$

This type of controller is called feedback linearizing because feedback of θ and $\dot{\theta}$ is used to transform the nonlinear control system to a linear one. The $\widehat{h}(\theta, \dot{\theta})$ term cancels dynamics dependent only on the state, and the inertia model $\widehat{M}(\theta)$ converts desired joint accelerations into joint torques, realizing the simple linear error dynamics (11.17). This kind of controller is sometimes called a *computed torque* controller.

A block diagram of the 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.
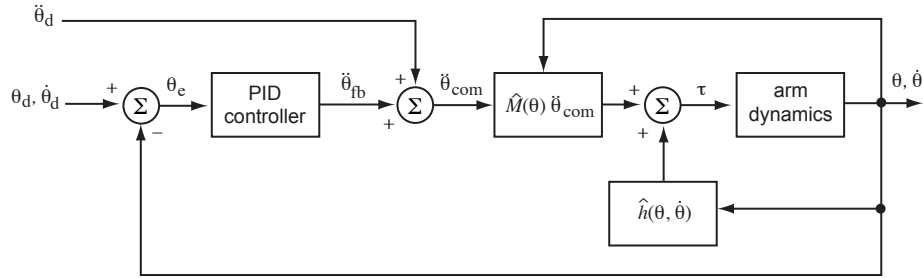
Figure 11.11: Feedback linearizing control. The feedforward acceleration $\ddot{\theta}_d$ is added to the acceleration $\ddot{\theta}_{\text{fb}}$ computed by the PID feedback controller to create the commanded acceleration $\ddot{\theta}_{\text{com}}$.
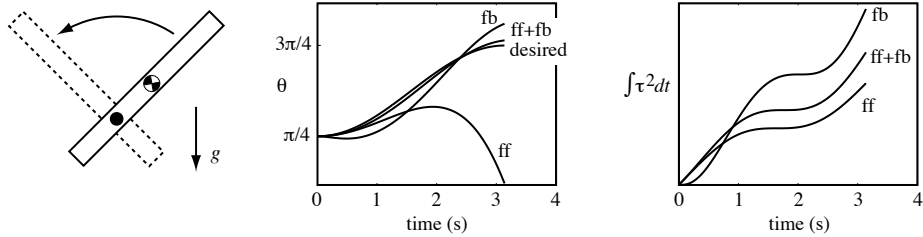
Figure 11.12: Performance of feedforward only (ff), feedback only (fb), and feedback linearizing control (ff+fb). PID gains are taken from Figure 11.6, and the feedforward modeling error is taken from Figure 11.10. The desired motion is Task 2 from Figure 11.10. The middle plot shows the tracking performance 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 feedback linearizing controller yields better tracking than either feedforward or feedback alone, with less control effort than feedback alone.

11.2.1.4 A Closer Look at Friction

For simplicity, we have been assuming a viscous model of friction at bearings and gears. In reality, 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 forces may be a function of the loading force at the contact, the time the contact has been at rest, position (due to spring-like forces before sliding begins, or non-uniformity in bearings), velocity, temperature, etc.

One noteworthy limitation of a viscous friction model is its implication of zero friction force at zero velocity. In fact, common experience indicates that friction forces can be large at zero velocity. For example, you can apply significant horizontal forces to a book on a table before it begins to slide. The force resisting motion at rest is known as *static friction* and is not included in

```
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 = Mhat(q)*(qdotdotd + Kp*e + Kd*edot + Ki*eint) + hhat(q,qdot)
  commandTorque(tau)

  time = time + dt
end loop
```

Figure 11.13: Pseudocode for the feedback linearizing controller.

the viscous model. For a robot joint, nonzero static friction implies that some torque can be applied to the joint at rest without causing any motion. The discontinuity of friction force at zero velocity significantly complicates the problem of controlling low-speed motions, particularly motions involving switches in direction. To address this issue, a more sophisticated model of joint friction can be included in the nonlinear dynamics compensation term $\widehat{h}(\theta,\dot{\theta})$ in feedforward or feedback linearizing control. See Figure 11.14 for some examples of velocity-dependent friction. Other ways of dealing with friction include "dithering" (imposing a high-frequency zero-mean control signal on top of the nominal control signal, to smooth the friction discontinuity at zero velocity) and using larger PID gains in the neighborhood of zero velocity.

11.2.1.5 The Effect of Gearing

Until now we have been considering our actuator as a source of torque, without considering how the actuator generates that torque. For example, if we choose a DC motor with an appropriate power rating for our robot joint, we will likely find that it is capable of producing high speed, up to 10,000 RPM or more, but only low torque. Most robotic applications require significantly lower speeds and higher torques. Therefore, gears, belts and pulleys, and other transmissions are usually used to reduce speed by a gear ratio $G > 1$, while ideally increasing the
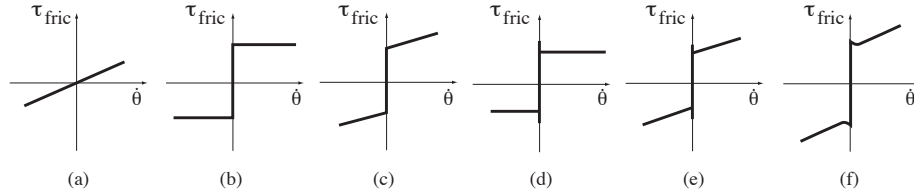
Figure 11.14: Examples of velocity-dependent friction models. (a) Viscous friction. (b) Coulomb friction, $\tau_{\text{fric}} = b\,\text{sgn}(\dot{\theta})$. τ_{fric} can take any value in $[-b, b]$ at zero velocity. (c) Static plus viscous friction, $\tau_{\text{fric}} = b_{\text{static}}\,\text{sgn}(\dot{\theta}) + b_{\text{viscous}}\dot{\theta}$. (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.

available torque by a factor of G, thereby preserving power:

$$\dot{\theta}_{\text{out}} = \frac{\dot{\theta}_{\text{in}}}{G}, \quad \tau_{\text{out}} = G\tau_{\text{in}}, \quad P_{\text{out}} = \tau_{\text{out}}\dot{\theta}_{\text{out}} = (G\tau_{\text{in}})(\dot{\theta}_{\text{in}}/G) = P_{\text{in}}.$$

In practice, however, some power is dissipated due to friction in the gearing, and the torque available at the output is less than $G\tau_{\text{in}}$. This effect tends to increase with the gear ratio G. Typical choices of G are from single digits to over one hundred.

Another option is to directly drive a robot joint with $G = 1$. Motors used in *direct-drive* configurations typically have much higher power ratings than needed for the application, so that they can provide sufficient torque without gearing. These motors never approach their top-end speeds in typical applications.

Now consider the inertia M of our one-joint robot. It is actually a lumped parameter capturing not only the inertia of the link, but also the inertia of the motor's rotor. Typically the motor's inertia I_{motor} is much smaller than the inertia of the link I_{link}. When there is gearing $G > 1$, however, the motor spins faster than the link; if the link angular velocity is $\dot{\theta}$, then the motor speed is $G\dot{\theta}$. We can write the kinetic energy of the link-rotor system as

$$K = \frac{1}{2}\left(I_{\text{link}}\dot{\theta}^2 + I_{\text{motor}}(G\dot{\theta})^2\right) = \frac{1}{2}\underbrace{(I_{\text{link}} + G^2 I_{\text{motor}})}_{M}\dot{\theta}^2,$$

where $G^2 I_{\text{motor}}$ is the inertia of the motor seen from the gearing output shaft. This is called the motor's *reflected inertia* through the gearbox: the effective inertia of the motor as seen at the output of the gearbox. The time derivative of the kinetic energy $\dot{K}$ is the torque driving the link-rotor system multiplied by the joint velocity, or

$$\dot{K} = (I_{\text{link}} + G^2 I_{\text{motor}})\ddot{\theta}\dot{\theta}.$$

As an example, consider $I_{\text{link}} = 1$ kgm^2 and $I_{\text{motor}} = 10^{-3}$ kgm^2. For $G = 1$, 99.9% of the total inertia is due to the link; only 0.1% of the acceleration torque

accelerates the motor. With a gear ratio $G = 100$, however, the effective inertia of the geared motor is ten times the inertia of the link.

When G is chosen so that $G = \sqrt{I_{\text{link}}/I_{\text{motor}}}$, half of the torque is used to accelerate each of the motor and the link, and the system is said to be *inertia matched*, maximizing the power transferred to the link (Exercise 9).

Summarizing, we can make two observations comparing direct-drive and highly geared systems:

- The behavior of geared systems is generally less sensitive to changes in the link inertia, as when the arm carries a load, since the link inertia is a smaller percentage of the total inertia due to the high reflected inertia of geared motor.

- Friction forces are larger in highly geared systems. In the limit where friction forces dominate inertial forces, the joint dynamics may be closer to a first-order viscous system than a second-order inertial system.

These properties play an important role in the analysis of control laws for multi-joint systems, discussed next.

11.2.2   Multi-Joint Motion Control

The methods applied above for a single-joint robot carry over directly to n-joint robots. The difference is that the dynamics (11.4) now take the more general vector-valued form

$$\tau = M(\theta)\ddot{\theta} + h(\theta, \dot{\theta}),$$

where the $n \times n$ positive-definite inertia matrix M is now a function of the configuration θ. We will sometimes find it convenient to explicitly state the components of the term $h(\theta, \dot{\theta})$:

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + g(\theta) + b(\dot{\theta}), \tag{11.20}$$

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 (11.20) are *coupled*—the force or torque at a joint may be a function of the positions, velocities, and accelerations of 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.

11.2.2.1   Decentralized Multi-Joint Control

The simplest method for controlling a multi-joint robot, and one that is often used, is to apply an independent controller at each joint. Decentralized control is appropriate when the dynamics of the joints can be decoupled, or at least approximately decoupled. The dynamics are decoupled when the acceleration

of each joint depends only on the torque applied at that joint. This occurs when the inertia matrix is 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 inertia matrix $M(\theta)$ is nearly diagonal, as it is dominated by the reflected inertias of the motors themselves. Variations in $M(\theta)$ due to different joint configurations are small. Significant friction at the individual joints also contributes to the decoupling of the dynamics.

11.2.2.2 Centralized Multi-Joint Control

When gravity forces and torques are significant and coupled, or when the inertia matrix $M(\theta)$ is not well approximated by a diagonal matrix, decentralized control may no longer yield acceptable performance. In this case, the feedback linearizing control law (11.19) of Figure 11.11 can be generalized. The configurations θ and θ_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:

$$\tau = \widehat{M}(\theta)\left(\ddot{\theta}_d + K_p\theta_e + K_i\int\theta_e(t)dt + K_d\dot{\theta}_e\right) + \widehat{h}(\theta,\dot{\theta}). \qquad (11.21)$$

Typically we choose the gain matrices as $k_p\mathcal{I}, k_i\mathcal{I}, k_d\mathcal{I}$, where $\mathcal{I}$ is the $n \times n$ identity matrix and k_p, k_i, and k_d are nonnegative scalars. In the case of an exact dynamics model $\widehat{M}$ and $\widehat{h}$, the dynamics of each joint reduces to the linear dynamics (11.17). The block diagram and pseudocode for this control algorithm are found in Figures 11.11 and 11.13, respectively.

Implementing the control law (11.21) 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 (11.21) can be obtained using only PID control and gravity compensation:

$$\tau = K_p\theta_e + K_i\int\theta_e(t)dt + K_d\dot{\theta}_e + \widehat{g}(\theta). \qquad (11.22)$$

With zero friction, perfect gravity compensation, and PD setpoint control ($K_i = 0$ and $\dot{\theta}_d = \ddot{\theta}_d = 0$), the controlled dynamics can be written

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} = K_p\theta_e - K_d\dot{\theta}, \qquad (11.23)$$

where the Coriolis and centripetal terms are written $C(\theta,\dot{\theta})\dot{\theta}$, and any viscous friction effects are included in K_d for simplicity. We can now define a virtual

"error energy," the sum of an "error potential energy" stored in the virtual spring and an "error kinetic energy":

$$V(\theta_e, \dot{\theta}_e) = \frac{1}{2}\theta_e^T K_p \theta_e + \frac{1}{2}\dot{\theta}_e^T M(\theta)\dot{\theta}_e. \tag{11.24}$$

Since $\dot{\theta}_d = 0$, this reduces to

$$V(\theta_e, \dot{\theta}) = \frac{1}{2}\theta_e^T K_p \theta_e + \frac{1}{2}\dot{\theta}^T M(\theta)\dot{\theta}. \tag{11.25}$$

Taking the time derivative and plugging in (11.23), we get

$$\begin{aligned}
\dot{V} &= -\dot{\theta}^T K_p \theta_e + \dot{\theta}^T M(\theta)\ddot{\theta} + \frac{1}{2}\dot{\theta}^T \dot{M}(\theta)\dot{\theta} \\
&= -\dot{\theta}^T K_p \theta_e + \dot{\theta}^T \left(K_p \theta_e - K_d \dot{\theta} - C(\theta, \dot{\theta})\dot{\theta} \right) + \frac{1}{2}\dot{\theta}^T \dot{M}(\theta)\dot{\theta}. \tag{11.26}
\end{aligned}$$

Rearranging, and using the fact that $\dot{M} - 2C$ is skew-symmetric, we get

$$\begin{aligned}
\dot{V} &= -\dot{\theta}^T K_p \theta_e + \dot{\theta}^T \left(K_p \theta_e - K_d \dot{\theta} \right) + \frac{1}{2}\dot{\theta}^T \left(\dot{M}(\theta) - 2C(\theta, \dot{\theta}) \right)\dot{\theta} \\
&= -\dot{\theta}^T K_d \dot{\theta} \ \leq 0. \tag{11.27}
\end{aligned}$$

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 $\ddot{\theta} \neq 0$, so $\dot{\theta}_e$ will again become nonzero and more energy will be dissipated. Thus by the Krasovskii-LaSalle invariance principle (Exercise 13), the total error energy decreases monotonically and the robot converges to rest at θ_d ($\theta_e = 0$) from any initial state.

11.2.3 Task Space Motion Control

In Section 11.2.2, 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 $(X(t), \mathcal{V}(t))$, where $X \in SE(3)$ or $X \in \mathbb{R}^n$, for example, with $\mathcal{V} \in \mathbb{R}^n$ the velocity. Provided the corresponding trajectory in joint space is feasible and does not pass through a dynamic singularity where $M(\theta)$ loses rank, we now have two options for control: (1) convert to a joint-space trajectory and proceed with control as in Section 11.2.2 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 $\mathcal{V} = J(\theta)\dot{\theta}$, where $J(\theta)$ is the appropriate Jacobian

based on the chosen velocity representation $\mathcal{V}$. Then the joint space trajectory is obtained from the task space trajectory by

$$\text{(inverse kinematics)} \quad \theta(t) = f^{-1}(X(t)) \tag{11.28}$$

$$\dot{\theta}(t) = J^{-1}(\theta(t))\mathcal{V}(t) \tag{11.29}$$

$$\ddot{\theta}(t) = J^{-1}(\theta(t))\left(\dot{\mathcal{V}}(t) - \dot{J}(\theta(t))\dot{\theta}(t)\right). \tag{11.30}$$

If the robot is redundant, i.e., $J(\theta)$ has more columns than rows, a redundancy resolution scheme must be used to solve for f^{-1} and J^{-1}.

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

$$\mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \gamma(\theta, \mathcal{V}) + \eta(\theta).$$

The joint forces and torques τ are related to the forces $\mathcal{F}$ expressed in the end-effector frame by $\tau = J^T(\theta)\mathcal{F}$.

We can now write a control law in task coordinates inspired by the feedback linearizing control law in joint coordinates (11.21),

$$\boxed{\tau = J^T(\theta)\left(\widehat{\Lambda}(\theta)\left(\dot{\mathcal{V}}_d + K_pX_e + K_i\int X_e(t)dt + K_d\mathcal{V}_e\right) + \widehat{\gamma}(\theta, \mathcal{V}) + \widehat{\eta}(\theta)\right),}$$

$$\tag{11.31}$$

where $\dot{\mathcal{V}}_d$ is the desired acceleration and $\widehat{\Lambda}$, $\widehat{\gamma}$, and $\widehat{\eta}$ represent the controller's dynamics model.

The task-space control law (11.31) makes use of the configuration error X_e and velocity error $\mathcal{V}_e$. When X is expressed in a minimal set of coordinates ($X \in \mathbb{R}^n$) and $\mathcal{V} = \dot{X}$, a natural choice is $X_e = X_d - X, \mathcal{V}_e = \mathcal{V}_d - \mathcal{V}$. When $X = (R, p) \in SE(3)$, however, there are a number of possible choices, including the following:

- $\mathcal{V} = \mathcal{V}_b$ and $J(\theta) = J_b(\theta)$ in the end-effector frame $\{b\}$. A natural choice would be $X_e = \log_{SE(3)}(X^{-1}X_d)$ and $\mathcal{V}_e = \text{Ad}_{X^{-1}X_d}\mathcal{V}_d - \mathcal{V}$. The expression for X_e gives the body-fixed "direction" from the current configuration X to the desired configuration X_d in the end-effector frame. The transform $\text{Ad}_{X^{-1}X_d}$ transports the desired velocity $\mathcal{V}_d$ from X_d to a velocity expressed in the end-effector frame at X.

- $\mathcal{V} = \mathcal{V}_s$ and $J(\theta) = J_s(\theta)$ in the space frame $\{s\}$. A natural choice would be $X_e = \log_{SE(3)}(X_dX^{-1})$ and $\mathcal{V}_e = \mathcal{V}_d - \mathcal{V}$.

- $\mathcal{V}$ and $J(\theta)$ chosen so that $\mathcal{V} = (\omega, v)$, where ω is the angular velocity of the end-effector relative to $\{s\}$ and $v = \dot{p}$. A natural choice would be

$$X_e = \begin{bmatrix} \log_{SO(3)}(R_dR^T) \\ p_d - p \end{bmatrix} \quad \text{and} \quad \mathcal{V}_e = \mathcal{V}_d - \mathcal{V}.$$

These choices lead to different behaviors of the robot. In particular, the last 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-damper 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 the next section.

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 $\mathcal{F}_{\mathrm{app}}$ be the force that the manipulator applies to the environment. The manipulator dynamics can be written

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) + b(\dot{\theta}) + J^T(\theta)\mathcal{F}_{\mathrm{app}} = \tau, \qquad (11.32)$$

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

$$g(\theta) + J^T(\theta)\mathcal{F}_{\mathrm{app}} = \tau. \qquad (11.33)$$

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

$$\tau = \widehat{g}(\theta) + J^T(\theta)\mathcal{F}_d, \qquad (11.34)$$

where $\widehat{g}(\theta)$ is the model of gravitational torques and $\mathcal{F}_d$ is the desired force. 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 direct-drive joint, 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 degrades 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.

A more common 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 forces $\mathcal{F}_{\mathrm{app}}$ (Figure 11.15). Force-torque measurements are often noisy, so the time derivative of these measurements may not be meaningful. In addition, the desired force $\mathcal{F}_d$ is typically constant or only slowly changing.
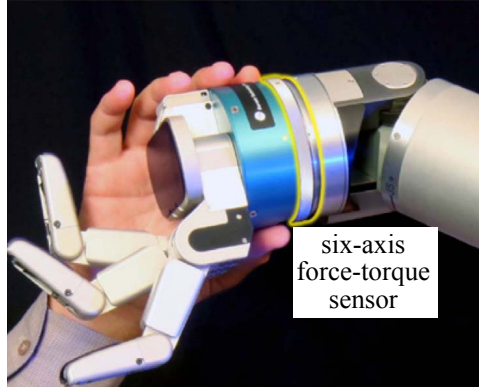
Figure 11.15: A six-axis force-torque sensor, highlighted in yellow, mounted between the Barrett WAM robot arm and its end-effector.

These characteristics suggest a PI controller with a feedforward term and gravity compensation,

$$\tau = \widehat{g}(\theta) + J^T(\theta)\left(\mathcal{F}_d + K_{fp}\mathcal{F}_e + K_{fi}\int \mathcal{F}_e(t)dt\right), \qquad (11.35)$$

where $\mathcal{F}_e = \mathcal{F}_d - \mathcal{F}_{\text{app}}$ 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 (11.35) into the dynamics (11.33), we get the error dynamics

$$K_{fp}\mathcal{F}_e + K_{fi}\int \mathcal{F}_e(t)dt = 0. \qquad (11.36)$$

In the case of a constant force disturbance on the right-hand side of (11.36), arising from an incorrect model of $\widehat{g}(\theta)$, for example, we take the derivative to get

$$K_{fp}\dot{\mathcal{F}}_e + K_{fi}\mathcal{F}_e = 0, \qquad (11.37)$$

showing that $\mathcal{F}_e$ converges to zero for positive-definite K_{fp} and K_{fi}.

The control law (11.35) 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

$$\tau = \widehat{g}(\theta) + J^T(\theta)\left(\mathcal{F}_d + K_{fp}\mathcal{F}_e + K_{fi}\int \mathcal{F}_e(t)dt - K_{\text{damp}}\mathcal{V}\right), \qquad (11.38)$$

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 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, $\mathcal{F} = B_{\mathrm{env}}\mathcal{V}$, where

$$B_{\mathrm{env}} = \left[\begin{array}{cc} 2 & 1 \\ 1 & 1 \end{array} \right].$$

Defining the components of $\mathcal{V}$ and $\mathcal{F}$ as $(\mathcal{V}_1, \mathcal{V}_2)$ and $(\mathcal{F}_1, \mathcal{F}_2)$, we have $\mathcal{F}_1 = 2\mathcal{V}_1 + \mathcal{V}_2, \mathcal{F}_2 = \mathcal{V}_1 + \mathcal{V}_2$. We have $n = 2$ freedoms to choose among the $2n = 4$ velocities and forces at any time. For example, we can specify both $\mathcal{F}_1$ and $\mathcal{V}_1$ independently, because B_{env} is not diagonal. Then $\mathcal{V}_2$ and $\mathcal{F}_2$ are determined by B_{env}. We cannot independently control both $\mathcal{F}_1$ and $2\mathcal{V}_1 + \mathcal{V}_2$, 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 force and torque that has zero moment about the axis of the hinges.

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 determinant in 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
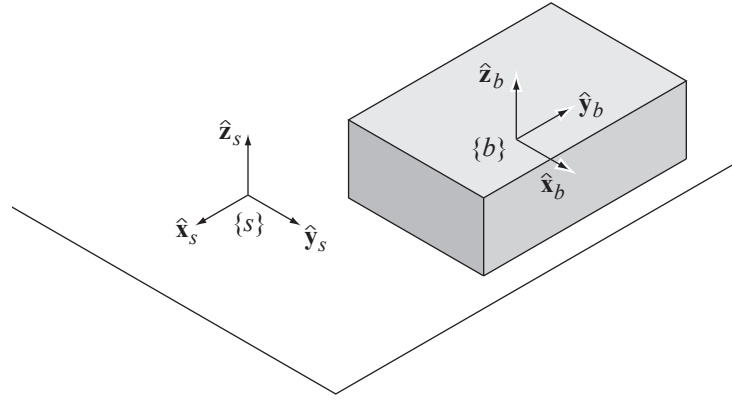
Figure 11.16: The fixed space frame $\{s\}$ attached to the chalkboard and the body frame $\{b\}$ attached to the eraser.

robot's configuration, and all constraints are equality constraints. For example, the inequality velocity constraint of the board (the chalk cannot penetrate 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 chalkboard using an eraser modeled as a rigid block (Figure 11.16). Let the configuration $X(t)$ be expressed in coordinates $q = (\phi, p) = (\phi_x, \phi_y, \phi_z, x, y, z)$, where ϕ are exponential coordinates for the orientation. The velocity is represented as $\mathcal{V} = \dot{q}$. When the eraser is in contact with the board, its configuration $X(t)$ is subject to the constraints

$$\phi_x = 0$$
$$\phi_y = 0$$
$$z = c$$

where c is half the thickness of the eraser. These constraints can be expressed differentially as

$$\dot{\phi}_x = 0$$
$$\dot{\phi}_y = 0$$
$$\dot{z} = 0$$

In the language of Chapter 2, these constraints are *holonomic*—the differential constraints can be integrated to give the configuration constraints.

These constraints are called *natural constraints*, specified by the environment. In light of the natural constraints, we can specify any motion of the eraser satisfying these $k = 3$ velocity constraints, giving $6 - k = 3$ motion freedoms. We are also free to specify $k = 3$ forces, $\mathcal{F}_z, \mathcal{F}_{\phi_x}$, and $\mathcal{F}_{\phi_y}$. These motion and force specifications are called *artificial constraints*. Below is a typical set of

artificial constraints corresponding to the natural constraints:

natural constraints	artificial constraints
$\dot{\phi}_x = 0$	$\mathcal{F}_{\phi_x} = 0$
$\dot{\phi}_y = 0$	$\mathcal{F}_{\phi_y} = 0$
$\mathcal{F}_{\phi_z} = 0$	$\dot{\phi}_z = 0$
$\mathcal{F}_x = 0$	$\dot{x} = k_1$
$\mathcal{F}_y = 0$	$\dot{y} = 0$
$\dot{z} = 0$	$\mathcal{F}_z = k_2$

The artificial constraints cause the eraser to move with an x-velocity k_1 while applying a constant force k_2 against the board.

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

$$A(X)\mathcal{V} = 0, \tag{11.39}$$

where $A(X) \in \mathbb{R}^{k \times n}$. (Alternatively, these constraints could be written in a minimal set of task coordinates as $A(q)\dot{q} = 0$ or in joint coordinates as $A(\theta)\dot{\theta} = 0$.) This formulation includes holonomic and nonholonomic contact constraints with the environment, as well as closure constraints in parallel mechanisms.

If the task-space dynamics of the robot, in the absence of constraints, are

$$\mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \gamma(\theta, \mathcal{V}) + \eta(\theta),$$

then the constrained dynamics are

$$\mathcal{F} = \Lambda(\theta)\dot{\mathcal{V}} + \gamma(\theta, \mathcal{V}) + \eta(\theta) + \underbrace{A^T(X)\lambda}_{\mathcal{F}_{\text{app}}}, \tag{11.40}$$

where $\lambda \in \mathbb{R}^k$ are Lagrange multipliers and $\mathcal{F}_{\text{app}}$ are forces that the robot applies against the constraints. The requested force $\mathcal{F}_d$ must lie in the column space of $A^T(X)$.

Since (11.39) must be satisfied at all times, we can replace (11.39) by the time derivative

$$A(X)\dot{\mathcal{V}} + \dot{A}(X)\mathcal{V} = 0. \tag{11.41}$$

Now, solving (11.40) for $\dot{\mathcal{V}}$, plugging the result into (11.41), and solving for λ, we get

$$\begin{aligned}
\lambda &= (A\Lambda^{-1}A^T)^{-1}(A\Lambda^{-1}(F - \gamma - \eta) + \dot{A}\mathcal{V}) \\
&= (A\Lambda^{-1}A^T)^{-1}(A\Lambda^{-1}(F - \gamma - \eta) - A\dot{\mathcal{V}}), \tag{11.42}
\end{aligned}$$

where we have plugged in $-A\dot{\mathcal{V}} = \dot{A}\mathcal{V}$ by (11.41). With (11.42), we can calculate the forces $\mathcal{F}_{\text{app}} = A^T(q)\lambda$ the robot applies against the constraints.

Plugging (11.42) into (11.40) and manipulating, the n equations of the constrained dynamics (11.40) can be expressed as the $n - k$ independent motion equations

$$P(X)\mathcal{F} = P(X)(\Lambda(\theta)\dot{\mathcal{V}} + \gamma(\theta, \mathcal{V}) + \eta(\theta)) \tag{11.43}$$

where

$$P = \mathcal{I} - A^T(A\Lambda^{-1}A^T)^{-1}A\Lambda^{-1} \tag{11.44}$$

and $\mathcal{I}$ 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 $\mathcal{I} - P(X)$ 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 feedback linearizing control law (11.31), and a task-space force controller (11.35), each projected to generate forces in its appropriate subspace:

$$\tau = J^T(\theta)\Bigg[\underbrace{P(X)\left(\widehat{\Lambda}(\theta)\left[\dot{\mathcal{V}}_d + K_p X_e + K_i \int X_e(t)dt + K_d \mathcal{V}_e\right]\right)}_{\text{motion control}}$$

$$+ \underbrace{(\mathcal{I} - P(X))\left(\mathcal{F}_d + K_{fp}\mathcal{F}_e + K_{fi}\int \mathcal{F}_e(t)dt\right)}_{\text{force control}}$$

$$+ \underbrace{\widehat{\gamma}(\theta, \mathcal{V}) + \widehat{\eta}(\theta)}_{\text{nonlinear compensation}} \Bigg]. \tag{11.45}$$

Because the dynamics of the two controllers are decoupled by the orthogonal projections P and $\mathcal{I} - P$, 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 (11.45) 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.
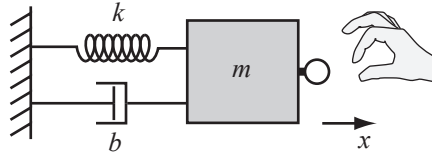
Figure 11.17: A robot creating a one-degree-of-freedom mass-spring-damper virtual environment. The human hand applies a force f to the haptic interface.

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 mimics particular mass, spring, and damper properties.[1] 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 tissues.

A one-degree-of-freedom environment can be written

$$m\ddot{x} + b\dot{x} + kx = f, \tag{11.46}$$

where x is the position, m is the mass, b is the damping, and k is the stiffness, and f is the force applied to the environment (Figure 11.17). Collectively, we refer to the parameters $\{m, b, k\}$ as the impedance. Loosely, we say that the environment has high impedance if one or more of these 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 (11.46), we get

$$(ms^2 + bs + k)X(s) = F(s), \tag{11.47}$$

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

---

[1] A popular subcategory of impedance control is *stiffness* or *compliance control*, where the robot mimics a virtual spring only.

ΔF produce only small position perturbations ΔX. 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

$$D\dot{\mathcal{V}} + B\mathcal{V} + KX = \mathcal{F}_{\text{ext}}, \tag{11.48}$$

where $X \in \mathbb{R}^n$ and $\mathcal{V} = \dot{X}$ are the task-space position and velocity; D, B, and K are the positive-definite virtual inertia, damping, and stiffness to be created by the robot; and $\mathcal{F}_{\text{ext}}$ is a force applied to the robot, perhaps by a user. The values of D, 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 can replace X and $\mathcal{V}$ in (11.48) by $X(t) - X_{\text{ref}}(t)$ and $\mathcal{V}(t) - \mathcal{V}_{\text{ref}}(t)$. If $X_{\text{ref}}(t)$ is time-varying, it will "pull" the robot approximately along the trajectory. This allows impedance control to achieve motion control, much like a task-space PD controller. Larger stiffness K and damping B achieve more precise trajectory tracking.

There are at least two ways to achieve the behavior (11.48):

- The robot senses motions and commands joint torques to create $-\mathcal{F}_{\text{ext}}$, 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. These robots tend to be lightweight and backdrivable, and are often cable-driven. They are typically good at displaying low impedances, and not as good at displaying stiff, high impedance environments.

- The robot senses $\mathcal{F}_{\text{ext}}$ 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. These robots tend to be highly geared. They are typically good at displaying stiff, high impedance environments, and not as good at displaying low impedances.

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 these robots are not equipped with a force-torque sensor, and instead rely on their ability to precisely control joint torques with little friction to display the appropriate end-effector force $-\mathcal{F}_{\text{ext}}$ (from (11.48)) to the user. An ideal control law might be

$$\tau = J^T(\theta) \left[\underbrace{\widehat{\Lambda}(\theta)\dot{\mathcal{V}} + \widehat{\gamma}(\theta, \mathcal{V}) + \widehat{\eta}(\theta)}_{\text{arm dynamics compensation}} - \underbrace{(D\dot{\mathcal{V}} + B\mathcal{V} + KX)}_{\mathcal{F}_{\text{ext}}} \right]. \tag{11.49}$$

Addition of an end-effector force-torque sensor allows the use of feedback terms to more closely achieve the desired interaction force $-\mathcal{F}_{\text{ext}}$.

In the control law (11.49), it is assumed that $\dot{\mathcal{V}}$, $\mathcal{V}$, and X are measured directly. Measurement of the acceleration $\dot{\mathcal{V}}$ is likely to be noisy, however, and there is the problem of attempting to compensate for the robot's inertia after the acceleration has been sensed. Therefore, it is not uncommon to eliminate the inertial compensation term $\widehat{\Lambda}(\theta)\dot{\mathcal{V}}$ and to set $D = 0$. The inertia of the arm will be apparent to the user, but impedance-controlled manipulators are often designed to be lightweight.

Additionally, instability can arise when (11.49) 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 in stiff virtual environments. On the other hand, the effective gains are low when emulating low impedance environments. A lightweight backdrivable manipulator can excel at impedance control.

11.5.2 Admittance Control Algorithm

In an admittance control algorithm, the force $\mathcal{F}_{\text{ext}}$ applied by the user is sensed by the wrist load cell, and the robot responds with an end-effector acceleration satisfying (11.48). A simple approach is to calculate the desired end-effector acceleration $\dot{\mathcal{V}}_d$ according to

$$D\dot{\mathcal{V}}_d + B\mathcal{V} + KX = \mathcal{F}_{\text{ext}},$$

where $(X, \mathcal{V})$ is the current state. Solving, we get

$$\dot{\mathcal{V}}_d = D^{-1}(\mathcal{F}_{\text{ext}} - B\mathcal{V} - KX). \tag{11.50}$$

Given $\dot{\mathcal{V}}_d$, $\mathcal{V}$, and X, integration over the servo timestep Δt yields the desired position and velocity X_d and $\mathcal{V}_d$. The reference $X_d, \mathcal{V}_d, \dot{\mathcal{V}}_d$ can be plugged into the feedback linearizing control law in task coordinates (11.31). To make the response smoother, the force readings can be low-pass filtered.

Simulating a low impedance environment is challenging for an admittance control algorithm, 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 inertia 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. For example, 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 focused on a single trajectory.

Passive Compliance and Flexible Manipulators   All robots unavoidably have some passive compliance. Models of this compliance could be as simple as a single torsional spring at each revolute joint (e.g., to account for finite stiffness in the flexsplines 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 position control performance in favor of safety.

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 comprise two motors, allowing the actuator to independently control the mechanical stiffness of the joint (e.g., based on the setpoint of an internal nonlinear spring) and the torque produced by the actuator.

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 θ_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

$$a_n \theta_e^{(n)} + a_{n-1} \theta_e^{(n-1)} + \ldots + a_2 \ddot\theta_e + a_1 \dot\theta_e + a_0 \theta_e = 0$$

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., $\mathrm{Re}(s_i) < 0$ for all $i = 1 \ldots n$.

- 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 ζ is the damping ratio and ω_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 generally characterized by its overshoot, 2% settling time, and steady-state error.

- Highly geared robots tend to have greater decoupling of their dynamics due to high friction and large reflected motor inertias.

- The joint-space feedback linearizing (or computed torque) controller is

$$\tau = \widehat{M}(\theta) \left(\ddot\theta_d + K_p \theta_e + K_i \int \theta_e(t) dt + K_d \dot\theta_e \right) + \widehat{h}(\theta, \dot\theta).$$

This controller cancels nonlinear terms, uses feedforward control to anticipate the desired acceleration, and uses linear feedback control for stabilization.

- Joint-space PD control plus gravity compensation

$$\tau = K_p \theta_e + K_d \dot\theta_e + \widehat{g}(\theta)$$

yields global convergence to $\theta_e = 0$ by the Krasovskii-LaSalle invariance principle.

- The task-space feedback linearizing motion controller is

$$\tau = J^T(\theta)\left(\widehat{\Lambda}(\theta)\left(\dot{\mathcal{V}}_d + K_p X_e + K_i \int X_e(t)dt + K_d \mathcal{V}_e\right) + \widehat{\gamma}(\theta, \mathcal{V}) + \widehat{\eta}(\theta)\right).$$

  There are a number of possible choices for calculating X_e and $\mathcal{V}_e$.

- Task-space force control can be achieved by the controller

$$\tau = \widehat{g}(\theta) + J^T(\theta)\left(\mathcal{F}_d + K_{fp}\mathcal{F}_e + K_{fi}\int \mathcal{F}_e(t)dt - K_{\text{damp}}\mathcal{V}\right),$$

  consisting of gravity compensation, feedforward force control, PI force feedback, and damping to prevent fast motion.

- In an n-dimensional task space (where typically $n = 6$), rigid constraints specify $n - k$ free motion directions and k constraint directions in which forces can be applied. These constraints can be represented as $A(X)\mathcal{V} = 0$. A force $\mathcal{F}$ can be partitioned as $\mathcal{F} = P(X)\mathcal{F} + (\mathcal{I} - P(X))\mathcal{F}$, where $P(X)$ projects to forces that move the end-effector tangent to the constraints and $\mathcal{I} - P(X)$ projects to forces against the constraints. The projection matrix $P(X)$ is written in terms of the task-space inertia matrix $\Lambda(\theta)$ and constraints $A(X)$ as

$$P = \mathcal{I} - A^T(A\Lambda^{-1}A^T)^{-1}A\Lambda^{-1}.$$

- A hybrid motion-force controller using the projection $P(X)$ can be written

$$\tau = J^T(\theta)\Bigg[\underbrace{P(X)\left(\widehat{\Lambda}(\theta)\left[\dot{\mathcal{V}}_d + K_p X_e + K_i \int X_e(t)dt + K_d \mathcal{V}_e\right]\right)}_{\text{motion control}}$$

$$+ \underbrace{(\mathcal{I} - P(X))\left(\mathcal{F}_d + K_{fp}\mathcal{F}_e + K_{fi}\int \mathcal{F}_e(t)dt\right)}_{\text{force control}}$$

$$+ \underbrace{\widehat{\gamma}(\theta, \mathcal{V}) + \widehat{\eta}(\theta)}_{\text{nonlinear compensation}}\Bigg].$$

- 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   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 $t = 4/(\zeta\omega_n)$, based on $\exp(-\zeta\omega_n t) = 0.02$. 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 $g = 10$ 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 $b = 0.1$ Nms/rad.

(i) Write the equation of motion of the pendulum in terms of θ, 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 θ 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 $\ddot{x}_e$, $\dot{x}_e$, 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 11.2.1. 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 Δt. A first-order Euler integration method suffices for this problem (e.g., $\theta(k+1) = \theta(k) + \dot{\theta}(k)\Delta t$, $\dot{\theta}(k+1) = \dot{\theta}(k) + \ddot{\theta}(k)\Delta t$). 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 Δt 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. Consider a motor with rotor inertia I_{motor} connected through a gearhead of gear ratio G to a load of inertia I_{link}. The load and motor are said to be inertia matched if, for any given torque τ_m at the motor, the acceleration of the load is maximized. The acceleration of the load can be written

$$\ddot{\theta} = \frac{G\tau_m}{I_{\text{link}} + G^2 I_{\text{motor}}}.$$

Solve for the inertia matching gear ratio $\sqrt{I_{\text{link}}/I_{\text{motor}}}$ by solving $d\ddot{\theta}/dG = 0$. Show your work.

10. Simulation of a two-degree-of-freedom robot and robot controller (Figure 11.18).

(i) *Dynamics.* Derive the dynamics of a 2R robot in gravity (Figure 11.18). The mass of link i is m_i, the center of mass is a distance r_i from the joint, the 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_i^{stator} and inertia I_i^{stator} of the stator and the mass m_i^{rotor} and inertia I_i^{rotor} 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, $g = 9.81$ m/s^2 in the direction indicated, $L_i = 1$ m, $r_i = 0.5$ m, $m_1 = 3$ kg, $m_2 = 2$ kg, $I_1 = 2$ kgm^2, and $I_2 = 1$ 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 $t < 1$ s and $(\theta_1, \theta_2) = (0, -\pi/2)$ for $t \geq 1$ 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, $\tau_i \leq |\tau_i^{\text{max}}|$. Assume $\tau_1^{\text{max}} = 100$ Nm and $\tau_2^{\text{max}} = 20$ 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 $b_i = 1$ Nms/rad to each joint and rerun the previous PID control simulation.
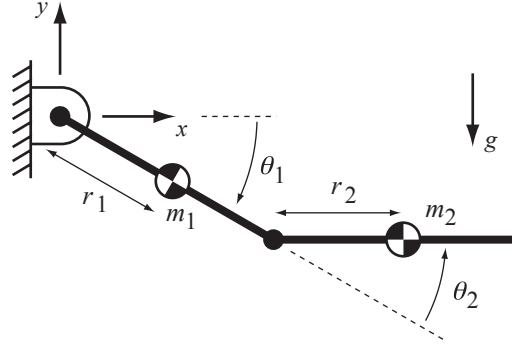
Figure 11.18: A two-link robot arm. The length of link i is L_i and its inertia about the joint is I_i. Gravity is $g = 9.81$ m/s^2.

11. For the two-joint robot of Exercise 10, 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

```
[qd,qdotd,qdotdotd] = trajectory(time)
```

returns the desired position, velocity, and acceleration of each joint at time `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

$$\theta_d(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5. \qquad (11.51)$$

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 (11.51) 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.

12. For the two-joint robot of Exercise 10 and fifth-order polynomial trajectory of Exercise 11, write a feedback linearizing 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 feedback linearizing controller as well as PID control only.

13. 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 $x \neq 0$;

- $V(x) \to \infty$ as $x \to \infty$;

- $V(0) = \dot{V}(0) = 0$; and

- $\dot{V}(x) \leq 0$ along all trajectories of the system.

Let $\mathcal{S}$ be the largest set of $\mathbb{R}^n$ such that $\dot{V}(x) = 0$ and trajectories beginning in $\mathcal{S}$ remain in $\mathcal{S}$ for all time. Then if $\mathcal{S}$ 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 (11.24), 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.

14. The two-joint robot of Exercise 10 can be controlled in task space using the endpoint task coordinates $X = (x, y)$, as shown in Figure 11.18. The task space velocity is $\mathcal{V} = \dot{X}$. Give the Jacobian $J(\theta)$ and the task-space dynamics models $\widehat{\Lambda}(\theta)$, $\widehat{\gamma}(\theta, \mathcal{V})$, and $\widehat{\eta}(\theta)$ in the feedback linearizing control law (11.31).

15. 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.

16. Assume the end-effector of the two-joint robot in Figure 11.18 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 $A(X)\mathcal{V} = 0$, where $X = (x, y)$ and $\mathcal{V} = \dot{X}$. (b) Write
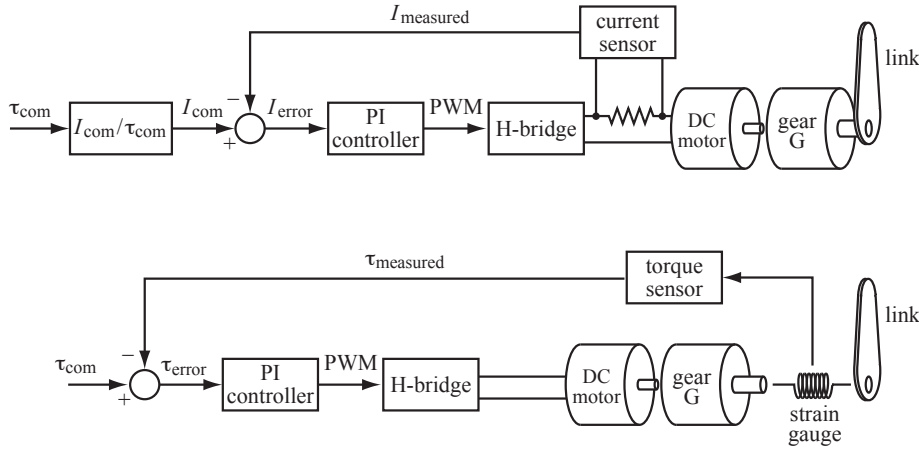
Figure 11.19: Two methods for controlling the torque at a joint driven by a geared DC motor. (Top) The current to the motor is measured by measuring the voltage across a small resistance in the current path. A PI controller works to make the actual current better match the requested current I_{com}. (Bottom) The actual torque delivered to the link is measured by a strain gauge.

the constraint as $A(\theta)\dot{\theta} = 0$.

17. Derive the constrained motion equations (11.43) and (11.44). Show all the steps.

18. 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 11.19 shows two possibilities for a DC electric motor, where the torque τ 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_{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 τ_{com}. Since a strain gauge measures deflection, the element it is mounted on must have a finite torsional stiffness. Actuators called *series elastic actuators* are designed to have particularly flexible torsional elements.

(i) For the current sensing scheme, what multiplicative factor should go in the block labeled $I_{\mathrm{com}}/\tau_{\mathrm{com}}$? Even if the PI current controller does its job perfectly ($I_{\mathrm{error}} = 0$) 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.

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; perhaps stationary constraints such as floors, walls, and fixtures; and one or more controlled manipulators (e.g., "fingers") which could be under motion, force, hybrid motion-force, or impedance control.

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:

d	$\dot{d}$	$\ddot{d}$	$\ldots$	
> 0				no contact
< 0				infeasible (penetration)
$= 0$	> 0			in contact, but breaking free
$= 0$	< 0			infeasible (penetration)
$= 0$	$= 0$	> 0		in contact, but breaking free
$= 0$	$= 0$	< 0		infeasible (penetration)
etc.				

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

$$\dot{d} = \frac{\partial d}{\partial q} \dot{q} \tag{12.1}$$

$$\ddot{d} = \dot{q}^T \frac{\partial^2 d}{\partial q^2} \dot{q} + \frac{\partial d}{\partial q} \ddot{q}. \tag{12.2}$$

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 12.1). 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
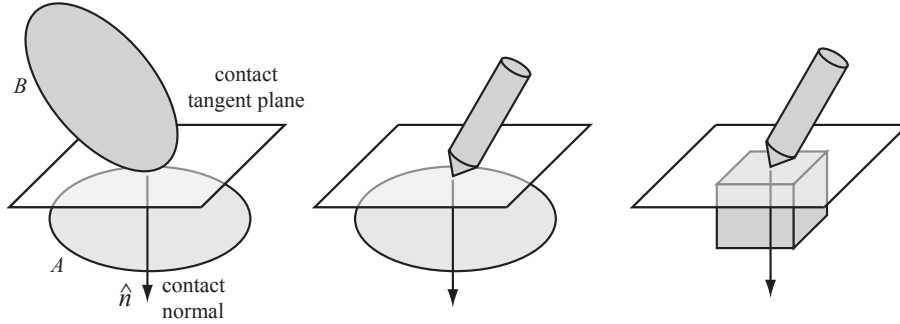
Figure 12.1: (Left) The parts A and B in single-point contact define a contact tangent plane and a contact normal vector $\hat{n}$ perpendicular to the tangent plane. By default, the positive direction of the normal is chosen into body A. Since contact curvature is not addressed in this chapter, the contact places the same restrictions on the motions of the rigid bodies in the middle and right panels.

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 12.1 are treated identically.

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, as discussed in Chapter **??**. 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 12.1). Let $\hat{n}$ be a unit vector aligned with the contact normal, expressed in a world frame. Let p_A be a representation of the point in contact on part A in the world frame, and p_B 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.[1] 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 $\mathcal{F} = (m, f)$ corresponding to a unit force applied along the contact normal:

$$\mathcal{F} = (p_A \times \hat{n}, \hat{n}) = ([p_A]\hat{n}, \hat{n}).$$

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

$$\textbf{(impenetrability constraint)} \qquad \mathcal{F}^T(\mathcal{V}_A - \mathcal{V}_B) \geq 0. \qquad (12.5)$$

(See Exercise 1). If

$$\textbf{(active constraint)} \qquad \mathcal{F}^T(\mathcal{V}_A - \mathcal{V}_B) = 0, \qquad (12.6)$$

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

$$\mathcal{F}^T\mathcal{V}_A \geq 0. \qquad (12.7)$$

If the condition (12.7) is satisfied, $\mathcal{F}$ and $\mathcal{V}_A$ are said to be **repelling**. If $\mathcal{F}^T\mathcal{V}_A = 0$, $\mathcal{F}$ and $\mathcal{V}_A$ are said to be **reciprocal** and the constraint is active.

Twists $\mathcal{V}_A$ and $\mathcal{V}_B$ satisfying (12.6) are called **first-order roll-slide motions**—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:

$$\textbf{(rolling constraint)} \qquad \dot{p}_A = v_A + [\omega_A]p_A = v_B + [\omega_B]p_B = \dot{p}_B. \qquad (12.8)$$

Note that "rolling" contacts include those where the two parts remain stationary relative to each other, i.e., no relative rotation.

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 important when we consider friction forces in Section 12.2.

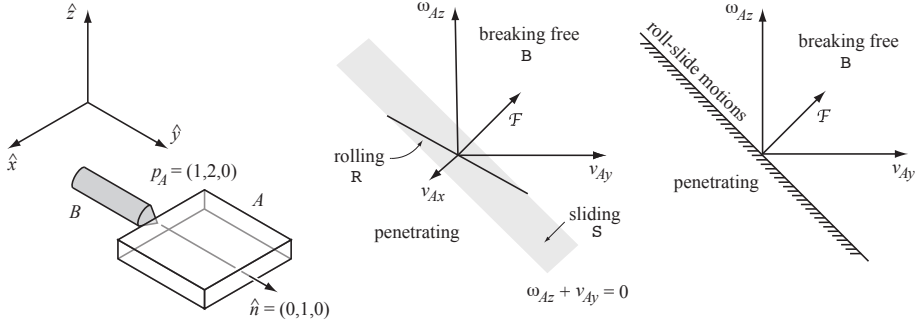[1]All twists and wrenches are expressed in an inertial space frame in this chapter.

Figure 12.2: Example 12.1. (Left) The body B makes contact with A at $p_A = p_B = (1, 2, 0)$ with normal $\hat{n} = (0, 1, 0)$. (Middle) The velocities $\mathcal{V}_A$ and their corresponding contact labels for B stationary and A confined to a plane. The contact normal force $\mathcal{F}$ is $m_z = 1, f_x = 0, f_y = 1$. (Right) Looking down the $-v_{Ax}$ axis.

Example 12.1. Consider the contact shown in Figure 12.2. Parts A and B are in contact at $p_A = p_B = (1, 2, 0)$ with contact normal direction $\hat{n} = (0, 1, 0)$. The impenetrability constraint (12.5) is

$$\mathcal{F}^T(\mathcal{V}_A - \mathcal{V}_B) \geq 0$$
$$([p_A]\hat{n}, \hat{n})^T(\omega_A - \omega_B, v_A - v_B) \geq 0$$
$$[0, 0, 1, 0, 1, 0]\,[\omega_{Ax} - \omega_{Bx}, \omega_{Ay} - \omega_{By}, \omega_{Az} - \omega_{Bz},$$
$$v_{Ax} - v_{Bx}, v_{Ay} - v_{By}, v_{Az} - v_{Bz}]^T \geq 0$$
$$\omega_{Az} - \omega_{Bz} + v_{Ay} - v_{By} \geq 0,$$

and therefore roll-slide twists satisfy

$$\omega_{Az} - \omega_{Bz} + v_{Ay} - v_{By} = 0. \tag{12.9}$$

Equation (12.9) defines an 11-dimensional hyperplane in the 12-dimensional space of twists $(\mathcal{V}_A, \mathcal{V}_B)$.

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

$$v_{Ax} + 2\omega_{Az} = v_{Bx} + 2\omega_{Bz} \tag{12.10}$$
$$v_{Ay} + \omega_{Az} = v_{By} + \omega_{Bz} \tag{12.11}$$
$$v_{Az} + 2\omega_{Ax} - \omega_{Ay} = v_{Bz} + 2\omega_{Bx} - \omega_{By}. \tag{12.12}$$

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 ($\mathcal{V}_B = 0$) and A is confined to the $z = 0$ plane, i.e., $\mathcal{V}_A = (\omega_{Ax}, \omega_{Ay}, \omega_{Az}, v_{Ax}, v_{Ay}, v_{Az}) = (0, 0, \omega_{Az}, v_{Ax}, v_{Ay}, 0)$. The wrench $\mathcal{F}$ is written $(m_z, f_x, f_y) = (1, 0, 1)$. The roll-slide constraint (12.9) reduces to

$$\omega_{Az} + v_{Ay} = 0,$$

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 $\mathcal{V}_B = 0$, the constraint surfaces pass through the origin $\mathcal{V}_A = 0$. If $\mathcal{V}_B \neq 0$, this is no longer the case in general.

Figure 12.2 graphically shows that nonpenetrating twists $\mathcal{V}_A$ must have a nonnegative dot product with the constraint wrench $\mathcal{F}$ when $\mathcal{V}_B = 0$.

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 $\mathcal{V}_A$ to a half-space of its six-dimensional twist space bounded by a five-dimensional hyperplane of the form $\mathcal{F}^T \mathcal{V}_A = \mathcal{F}^T \mathcal{V}_B$. 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 $\mathcal{V}_A$ space, written

$$V = \{\mathcal{V}_A \mid \mathcal{F}_i^T (\mathcal{V}_A - \mathcal{V}_i) \geq 0 \ \forall i\},$$

where $\mathcal{F}_i$ corresponds to the ith contact normal and $\mathcal{V}_i$ 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 $\mathcal{F}_i$ 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 $\mathcal{V}_A$ 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 $\mathcal{V}_A$ space. We call such a constraint **homogeneous**. The feasible twist set becomes

---

[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.

a cone rooted at the origin, called a (homogeneous) **polyhedral convex cone**. Let $\mathcal{F}_i$ be the constraint wrench of stationary contact i. Then the feasible twist cone V is

$$V = \{\mathcal{V}_A \mid \mathcal{F}_i^T \mathcal{V}_A \geq 0 \ \ \forall i\}.$$

If the $\mathcal{F}_i$ positively span the six-dimensional wrench space, or, equivalently, the convex hull of the $\mathcal{F}_i$ 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 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 12.3 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 $\mathcal{V}_A = (0, 0, 0, v_{Ax}, v_{Ay}, 0)$. In Figure 12.3(a), the single stationary finger creates a contact wrench $\mathcal{F}_1$ that can be drawn in the $\mathcal{V}_A$ space. All feasible twists have a nonnegative component in the direction of $\mathcal{F}_1$. Roll-slide twists satisfying $\mathcal{F}_1^T \mathcal{V}_A = 0$ lie on the constraint line. Since no rotations are allowed, the only twist yielding a rolling contact is $\mathcal{V}_A = 0$. In Figure 12.3(b), the union of the constraints due to two stationary fingers creates a cone of feasible twists. Figure 12.3(c) shows three fingers in contact, one of which is moving with twist $\mathcal{V}_3$. Because the moving finger has nonzero velocity, its constraint half-space is displaced from the origin by $\mathcal{V}_3$. The result is a closed polygon of feasible twists.

Example 12.3. Figure 12.4 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

$$
\begin{aligned}
v_{Ay} - 2\omega_{Az} &\geq 0 \\
-v_{Ax} + \omega_{Az} &\geq 0 \\
v_{Ax} + \omega_{Az} &\geq 0.
\end{aligned}
$$

These constraints describe a polyhedral convex cone of feasible twists rooted at the origin, as illustrated in Figure 12.4.
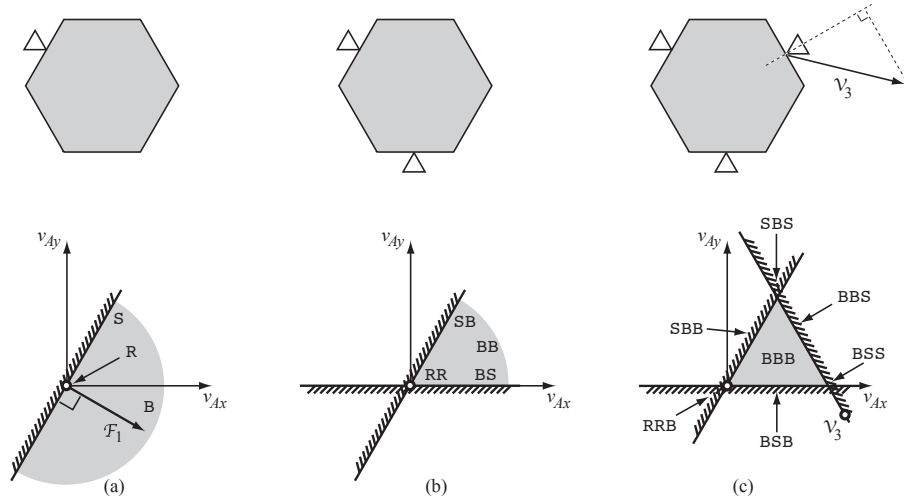
Figure 12.3: Motion-controlled fingers contacting a hexagon that is constrained to translate in a plane only (Example 12.2). (a) A single stationary finger provides a single half-space constraint on the hexagon's twist $\mathcal{V}_A$. The feasible motion half-space is shaded gray. The two-dimensional set of twists corresponding to breaking contact B, the one-dimensional set corresponding to sliding contact S, and the zero-dimensional set corresponding to rolling (fixed) contact 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: RR, SB, BS, and BB. 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 ∘ 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 RRR.

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

$$\mathcal{F}^T(\mathcal{V}_i - \mathcal{V}_j) \geq 0 \tag{12.13}$$

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
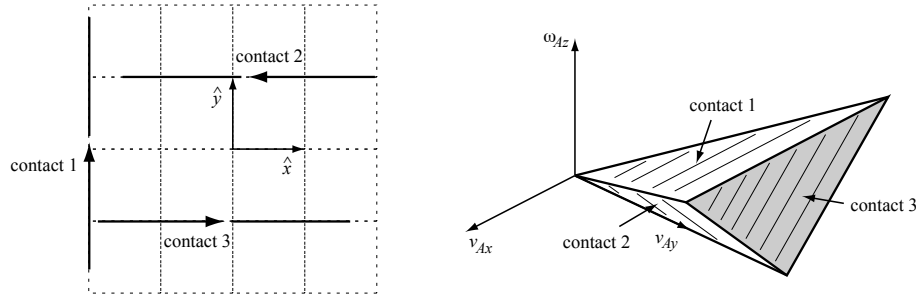
Figure 12.4: Example 12.3. (Left) The lines of force corresponding to three stationary contacts on a planar body. If we are only concerned with feasible motions, and do not distinguish between rolling and sliding, contacts anywhere along the lines, with the contact normals shown, are equivalent. (Right) 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 an active constraint.

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 moving contacts whose motion is prescribed, e.g., robot fingers, the constraints on the motion of the remaining parts are no longer homogeneous. As a result, the polyhedral feasible 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 12.5(a), where at least one of the bodies in contact uniquely defines the contact normal. Figures 12.5(b)-(e) show other types of contact. The kinematic constraints provided by the convex-concave vertex, line, and plane contacts of Figures 12.5(b)-(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 12.5(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 contact. Normal and tangential contact forces are subject to constraints: the normal force must be pushing into a part, not pulling, and the maximum friction

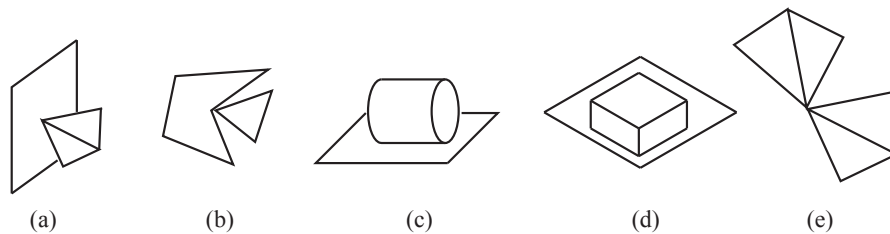(a)   (b)   (c)   (d)   (e)

Figure 12.5: (a) Vertex-face contact. (b) A convex vertex contacting a concave vertex can be treated as multiple point contacts, one at each face adjacent to the concave vertex. These faces define the contact normals. (c) A line contact can be treated as two point contacts at either end of the line. (d) A plane contact can be treated as point contacts at the corners of the convex hull of the contact area. (e) Convex vertex-vertex contact. This case is degenerate and not considered.

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 **hard-finger contact** (or point contact with friction), and a **soft-finger contact**. A frictionless point contact enforces only the roll-slide constraint (12.5). A hard-finger contact also enforces the rolling constraints (12.8), implicitly modeling friction sufficient to prevent slip at the contact. A soft-finger 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 small but nonzero contact area between the two bodies. For planar problems, a hard-finger contact and a soft-finger contact are identical.

While these kinematic models are convenient for studying the kinematic mobility of bodies in contact, we will not use them in the rest of this 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 12.4. 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 $\mathcal{V} = (\omega_z, v_x, v_y)$ is as a **center of rotation** (CoR) at $(-v_y/\omega_z, v_x/\omega_z)$ plus the angular velocity ω_z. The CoR is the point in the (projective) plane that remains stationary under the motion.[3] In the case that the speed of motion is immaterial, we may simply label the CoR with a $+$, $-$, or 0 sign representing the direction of rotation (Figure 12.6).

---

[3]Note that the case $\omega_z = 0$ must be treated with care, as it corresponds to a CoR at infinity.
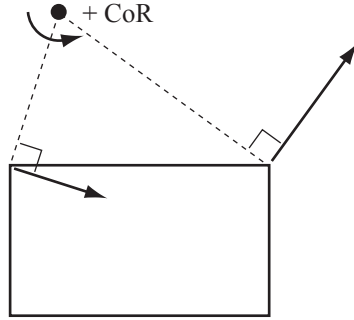
Figure 12.6: Given the velocity of two points on the part, the lines normal to the velocities intersect at the CoR. The CoR shown is labeled + corresponding to the (counterclockwise) positive angular velocity of the part.
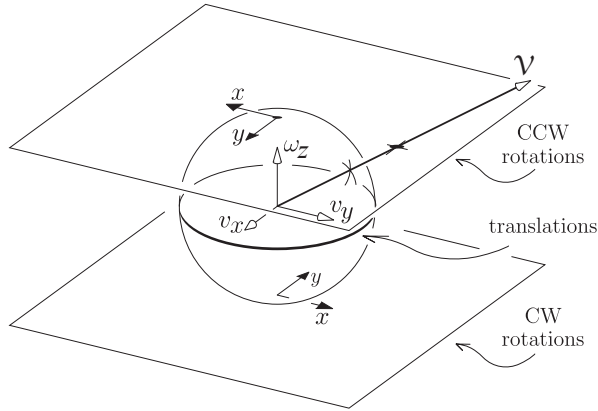

Figure 12.7: Mapping a planar twist $\mathcal{V}$ to a CoR. The ray containing a vector $\mathcal{V}$ intersects either the plane of $+$ CoRs at $\omega_z = 1$, the plane of $-$ CoRs at $\omega_z = -1$, or the circle of translation directions.

The mapping from planar twists to CoRs is illustrated in Figure 12.7, 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 $\mathrm{CoR}(\mathcal{V}_1)$ and $\mathrm{CoR}(\mathcal{V}_2)$. Since k_1 and k_2 can have either sign, if either ω_{1z} or ω_{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 nonnegative linear combination of these two velocities is written

$$V = \mathrm{pos}(\{\mathcal{V}_1, \mathcal{V}_2\}) = \{k_1\mathcal{V}_1 + k_2\mathcal{V}_2 \mid k_1, k_2 \geq 0\},$$
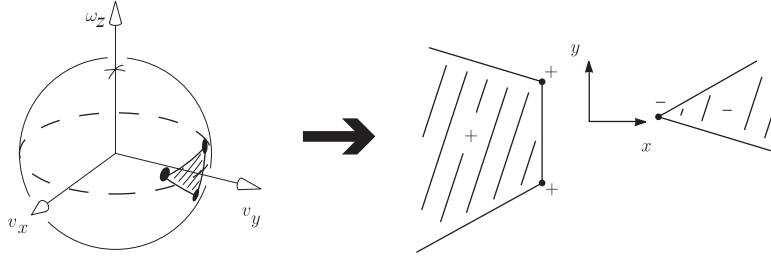
Figure 12.8: The intersection of a twist cone with the unit twist sphere, and the representation of the cone as a set of CoRs.
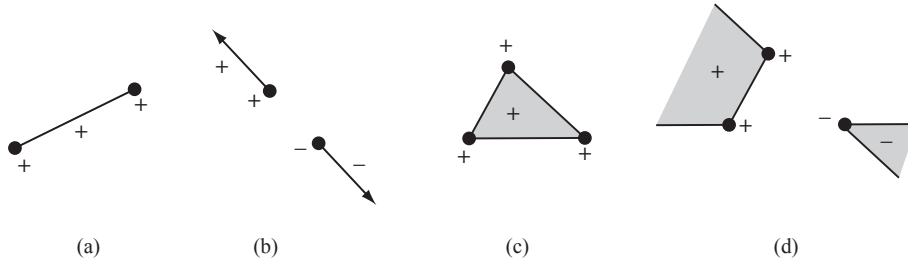

Figure 12.9: (a) Positive linear combination of two CoRs labeled $+$. (b) Positive linear combination of a $+$ CoR and a $-$ CoR. (c) Positive linear combination of three $+$ CoRs. (d) Positive linear combination of two $+$ CoRs and a $-$ CoR.

a polyhedral convex twist cone rooted at the origin, with $\mathcal{V}_1$ and $\mathcal{V}_2$ defining the edges of the cone. If ω_{1z} and ω_{2z} have the same sign, then the CoRs of their nonnegative linear combinations $\text{CoR}(\text{pos}(\{\mathcal{V}_1, \mathcal{V}_2\}))$ all have that sign, and lie on the line segment between the two CoRs. If $\text{CoR}(\mathcal{V}_1)$ and $\text{CoR}(\mathcal{V}_2)$ are labeled $+$ and $-$, respectively, then $\text{CoR}(\text{pos}(\{\mathcal{V}_1, \mathcal{V}_2\}))$ 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 $\text{CoR}(\mathcal{V}_1)$, a ray of CoRs labeled $-$ attached to $\text{CoR}(\mathcal{V}_2)$, 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 12.8 and 12.9 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. This cone can be represented uniquely by a set of CoRs with $+$, $-$, and 0 labels. A general twist polytope, as in the case of moving constraints, cannot be uniquely represented 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
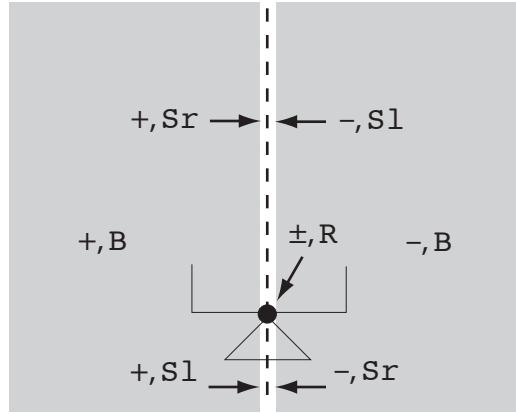
Figure 12.10: The stationary triangle makes contact with a movable part. CoRs to the left of the contact normal are labeled +, to the right are labeled −, and on the normal labeled ±. Also given are the contact types for the CoRs. For points on the contact normal, the sign assigned to `Sl` and `Sr` CoRs switches at the contact point.

on the contact normal ±, 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: `Sr`, where the moving body slips right relative to the fixed constraint, and `Sl`, where the moving body slips to the left. Figure 12.10 illustrates the labeling. If there is more than one contact, we simply union the constraints and contact labels from the individual contacts.

Example 12.4. Figure 12.11(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 12.11(b)). Each feasible CoR is labeled with a contact mode that concatenates the labels for the individual contacts (Figure 12.11(c)).

Now look more closely at the CoR labeled (+, `SrBSr`) in Figure 12.11(c). Is this motion really possible? It should be apparent that it is, in fact, *not* 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.
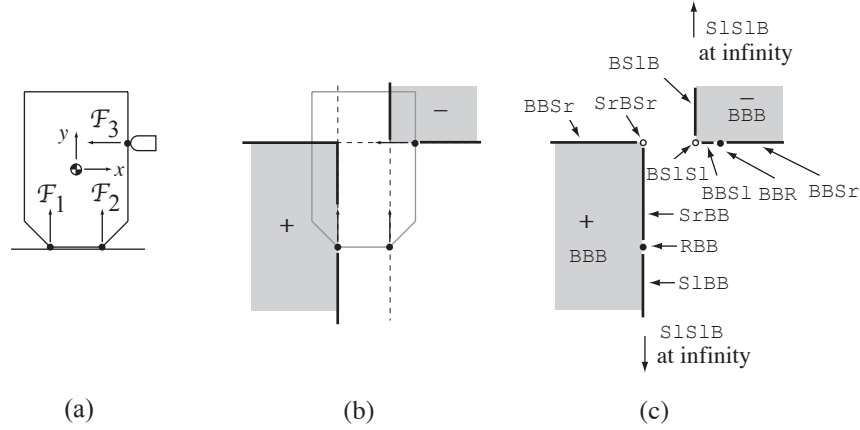
Figure 12.11: Example 12.4. (a) A part resting on a table and the three contact constraints. (b) The feasible twists represented as CoRs, shown in gray. Note that the lines that extend off to the left and to the bottom "wrap around" at infinity and come back in from the right and the top, respectively, 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.

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 nth-order analysis indicates that the contact is breaking or penetrating, then no analysis of order greater than n will change the conclusion, because the higher-order effects are smaller.

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**.

Before proceeding, we introduce a few definitions from linear algebra.

Definition 12.1. Given a set of j vectors $\mathcal{A} = a_1, \ldots a_j \in \mathbb{R}^n$, we define the **linear span**, or the 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**,
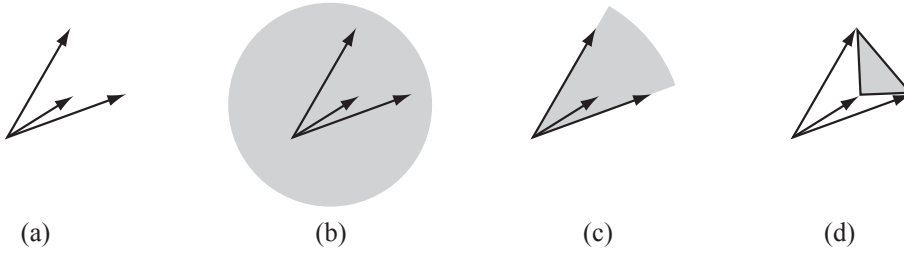
(a)   (b)   (c)   (d)

Figure 12.12: (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 combination is the polygon and its interior.

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 $\text{conv}(\mathcal{A}) \subset \text{pos}(\mathcal{A}) \subset \text{span}(\mathcal{A})$ (see Figure 12.12). 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.

3. From any set $\mathcal{A}$ of vectors positively spanning $\mathbb{R}^n$, a subset of no more than $2n$ vectors can be chosen that also positively spans $\mathbb{R}^n$.

The first fact is implicit in our use of n coordinates to represent $\mathbb{R}^n$. 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 $a_{n+1} = -\sum_{i=1}^{n} a_i$, we see that this set of $n + 1$ vectors positively spans $\mathbb{R}^n$. Fact 3 is stated without proof, and all three facts are illustrated in Figure 12.13.

These definitions and facts are useful in determining the number of contacts needed to immobilize an object by form closure.

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

$$\mathcal{F}_i^T \mathcal{V} \geq 0.$$

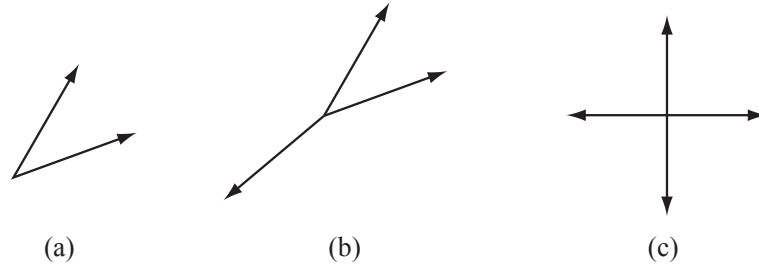(a)   (b)   (c)

Figure 12.13: (a) Two vectors spanning the plane but not positively spanning the plane. (b) Three vectors positively spanning the plane. (c) Four vectors positively spanning the plane from which no three vectors can be chosen which also positively span the plane.

Form closure holds if the only velocity $\mathcal{V}$ satisfying the constraints is the zero velocity. For j contacts, this condition is equivalent to

$$\text{pos}(\{\mathcal{F}_1, \ldots, \mathcal{F}_j\}) = \mathbb{R}^6$$

for parts in three dimensions. Therefore, by Fact 2 above, at least $6 + 1 = 7$ contacts are needed for first-order form closure of spatial parts. For planar parts, the condition is

$$\text{pos}(\{\mathcal{F}_1, \ldots, \mathcal{F}_j\}) = \mathbb{R}^3,$$

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 contacts are needed for first-order form closure. For a spatial part, at least seven 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.

Apart from these exceptional objects, we can directly use Fact 3 to show that no more than six contacts are needed for planar form closure and no more than twelve contacts for spatial form closure. With more detailed analysis, it is possible to reduce the numbers further (to four and seven contacts for planar and spatial cases, respectively), but we do not go into details here.

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 object. Our first-order analysis indicates that the parts in Figures 12.14(b) and 12.14(c) can each rotate about their centers in the three-finger grasps, but in
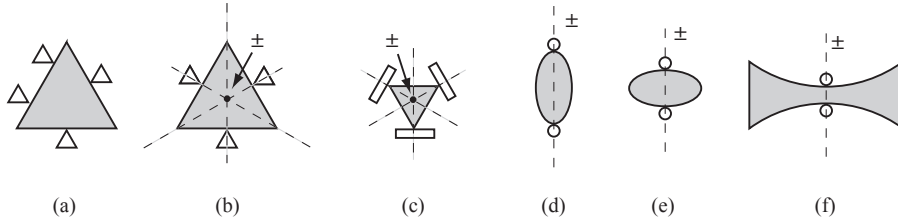
Figure 12.14: (a) Four fingers yielding planar form closure. The first-order analysis treats (b) and (c) identically, saying the triangle can rotate about its center in each case. A second-order analysis shows this is not possible for (b). The grasps in (d), (e), and (f) are identical by a first-order 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).

fact this is not possible for the part in Figure 12.14(b)—a second-order analysis would tell us that this part is actually immobilized. Finally, the first-order analysis tells us that the two-finger grasps in Figures 12.14(d)-(f) are identical, but in fact the part in Figure 12.14(f) is immobilized by only two fingers due to curvature effects.

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 $F = [\, \mathcal{F}_1 \mid \mathcal{F}_2 \mid \ldots \mid \mathcal{F}_j \,] \in \mathbb{R}^{n \times j}$ be a matrix whose columns are formed by the j contact wrenches. For spatial parts, $n = 6$, and for planar parts, $n = 3$ with $\mathcal{F}_i = [m_{iz}, f_{ix}, f_{iy}]^T$. The contacts yield form closure if there exists a vector of weights $k \in \mathbb{R}^j, k \geq 0$ such that $Fk = \mathcal{F}_{\text{ext}}$ for all $\mathcal{F}_{\text{ext}} \in \mathbb{R}^n$.

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 above 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:

$$\text{find } k \tag{12.14}$$
$$\text{minimizing } \mathbf{1}^T k$$
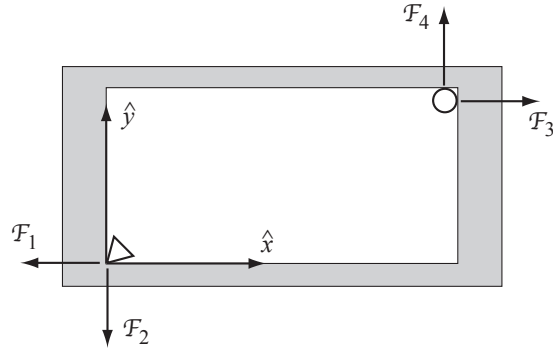$$\text{such that } Fk = 0$$
$$k \geq 1,$$

Figure 12.15: Two fingers grasping the interior of an object.

where $\mathbf{1}$ 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 $\mathbf{1}^T k$ is not strictly 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 $F = [\, \mathcal{F}_1 \mid \mathcal{F}_2 \mid \mathcal{F}_3 \mid \mathcal{F}_4 \,]$ 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 obviously requires a metric measuring the quality of a grasp. A **grasp metric** takes the set of contacts $\{\mathcal{F}_i\}$ and returns a single value $\mathrm{Qual}(\{\mathcal{F}_i\})$, where $\mathrm{Qual}(\{\mathcal{F}_i\}) < 0$ indicates that the grasp is not form closure, and larger positive values indicate better grasps.
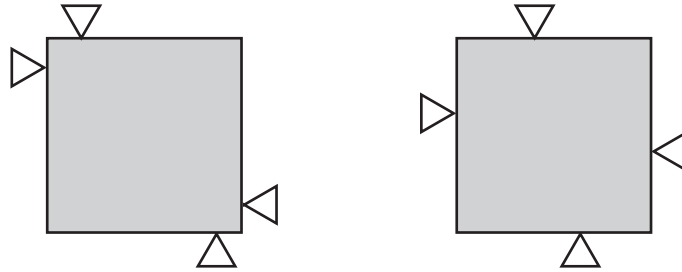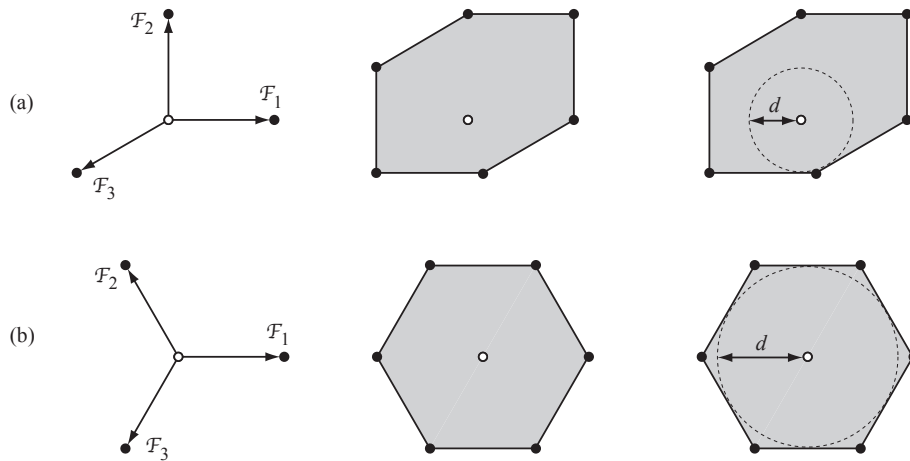
Figure 12.16: Both grasps are form closure, but which is better?

Figure 12.17: (a) A set of three wrenches in a two-dimensional wrench space, the set of all wrenches that can be applied by the contacts, and the radius d of the largest ball of wrenches centered at the origin that fits inside the wrench polygon. (b) A different set of three wrenches yielding a larger inscribed ball.

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 $f_{i,\max} > 0$, which may be set to 1. Then the total set of contact wrenches that can be applied by the j contacts is given by

$$CF = \left\{ \sum_{i=1}^{j} f_i \mathcal{F}_i \mid f_i \in [0, f_{i,\max}] \right\}. \tag{12.15}$$

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.

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 distur-

bance wrenches the part can be expected to experience. A more common choice is to set $\text{Qual}(\{\mathcal{F}_i\})$ 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 contat moments m to forces m/r. To address (2), typically the origin is 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 $\text{Qual}(\{\mathcal{F}_i\})$ (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 is likely to 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. A typical 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., $\partial \text{Qual}(p)/\partial p$, where p is a vector representing the grasp locations.

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 $f_\text{t} \leq \mu f_\text{n}$, where μ 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 $f_\text{t} = \mu f_\text{n}$, and the friction force opposes the motion. The friction force is independent of the speed of sliding.

Often two friction coefficients are defined, a static friction coefficient μ_s and a kinetic (or sliding) friction coefficient μ_k, where $\mu_\text{s} \geq \mu_\text{k}$. 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 dependences on factors such as the speed of sliding and the duration of static contact before sliding (for example, see Chapter 11.2.1). All of these are aggregate models of complex microscopic
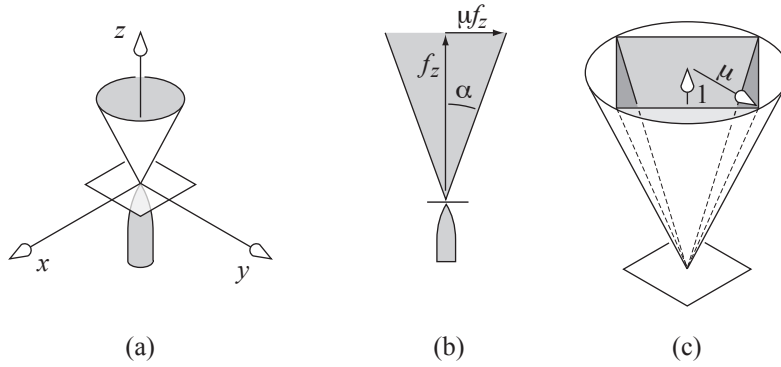
(a)   (b)   (c)

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 μ and the friction angle $\alpha = \tan^{-1} \mu$. (c) An inscribed polyhedral convex cone approximation to the circular friction cone.

behavior. For simplicity, we use the simplest Coulomb friction model with a single friction coefficient μ. 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 $+z$ direction, the set of forces that can be transmitted through the contact satisfies

$$\sqrt{f_x^2 + f_y^2} \leq \mu f_z, \ f_z \geq 0. \tag{12.16}$$

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. Correspondingly, the equal and opposite force the plane applies to the finger is inside the negative of the cone. 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 finger slips to the right, the force it applies lies on the right edge of the friction cone, with a magnitude determined by the normal force. If the contact is rolling, the force may be anywhere inside the cone.

To allow linear formulations of contact mechanics problems, it is often convenient to represent the convex circular cone by a polyhedral convex cone. Figure 12.18(c) shows an inscribed four-sided pyramidal approximation of the friction cone, defined by the nonnegative linear combinations 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 underapproximates the friction forces available, while a circumscribed cone overapproximates 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 underapproximate the friction forces available.
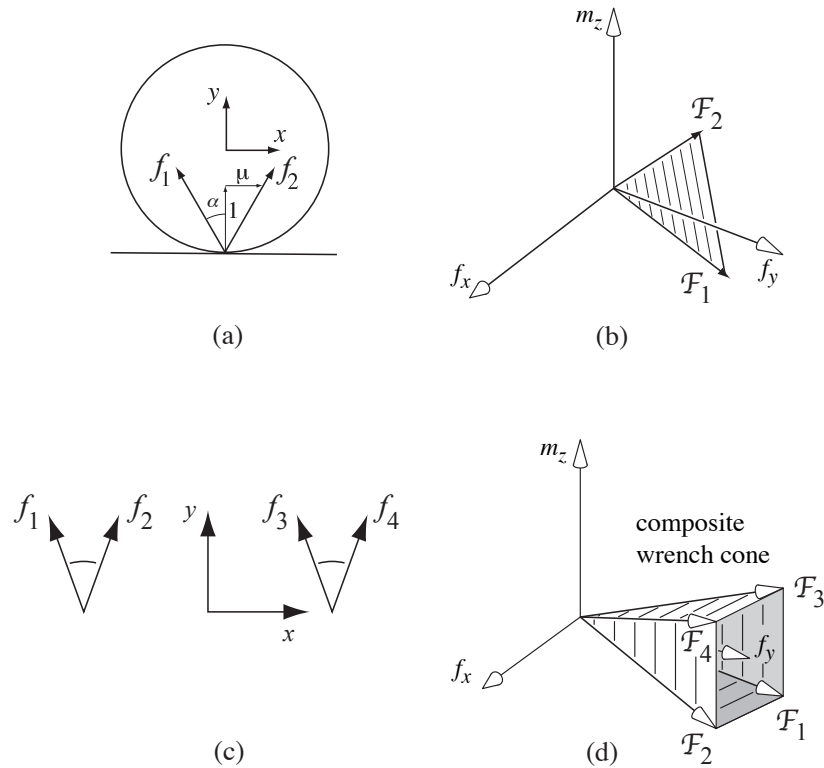
Figure 12.19: (a) A planar friction cone with friction coefficient μ and corresponding friction angle $\alpha = \tan^{-1}\mu$. (b) The corresponding wrench cone. (c) Two friction cones. (d) The corresponding composite wrench cone.

For planar problems, no approximation is necessary; a friction cone is exactly represented by the nonnegative linear combinations 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 $\mathcal{F} = ([p]f, f)$, 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 are all nonnegative linear combinations of basis vectors along these edges. If $\mathcal{F}_1$ and $\mathcal{F}_2$ are basis vectors for these wrench cone edges, we write the wrench cone as $\mathcal{WC} = \mathrm{pos}(\{\mathcal{F}_1, \mathcal{F}_2\})$.

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 nonnegative linear com-

bination of all the individual wrench cones $\mathcal{WC}_i$,

$$\mathcal{WC} = \mathrm{pos}(\{\mathcal{WC}_i\}) = \left\{ \sum_i k_i \mathcal{F}_i \mid \mathcal{F}_i \in \mathcal{WC}_i, k_i \geq 0 \right\}.$$

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 have been 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 $\mathcal{F}_{\mathrm{ext}}$ specified by the controller must lie within the composite wrench cone corresponding to those contacts. Because these force-controlled contacts choose a nonzero wrench from this wrench cone, the set of wrenches that can act on the part, including other non-force-controlled contacts, may no longer be a homogeneous cone rooted at the origin. This is analogous to the case of velocity-controlled contacts in Section 12.1.3, where moving constraints result in feasible part twists which are not necessarily a cone rooted at the origin.

12.2.2 Planar Graphical Methods

Just as homogeneous twist cones for planar problems can be represented as convex signed CoR regions in the plane, homogeneous wrench cones for planar problems can be represented as convex signed regions in the plane. One method for doing so is called **moment labeling**. Given a collection of lines of force in the plane (e.g., the edges of friction cones from a set of point contacts), the composite wrench cone can be represented by labeling each point in the plane with either a '+' if all wrenches in the cone make nonnegative moment about that point, a '−' if all make nonpositive moment about that point, a '±' if all make zero moment about that point, and a blank label if there exist wrenches making positive moment and wrenches making negative moment about that point.

The idea is best illustrated by an example. In Figure 12.20(a), a single line of force $\mathcal{F}_1$ 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 ±. In Figure 12.20(b), another line of force 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 line of force is added in Figure 12.20(c). The result is a single region labeled +. A nonnegative linear combination of the three lines of force can create any line of force in the plane that passes around this region in a counterclockwise sense; no other forces can be created. This representation is equivalent to a homogeneous convex wrench cone representation of $\mathrm{pos}(\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\})$.
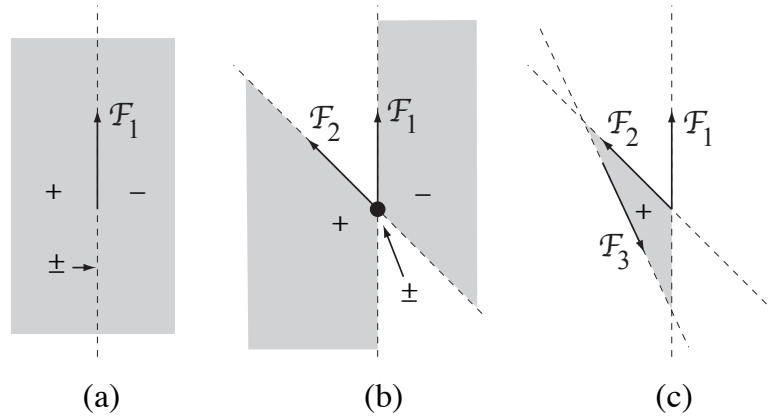
Figure 12.20: (a) Representing a line of force by moment labels. (b) Representing the nonnegative linear combinations of two lines of force by moment labels. (c) Nonnegative linear combinations of three lines of force.

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 $\mathcal{F}_{\text{ext}}$ on the object can be balanced by contact forces.

We can derive a simple linear test for force closure which is exact for planar cases and approximate for spatial cases. Let $\mathcal{F}_i, i = 1 \ldots j$ 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 (see Figure 12.18(d)), depending on the polyhedral approximation chosen. The columns of the $n \times j$ matrix F are the $\mathcal{F}_i$, where $n = 3$ for planar problems and $n = 6$ for spatial problems. As with the test for form closure in Section 12.1.7.2, the part is in force closure if there is a vector of weights $k \in \mathbb{R}^j, k \geq 0$ such that $Fk = \mathcal{F}_{\text{ext}}$ for all $\mathcal{F}_{\text{ext}} \in \mathbb{R}^n$. 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 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 12.21).
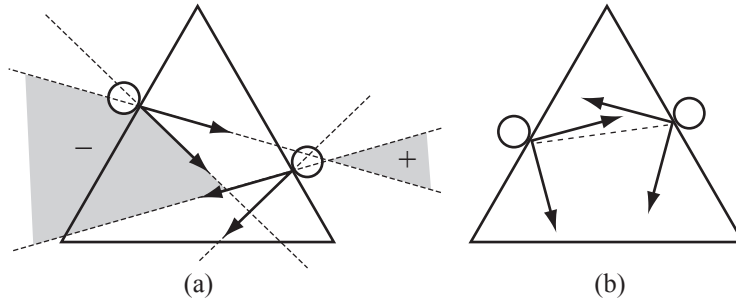
Figure 12.21: An equilateral triangle can be force-closure grasped by two fingers on edges of the triangle if $\mu \geq \tan 30° \approx 0.577$. (a) This grasp with $\mu = 0.25$ is not force closure, as indicated by the consistently labeled moment-labeling region. (b) This grasp is force closure with $\mu = 1$.

It is important to note that force closure simply means that forces at the contacts are capable of generating any wrench. It does not necessarily mean that the object will not move, however. For the example of Figure 12.21(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 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. [16] reduces the force closure analysis of spatial frictional grasps into a planar force closure problem. Referring to Figure 12.22, suppose a rigid body is constrained by three point contacts with friction. If the three contact points happened 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 12.23):

- 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.*
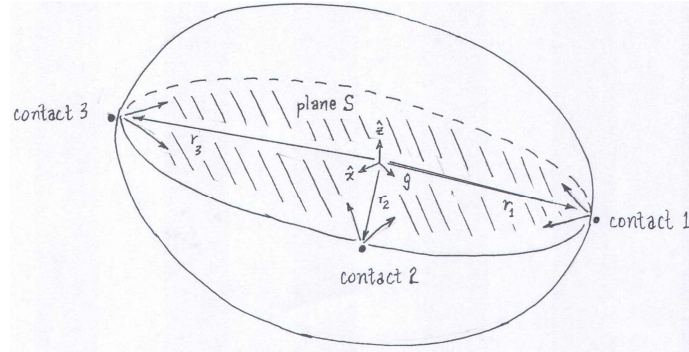
Figure 12.22: A spatial rigid body restrained by three point contacts with friction.
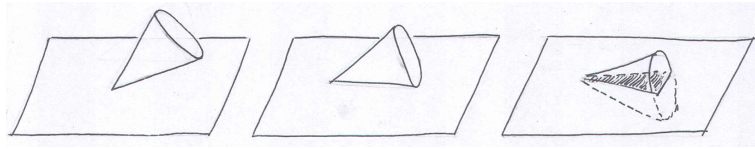

Figure 12.23: Three possibilities for the intersection between a friction cone and a plane.

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 12.22). Denoting the contact force at i by $f_i \in \mathbb{R}^3$, the contact spatial force $F_i \in \mathbb{R}^6$ is then of the form

$$\mathcal{F}_i = \left[\begin{array}{c} f_i \\ m_i \end{array} \right], \tag{12.17}$$

where each $m_i = r_i \times f_i$, $i = 1, 2, 3$. Denote the arbitrary external spatial force

$\mathcal{F}_{\text{ext}} \in \mathbb{R}^6$ by

$$\mathcal{F}_{\text{ext}} = \begin{bmatrix} f_{\text{ext}} \\ m_{\text{ext}} \end{bmatrix} \in \mathbb{R}^6. \tag{12.18}$$

Force closure then requires that there exist contact spatial forces $\mathcal{F}_i$, $i = 1, 2, 3$, each lying inside its respective friction cone, such that for any external spatial disturbance force F_{ext}, the following equality is satisfied:

$$\mathcal{F}_1 + \mathcal{F}_2 + \mathcal{F}_3 + \mathcal{F}_{\text{ext}} = 0, \tag{12.19}$$

or equivalently,

$$f_1 + f_2 + f_3 + f_{\text{ext}} = 0 \tag{12.20}$$
$$(r_1 \times f_1) + (r_2 \times f_2) + (r_3 \times f_3) + m_{\text{ext}} = 0, \tag{12.21}$$

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

$$f_{1S} + f_{2S} + f_{3S} = -f_{\text{ext},S} \tag{12.22}$$
$$(r_1 \times f_{1S}) + (r_2 \times f_{2S}) + (r_3 \times f_{3S}) = -m_{\text{ext},S} \tag{12.23}$$
$$f_{1N} + f_{2N} + f_{3N} = -f_{\text{ext},N} \tag{12.24}$$
$$(r_1 \times f_{1N}) + (r_2 \times f_{2N}) + (r_3 \times f_{3N}) = -m_{\text{ext},N}. \tag{12.25}$$

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 hypothesis 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

$$\eta_1 + \eta_2 + \eta_3 = 0 \tag{12.26}$$
$$(r_1 \times \eta_1) + (r_2 \times \eta_2) + (r_3 \times \eta_3) = 0. \tag{12.27}$$

(To see why such η_i 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 η_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

$$
\begin{aligned}
f_1 &= f_{1N}^* + f_{1S} + \eta_1 & (12.28) \\
f_2 &= f_{2N}^* + f_{2S} + \eta_2 & (12.29) \\
f_3 &= f_{3N}^* + f_{3S} + \eta_3 & (12.30)
\end{aligned}
$$

all lie inside their respective friction cone. This completes the proof of the sufficiency condition. $\qquad\square$

12.2.3.2 Measuring the Quality of a Force-Closure Grasp

Friction forces are not always repeatable. 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 μ, 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 opposite 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 parts that are too large to be grasped and too heavy to be lifted, or to send objects outside the workspace of the end-effector.

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

$$\mathcal{F}_{\text{ext}} + \sum k_i \mathcal{F}_i = G\dot{\mathcal{V}} - [\text{ad}_\mathcal{V}]^T \mathcal{P}, \quad k_i \geq 0, \ \mathcal{F}_i \in \mathcal{WC}_i, \qquad (12.31)$$

where $\mathcal{V}$ is the part's twist, $\mathcal{P}$ is its momentum, $\mathcal{F}_{\text{ext}}$ is the external wrench acting on the part due to gravity, etc., $\mathcal{WC}_i$ is the set of possible wrenches acting on the object due to contact i, and $\sum k_i \mathcal{F}_i$ is the wrench due to the contacts. Now, given a set of motion- or force-controlled contacts acting on the part, and the initial state of the system, the general method for solving for the motion of the part is the following:

(i) Enumerate the set of possible contact modes consisting of the contact labels R, S, and B at each contact.

(ii) For each contact mode, determine if there exists a contact wrench $\sum k_i \mathcal{F}_i$ that is consistent with the contact mode and Coulomb's law, and an acceleration $\dot{\mathcal{V}}$ 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.

For a given contact mode, the set of kinematically consistent accelerations is described by a set of linear constraints on $\dot{\mathcal{V}}$. Therefore, for planar problems and spatial problems with approximate polyhedral friction cones, step (ii) above is a linear constraint satisfaction problem (LCSP).

This kind of "case analysis" may sound unusual; we are not simply solving a set of equations. It also seems to leave 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. For many problems the method described above will yield a unique motion.

A special case of the dynamics (12.31) are **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

$$\mathcal{F}_{\text{ext}} + \sum k_i \mathcal{F}_i = 0, \quad k_i \geq 0, \ \mathcal{F}_i \in \mathcal{WC}_i. \tag{12.32}$$

The same case-enumeration method is used to solve such problems, except now we often solve for the part's twist $\mathcal{V}$ instead of its acceleration $\dot{\mathcal{V}}$. The part's twist is relevant for determining whether individual contacts are sliding or not, which places constraints on the contact forces that can be applied.

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 12.24(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 $\text{pos}(\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3\})$, as shown using moment labeling in Figure 12.24(a).

Our first question is whether the stationary fingers can keep the block at rest. To do so, the fingers must provide a wrench $\mathcal{F} = (m_z, f_x, f_y) = (0, 0, \mathfrak{mg})$ to balance $\mathcal{F}_{\text{ext}} = (0, 0, -\mathfrak{mg})$ due to gravity. As shown in Figure 12.24(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 $2\mathfrak{g}$. In this case, the contact mode RR requires that the block also accelerate to the left at $2\mathfrak{g}$. The wrench needed to cause this acceleration is $(0, -2\mathfrak{mg}, 0)$. Therefore the total wrench that the fingers must apply to the block is $(0, -2\mathfrak{mg}, 0) - \mathcal{F}_{\text{ext}} = (0, -2\mathfrak{mg}, \mathfrak{mg})$. As shown in Figures 12.24(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 $2\mathfrak{g}$.

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, 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
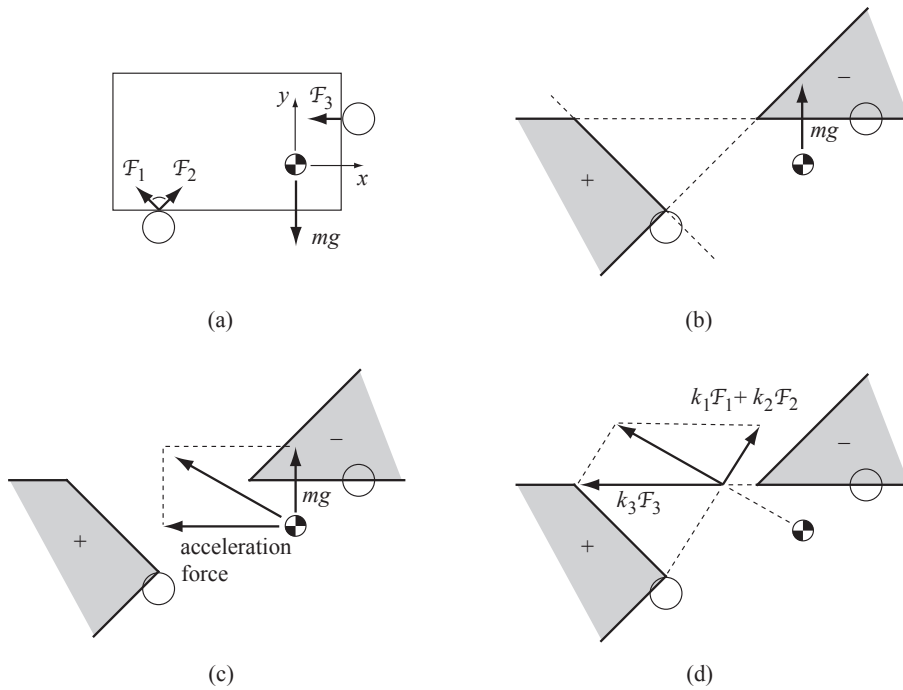
(a)

(b)

(c)

(d)

Figure 12.24: (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 does not make nonpositive moment with respect to all 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 $k_1 \mathcal{F}_1 + k_2 \mathcal{F}_2$ and $k_3 \mathcal{F}_3$ provided by the fingers.

wrenches

$$\mathcal{F}_1 = \frac{1}{\sqrt{2}}(-4, -1, 1)$$

$$\mathcal{F}_2 = \frac{1}{\sqrt{2}}(-2, 1, 1)$$

$$\mathcal{F}_3 = (1, -1, 0).$$

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

$$k_1 \mathcal{F}_1 + k_2 \mathcal{F}_2 + k_3 \mathcal{F}_3 + (0, 0, -\mathfrak{m}\mathfrak{g}) = (0, \mathfrak{m}a_x, 0). \tag{12.33}$$

This yields three equations in the three unknowns, k_1, k_2, k_3. Solving, we get **blah blah**. For the k_i to be nonnegative, we need **some range for** a_x. 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 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 12.25 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 motion. Figure 12.25 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 (Sl) 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 SlR 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 over 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 SlSr 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, requiring large accelerations of the stick to maintain the 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 SlSr contact mode, but instead switches rapidly between the SlR and RSr contact modes. If there is a large difference between the static and kinetic friction coefficients, the time between switches is increased.

Example 12.8. Stability of an assembly.

Consider the arch in Figure 12.26. Is it stable under gravity?

For a problem like this, graphical planar methods are difficult to use. Instead we test algebraically for consistency of the contact mode with all contacts
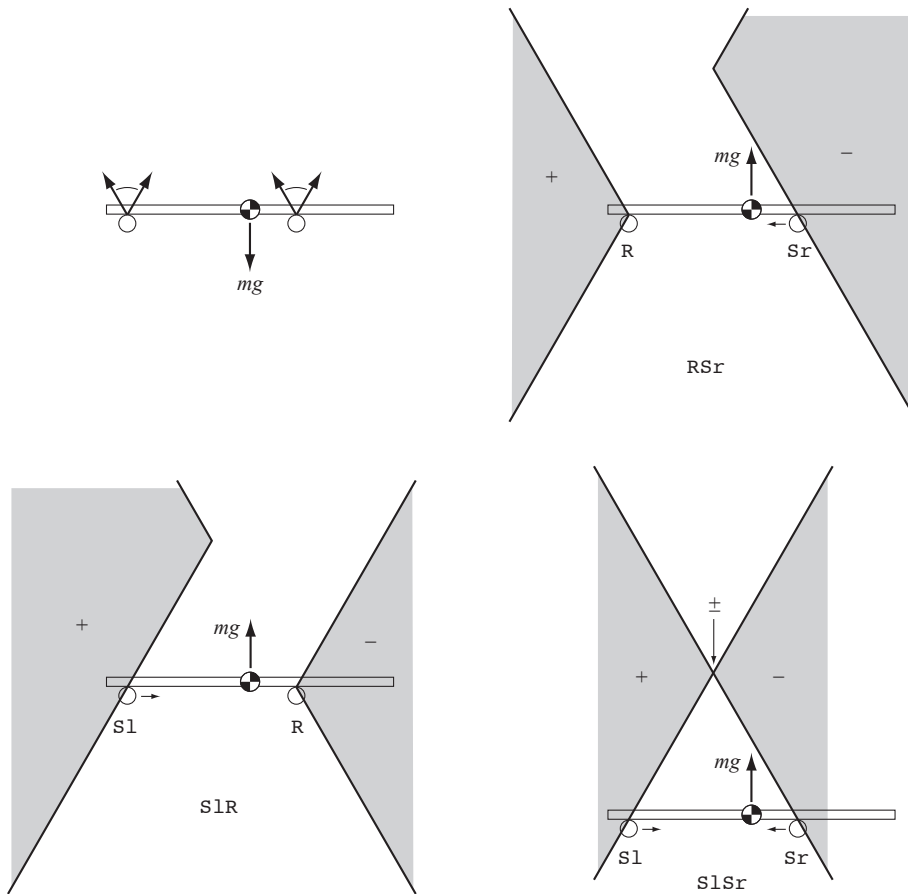
Figure 12.25: Top left: Two frictional fingers supporting a meter stick in gravity. The other three panels show the moment labels for the `RSr`, `SlR`, and `SlSr` contact modes. Only the `SlR` contact mode yields force balance.
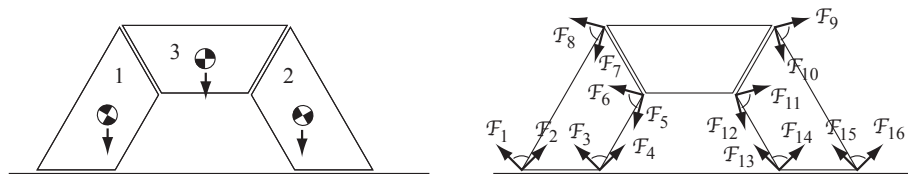

Figure 12.26: Left: An arch in gravity. Right: The friction cones at the contacts of the stones 1 and 2.

labeled `R`. The friction cones are shown in Figure 12.26. 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
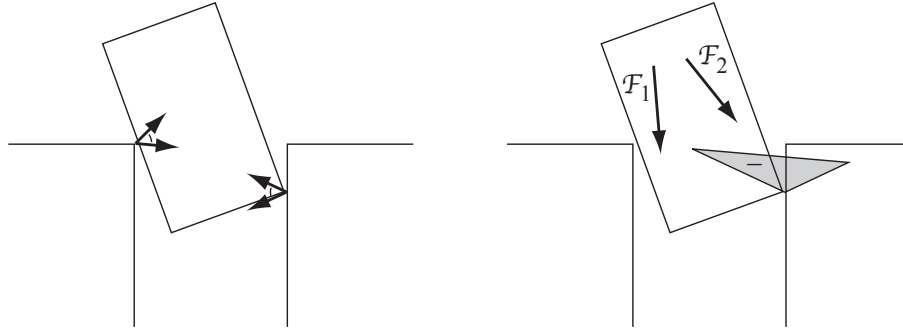
Figure 12.27: Left: A peg in two-point contact with a hole. Right: The wrench $\mathcal{F}_1$ may cause the peg to jam, while the wrench $\mathcal{F}_2$ continues to push the peg into the hole.

equations, three for each body:

$$\sum_{i=1}^{8} k_i \mathcal{F}_i + \mathcal{F}_{\text{ext1}} = 0$$

$$\sum_{i=9}^{16} k_i \mathcal{F}_i + \mathcal{F}_{\text{ext2}} = 0$$

$$-\sum_{i=5}^{12} k_i \mathcal{F}_i + \mathcal{F}_{\text{ext3}} = 0.$$

The last set of equations comes from the fact that forces 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.

Example 12.9. Peg insertion.
Figure 12.27 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 $\mathcal{F}_1$ to the peg, it may **jam**—the hole may generate contact forces that balance $\mathcal{F}_1$. Therefore the peg may get stuck in this position. If the force controller applies the wrench $\mathcal{F}_2$, 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. The peg is said to be **wedged**.
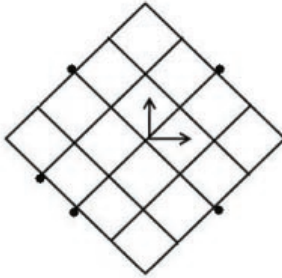
Figure 12.28: A 4×4 planar square restrained by five frictionless point contacts.

12.4 Exercises

1. Prove that the impenetrability constraint (12.4) is equivalent to the constraint (12.7).

2. note that any contact with the same normal provides the same interpenetrability constraints, no matter where it acts along the line of action

3. write SFC and PCwF contact velocity constraints in a coordinate frame at the contact, with z-axis aligned with normal.

4. follow up on the examples at the end of the chapter. show that other contact modes are not possible for the meter stick trick.

5. A finger toppling a block.

6. Vibratory transport.

7. pushing? 3 support points, CoRs of pushing and relative motion

8. holding a rod by over-under contacts. table supported by four legs. parallel jaw gripper. ambiguity and inconsistency example. waiter's problem. 3d example. LCSP solution. limit surface. two leg planar table.
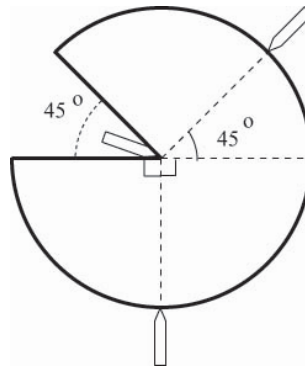
Figure 12.29: A planar disk restrained by three frictionless point contacts.

9. (a) The planar grasp of Figure 12.28 consists of five frictionless point contacts. The square is of size 4×4. 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.

10. Assume all contacts shown in Figure 12.29 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?
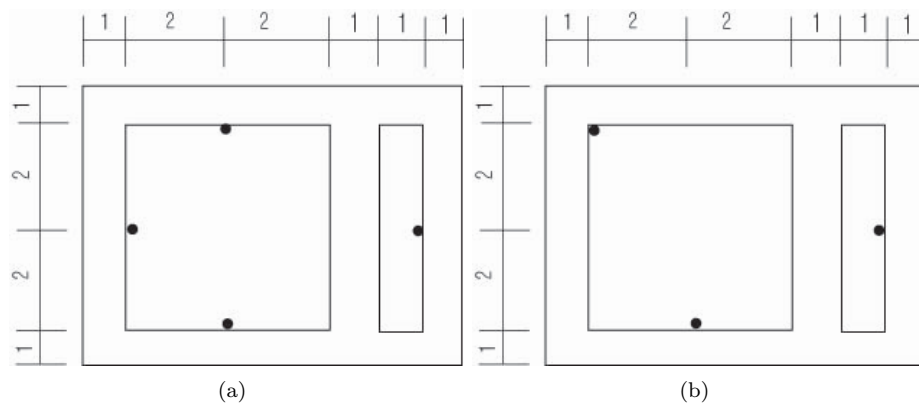

Figure 12.30: A planar rigid body with two square holes.

11. (a) In Figure 12.30-(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.

(b) In Figure 12.30-(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.
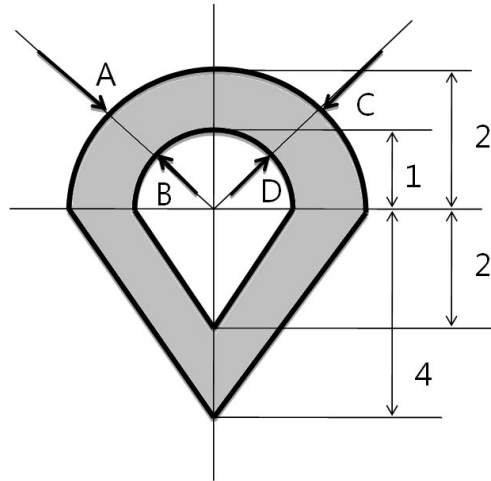

Figure 12.31: A planar wedge-like object with a hole.

12. The planar object of Figure 12.31 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.

13. (a) Determine whether the grasp of Figure 12.32-(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 grasp.
(b) Now place two frictionless point contacts at the corners as shown in Figure 12.32-(b). Determine if this grasp is force closure.
(c) In the grasp of Figure 12.32-(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.

14. Determine whether the grasp of Figure 12.33 is force closure. Assume all contacts are point contacts with a friction coefficient $\mu = 1$.

15. (a) In the planar triangle of Figure 12.34-(a), contacts A, B, and C are
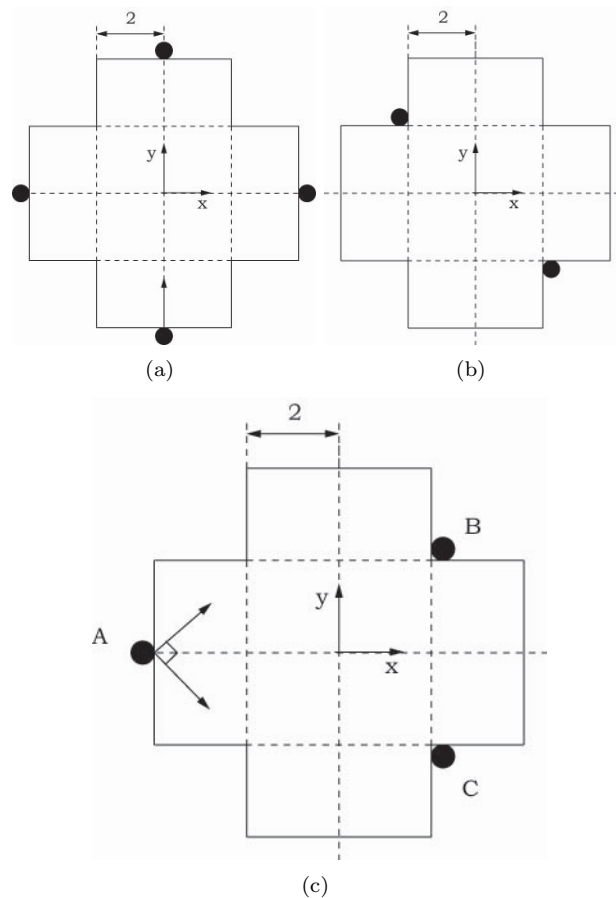
Figure 12.32: A planar rigid body restrained by point contacts.

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 12.34-(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 12.34-(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 12.35. 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 12.35. Is Bill correct? Justify your answer with a derivation of the appropriate force closure condition.
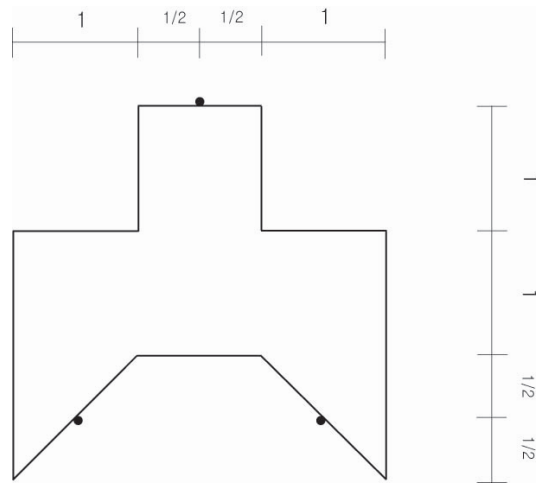
Figure 12.33: A planar rigid body restrained by three point contacts with friction.
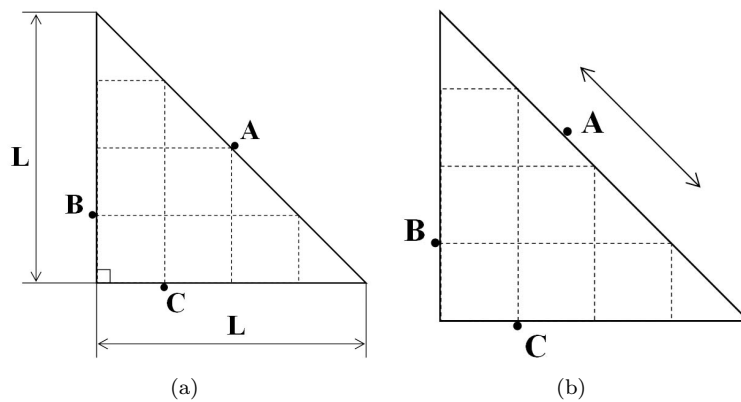
Figure 12.34: A planar triangle constrained by three point contacts.

16. Consider the L-shaped planar object of Figure 12.36.
(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 force closure.
(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.
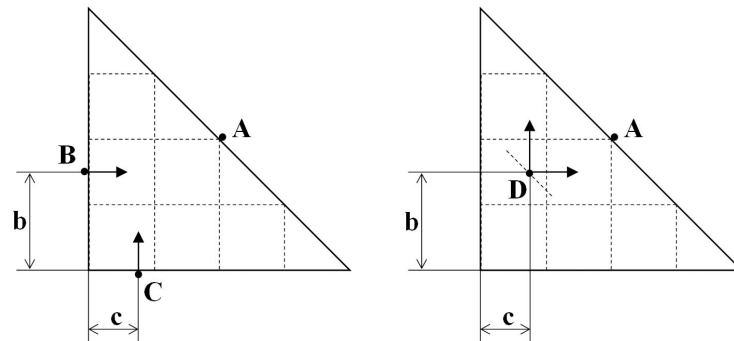
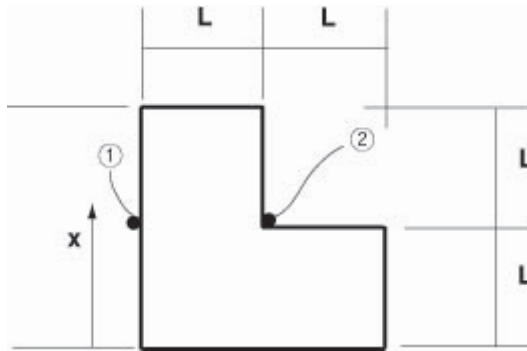Figure 12.35: A planar triangle constrained by three point contacts.

Figure 12.36: An L-shaped planar object restrained by two point contacts with friction.

17. A square is restrained by three point contacts as shown in Figure 12.37: contact f_1 is a point contact with friction coefficient μ, 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 μ such that grasp is force closure.

18. In the grasp of Figure 12.38, 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.

19. A single point contact with friction coefficient $\mu = 1$ is applied to the left side of the square doughnut as shown in Figure 12.39. 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.
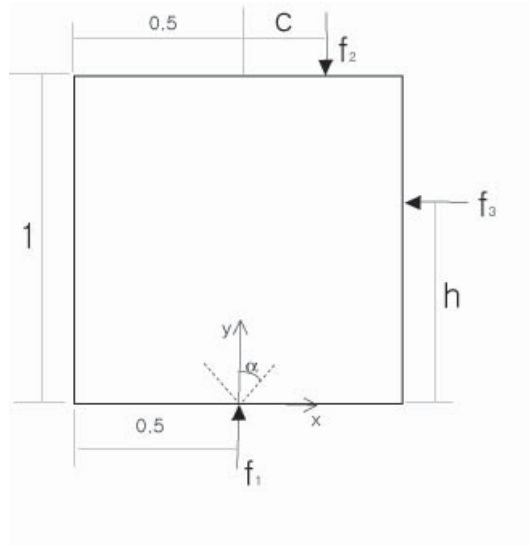
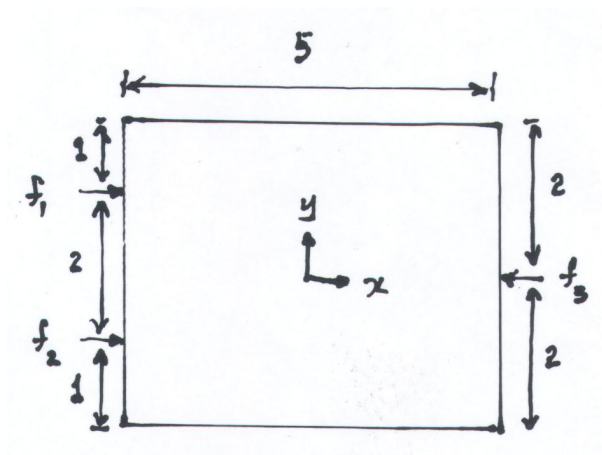Figure 12.37: A square restrained by three point contacts.

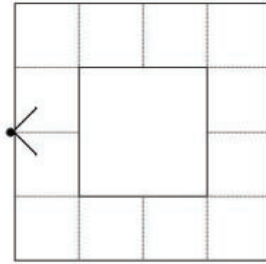Figure 12.38: A rectangle restrained by three point contacts.

Figure 12.39: A square doughnut.

Appendix A

Summary of Useful Formulas

Chapter 2

- Grübler's formula for the DOF of 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:

$$\text{dof} = m(N - 1 - J) + \sum_{i=1}^{J} f_i$$

- Pfaffian velocity constraints take the form $A(\theta)\dot{\theta} = 0$.

Chapter 3

- An element R of $SO(3)$ satisfies $R^T R = I$ and $\det R = 1$, and therefore $R^{-1} = R^T$. Also $R_{ab} = R_{ba}^{-1}$ and $R_{ab}v_b = v_a$, while $R_{ab}v_a = v'_a$, which is the original vector v_a rotated by the rotation that takes $\{a\}$ to $\{b\}$.

- Let R_1 be the orientation achieved when rotating about a fixed axis ω ($\|\omega\| = 1$) a distance θ from an initial orientation $R = I$. Then $R_1 R_a$ is the orientation achieved by rotating $\{a\}$ about ω intrepreted as a space frame angular velocity, while $R_a R_1$ is the orientation achieved by rotating $\{a\}$ about ω interpreted as a body frame angular velocity.

- $\dot{x}(t) = Ax(t)$ has solution $x(t) = e^{At}x_0$. A can be viewed as a constant angular velocity or rigid-body twist (angular and linear velocity), in the body or space frame.

- For $\omega \in \mathbb{R}^3$, we have $\omega \times x = [\omega]x$, where

$$[\omega] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}.$$

- Rodrigues' formula, integrating a rotation with an angular velocity ω with $\|\omega\| = 1$ for time (or angle) θ: $e^{[\omega]\theta} = I + \sin\theta[\omega] + (1 - \cos\theta)[\omega]^2$. ω and θ together are called the axis-angle representation of an orientation of an element of $SO(3)$, and $\omega\theta \in \mathbb{R}^3$ is the exponential coordinate representation of an an element of $SO(3)$.

- The matrix log of R, in the general case, is given by: $\theta = \cos^{-1}((\text{trace}(R) - 1)/2) \in [0, \pi)$ and $[\omega] = (R - R^T)/(2\sin\theta)$. If $R = I$, then $\theta = 0$. If $\text{trace}(R) = -1$, then $\theta = \pi$. We write $\log(R) = [\omega]\theta$.

- A rigid-body configuration is written $T \in SE(3)$ with the form

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}$$

where $R \in SO(3)$ and $p \in \mathbb{R}^3$. Also,

$$T^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix},$$

$T_{ab}T_{bc} = T_{ac}$, $T_{ab}^{-1} = T_{ba}$, and $x_a = T_{ab}x_b$.

- A spatial velocity, or twist, is written $\mathcal{V} = (\omega, v) \in \mathbb{R}^6$, which we can also write in the matrix form

$$[\mathcal{S}] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4}.$$

- Consider a screw motion following the twist $\mathcal{S}' = (\omega', v')$ for duration 1. We can write this as $\mathcal{S}' = \mathcal{S}\theta$, where $\mathcal{S} = (\omega, v)$ and θ is the "distance" of motion along the screw axis $\mathcal{S}$. If $\omega' \neq 0$, then $\mathcal{S} = \mathcal{S}'/\|\omega\|$ and θ is the net rotation about the screw axis. If $\omega' = 0$, then $\mathcal{S} = \mathcal{S}'/\|v'\|$ and θ is the translation along the axis.

  The net displacement obtained by motion along the screw axis $[\mathcal{S}]$ by θ from the identity element of $SE(3)$, in either the body or space frame (since they are initially aligned with each other), is

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

  For $\omega = 0$, i.e., $\mathcal{S} = (0, v)$, then

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

  For $T = e^{[\mathcal{S}]\theta}$, $\mathcal{S}\theta \in \mathbb{R}^6$ are the exponential coordinates of T.

- The matrix log of $T = (R, p)$, for the general case, is given by

$$\theta = \cos^{-1}\left(\frac{\text{trace}(R) - 1}{2}\right) \in [0, \pi)$$

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

$$v = \left(\frac{1}{\theta}I - \frac{1}{2}[\omega] + \left(\frac{1}{\theta} - \frac{1}{2}\cot\frac{\theta}{2}\right)[\omega]^2\right)p. \qquad (A.1)$$

If $R = I$, then $\omega = 0$, $v = p/\|p\|$, and $\theta = \|p\|$. If $\text{trace}(R) = -1$, then $\theta = \pi$, and $[\omega] = \log R$. We write $\log(T) = [\mathcal{S}]\theta$.

- The quantity $T' = e^{[\mathcal{S}]\theta}T$ is the new configuration after T undergoes a screw motion $\mathcal{S}\theta$ in the space frame. The quantity $T' = Te^{[\mathcal{S}]\theta}$ is the new configuration after T undergoes a screw motion $\mathcal{S}\theta$ in the body frame.

- Given frames $\{s\}$ and $\{b\}$, a particular spatial velocity can be represented in these frames as $\mathcal{V}_s$ or $\mathcal{V}_b$, and these are related by the Adjoint transformation

$$\mathcal{V}_s = \text{Ad}_{T_{sb}}(\mathcal{V}_b),$$

where $\text{Ad}_{T_{sb}}(\mathcal{V}_b) = [\text{Ad}_{T_{sb}}]\mathcal{V}_b$ and

$$[\text{Ad}_T] = \left[\begin{array}{cc} R & 0 \\ [p]R & R \end{array}\right] \in \mathbb{R}^{6 \times 6}.$$

The expression $\mathcal{V}_s = \text{Ad}_{T_{sb}}(\mathcal{V}_b)$ is equivalent to $[\mathcal{V}_s] = T_{sb}[\mathcal{V}_b]T_{sb}^{-1}$.

- $\text{Ad}_T^{-1} = \text{Ad}_{T^{-1}}$ and $\text{Ad}_{T_1}(\text{Ad}_{T_2}(\mathcal{V})) = \text{Ad}_{T_1 T_2}(\mathcal{V})$.

- $\dot{T}T^{-1} = [\mathcal{V}_s]$, the spatial velocity (twist) in space coordinates, and $T^{-1}\dot{T} = [\mathcal{V}_b]$, the spatial velocity (twist) in body coordinates.

- A wrench in space coordinates is written $\mathcal{F}_s = (m_s, f_s) \in \mathbb{R}^6$ and a wrench in body coordinates is written $\mathcal{F}_b = (m_b, f_b)$. $\mathcal{F}_b$ and $\mathcal{F}_s$ are related by

$$\mathcal{F}_b = \text{Ad}_{T_{sb}}^T(\mathcal{F}_s) = [\text{Ad}_{T_{sb}}]^T\mathcal{F}_s$$
$$\mathcal{F}_s = \text{Ad}_{T_{bs}}^T(\mathcal{F}_b) = [\text{Ad}_{T_{bs}}]^T\mathcal{F}_b,$$

derived from the relationship between space and body velocities and the fact that power, $\mathcal{F}_s^T\mathcal{V}_s$ and $\mathcal{F}_b^T\mathcal{V}_b$, must be the same in both frames.

Chapter 4

- The product of exponentials formula for a serial chain manipulator is

$$\text{space frame:} \quad T = e^{[\mathcal{S}_1]\theta_1}\ldots e^{[\mathcal{S}_n]\theta_n}M$$
$$\text{body frame:} \quad T = Me^{[\mathcal{B}_1]\theta_1}\ldots e^{[\mathcal{B}_n]\theta_n}$$

where M is the frame of the end-effector in the space frame when the manipulator is at its home position, $[\mathcal{S}_i]$ is the velocity of the space frame in space coordinates when joint i rotates (or translates) at unit speed while all other joints are fixed, and $[\mathcal{B}_i]$ is the velocity of the body frame in body coordinates when all other joints are fixed.

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 $\dot{x} = \frac{\partial f}{\partial \theta} \dot{\theta} = J(\theta)\dot{\theta}$, where $J(\theta)$ is the manipulator Jacobian.

- In spatial velocities, the relation is $\mathcal{V}_* = J_*(\theta)\dot{\theta}$, where $*$ is either s (space Jacobian) or b (body Jacobian). The columns J_{si} of the space Jacobian are

$$J_{si}(\theta) = \mathrm{Ad}_{e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_{i-1}]\theta_{i-1}}}(\mathcal{S}_i)$$

and the columns J_{bi} of the body Jacobian are

$$J_{bi}(\theta) = \mathrm{Ad}_{e^{-[\mathcal{B}_n]\theta_n} \cdots e^{-[\mathcal{B}_{i+1}]\theta_{i+1}}}(\mathcal{B}_i).$$

As expected, the space motion caused by $\mathcal{S}_i$ is only altered by the configurations of joints inboard from joint i (between the joint and the space frame), while the body motion caused by $\mathcal{B}_i$ is only altered by the configurations of joints outboard from joint i (between the joint and the body frame).

The two Jacobians are related by

$$J_b(\theta) = \mathrm{Ad}_{T_{bs}}(\theta)(J_s(\theta)) \quad , \quad J_s(\theta) = \mathrm{Ad}_{T_{sb}}(\theta)(J_b(\theta)).$$

- Generalized forces at the joints τ are related to wrenches expressed in the space or end-effector body frame by

$$\tau = J_*^T(\theta)\mathcal{F}_*,$$

where $*$ is s (space frame) or b (body frame).

- Singularities occur at manipulator configurations where the rank of the Jacobian drops below its maximum value. Often we only care about end-effector motions in a particular subspace, and a singularity is defined when the set of feasible motions in that subspace loses rank.

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 γ is the interior angle opposite side c. This formula is often useful to solve inverse kinematics problems.

- Many inverse problems can be stated as finding θ such that $x = f(\theta)$, where x and θ are vectors. Such problems can have many or no solutions, and often admit no closed-form solution. Newton-Raphson iterative numerical root-finding attempts to find a "close by" solution to an initial guess. Starting with an initial guess $\theta(0)$, the iteration is defined by

$$\theta(i+1) = \theta(i) + \left(\frac{\partial f}{\partial \theta}\big|_{\theta(i)}\right)^{-1}(x - f(\theta(i))),$$

where the expression $x - f(\theta(i))$ is the vector from the current guess to the desired value.

- For inverse kinematics with a desired end-effector configuration $X \in SE(3)$, the direction from the current configuration $T(\theta(i))$ to X, expressed in the end-effector body frame, is given by $[\mathcal{S}] = \log T^{-1}X$. The Newton-Raphson iteration beomes

$$\theta(i+1) = \theta(i) + \underbrace{(J_b(\theta(i))^{-1}\mathcal{S}}_{\Delta\theta_i}.$$

- If the Jacobian is not square (i.e., the number of joints n differs from the degrees of freedom of the end-effector m), then $J_b^{-1}(\theta)$ does not exist. The right generalized inverse $J_b^{-\text{right}} = J_b^T(J_bJ_b^T)^{-1}$ can be used for $n > m$ and the left generalized inverse $J_b^{-\text{left}} = (J_b^TJ_b)^{-1}J_b^T$ can be used for $n < m$.

Chapter 8

- The Lagrangian is the kinetic minus the potential energy, $\mathcal{L}(q,\dot{q}) = K(q,\dot{q}) - U(q)$.

- The Euler-Lagrange equations are

$$\tau = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q}.$$

- The kinetic energy of a mechanical system is $K(q,\dot{q}) = \frac{1}{2}\dot{q}^TM(q)\dot{q}$, where M is the mass or inertia matrix.

- The equations of motion of a manipulator can be written

$$\tau = M(\theta)\ddot{\theta} + c(\theta,\dot{\theta}) + \frac{\partial U}{\partial \theta} \tag{A.2}$$
$$= M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) \tag{A.3}$$
$$= M(\theta)\ddot{\theta} + \dot{\theta}^T\Gamma(\theta)\dot{\theta} + g(\theta) \tag{A.4}$$

where $g(\theta)$ are the potential terms (typically due to gravity) and $c(\theta,\dot{\theta})$ is the vector of quadratic velocity terms (Coriolis and centrifugal terms).

These quadratic terms are sometimes written as a Coriolis matrix $C(\theta, \dot{\theta})$ multiplied by the linear velocity $\dot{\theta}$, or more insightfully as a quadratic form in terms of the three-dimensional matrix of Christoffel symbols of the mass matrix.

- The Lie bracket of twists $\mathcal{V}_1$ and $\mathcal{V}_2$, i.e., the derivative of $\mathcal{V}_2$ in the direction of $\mathcal{V}_1$, is written

$$[\mathcal{V}_1, \mathcal{V}_2] = \mathrm{ad}_{\mathcal{V}_1}(\mathcal{V}_2) = [\mathrm{ad}_{\mathcal{V}_1}]\mathcal{V}_2,$$

where

$$[\mathrm{ad}_{\mathcal{V}}] = \begin{bmatrix} [\omega] & 0 \\ [v] & [\omega] \end{bmatrix} \in \mathbb{R}^{6 \times 6}.$$

- The body-frame 6×6 mass matrix of a rigid-body is

$$\mathcal{G}_b = \begin{bmatrix} \mathcal{I}_b & 0 \\ 0 & \mathfrak{m}I \end{bmatrix},$$

where $\mathcal{I}_b$ is the inertia matrix in the body frame and $\mathfrak{m}$ is the mass.

- The equations of motion of a rigid body, expressed in the body frame, are

$$\mathcal{F}_b = \mathcal{G}_b \dot{\mathcal{V}}_b - [\mathrm{ad}_{\mathcal{V}_b}]^T \mathcal{G}_b \mathcal{V}_b.$$

Appendix B

Other Representations of Rotations

B.1 Euler Angles

As we established earlier, a rotation matrix can be parametrized by three independent coordinates. Here we introduce one popular three-parameter representation of rotations, the **ZYX Euler angles**. One way to visualize these angles is through the wrist mechanism shown in Figure B.1. The ZYX Euler angles (α, β, γ) 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. We now consider the relative frame orientations R_{01}, R_{12}, and R_{23}. First, it can be seen that R_{01} depends only on the angle α: rotating about the $\hat{z}$-axis of frame $\{0\}$ by an angle α (a positive rotation about an axis is taken by aligning the thumb of the right hand along the axis, and rotating in the direction of the fingers curling about the axis), it can be seen that

$$R_{01} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathrm{Rot}(\hat{z}, \alpha). \qquad (\mathrm{B.1})$$

The notation $\mathrm{Rot}(\hat{z}, \alpha)$ describes a rotation about the $\hat{z}$-axis by angle α. Similarly, R_{12} depends only on β, and is given by

$$R_{12} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} = \mathrm{Rot}(\hat{y}, \beta), \qquad (\mathrm{B.2})$$
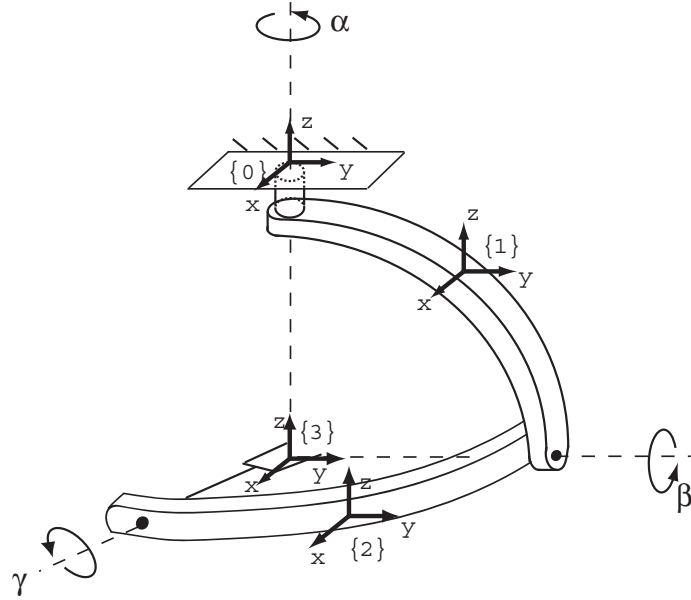
375

Figure B.1: Wrist mechanism illustrating the ZYX Euler angles.

where the notation $\mathrm{Rot}(\hat{\mathbf{y}}, \beta)$ describes a rotation about the $\hat{\mathbf{y}}$-axis by angle β. Finally, R_{23} depends only on γ, and is given by

$$R_{23} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} = \mathrm{Rot}(\hat{\mathbf{x}}, \gamma), \tag{B.3}$$

where the notation $\mathrm{Rot}(\hat{\mathbf{x}}, \gamma)$ describes a rotation about the $\hat{\mathbf{x}}$-axis by angle γ. $R_{03} = R_{01} R_{12} R_{23}$ is therefore given by

$$R_{03} = \begin{bmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix}, \tag{B.4}$$

where s_α is shorthand for $\sin\alpha$, c_α for $\cos\alpha$, etc.

We now ask the following question: given an arbitrary rotation matrix R, does there exist (α, β, γ) such that Equation (B.4) is satisfied? If the answer is yes, then the wrist mechanism of Figure B.1 can reach any orientation. This is indeed the case, and we prove this fact constructively as follows. Let r_{ij} be the ij-th element of R. Then from Equation (B.4) we know that $r_{11}^2 + r_{21}^2 = \cos^2\beta$; as long as $\cos\beta \neq 0$, or equivalently $\beta \neq \pm 90°$, we have

$$\beta = \tan^{-1}\left(\frac{\sin\beta}{\cos\beta}\right) = \tan^{-1}\left(\frac{-r_{31}}{\pm\sqrt{r_{11}^2 + r_{21}^2}}\right).$$
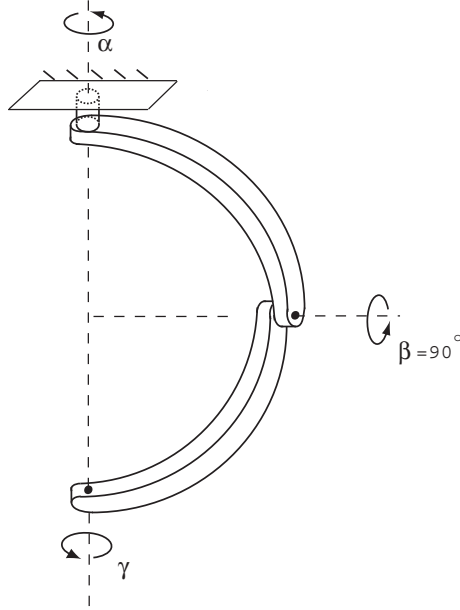
Figure B.2: Configuration corresponding to $\beta = 90°$ for ZYX Euler angles.

We now define the atan2 function, which is a two-argument function implemented in a variety of computer languages for computing the arctangent. Specifically, the function $\text{atan2}(y, x)$ evaluates $\tan^{-1}(y/x)$ by taking into account the signs of x and y. For example, $\text{atan2}(1, 1) = \pi/4$, while $\text{atan2}(-1, -1) = -3\pi/4$. Using atan2, the possible values for β can be expressed as

$$\beta = \text{atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right)$$

and

$$\beta = \text{atan2}\left(-r_{31}, -\sqrt{r_{11}^2 + r_{21}^2}\right).$$

In the first case β lies in the range $[-90°, 90°]$, while in the second case β lies in the range $[90°, 270°]$. Assuming the β obtained above is not $\pm 90°$, α and γ can then be determined from the following relations:

$$
\begin{aligned}
\alpha &= \text{atan2}(r_{21}, r_{11}) \\
\gamma &= \text{atan2}(r_{32}, r_{33})
\end{aligned}
$$

In the event that $\beta = \pm 90°$, there exists a one-parameter family of solutions for α and γ. This is most easily seen from Figure B.2. If $\beta = 90°$, then α and γ represent rotations (in the opposite direction) about the same vertical axis. Hence, if $(\alpha, \beta, \gamma) = (\bar{\alpha}, 90°, \bar{\gamma})$ is a solution for a given rotation R, then any triple $(\bar{\alpha}', 90°, \bar{\gamma}')$ where $\bar{\alpha}' - \bar{\gamma}' = \bar{\alpha} - \bar{\gamma}$ is also a solution.

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

$$R = \begin{bmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix}, \tag{B.5}$$

where s_α is shorthand for $\sin \alpha$, c_α for $\cos \alpha$, etc. Denote by r_{ij} the ij-th entry of R.

(i) If $r_{31} \neq \pm 1$, set

$$\beta = \text{atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right) \tag{B.6}$$

$$\alpha = \text{atan2}(r_{21}, r_{11}) \tag{B.7}$$

$$\gamma = \text{atan2}(r_{32}, r_{33}), \tag{B.8}$$

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 α and γ exists. One possible solution is $\alpha = 0$ and $\gamma = \text{atan2}(r_{12}, r_{22})$.

(iii) If $r_{31} = -1$, then $\beta = -\pi/2$, and a one-parameter family of solutions for α and γ exists. One possible solution is $\alpha = 0$ and $\gamma = -\text{atan2}(r_{12}, r_{22})$.

From the earlier wrist mechanism illustration of the ZYX Euler angles it should be evident that the choice of zero position for β 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.2; this would then lead to another three-parameter representation (α, β, γ) for $SO(3)$. Figure B.2 is precisely the definition of the **ZYZ Euler angles**. The resulting rotation matrix can be obtained via the following sequence of rotations:

$$\begin{aligned} R(\alpha, \beta, \gamma) &= \text{Rot}(\hat{z}, \alpha) \cdot \text{Rot}(\hat{y}, \beta) \cdot \text{Rot}(\hat{z}, \gamma) \\ &= \begin{bmatrix} c_\alpha & -s_\alpha & 0 \\ s_\alpha & c_\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{bmatrix} \begin{bmatrix} c_\gamma & -s_\gamma & 0 \\ s_\gamma & c_\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} c_\alpha c_\beta c_\gamma - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ s_\alpha c_\beta c_\gamma + c_\alpha s_\gamma & -s_\alpha c_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{bmatrix}. \end{aligned} \tag{B.9}$$

Just as before, we can show that for every rotation $R \in SO(3)$, there exists a triple (α, β, γ) that satisfies $R = R(\alpha, \beta, \gamma)$ for $R(\alpha, \beta, \gamma)$ as given in Equation (B.9). (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
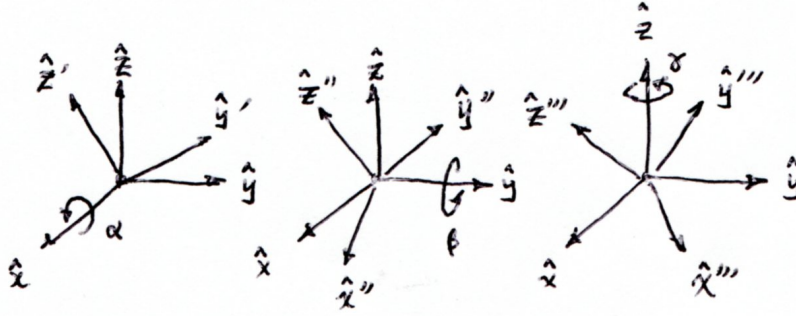
Figure B.3: Illustration of XYZ roll-pitch-yaw angles.

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

$$\text{Rot}(\text{axis1}, \alpha) \cdot \text{Rot}(\text{axis2}, \beta) \cdot \text{Rot}(\text{axis3}, \gamma), \tag{B.10}$$

where axis1 is orthogonal to axis2, and axis2 is orthogonal to axis3, can serve as a valid three-parameter representation for $SO(3)$. Later in this chapter we see how to express a rotation about an arbitrary axis that is not a unit axis of the reference frame.

B.2 Roll-Pitch-Yaw Angles

Earlier in the chapter we asserted that a rotation matrix can also be used to describe a transformation of a rigid body from one orientation to another. Here we use this prescriptive viewpoint to derive another three-parameter representation for rotation matrices, the **roll-pitch-yaw angles**. Referring to Figure B.3, given a frame in the identity configuration (that is, $R = I$), we first rotate this frame by an angle γ about the $\hat{x}$-axis of the fixed frame, followed by an angle β about the $\hat{y}$-axis of the fixed frame, and finally by an angle α about the $\hat{z}$-axis of the fixed frame.

Let us derive the explicit form of a vector $v \in \mathbb{R}^3$ (expressed as a column vector using fixed frame coordinates) that is rotated about the fixed frame $\hat{x}$-axis by an angle γ. The rotated vector, denoted v', will be

$$v' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} v. \tag{B.11}$$

If v' is now rotated about the fixed frame $\hat{y}$-axis by an angle β, then the rotated vector v'' can be expressed in fixed frame coordinates as

$$v'' = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} v'. \tag{B.12}$$

Finally, rotating v'' about the fixed frame $\hat{z}$-axis by an angle α yields the vector

$$
v''' = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} v''. \tag{B.13}
$$

If we now take v to successively be the three unit axes of the reference frame in the identity orientation $R = I$, then after applying the above sequence of rotations to the three axes of the reference frame, its final orientation will be

$$
\begin{aligned}
R(\alpha,\beta,\gamma) &= \mathrm{Rot}(\hat{z},\alpha)\mathrm{Rot}(\hat{y},\beta)\cdot\mathrm{Rot}(\hat{x},\gamma)\cdot\cdot I \\
&= \begin{bmatrix} c_\alpha & -s_\alpha & 0 \\ s_\alpha & c_\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\beta & 0 & s_\beta \\ 0 & 1 & 0 \\ -s_\beta & 0 & c_\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\gamma & -s_\gamma \\ 0 & s_\gamma & c_\gamma \end{bmatrix} \cdot I \\
&= \begin{bmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix}. \tag{B.14}
\end{aligned}
$$

This product of three rotations is exactly the same as that for the ZYX Euler angles given in (B.5). 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 (α,β,γ) of Equation (B.14).

In fact, for any sequence of rotations of the form (B.10) in which consecutive axes are orthogonal, a similar descriptive-prescriptive interpretation exists for the corresponding Euler angle formula. Euler angle formulas can be defined in a number of ways depending on the choice and order of the rotation axes, but their common features are:

- The angles represent three successive rotations taken about the axes of either the body frame or the fixed frame.

- The first axis must be orthogonal to the second axis, and the second axis must be orthogonal to the third axis.

- The angle of rotation for the first and third rotations ranges in value over a 2π interval, while that of the second rotation ranges in value over an interval of length π.

B.3 Unit Quaternions

One disadvantage with 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 θ. The **unit quaternions** are an alternative representation of rotations that alleviates some of these numerical difficulties, 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 exponential coordinate representation $R = e^{[\omega]\theta}$, where as usual $\|\omega\| = 1$ and $\theta \in [0, \pi]$. The unit quaternion representation of R is constructed as follows. Define $q \in \mathbb{R}^4$ according to

$$q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos \frac{\theta}{2} \\ \omega \sin \frac{\theta}{2} \end{bmatrix} \in \mathbb{R}^4. \tag{B.15}$$

q as defined clearly satifies $\|q\| = 1$. 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 $1 + 2\cos\theta = \mathrm{tr}\, R$, 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:

$$q_0 = \frac{1}{2}\sqrt{1 + r_{11} + r_{22} + r_{33}} \tag{B.16}$$

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \frac{1}{4q_0} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - 2_{12} \end{bmatrix}. \tag{B.17}$$

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,

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_0q_2 + q_1q_3) \\ 2(q_0q_3 + q_1q_2) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_0q_1 + q_2q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}. \tag{B.18}$$

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 2×2 complex matrices:

$$Q = \begin{bmatrix} q_0 + iq_1 & q_2 + ip_3 \\ -q_2 + iq_3 & q_0 - iq_1 \end{bmatrix}, \quad P = \begin{bmatrix} p_0 + ip_1 & p_2 + ip_3 \\ -p_2 + ip_3 & p_0 - ip_1 \end{bmatrix}, \quad \text{(B.19)}$$

where i denotes the imaginary unit. Now take the product $N = QP$, where the entries of N are written

$$N = \begin{bmatrix} n_0 + in_1 & n_2 + in_3 \\ -n_2 + in_3 & n_0 - in_1 \end{bmatrix}. \quad \text{(B.20)}$$

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:

$$\begin{bmatrix} n_0 \\ n_1 \\ n_2 \\ n_3 \end{bmatrix} = \begin{bmatrix} q_0 p_0 - q_1 p_1 - q_2 p_2 - q_3 p_3 \\ q_0 p_1 + p_0 q_1 + q_2 p_3 - q_3 p_2 \\ q_0 p_2 + p_0 q_2 - q_1 p_3 + q_3 p_1 \\ q_0 p_3 + p_0 q_3 + q_1 p_2 - q_2 p_1 \end{bmatrix}. \quad \text{(B.21)}$$

Verification of this formula is left as an exercise at the end of this chapter.

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 labelled 0, and the end-effector frame is attached to some point on link n. Reference frames attached to the links are also correspondingly labelled from $\{0\}$ (the fixed frame) to $\{n\}$ (the end-effector frame). The joint variable corresponding to the i-th joint is denoted θ_i. The forward kinematics of the n-link open chain can then be expressed as

$$T_{0n}(\theta_1, \ldots, \theta_n) = T_{01}(\theta_1)T_{12}(\theta_2) \cdots T_{n-1,n}(\theta_n), \tag{C.1}$$

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, for open chains each $T_{i-1,i}$ can be obtained in a straightforward fashion.
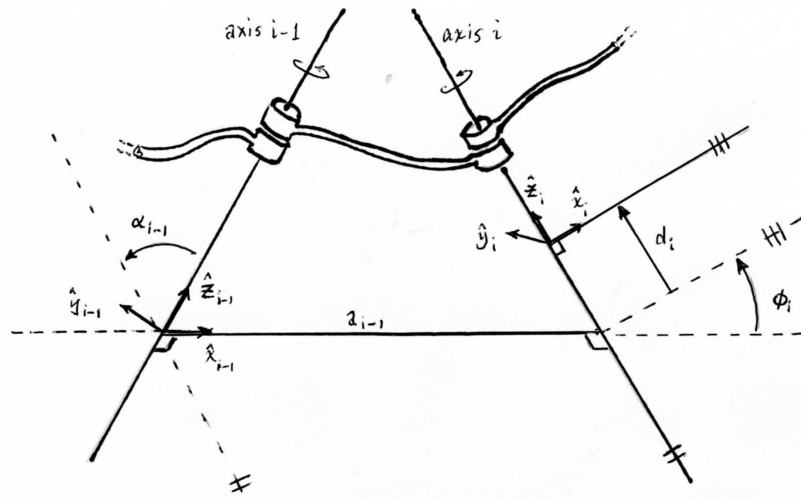
Figure C.1: Illustration of Denavit-Hartenberg parameters.

C.1.1   Assigning Link Frames

Rather than attaching reference frames to each link in some 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 $\hat{z}$-axis coincides with joint axis i, and $\hat{z}_{i-1}$ coincides with joint axis $i-1$. The direction of each link frame $\hat{z}$-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 link frames i and $i-1$ chosen according to this convention.

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** α_{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** ϕ_i is the angle from $\hat{x}_{i-1}$ to $\hat{x}_i$, measured about the $\hat{z}_i$-axis in the right-hand sense.

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 α_{i-1}, and offset parameters d_i are all constant, while the joint angle parameters ϕ_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 α_{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-parameter family of mutual perpendicular lines is said to exist). Again, it is important to detail precisely how the link frames are assigned. 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).

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 parameter d_i now becomes the joint variable (see
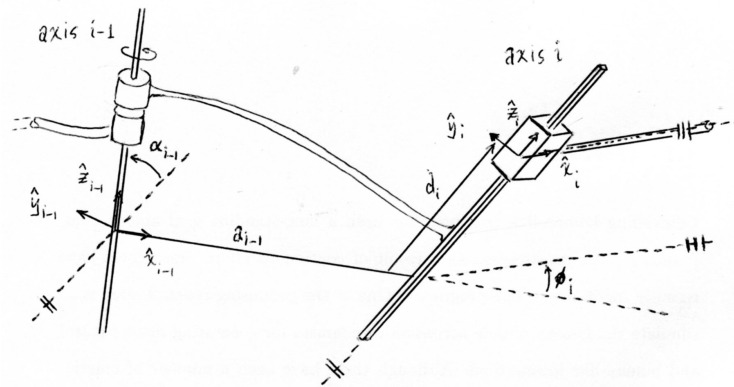
Figure C.2: Link frame assignment convention for prismatic joints. Joint $i - 1$ is a revolute joint, while joint i is a prismatic joint.

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 typically 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.1.2   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,
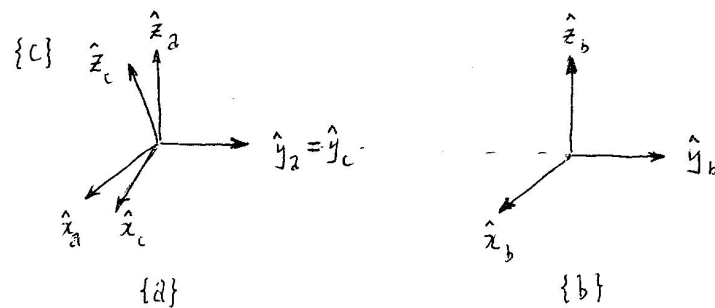
Figure C.3: An example of three frames $\{a\}$, $\{b\}$, and $\{c\}$, in which the transformations T_{ab} and T_{ac} cannot be described by any set of Denavit-Hartenberg parameters.

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 (α, a, d, θ) 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 transformations 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 [7].

We end this section with a reminder that there are alternative conventions 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

$$
\begin{aligned}
T_{i-1,i} &= \mathrm{Rot}(\hat{x}, \alpha_{i-1}) \cdot \mathrm{Trans}(\hat{x}, a_{i-1}) \cdot \mathrm{Trans}(\hat{z}, d_i) \cdot \mathrm{Rot}(\hat{z}, \phi_i) \\
&= \begin{bmatrix}
\cos\phi_i & -\sin\phi_i & 0 & a_{i-1} \\
\sin\phi_i \cos\alpha_{i-1} & \cos\phi_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} & -d_i \sin\alpha_{i-1} \\
\sin\phi_i \sin\alpha_{i-1} & \cos\phi_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i \cos\alpha_{i-1} \\
0 & 0 & 0 & 1
\end{bmatrix},
\end{aligned}
$$

where

$$
\mathrm{Rot}(\hat{x}, \alpha_{i-1}) = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & \cos\alpha_{i-1} & -\sin\alpha_{i-1} & 0 \\
0 & -\sin\alpha_{i-1} & \cos\alpha_{i-1} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \tag{C.2}
$$

$$
\mathrm{Trans}(\hat{x}, a_{i-1}) = \begin{bmatrix}
1 & 0 & 0 & a_{i-1} \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \tag{C.3}
$$

$$
\mathrm{Trans}(\hat{z}, d_i) = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & d_i \\
0 & 0 & 0 & 1
\end{bmatrix} \tag{C.4}
$$

$$
\mathrm{Rot}(\hat{z}, \phi_i) = \begin{bmatrix}
\cos\phi_{i-1} & -\sin\phi_{i-1} & 0 & 0 \\
-\sin\phi_{i-1} & \cos\phi_{i-1} & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}. \tag{C.5}
$$

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 α_{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 ϕ_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 $T_{i-1,i}$.
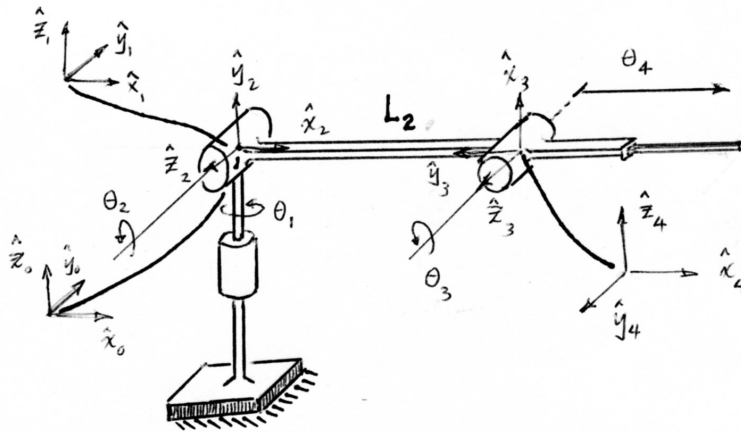
Figure C.4: An RRRP spatial open chain.

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:

i	α_{i-1}	a_{i-1}	d_i	ϕ_i
1	0	0	0	θ_1
2	$90°$	L_1	0	$\theta_2 - 90°$
3	$-90°$	L_2	0	θ_3

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 θ_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:
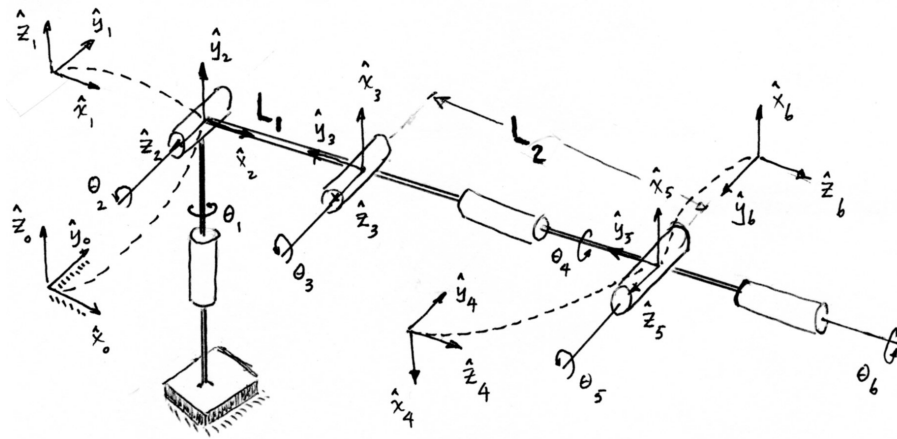
Figure C.5: A 6R spatial open chain.

i	α_{i-1}	a_{i-1}	d_i	ϕ_i
1	0	0	0	θ_1
2	90°	0	0	θ_2
3	0	L_2	0	$\theta_3 + 90°$
4	90°	0	θ_4	0

The four joint variables are $(\theta_1, \theta_2, \theta_3, \theta_4)$, where θ_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 *ZYZ* type wrist. The link frames are shown in the figure, and the corresponding Denavit-Hartenberg parameters are listed in the following table:

i	α_{i-1}	a_{i-1}	d_i	ϕ_i
1	0	0	0	θ_1
2	90°	0	0	θ_2
3	0	L_1	0	$\theta_3 + 90°$
4	90°	0	L_2	$\theta_4 + 180°$
5	90°	0	0	$\theta_5 + 180°$
6	90°	0	0	θ_6

C.1.5 Relation between the Product of Exponential and the Denavit-Hartenberg Representations

The product of exponentials formula can also 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

$$T_{i-1,i} = \text{Rot}(\hat{x}, \alpha_{i-1}) \cdot \text{Trans}(\hat{x}, a_{i-1}) \cdot \text{Trans}(\hat{z}, d_i) \cdot \text{Rot}(\hat{z}, \phi_i).$$

If joint i is revolute, the first three matrices can be regarded as constant, and ϕ_i becomes the revolute joint variable. Define $\theta_i = \phi_i$, and

$$M_i = \text{Rot}(\hat{x}, \alpha_{i-1}) \cdot \text{Trans}(\hat{x}, a_{i-1}) \cdot \text{Trans}(\hat{z}, d_i), \tag{C.6}$$

and write $\text{Rot}(\hat{z}, \theta_i)$ as the following matrix exponential:

$$\text{Rot}(\hat{z}, \theta_i) = e^{[\mathcal{A}_i]\theta_i}, \quad [\mathcal{A}_i] = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{C.7}$$

With the above definitions we can write $T_{i-1,i} = M_i e^{[\mathcal{A}_i]\theta_i}$.

If joint i is prismatic, then d_i becomes the joint variable, ϕ_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 $T_{i-1,i} = M_i e^{[\mathcal{A}_i]\theta_i}$, where $\theta_i = d_i$ and

$$M_i = \text{Rot}(\hat{x}, \alpha_{i-1})\text{Trans}(\hat{x}, a_{i-1})\text{Rot}(\hat{z}, \phi_i) \tag{C.8}$$

$$[\mathcal{A}_i] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{C.9}$$

Based on the above, for an n-link open chain containing both revolute and prismatic joints, the forward kinematics can be written

$$T_{0,n} = M_1 e^{[\mathcal{A}_1]\theta_1} M_2 e^{[\mathcal{A}_2]\theta_2} \cdots M_n e^{[\mathcal{A}_n]\theta_n} \tag{C.10}$$

where θ_i denotes joint variable i, and $[\mathcal{A}_i]$ 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 $Me^P M^{-1} = e^{MPM^{-1}}$, 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 $Me^P = e^{MPM^{-1}}M$. 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:

$$\begin{aligned} T_{0n} &= e^{M_1[\mathcal{A}_1]M_1^{-1}\theta_1}(M_1 M_2)e^{[\mathcal{A}_2]\theta_2} \cdots e^{[\mathcal{A}_n]\theta_n} \\ &= e^{M_1[\mathcal{A}_1]M_1^{-1}\theta_1} e^{(M_1 M_2)[\mathcal{A}_2](M_1 M_2)^{-1}\theta_2}(M_1 M_2 M_3)e^{[\mathcal{A}_3]\theta_3} \cdots e^{[\mathcal{A}_n]\theta_n} \\ &= e^{[\mathcal{S}_1]\theta_1} \cdots e^{[\mathcal{S}_n]\theta_n} M, \end{aligned} \tag{C.11}$$

where

$$[\mathcal{S}_i] \;=\; (M_1 \cdots M_{i-1})[\mathcal{A}_i](M_1 \cdots M_{i-1})^{-1}, \quad i = 1, \ldots, n \qquad \text{(C.12)}$$

$$M \;=\; M_1 M_2 \cdots M_n. \qquad \text{(C.13)}$$

We now re-examine the physical meaning of the $\mathcal{S}_i$ by recalling how a screw twist transforms under a change of reference frames. If $\mathcal{S}_a$ represents the screw twist for a given screw motion with respect to frame $\{a\}$, and $\mathcal{S}_b$ represents the screw twist for the same physical screw motion but this time with respect to frame $\{b\}$, then recall that $\mathcal{S}_a$ and $\mathcal{S}_b$ are related by

$$[\mathcal{S}_b] = T_{ba}[\mathcal{S}_a]T_{ba}^{-1}, \qquad \text{(C.14)}$$

or using the adjoint notation $\text{Ad}_{T_{ba}}$,

$$\mathcal{S}_b = \text{Ad}_{T_{ba}}(\mathcal{S}_a). \qquad \text{(C.15)}$$

Seen from the perspective of this transformation rule, Equation (C.13) suggests that $\mathcal{A}_i$ is the screw twist for joint axis i as seen from link frame $\{i\}$, while $\mathcal{S}_i$ is the screw twist for joint axis i as seen from the fixed frame $\{0\}$.

Bibliography

[1] J. Angeles, *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms.* Springer, 2006.

[2] R. S. Ball, *A Treatise on the Theory of Screws.* Cambridge University Press, 1998.

[3] O. Bottema and B. Roth, *Theoretical Kinematics.* Dover Publications, 1990.

[4] R. W. Brockett, "Robotic manipulators and the product of exponentials formula," *Proc. Int. Symp. Math. Theory of Networks and Systems*, Beer Sheba, Israel, 1983.

[5] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations.* Cambridge, MA: MIT Press, 2005.

[6] J. Craig, *Introduction to Robotics: Mechanics and Control.* Upper Saddle River, New Jersey: Prentice-Hall, 2004.

[7] J. Denavit and R.S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Trans. ASME J. Appl. Mech*, vol. 23, pp. 215-221, 1955.

[8] M. Do Carmo, *Differential Geometry of Curves and Surfaces.* Upper Saddle River, New Jersey: Prentice-Hall, 1976.

[9] A. G. Erdman and G. N. Sandor, *Advanced Mechanism Design: Analysis and Synthesis Vol. I and II.* Upper Saddle River, New Jersey: Prentice-Hall, 1996.

[10] R. Featherstone, *Rigid Body Dynamics Algorithms.* Upper Saddle River, New Jersey: Springer, 2008.

[11] D. T. Greenwood, *Advanced Dynamics.* Cambridge: Cambridge University Press, 2003.

[12] J. Hong, G. Lafferiere, B. Mishra, and X. Tan, "Fine manipulation with multifinger hands," *Proc. IEEE Int. Conf. Robotics Autom.*, pp. 1568-1573, 1990.

[13] O. Khatib, "A unified approach to motion and force control of robot manipulators: the operational space formulation," *IEEE Int. J. Robotics Autom.*, vol. 3, no. 1, pp. 43-53, 1987.

[14] S. M. LaValle, *Planning Algorithms*. Cambridge: Cambridge University Press, 2006.

[15] H.Y. Lee and C.G. Liang, "A new vector theory for the analysis of spatial mechanisms," *Mech. Machine Theory*, vol. 23, no. 3, pp. 209-217, 1988.

[16] J.-W. Li, H. Liu, and H.-G. Cai, "On computing three-finger force-closure graps of 2-D and 3-D objects," *IEEE Trans. Robotics Autom.*, vol. 19, no. 1, 2003, pp. 155-161.

[17] K. M. Lynch and M. T. Mason, "Pulling by pushing, slip with infinite friction, and perfectly rough surfaces," *Int. J. Robotics Res.*, vol. 14, no. 2, pp. 174-183, 1995.

[18] D. Manocha and J. Canny, "Real time inverse kinematics for general manipulators," *Proc. IEEE Int. Conf. Robotics Automation*, vol. 1, pp. 383-389, 1989.

[19] X. Markenscoff, L. Ni, and C. H. Papadimitriou, "The geometry of grasping," *Int. J. Robotics Res.*, vol. 9, no. 1, pp. 61-74, 1990.

[20] J. M. McCarthy, *Introduction to Theoretical Kinematics*. Cambridge: MIT Press, 1990.

[21] J. P. Merlet, *Parallel Robots*. Berlin: Springer, 2006.

[22] R. S. Millman and G. D. Parker, *Elements of Differential Geometry*. Upper Saddle River, New Jersey: Prentice-Hall, 1977.

[23] B. Mishra, J. T. Schwartz, and M. Sharir, "On the existence and synthesis of multifinger positive grips," *Algorithmica*, vol. 2, no. 4, pp. 541-558, 1987.

[24] M. T. Mason, *Mechanics of Robot Manipulation*. Cambridge: MIT Press, 2001.

[25] R. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, Florida: CRC Press, 1994.

[26] V.-D. Nguyen, "Constructing force-closure grasps," *Int. J. Robotics Research*, vol. 7, no. 3, pp. 3-16, 1988.

[27] F.C. Park, "Computational aspects of the product of exponentials formula for robot kinematics," *IEEE Trans. Automatic Control*, Vol. 39, No. 3, pp. 643-647, 1994.

[28] F.C. Park, J.E. Bobrow, and S.R. Ploen, "A Lie group formulation of robot dynamics," *International Journal of Robotics Research,* vol. 14, no. 6, pp. 609-618, 1995.

[29] F. C. Park, Jihyeon Choi, and S. R. Ploen, "Symbolic formulation of closed chain dynamics in independent coordinates," *Journal of Mechanism and Machine Theory*, vol. 34, no. 5, pp. 731-751, July 1999.

[30] F. C. Park and Jinwook Kim, "Manipulability of closed kinematic chains," *ASME J. Mechanical Design*, vol. 120, no. 4, pp. 542-548, 1998.

[31] F. C. Park and Jinwook Kim, "Singularity analysis of closed kinematic chains," *ASME J. Mechanical Design*, vol. 121, no. 1, pp. 32-38, 1999.

[32] F. C. Park and I. G. Kang, "Cubic spline algorithms for orientation interpolation," *Int. .J. Numerical Methods in Engineering*, vol. 46, pp. 45-64, 1999.

[33] M. Raghavan and B. Roth, "Kinematic analysis of the 6R manipulator of general geometry," *5th Int. Symp. Robotics Research*, 1990.

[34] E. Rimon and J.W. Burdick, "Mobility of bodies in contact. I. A second-order mobility index for multiple-finger grasps,"a *IEEE Trans. Robotics Autom.*, vol. 14, no. 5, pp. 696-708, 1998.

[35] T. Shamir and Y. Yomdin, "Repeatability of redundant manipulators: mathematical solution to the problem," *IEEE Trans. Automatic Control*, vol. 33, no. 11, pp. 1004-1009.

[36] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Berlin-Heidelberg: Springer, 2008.

[37] E.T. Whittaker, *A Treatise on the Analytical Dynamics of Particles and Rigid Bodies*. Cambridge, Cambridge University Press, 1917.