Chapter 3

Kinematics

Kinematics is a geometric approach to robot motion. It is the study of positions and linear velocities, of angles and angular velocities. In kinematics, we create abstractions to simplify our analysis of the motion of billions of particles into a small number of motions that we have designed into the system as joints. The joints connect links that are rigid bodies, and we assume they are unbendable. In a classic mechanical system such as this, we call the joint angles freedoms, and we count the number of degrees of freedom of the whole system such that a vector can completely describe the configuration of the robot at some moment in time.

We start with the basics of rigid body motion, namely translation and rotation. Then we introduce the configuration space and define forward and inverse kinematic mappings, which address questions of where the robot is and where we want it to be. Lastly, we turn to velocity kinematics.

It should be noted that kinematics is not concerned with why things move; that question is addressed in mechanics by the introduction of forces, which induce accelerations. In kinematics, we are only concerned with position and its first derivative, velocity. When we talk about velocity kinematics, we may be describing the rate of change of position of an object with respect to time or with respect to that motion expressed by other coordinates.

In this chapter, we begin with a gentle introduction to kinematics for those readers without a mechanical background. If you have seen kinematics before, you may be comfortable jumping straight to Section 3.2.

3.1 A Gentle Introduction to Kinematics

This section is designed for those without a background in mechanics to receive a primer in the basic concepts of kinematics by presenting concepts in a simplified
fashion in two dimensions only. The structure of this section partly mirrors the structure of the rest of the kinematics chapter, so that you can easily jump back and forth between the basics and the full treatment.

### 3.1.1 The Rigid Body Assumption

A **rigid body** is a set of particles that move together because they are attached. When I move one particle in a rigid body by pushing on it, the other particles will all follow in some coherent way. This concept is an abstraction because it assumes that the object is perfectly rigid and inflexible. Since no object is completely rigid, the rigid body assumption holds only under a set of other reasonable assumptions: no unusually-large forces or deformations are imparted on the object, the object is not broken, cut, or otherwise disassembled.

Examples of objects that fit the rigid body assumption well include a brick, a metal crowbar, or a plastic shovel. Some objects that do not fit the rigid body assumption well at all include a piece of paper (it bends too easily), a body of water (the particles freely rearrange and flow), and a Jenga tower (the blocks fall apart easily). Some objects are in between these two extremes, and we may choose to apply the rigid body assumption anyway, knowing that it is imperfect. For example, a beach ball stays mostly spherical, but it is soft and deforms where it touches objects. A building is rigid and largely fixed, but the doors can swing independently of the building; we just neglect this detail due to the difference in scale.

When we talk about a robot, it is important to be clear what assumptions we are making for the sake of a clean mathematical analysis. In order to describe a rigid body, we need to introduce some formalisms.

A **reference frame** is a set of axes fixed to a point (or particle). In Fig. 3.1, we see two reference frames. The **global reference frame**, called \( \{G\} \), is fixed to a point \( O_G \) in the world that does not move. The reference frame \( \{B\} \) is fixed to a certain point \( O_B \) on a rigid body object called \( B \). Our convention is that subscripts are names, such as the \( B \)-origin or the \( G \)-origin. Thus, we know that if the point \( O_B \) moves, then the frame moves with it. But we can say more than this thanks to the rigid body assumption. We can pick any arbitrary point \( A \) on the rigid body \( B \), and we can express its **coordinates** in the frame \( \{B\} \) as

\[
P^B_A = \begin{bmatrix} p^B_x \\ p^B_y \end{bmatrix}.
\]  

We read this expression as the position of the point \( A \) in the frame \( \{B\} \) is defined by the coordinates \( (p^B_x, p^B_y) \), each expressed in the \( \{B\} \) frame. By convention, superscripts define the frame in which something is expressed. Thus, \( P^B_A \) and \( P^G_A \) both refer to the position of the same point named \( A \), but the values of the vectors
3.1. A GENTLE INTRODUCTION TO KINEMATICS

will differ because they are defined with respect to different frames. Since the coordinates of $P_A^B$ are expressed in the body frame of the rigid body and describe a point on the rigid body, we know that the coordinates are constant regardless of how the rigid body moves.

Rigid body motion is described by a combination of two types of motion: translation and rotation. Translation describes a change in position, whereas rotation describes a change in orientation. For example, if the rigid body $B$ used to be at the global origin $O_G$, then we would say that it was translated to its current position by a motion of $P_G^B$. Often, the terms position and translation will be used interchangeably when we don’t care about the history of motions that were applied to $B$ – that is, its current position is the sum of all the translation operations that were applied to it. Similarly, the terms rotation and orientation are sometimes used synonymously because you can combine rotations to define a rigid body’s current orientation. The key concept here is that any arbitrary sequence of translation and rotation operations can be combined and expressed as a single translation and a single rotation.

Note that a particle can translate only – it cannot rotate because it has no orientation. However, when we define a rigid body’s translation and rotation, we define the position of every particle within the rigid body. In Fig. 3.1, we see that the position of frame $\{B\}$ is expressed with respect to the global frame $\{G\}$ as

$$P_G^B = \begin{bmatrix} q_G^x \\ q_G^y \end{bmatrix}^T$$

(3.2)

Vectors are always columns, but here, we have written the vector in transposed form for compactness. Here, the coordinates $(q_G^x, q_G^y)$ describe the position of the
CHAPTER 3. KINEMATICS

\[ O_B = O_G \]

\[ \hat{x}_B = \hat{x}_G \]

\[ \hat{y}_B = \hat{y}_G \]

\[ A \]

\[ \theta \]

\[ \hat{x} \]

\[ \hat{y} \]

\[ \hat{z} \]

\[ B \]

\[ \{B\} \]

\[ \{G\} \]

\[ P_A \]

\[ O_B = O_G \]

\[ \hat{x}_G \]

\[ \hat{y}_G \]

\[ \hat{z}_G \]

\[ \theta \]

\[ \text{Figure 3.2: At left, a rigid body } B \text{ before rotation. At right, the same body after being rotated by angle } \theta. \text{ We see that rotating } B \text{ by } \theta \text{ causes the point } A \text{ to move to a new position in the global reference frame } \{G\}, \text{ although it is still in the same position in the body frame } \{B\}. \]

\{B\} frame in \{G\}-frame coordinates. Since \( B \) is a rigid body that can move with respect to the global frame, we know that these coordinates can vary. To find the values of the coordinates, we can project the point \( O_B \) onto the \{G\}-frame axes \( \hat{x}_G \) and \( \hat{y}_G \).

A fuller treatment of the rigid body assumption in 3D occurs in Section 3.2, including more detailed terminology.

3.1.2 Rotation

In Section 3.1.1, we saw how a position could be expressed in 2D using two coordinates. Rotation in 2D only requires a single parameter, but we will see that there are reasons to sometimes use more than one parameter anyway.

The simplest expression of an orientation or a rotation in 2D is to specify an angle in degrees. Just as with position and translation, we must specify the frame in which an orientation is being measured. Suppose that, as in Fig. 3.2, a point \( A \) is located on rigid body \( B \). If we say that \( B \)'s frame \( \{B\} \) is rotated by angle \( \theta \) with respect to global frame \( \{G\} \), then we can use this information to compute the global coordinates (position) of \( A \). Remember that the point \( A \) does not have an innate orientation, but the act of applying a pure rotation to \( B \) causes all points in \( B \) to rotate about \( B \)'s origin, \( O_B \). The operation of rotating \( A \) about \( O_B \) causes \( A \) to translate to a new position.

If we want to compute the new position of \( A \) after applying the rotation operation to \( B \), then we need to use some trigonometry. If we know the unchanging coordinates of \( A \) in the \( B \) frame as \( P^B_A = (p^B_x, p^B_y) \), then the transformed coordinates following the rotation are \( P^G_A = (p^B_x \cos \theta - p^B_y \sin \theta, p^B_x \sin \theta + p^B_y \cos \theta) \).

This result can be derived from inspection, but it may not seem intuitively obvious.
yet. We return to it from a more principled stance shortly.

Using the angle $\theta$ to describe an orientation is convenient and intuitive, but it has a flaw that will cause mathematical complications later. The set possible values of rotations is $\mathbb{R}$, the set of real numbers. However, if we rotate a rigid body object by some multiple of 360° or $2\pi$ radians, then every point in the object is in precisely the same position. To avoid this redundancy, we might wish to restrict the set of valid orientations to something like $[0, 2\pi)$ – that is, the interval of the real numbers between 0 and $2\pi$ radians that is closed on the bottom end (includes 0) and open on the top end (includes every number up to but not including $2\pi$). Introducing the interval eliminates the redundancy problem, but it introduces a second problem. If we wish to differentiate an angle to compute an angular velocity, then we will periodically encounter a discontinuity that makes it impossible to differentiate.

To resolve these issues, we introduce an alternative representation of rotations called a rotation matrix based on the observation above that we need to use trigonometry to compute the displacement of a point on a rigid body under rotation. In 2D, the rotation matrix is a $2 \times 2$ matrix,

$$R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}. \quad (3.3)$$

From this definition, it immediately follows that we can rotate a point simply by matrix multiplication,

$$P_A = R_B P_A \quad (3.4)$$

$$= \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} p_x^B \\ p_y^B \end{bmatrix} \quad (3.5)$$

$$= \begin{bmatrix} p_x^B \cos \theta - p_y^B \sin \theta \\ p_x^B \sin \theta + p_y^B \cos \theta \end{bmatrix}. \quad (3.6)$$

This result matches the intuitive trigonometric expression we arrived at earlier. However, the rotation matrix encodes a lot of information that can be quickly and easily interpreted. For example, if we consider the coordinate frame axes of $\{B\}$ as expressed in the $\{G\}$ frame, we can write the rotation matrix

$$R_G^B = \begin{bmatrix} \hat{x}_B^G & \hat{y}_B^G \end{bmatrix} \quad (3.7)$$

that corresponds to the rotation from the source frame $\{G\}$ to the destination frame $\{B\}$.

**Example 3.1** Interpreting a Rotation Matrix
Problem: Suppose you are given the rotation matrix

\[
R = \begin{bmatrix}
0.866 & -0.500 \\
0.500 & 0.866
\end{bmatrix}.
\] (3.8)

What are the axes of the destination frame as expressed in the origin frame?

Solution: We read the axes of the destination frame as the column vectors of \( R \), so \( \hat{x} = [0.866 \ 0.5]^T \) and \( \hat{y} = [-0.5 \ 0.866]^T \).

A full treatment of rotations in 3D, including several alternate representations, can be found in Section 3.3. That section also provides details on the properties of rotations and comparisons of the various representational alternatives.

3.2 The Rigid Body Assumption

Let \( \{G\} \) represent a *global reference frame*, i.e. a coordinate frame fixed in a 3-dimensional Cartesian space \( \mathbb{R}^3 \) at a point \( O \) with axes defined by the unit vectors \( \{\hat{x}_G, \hat{y}_G, \hat{z}_G\} \). The kinematic state of a point, or particle, can be fully specified by its position with respect to \( \{G\} \). The position of a particle \( A \) with respect to \( \{G\} \) can be given by a 3-vector,

\[
P_A^G = [p_x \ p_y \ p_z]^T,
\] (3.9)

which is called a *position vector*. The components of the position vector denote the projection of \( P_A^G \) onto each axis of \( \{G\} \) (see Fig. 3.3).
3.2. THE RIGID BODY ASSUMPTION

A rigid body can be defined as a collection of particles such that the distance between any two particles remains fixed, regardless of any motions of the body or forces exerted on the body. In other words, a rigid body is an idealized, completely undeformable solid body. We use the rigid body assumption in robotics to model a robot’s whole body or its components, such as the links of a manipulator.
In order to fully specify the kinematic state of a rigid body, we need descriptions of both its position and its orientation. To find these descriptions, we attach a frame \( \{ B \} \) to the body called the body frame (see Fig. 3.4). Let \( \{ B \} \) be attached at a point \( C \) and comprise the axes defined by the unit vectors \( \{ \hat{x}_B, \hat{y}_B, \hat{z}_B \} \). Then the position vector \( P^G_C \) represents the position of the body, and the orientation of \( \{ B \} \) with respect to \( \{ G \} \) represents the orientation of the body.

The motion of a rigid body in space can be fully described by its position with respect to a coordinate frame at every instant in time. A rigid motion of a body is a continuous movement of its particles under the constraint that the distance between any two particles remains invariant. The net movement of a body via rigid motion is called a rigid displacement, which may result from translation (linear displacement), rotation (angular displacement), or both at the same time.

A translation has only one common representation, which is a vector that looks just like a position vector. The vector \( P^G_B \) can be interpreted as either (1) a position

\[ P^G_B = \begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix} \]

1. A body frame can be attached anywhere in the body, although for most applications the center of mass is usually selected.

2. These vectors are expressed with respect to \( \{ G \} \), but the superscripts are often omitted when the global frame of reference is understood.
vector giving the location of the origin of the \{B\} frame with respect to the \{G\} frame, or (2) a translation of a rigid body into the \{B\} frame from the \{G\} frame. Translations are composed by vector addition.

Unlike translations, rotations are complex, and many different ways to represent them have been devised. In the following section, we describe several of these representations as well as their advantages and disadvantages.

3.3 Rotation Representations

For readers without a mechanical engineering background, a gentle introduction to rotation representation conventions can be found in Section 3.1.2.

3.3.1 The Rotation Matrix

If we take the axes of a body frame \{B\} expressed relative to the global frame \{G\} and use them as the columns of a matrix, the result is a $3 \times 3$ matrix called the rotation matrix:

$$R^G_B = \begin{bmatrix} \hat{x}_B^G & \hat{y}_B^G & \hat{z}_B^G \end{bmatrix}.$$ \hspace{1cm} (3.10)

Since the columns of $R^G_B$ are the axes of \{B\}, they are orthogonal unit vectors, and the rotation matrix is therefore an orthogonal matrix.

As noted in the previous section, a position vector is formed by its projections onto the axes of a selected reference frame. Therefore, each column in the rotation matrix contains the projections of the corresponding axis of \{B\} onto the axes of \{G\}. Since the axes are unit vectors, the projection of an axis of \{B\} onto an axis of \{G\} is the cosine of the angle between those two axes (often called a direction cosine). Thus,

$$R^G_B = \begin{bmatrix} \cos(\hat{x}_B, \hat{x}_G) & \cos(\hat{y}_B, \hat{x}_G) & \cos(\hat{z}_B, \hat{x}_G) \\ \cos(\hat{x}_B, \hat{y}_G) & \cos(\hat{y}_B, \hat{y}_G) & \cos(\hat{z}_B, \hat{y}_G) \\ \cos(\hat{x}_B, \hat{z}_G) & \cos(\hat{y}_B, \hat{z}_G) & \cos(\hat{z}_B, \hat{z}_G) \end{bmatrix}.$$ \hspace{1cm} (3.11)

Alternatively, we can write the direction cosines as dot products, yielding

$$R^G_B = \begin{bmatrix} \hat{x}_B \cdot \hat{x}_G & \hat{y}_B \cdot \hat{x}_G & \hat{z}_B \cdot \hat{x}_G \\ \hat{x}_B \cdot \hat{y}_G & \hat{y}_B \cdot \hat{y}_G & \hat{z}_B \cdot \hat{y}_G \\ \hat{x}_B \cdot \hat{z}_G & \hat{y}_B \cdot \hat{z}_G & \hat{z}_B \cdot \hat{z}_G \end{bmatrix}.$$ \hspace{1cm} (3.12)

We can use the fact that the dot product is commutative to derive an interesting property of the rotation matrix. If we transpose $R^G_B$ and then use commutativity to
change the order of the vectors in each dot product, we see that

\[(R_B^G)^T = \begin{bmatrix}
\hat{x}_G \cdot \hat{x}_B & \hat{y}_G \cdot \hat{x}_B & \hat{z}_G \cdot \hat{x}_B \\
\hat{x}_G \cdot \hat{y}_B & \hat{y}_G \cdot \hat{y}_B & \hat{z}_G \cdot \hat{y}_B \\
\hat{x}_G \cdot \hat{z}_B & \hat{y}_G \cdot \hat{z}_B & \hat{z}_G \cdot \hat{z}_B
\end{bmatrix} = R_G^B. \tag{3.13} \]

The transpose of a rotation matrix also has an important relationship with its inverse. If we multiply \(R_B^G\) by \((R_B^G)^T\), we see that

\[R_B^G(R_B^G)^T = \begin{bmatrix}
\hat{x}_B^G & \hat{y}_B^G & \hat{z}_B^G
\end{bmatrix} \begin{bmatrix}
(\hat{x}_B^G)^T \\
(\hat{y}_B^G)^T \\
(\hat{z}_B^G)^T
\end{bmatrix} = I_3 \tag{3.14} \]

since the columns of \(R_B^G\) are mutually orthogonal. Thus,

\[(R_B^G)^T = (R_B^G)^{-1}. \tag{3.15} \]

Another notable property of a rotation matrix \(R_B^G\) is that

\[\det(R_B^G) = 1 \tag{3.16} \]

under the assumption of right-handed coordinates. (In general, it can be ±1.)

**Interpretations of the Rotation Matrix**

The rotation matrix has different interpretations depending on the application. In particular, the rotation matrix can be used to

- describe the orientation of a coordinate frame with respect to another frame,
- map the coordinates of a point located in one frame to another frame, and
- rotate a vector to a new orientation in the same coordinate frame.

We provide an example for each of these interpretations below.

**Example 3.2** Describing the orientation of a frame

Consider a planar rotation in which a frame is rotated about its \(z\)-axis for an angle \(\theta\) to transition from an initial orientation \(\{0\}\) to a final orientation \(\{1\}\) (see Fig. 3.5). We would like to describe \(\{1\}\) relative to \(\{0\}\). Hence, we use (3.11) to derive the expression

\[R_z(\theta) = R_1^0 = \begin{bmatrix}
\hat{x}_1^0 & \hat{y}_1^0 & \hat{z}_1^0
\end{bmatrix} = \begin{bmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}, \tag{3.17} \]
3.3. ROTATION REPRESENTATIONS

where the notation \( R_z(\theta) \) is used to denote rotation about the \( z \)-axis for an angle \( \theta \).

![Diagram of rotation about the z-axis](image)

Figure 3.5: Example of a planar rotation about the \( z \)-axis from a side (left) and top view.

Similar computations yield the expressions for the rotation matrix describing a rotation about the \( y \)-axis,

\[
R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}, \quad (3.18)
\]

and the \( x \)-axis,

\[
R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}. \quad (3.19)
\]

**Example 3.3** Mapping the coordinates of a point

Assume we are given two frames \( \{1\} \) and \( \{2\} \) as well as the coordinates of a point \( A \) with respect to frame \( \{2\} \) in the form of a position vector \( P_A^{2} = [u \ v \ w]^T \) (see Fig. 3.6). Our goal is to determine the coordinates of \( A \) with respect to frame \( \{1\} \); that is, we wish to find the position vector \( P_A^{1} \).
Figure 3.6: The rotation matrix can be used to express a point $A$ with respect to different coordinate frames.

Since the elements of $P_A^2$ are its projections onto the axes of frame $\{2\}$, we can rewrite $P_A^2$ as

$$P_A^2 = u\hat{x}_2 + v\hat{y}_2 + w\hat{z}_2.$$  \hspace{1cm} (3.20)

We can write a similar expression for $P_A^1$,

$$P_A^1 = \begin{bmatrix} P_A^2 \cdot \hat{x}_1 \\ P_A^2 \cdot \hat{y}_1 \\ P_A^2 \cdot \hat{z}_1 \end{bmatrix}.$$ \hspace{1cm} (3.21)

From equations (3.21) and (3.20), we have

$$P_A^1 = \begin{bmatrix} (u\hat{x}_2 + v\hat{y}_2 + w\hat{z}_2) \cdot \hat{x}_1 \\ (u\hat{x}_2 + v\hat{y}_2 + w\hat{z}_2) \cdot \hat{y}_1 \\ (u\hat{x}_2 + v\hat{y}_2 + w\hat{z}_2) \cdot \hat{z}_1 \end{bmatrix} = \begin{bmatrix} u\hat{x}_2 \cdot \hat{x}_1 + v\hat{y}_2 \cdot \hat{x}_1 + w\hat{z}_2 \cdot \hat{x}_1 \\ u\hat{x}_2 \cdot \hat{y}_1 + v\hat{y}_2 \cdot \hat{y}_1 + w\hat{z}_2 \cdot \hat{y}_1 \\ u\hat{x}_2 \cdot \hat{z}_1 + v\hat{y}_2 \cdot \hat{z}_1 + w\hat{z}_2 \cdot \hat{z}_1 \end{bmatrix} = \begin{bmatrix} \hat{x}_2 \cdot \hat{x}_1 & \hat{y}_2 \cdot \hat{x}_1 & \hat{z}_2 \cdot \hat{x}_1 \\ \hat{x}_2 \cdot \hat{y}_1 & \hat{y}_2 \cdot \hat{y}_1 & \hat{z}_2 \cdot \hat{y}_1 \\ \hat{x}_2 \cdot \hat{z}_1 & \hat{y}_2 \cdot \hat{z}_1 & \hat{z}_2 \cdot \hat{z}_1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}.$$ \hspace{1cm} (3.22)

The matrix in the final step of (3.22) is the rotation matrix $R^1_2$ and therefore

$$P_A^1 = R^1_2 P_A^2.$$ \hspace{1cm} (3.23)
3.3. ROTATION REPRESENTATIONS

Equation (3.23) is a useful expression that allows us to map the coordinates of a point from one frame to another.

**Example 3.4** Rotating a position vector

Consider the block of Fig. 3.7a in which one corner of the block is located at point $C$ in space. After rotating the block about the axis $\hat{z}_G$ for $\pi$ radians, the block is configured as shown in Fig. 3.7b, and the same corner of the block is now located at point $D$. We would like to describe the change in the position of the corner as the rotation of a position vector with respect to the global frame.

To do this, we attach a body frame $\{B\}$ to the block such that $\{B\}$ is coincident with $\{G\}$ before rotation, as shown in Fig. 3.7a. Using $R_z(\theta)$ from (3.17), we can derive the rotation matrix representing the final orientation of $\{B\}$ with respect to the global frame $\{G\}$ as follows:

$$R^G_B = R_z(\pi) = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.24)$$

Now we can use (3.23) from the previous example to relate the coordinates of $D$ within the body frame to the coordinates of $D$ within the global frame, as

$$P^G_D = R^G_B P^B_D. \quad (3.25)$$

But since the position of the corner never changes with respect to $\{B\}$, it must be the case that $P^B_D = P^G_C$ when $\{B\}$ and $\{G\}$ are aligned as in Fig. 3.7a. We can
therefore substitute $P_G^C$ into (3.25) to obtain

$$P_D^G = R_B^G P_C^G. \quad (3.26)$$

Equation (3.26) thus allows us to express the rotational motion of a vector within the same coordinate frame.

**Similarity Transformations**

A rotation matrix can be thought of as a change of basis from one frame to another. As noted earlier, the columns of the rotation matrix represent the unit vectors that define the new basis with respect to the old basis. It is often necessary to express a general linear transformation with respect to a different frame. This can be performed with a similarity transformation,

$$ A' = T^{-1} A T, \quad (3.27) $$

where $A$ is a linear transformation $A \in \mathbb{R}^{n \times n}$, expressed with respect to an initial basis and $T$ is a matrix formed by a set of basis vectors defining another coordinate frame.

For example, assume that $A$ is a linear transformation expressed with respect to a frame $\{0\}$ and that $A'$ is the same transformation expressed with respect to another frame $\{1\}$. Given the rotation matrix $R_0^1$, $A$ can be expressed with respect to $\{1\}$ as

$$ A' = (R_0^1)^{-1} A R_0^1. \quad (3.28) $$

Therefore, this relationship allows us to express a rotation operation described by a rotation matrix $A$ with respect to a desired frame.

**Composition of Rotations**

Most robotic mechanisms are composed of several rigid bodies attached to each other by joints of various kinds. As a result, to describe their kinematics, it is often necessary to compose the motions of their components. The typical approach to this problem is to attach a frame to each body, determine the relative displacements between successive bodies, and combine those displacements into a net displacement. Importantly, when composing a sequence of rotational displacements, the method of composition depends on whether the rotations were made with respect to the *current frame* or to the *fixed frame*. We consider both cases below.

---

3 A rotation matrix is an example of a linear transformation, when used as a rotation operator.
3.3. ROTATION REPRESENTATIONS

Current Frame

Consider three coordinate frames \{0\}, \{1\}, and \{2\}. Any point \( p \in \mathbb{R}^3 \) can be described with respect to each of those frames through multiplication with an appropriate rotation matrix as shown in the previous section. Consider for example the following relationships:

\[
p^0 = R_{1}^0 p^1, \quad (3.29)
\]

\[
p^1 = R_{2}^1 p^2, \quad (3.30)
\]

\[
p^0 = R_{2}^0 p^2. \quad (3.31)
\]

Substituting (3.30) into (3.29), we derive:

\[
p^0 = R_{1}^0 R_{2}^1 p^2. \quad (3.32)
\]

Looking at equations (3.31) and (3.32), we notice that:

\[
R_{2}^0 = R_{1}^0 R_{2}^1. \quad (3.33)
\]

This simple example reflects the composition law for composing rotational transformations: consecutive rotational transformations can be synthesized by multiplying their corresponding rotation matrices.

Fixed Frame

In some applications, we may need to perform a sequence of rotations about a single fixed frame rather than successive current frames. In those cases, in order to compose the individual rotations into a net rotation, we can use similarity transformations to reduce the problem to a sequence of rotations with respect to the current frame.

For example, assume that the rotation matrix \( R_{1}^0 \), representing the rotation of a frame \{1\} with respect to a fixed frame \{0\}, is available. Consider also another rotation \( R \) with respect to \{0\}. We can transform \( R \) into a rotation \( R_{2}^1 \) by taking the following similarity transformation:

\[
R_{2}^1 = (R_{1}^0)^{-1} R R_{1}^0. \quad (3.34)
\]

Then we can determine the total rotation \( R_{2}^0 \) as shown in the previous section:

\[
R_{2}^0 = R_{1}^0 (R_{1}^0)^{-1} R R_{1}^0 = RR_{1}^0. \quad (3.35)
\]

Therefore, when a rotation \( R \) is performed with respect to the global frame, it can be incorporated into a net rotation by premultiplying the current rotation matrix by \( R \).
Constraints on the Rotation Matrix

Recall that we originally introduced the rotation matrix as
\[ R_{GB} = [\hat{x}_B, \hat{y}_B, \hat{z}_B] \] (3.36)
where the columns of \( R_{GB} \) are orthogonal unit vectors. The fact that they are unit vectors gives us the dependencies
\[ \| \hat{x}_B \| = 1, \quad \| \hat{y}_B \| = 1, \quad \| \hat{z}_B \| = 1, \] (3.37)
and the fact that they are mutually orthogonal gives us the dependencies
\[ \hat{x}_B \cdot \hat{y}_B = 0, \quad \hat{x}_B \cdot \hat{z}_B = 0, \quad \hat{y}_B \cdot \hat{z}_B = 0. \] (3.38)
Thus we have six equations that constrain the nine elements of the rotation matrix. Mathematically, this indicates that only three of those elements are independent quantities, which represents the fact that a rigid body possesses at most three rotational degrees of freedom. It follows that any rotation can be expressed with three parameters. Several rotation representations make use of this fact, and we present a few notable ones below: Euler angles, the axis-angle pair, and quaternions.

3.3.2 Euler Angles

The Euler angles parameterization is one of the most common ways to represent rotations in robotics. The main idea behind Euler angles is to decompose a rotation into a sequence of three successive rotations. Consider a fixed frame \( \{0\} \) and a rotated frame \( \{1\} \). We can represent the orientation of \( \{1\} \) with respect to \( \{0\} \) with a set of Euler angles \( (\phi, \theta, \psi) \) as follows (the steps are depicted in Fig. 3.8):

1. Rotate \( \{0\} \) about the \( z \)-axis by the angle \( \phi \) to derive a new frame \( \{a\} \).

2. Rotate \( \{a\} \) about its \( y \)-axis by the angle \( \theta \) to derive a new frame \( \{b\} \).

3. Rotate \( \{b\} \) about its \( z \)-axis by the angle \( \psi \).

The aforementioned rotations can be represented as rotation matrices \( R_z(\phi), R_y(\theta), R_z(\psi) \) that describe rotations \emph{with respect to the current frame}. They can
3.3. ROTATION REPRESENTATIONS

Figure 3.8: Euler Angles \((\phi, \theta, \psi)\) representing the orientation of frame \(\{1\}\) with respect to frame \(\{0\}\).

be composed into a rotation matrix\(^4\) \(R_{\text{ZYX}}\) as follows:

\[
R_{\text{ZYX}} = R_z(\phi)R_y(\theta)R_z(\psi) \\
= \begin{bmatrix}
    c_\phi & -s_\phi & 0 \\
    s_\phi & c_\phi & 0 \\
    0 & 0 & 1
\end{bmatrix}
= \begin{bmatrix}
    c_\theta & 0 & s_\theta \\
    0 & 1 & 0 \\
    -s_\theta & 0 & c_\theta
\end{bmatrix}
= \begin{bmatrix}
    c_\psi & -s_\psi & 0 \\
    s_\psi & c_\psi & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

(3.39)

(3.40)

(3.41)

\[
\frac{\sum c_{\text{angle}} c_{\text{angle}} c_{\text{angle}} - s_{\text{angle}} s_{\text{angle}}}{-s_{\text{angle}} c_{\text{angle}} + c_{\text{angle}} s_{\text{angle}} + c_{\text{angle}} c_{\text{angle}} + s_{\text{angle}} s_{\text{angle}}}
\]

where the notation \(c_{\text{angle}}, s_{\text{angle}}\) denotes \(\cos(\text{angle})\) and \(\sin(\text{angle})\) respectively, with \(\text{angle} \in \{\phi, \theta, \psi\}\). The rotation matrix \(R_{\text{ZYX}}\) is called the ZYX Euler angle transformation. Depending on the sequence of rotations considered, various conventions for Euler angles can be defined (how many?).

Many times in robotics, we face the opposite problem, i.e., we need to determine a set of Euler angles, given an orientation, in the form of a rotation matrix. Consider a given rotation matrix in its general form:

\[
R = \begin{bmatrix}
    r_{11} & r_{12} & r_{13} \\
    r_{21} & r_{22} & r_{23} \\
    r_{31} & r_{32} & r_{33}
\end{bmatrix}
\]

(3.42)

We break the problem into two cases:

- **Case 1**: At most one of \(r_{13}\) and \(r_{23}\) is zero.

Then from (3.41), \(s_\theta \neq 0\), which implies that \(c_\theta \neq \pm 1\). Consequently, (a)

\(^4\)The subscript denotes the angle convention. Different conventions for sequences of individual rotations can be defined.
CHAPTER 3. KINEMATICS

$r_{31}$ and $r_{32}$ cannot both be zero and (b) $r_{33} \neq \pm 1$. From the trigonometric Pythagorean identity, we can derive the expression $s_\theta = \pm \sqrt{1 - r_{33}^2}$. Therefore

$$\theta = \text{atan2} \left( r_{33}, \sqrt{1 - r_{33}^2} \right)$$

or

$$\theta = \text{atan2} \left( r_{33}, -\sqrt{1 - r_{33}^2} \right).$$

If we choose (3.43), then it follows that $s_\theta > 0$ and therefore:

$$\phi = \text{atan2} \left( r_{13}, r_{23} \right),$$

$$\psi = \text{atan2} \left( -r_{31}, r_{32} \right).$$

Otherwise, if we choose (3.44), then $s_\theta < 0$ and therefore:

$$\phi = \text{atan2} \left( -r_{13}, -r_{23} \right),$$

$$\psi = \text{atan2} \left( r_{31}, -r_{32} \right).$$

Hence, depending on the sign of $\theta$, we get a different solution.

- Case 2: Both $r_{13}$ and $r_{23}$ are zero.
  Since $R$ is orthogonal, its columns are unit vectors; therefore $r_{33} = c_\theta = \pm 1$, which also implies that $s_\theta = 0$. Hence, $r_{31} = r_{32} = 0$. Thus $R$ has the following form:

$$R = \begin{bmatrix} r_{11} & r_{12} & 0 \\ r_{21} & r_{22} & 0 \\ 0 & 0 & \pm 1 \end{bmatrix}. \quad (3.49)$$

If $r_{33} = 1$, then $c_\theta = 1$ and $\theta = 0$. Substituting in (3.41) and applying some trigonometric sum identities, we get:

$$R = \begin{bmatrix} c_\phi c_\psi - s_\phi s_\psi & -c_\phi s_\psi - s_\phi c_\psi & 0 \\ s_\phi c_\psi + c_\phi s_\psi & -s_\phi s_\psi + c_\phi c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_{\phi+\psi} & -s_{\phi+\psi} & 0 \\ s_{\phi+\psi} & c_{\phi+\psi} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.50)$$

Therefore, we can get the following expression for the sum $\phi + \psi$:

$$\phi + \psi = \text{atan2}(r_{11}, r_{21}) = \text{atan2}(r_{11}, -r_{12})$$

which implies that there are infinitely many solutions for $\phi$ and $\psi$. 

Finally, if \( r_{33} = -1 \), then in a similar way we derive:

\[
R = \begin{bmatrix}
-c_{\phi-\psi} & -s_{\phi-\psi} & 0 \\
 s_{\phi-\psi} & c_{\phi-\psi} & 0 \\
 0 & 0 & -1
\end{bmatrix}
\]  

(3.52)

which leads to a similar expression for the difference of \( \phi - \psi \):

\[
\phi - \psi = \text{atan2}(-r_{11}, -r_{12}).
\]  

(3.53)

This also implies that there are infinitely many solutions for \( \phi \) and \( \psi \).

The main weakness of the Euler angles representation arises from its main idea of representing the rotation with three numbers. This leads to singularities, i.e. the lack of global, smooth solutions to the inverse problem of determining the Euler angles from a given rotation matrix. For example, in the case of the case of the \( ZYZ \) parameterization, singularities occur at \( R = I \). In particular, assuming Euler angles \((\phi, \theta, \psi)\) of the form \((\phi, 0, \psi)\) yields \( R = I \). Therefore there are infinitely many representations of the identity rotation in the \( ZYZ \) Euler angle parameterization. Such singularities occur in all conventions at different points.

### 3.3.3 Axis-Angle Representation

Another common way to represent rotations is by an axis-angle pair. This convention makes use of the fact that any rotation matrix can be represented by a single rotation about a suitable axis for a suitable angle.

Consider a fixed frame \( \{0\} \) and a unit vector \( \hat{k} = [k_x \ k_y \ k_z]^T \) expressed with respect to \( \{0\} \) that defines an axis (see Fig. 3.9). Assume that we want to perform a rotation about \( \hat{k} \) by an angle \( \theta \). To represent this rotation, we may derive a rotation matrix \( R_k(\theta) \). There are many different ways to do so. For example, note that a rotational transformation of the form \( R = R_z(\alpha)R_y(\beta) \) by appropriate angles \( \alpha \) and \( \beta \) first about the \( z \)- and then about the \( y \)-axis may align the \( z \)-axis of \( \{0\} \) with \( \hat{k} \). Performing a rotation about \( \hat{k} \) can then be done with a similarity transformation:

\[
R_k(\theta) = RR_z(\theta)R^{-1}
\]

\[
= R_z(\alpha)R_y(\beta)R_z(\theta)(R_z(\alpha)R_y(\beta))^{-1}
\]

\[
= R_z(\alpha)R_y(\beta)R_z(\theta)R^{-1}_y(\beta)R^{-1}_z(\alpha)
\]

\[
= R_z(\alpha)R_y(\beta)R_z(\theta)R_y(-\beta)R_z(-\alpha)
\]

(3.57)
We typically think of the axis-angle representation as requiring four real numbers: a 3-vector axis plus a scalar angle. However, the representation can be compressed into only three real numbers as \( \theta \hat{k} \). This compressed representation has two benefits. First, it eliminates the ambiguity presented by rotating by a non-zero angle about a zero-magnitude vector. Second, it removes redundancy inherent in the axis specification. Since values of \( \hat{k} \) are constrained to the surface of a unit sphere, \( \hat{k} \) has only two degrees of freedom represented by three values.

### 3.3.4 Quaternions

Quaternions can be thought of as a generalization of the complex numbers and can be used to represent rotations in a 3-dimensional Cartesian space in the same way that complex numbers can be used to represent planar rotations on the unit circle. Quaternions avoid the singularities of Euler angles by using four parameters to represent a rotation.

A quaternion \( Q \in \mathbb{Q} \subseteq \mathbb{R}^4 \) can be defined as the following vector:

\[
Q = q_0 + q_1 i + q_2 j + q_3 k
\]  

(3.58)

where \( q_0, q_1, q_2, q_3 \in \mathbb{R} \) and \( i, j \) and \( k \) are the fundamental unit quaternions. We usually describe a quaternion as \( Q = (q_0, \vec{q}) \) where \( q_0 \) is the scalar component of \( Q \) and \( \vec{q} = (q_1, q_2, q_3) \in \mathbb{R}^3 \) is its vector component.

\footnote{This similarity transformation changes the basis of \( R_x(\theta) \) from \( \{0\} \) to \( \hat{k} \).}
3.3. ROTATION REPRESENTATIONS

The set of quaternions $\mathbb{Q}$ forms a group with the operation of multiplication. The quaternion multiplication is distributive and associative but non-commutative. It also satisfies the following properties:

$$ai = ia \quad aj = ja \quad ak = ka \quad a \in \mathbb{R}$$ \hspace{1cm} (3.59)

$$i \cdot i = j \cdot j = k \cdot k = 1 \quad \text{i.j} = \text{j.k} = \text{k.i} = -1$$ \hspace{1cm} (3.60)

$$i \cdot j = -j \cdot i = k \quad j \cdot k = -k \cdot j = i \quad k \cdot i = -i \cdot k = j$$ \hspace{1cm} (3.61)

The conjugate of a quaternion $Q = (q_0, \mathbf{q})$ is given by $Q^* = (q_0, -\mathbf{q})$ and satisfies the following equation:

$$\|Q\|^2 = Q \cdot Q^* = q_0^2 + q_1^2 + q_2^2 + q_3^2.$$ \hspace{1cm} (3.62)

The inverse of a quaternion is given by $Q^{-1} = Q^*/\|Q\|^2$, and it can be seen that $Q = (1, 0)$ is the identity element for quaternion multiplication.

The product of two quaternions $Q = (q_0, \mathbf{q})$ and $P = (p_0, \mathbf{p})$ can be shown to be equal to:

$$Q \cdot P = (q_0p_0 - \mathbf{q} \cdot \mathbf{p}, q_0\mathbf{p} + p_0\mathbf{q} + \mathbf{q} \times \mathbf{p}).$$ \hspace{1cm} (3.63)

In order to represent and compose rotations, we usually make use of the subset of unit quaternions $\{Q \in \mathbb{Q} : \|Q\| = 1\}$. There is a nice correspondence between the axis-angle representation presented in the previous section and a unit quaternion. In particular, consider a rotation defined as the axis-angle pair $(\hat{k}, \theta)$. It can be shown that this rotation corresponds to the quaternion

$$Q = (\cos \frac{\theta}{2}, \hat{k} \sin \frac{\theta}{2}).$$ \hspace{1cm} (3.64)

For the inverse operation, given a unit quaternion $Q = (q_0, \vec{q})$, we can extract the corresponding axis-angle pair as:

$$\theta = 2 \cos^{-1} q_0 \quad \hat{k} = \begin{cases} \frac{\vec{q}}{\sin(\theta/2)} & \text{if } \theta \neq 0, \\ 0 & \text{otherwise}. \end{cases}$$ \hspace{1cm} (3.65)

Given a rotation matrix $R$, we may derive its corresponding quaternion as:

$$q_0 = \frac{1}{2} (1 + r_{11} + r_{22} + r_{33})^{1/2} \quad 0 \leq \theta \leq \pi$$ \hspace{1cm} (3.66)

$$\vec{q} = \frac{1}{4q_0} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}.$$ \hspace{1cm} (3.67)
In order to compose consecutive rotations, we can multiply their corresponding quaternions in the same way as we did in the rotation matrix representations. Consider the quaternions $Q_0^1$ and $Q_1^2$, representing respectively the orientation of frame $\{1\}$ with respect to frame $\{0\}$ and the orientation of frame $\{2\}$ with respect to frame $\{1\}$. We may derive the quaternion expressing the orientation of frame $\{2\}$ with respect to frame $\{0\}$ by the multiplication

$$Q_2^0 = Q_1^0 \cdot Q_2^1.$$  \hspace{1cm} (3.68)

### 3.3.5 Comparison of Representations

Why might a roboticist select one parametrization of rotations over another? In this section, we discuss some of the properties that we need out of any representation of rotations, and we consider the properties of various representations that may be useful or problematic in an implementation.

Some of the most important properties when writing software that computes rotations for a robot are

- **Dimensional generality.** We prefer that representations support rotations in both two and three dimensions.
- **Compactness.** As a rule, we like data structures to be no larger than they need to be. If a representation takes more than the minimal space to store information, we say that it has redundancy. Algorithms compute faster on smaller data structures because more of them fit into memory cache, etc. They also transmit more quickly over networks than larger data structures. With modern computers, this is a minimal concern. However, it is related to another similar property.
- **Uniqueness.** We prefer that for each possible rotation we might want to express, there should be one unique representational form of that rotation. Having redundant representations of the same rotation suggests that the data structure is less compact than necessary, and also that comparisons for equality are more difficult.
- **Numerical stability.** Computations with floating point numbers are subject to numerical error. For example, with repeated multiplications, numbers may lose precision in subtle ways. If a representation has any redundancy, then some possible values in the representation may not correspond to any physical rotation. A good representation should be numerically stable – that is, it should degrade gracefully in the face of accumulating numerical error.
- **Interpolability.** We may wish to interpolate from one orientation to another, with each orientation given by some parametrization of a rotation. The naive
approach to interpolation with a scalar, vector, or matrix is to linearly interpolate each value based on a parameter \( t \in [0, 1] \). Interpolation is not trivial, however, when a representation has redundancy since intermediate values under a linear interpolation do not in general correspond to physically meaningful orientations. Even with representations for which linear interpolation produces valid rotations, the result of linear interpolation may be counterintuitive.

- **Differentiability.** Besides composing discrete rotations, we may wish to describe rotational rates (covered in Section 3.9), in which case we would like it to be the case that for any orientation, a given rate of change of that orientation should correspond at least roughly to a constant rate of change in the parametrization.

The properties above are the metrics by which we should judge each representation for use in robotics applications. The table below shows how the various representations defined previously compare according to these metrics.

<table>
<thead>
<tr>
<th>Representation</th>
<th>Generality</th>
<th>Compactness</th>
<th>Uniqueness</th>
<th>Numerical Stability</th>
<th>Interpolability</th>
<th>Differentiability</th>
</tr>
</thead>
<tbody>
<tr>
<td>angle 2D</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y*</td>
</tr>
<tr>
<td>rotation matrix</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Euler angles 3D</td>
<td>Y</td>
<td>Y*</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>axis-angle 3D</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>quaternion 3D</td>
<td>N</td>
<td>N†</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

* true in general, but fails at a finite number of discrete parameter values.
† the set of unit quaternions is a double-cover of SO(3); that is, for each orientation, there is a corresponding quaternion with both positive and negative real value.

### 3.3.6 Long Form Example

**Example 3.5** Fixed Frame vs. Current Frame Euler Angles

**Concepts reviewed:** expressing coordinates in a new frame, coordinate transformation, composing rotations, Euler angles, body frame vs. global frame, current frame vs. fixed frame.
Problem: Suppose you want to open the treasure chest in Fig. 3.10a lying on its back with the latch at point $A$ identified by coordinates $p_A^B$ or $p_A^G$. You know the coordinates $p_L^G$ of a point $L$ which corresponds to the latch on the unrotated chest in Fig. 3.10b. You want to open the latch and reveal its treasure, but your robot is in the global frame $\{G\}$, and it cannot directly reach to the point $A$ without transforming it into the $\{G\}$ frame first. You are given the orientation of the box with Euler angles $\phi = \frac{\pi}{2}$, $\theta = \frac{\pi}{2}$, $\psi = \frac{\pi}{2}$ using the Euler angle convention $R_{XYZ}$, but you do not know whether it is current or fixed frame. Determine which one is correct and find $p_A^G$.

(a) Rotated body frame    (b) Unrotated global frame

Figure 3.10: Problem statement: a treasure chest is configured as shown in (a), but the position of the latch at point $A$ is only known in the global frame as in (b). Find $p_A^B$. The red, green, and blue axes represent the coordinate frame $\{B\}$ of the body of the treasure chest, and they are fixed to its three straight edges. The global frame $\{G\}$ is on the paper. In (b), the two frames are aligned.

Solution: We compute the Euler angle twice, as both fixed-frame and current-frame, and check which one fits the final orientation.

Note that this problem can be interpreted in two ways. First, we could interpret the problem as expressing coordinates in a new frame. We note that $p_A^B = p_L^B$ because the chest is a rigid body and points $L$ and $A$ both correspond to the same latch. Furthermore, $p_L^G = p_L^B$ because $L$ is defined when the body’s coordinate frame is aligned with the global one. Thus, we know $p_A^B$ and merely need to express it in the global coordinate frame as $p_A^G = R_B^G p_A^B$.

Alternatively, we could interpret the problem as a coordinate transformation by recognizing that at some point in the past, the chest was rotated from its canonical configuration to the way we found it. At that time, the latch moved from the point $L$ to the point $A$. Therefore, $p_A^G = R_B^G p_L^G$.

Fixed Frame. To perform a rotation in the fixed frame (which in our case is
3.3. **ROTATION REPRESENTATIONS**

the global frame), we premultiply. Thus,

\[
R^G_B = R_{XYZ} = R_Z(\psi)R_Y(\theta)R_X(\phi)
\]

\[
= \begin{bmatrix}
    c_\psi & -s_\psi & 0 \\
    s_\psi & c_\psi & 0 \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    c_\theta & 0 & s_\theta \\
    0 & 1 & 0 \\
    -s_\theta & 0 & c_\theta
\end{bmatrix}
\begin{bmatrix}
    1 & 0 & 0 \\
    0 & c_\phi & -s_\phi \\
    0 & s_\phi & c_\phi
\end{bmatrix}.
\]  

(3.69)

(3.70)

Let us consider these steps one at a time.

1. Premultiply

\[
R_X(\phi) = \begin{bmatrix}
    1 & 0 & 0 \\
    0 & c_\phi & -s_\phi \\
    0 & s_\phi & c_\phi
\end{bmatrix}
\]  

(3.71)

This rotates about the global +X axis, which is also the body +X axis.

2. Premultiply

\[
R_Y(\theta) = \begin{bmatrix}
    c_\theta & 0 & s_\theta \\
    0 & 1 & 0 \\
    -s_\theta & 0 & c_\theta
\end{bmatrix}
\]  

(3.72)

This rotates about the global +Y axis, which is also the body –Z axis.
3. Premultiply

\[ R_Z(\psi) = \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \]  

(3.73)

This rotates about the global +Z axis, which is also the body −X axis.

One can visually verify the computed rotation matrix by noting that its three column vectors match the three body-frame coordinate axes when expressed in the global frame. This configuration does not match the desired final configuration.

Current Frame. Rotations in the current frame are accomplished by postmultiplying. Thus,

\[ R^G_{B} = R^G_{X Y Z} = R_X(\phi)R_Y(\theta)R_Z(\psi) \]  

(3.78)

\[ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \]  

(3.79)
Let us again consider these steps one at a time.

1. Postmultiply

\[ R_X(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix} \tag{3.80} \]

This rotates about the body +X axis, which is also the global +X axis.

2. Postmultiply

\[ R_Y(\theta) = \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \tag{3.81} \]

This rotates about the body +Y axis, which is also the global +Z axis.

3. Postmultiply

\[ R_Z(\psi) = \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.82} \]
This rotates about the body $+Z$ axis, which is also the global $+X$ axis.

This time, we have arrived in the correct configuration, so we can conclude that this rotation is expressed in the current frame as

\[
\mathbf{p}_A^G = \mathbf{R}_{DB}^G \mathbf{p}_A^B = \mathbf{R}_{XYZ} \mathbf{p}_A^B
\]

\[
\mathbf{p}_A^B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_A^B
\]

\[
\mathbf{p}_A^B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_A^B
\]

\[
\mathbf{p}_A^B = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \mathbf{p}_A^B
\]

### 3.4 Properties of Rotations

#### 3.4.1 Algebraic Properties of Rotations

Recall that an algebraic group is a set with an operator. Rotations define a group under the composition operator, which means performing one rotation followed by another rotation. In two dimensions, the composition operator for angles is addition.
3.5. RIGID BODY MOTION

modulo $2\pi$, whereas for rotation matrices the composition operator is multiplication.

The rotation group under composition is called SO(2) in two dimensions and SO(3) in three dimensions. “SO” stands for the special orthogonal group. As with all groups, SO(2) and SO(3) have four properties:

- **Closure.** The composition of any two rotations is itself a valid rotation.
- **Identity.** There exists an element of the set of rotations, $I$, that when composed with any other rotation $R$ yields $R$. Thus, $IR = RI = R$.
- **Inverse.** For every element $R$ in the set of rotations, there exists an inverse element $R^{-1}$ such that when composed with $R$, the result is the identity. Thus, $RR^{-1} = R^{-1}R = I$.
- **Associativity.** When composing three rotations, we can simplify either pair first and get the same result. Thus, $ABC = (AB)C = A(BC)$.

Recall that the additional property of commutativity applies only in an Abelian group, thus $AB = BA$. As a general rule, rotation does not commute, and so SO(3) is not Abelian. That is, the order in which you apply rotations gives a different result in 3D.

We can observe that SO(2) is Abelian because real numbers (angles) commute over addition modulo $2\pi$. As a sanity check, we can confirm that 2D rotation matrices should also commute under multiplication. This is somewhat surprising because in general matrix multiplication does not commute, but interestingly the structure of 2D rotation matrices is special in a way that causes them to commute. We can see that this is true as follows:

\[
R(\alpha)R(\beta) = \begin{bmatrix}
\cos \alpha & -\sin \alpha \\
\sin \alpha & \cos \alpha
\end{bmatrix}
\begin{bmatrix}
\cos \beta & -\sin \beta \\
\sin \beta & \cos \beta
\end{bmatrix} = \begin{bmatrix}
\cos \alpha \cos \beta - \sin \alpha \sin \beta & -\cos \alpha \sin \beta - \sin \alpha \cos \beta \\
\cos \alpha \sin \beta + \sin \alpha \cos \beta & \cos \alpha \cos \beta - \sin \alpha \sin \beta
\end{bmatrix}
\]

(3.87)

\[
R(\beta)R(\alpha) = \begin{bmatrix}
\cos \beta & -\sin \beta \\
\sin \beta & \cos \beta
\end{bmatrix}
\begin{bmatrix}
\cos \alpha & -\sin \alpha \\
\sin \alpha & \cos \alpha
\end{bmatrix} = \begin{bmatrix}
\cos \alpha \cos \beta - \sin \alpha \sin \beta & -\cos \alpha \sin \beta - \sin \alpha \cos \beta \\
\cos \alpha \sin \beta + \sin \alpha \cos \beta & \cos \alpha \cos \beta - \sin \alpha \sin \beta
\end{bmatrix}
\]

(3.88)

3.5 Rigid Body Motion

As explained earlier, a rigid motion is in general the result of translation and rotation. So far we have focused on pure rotation. In this section we demonstrate how these concepts can be combined to fully describe the motion of a rigid body.
A rigid motion can be defined as a tuple \((d, R)\) where \(d \in \mathbb{R}^3\) and \(R \in SO(3)\) \((SO(3)\) is the group of all rotations about the origin of \(\mathbb{R}^3\) under the operation of composition\). The group of all rigid motions, also called the \textit{Special Euclidean Group}, is defined as \(SE(3) = \mathbb{R}^3 \times SO(3)\).

Consider the rotation matrix \(R_{01}\) that describes the orientation of a frame \(\{1\}\) with respect to a fixed frame \(\{0\}\). Let \(p^1\) denote the coordinates of a point \(p \in \mathbb{R}^3\) with respect to the frame \(\{1\}\) and \(d^0 \in \mathbb{R}^3\) be a position vector from the origin of \(\{0\}\) to the origin of \(\{1\}\). Then the coordinates of \(p\) with respect to \(\{0\}\) can be derived as follows:

\[
p^0 = R_{01}^1 p^1 + d^0.
\] (3.91)

Consider now three frames \(\{0\}, \{1\}, \text{ and } \{2\}\). Let \(d_{01}^0\) be the coordinates of the origin of frame \(\{1\}\) with respect to frame \(\{0\}\) and \(d_{12}^1\) be the coordinates of the origin of frame \(\{2\}\) with respect to frame \(\{1\}\). Then for a point \(p\) we can write the following equations:

\[
p^1 = R_{12}^1 p^2 + d_{12}^1
\] (3.92)

and

\[
p^0 = R_{01}^0 p^1 + d_{01}^0.
\] (3.93)

Composing these equations gives us the following expression for \(p^0\):

\[
p^0 = R_{01}^0 (R_{12}^1 p^2 + d_{12}^1) + d_{01}^0
\] (3.94)
\[
= R_{01}^0 R_{12}^1 p^2 + R_{01}^0 d_{12}^1 + d_{01}^0
\] (3.95)
\[
= R_{02}^0 p^2 + (d^0 + d_{12}^1)
\] (3.96)

Equation (3.96) demonstrates how a rigid motion can be decomposed into a rotation and a translation and introduces the main idea behind the \textit{homogeneous transformation}, a unified representation of a rigid motion consisting of a translation and a rotation.

3.5.1 The Homogeneous Transformation

The homogeneous transformation is a matrix representation of a rigid motion that allows us to reduce the composition of rigid motions into matrix multiplications. Consider a rotation matrix \(R_{1}^0\) representing the orientation of a frame \(\{1\}\) with respect to a frame \(\{0\}\) and a vector \(d_{1}^0\) describing the coordinates of the origin of \(\{1\}\) with respect to the origin of \(\{0\}\). The homogeneous transform describing the rigid motion (position and orientation) of the frame \(\{1\}\) with respect to frame \(\{0\}\) is defined as:

\[
T_{1}^0 = \begin{bmatrix} R_{1}^0 & d_{1}^0 \\ 0 & 1 \end{bmatrix}
\] (3.97)
3.5. RIGID BODY MOTION

where 0 is a zero row vector.

It can be shown that the inverse transformation, i.e., the transformation describing the rigid motion of frame \{0\} with respect to frame \{1\}, is given by the following equation:

\[
(T_0^1)^{-1} = T_0^1 = \begin{bmatrix}
(R_0^1)^T & -R_0^1 a_0^1 \\
0 & 1
\end{bmatrix}.
\] (3.98)

Homogeneous transformations can be used to perform rigid operations to position vectors in the same way that rotation matrices can be used to reorient vectors. However, since the homogeneous transformation is a matrix $4 \times 4$, we need to augment a given position vector $p$ as $P = \begin{bmatrix} p^T & 1 \end{bmatrix}^T$ in order to define the operation in the form of a multiplication. This augmented version of $p$ is called a homogeneous representation.

As an example, consider the homogeneous transform $T_1^0$ and a position vector $p^1$ which expresses the position of point $p$ with respect to the frame \{1\}. Upon augmenting $p^1$ to $P^1 = \begin{bmatrix} (p^1)^T & 1 \end{bmatrix}^T$, we can derive an expression for $P^0$ as:

\[
P^0 = T_1^0 P^1.
\] (3.99)

The composition of homogeneous transforms follows the same rules as the composition of rotations. Consider a homogeneous transformation $T_1^0$ relating frame \{1\} with frame \{0\} and a second transformation $T$ relative to the current frame. These transformations can be composed to form $T_2^0$ as:

\[
T_2^0 = T_1^0 T.
\] (3.100)

On the other hand, if $T$ is also expressed with respect to the fixed frame, then, as in the case of rotation with respect to the fixed frame, it can be shown that:

\[
T_2^0 = TT_1^0.
\] (3.101)

3.5.2 Long-Form Example

Example 3.6 Homogeneous Transforms – Document Viewer

Concepts reviewed: homogeneous transforms, expressing coordinates in a new frame, composing transforms, inverse of a transform.

Problem: Consider the diagram in Fig. 3.11. A document camera is centered one meter above a small object of negligible size that is centered on a square table. Find $H_0^1, H_1^2, H_3^1$. Using only these matrices, find $H_3^2$ – the position of the small object in the camera’s coordinate frame.
Solution: We note that coordinate frames \( \{0\} \) and \( \{1\} \) are aligned, allowing us to construct a pure translation:

\[
H^0_1 = \begin{bmatrix}
1 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\] (3.102)

For the remaining two homogeneous transform matrices, there are non-trivial rotation and translation components. These can be inferred from the diagram in two ways:

1. **By inspection.** Inspection works well only in special cases such as this, where the axes are aligned. Recall that the column vectors of a rotation matrix are the three coordinate frame axes of the child frame \( \{2\} \) in the
3.5. RIGID BODY MOTION

parent frame \{1\}.

\[
\begin{align*}
\hat{x}_2^1 &= \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^T \\
\hat{y}_2^1 &= \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^T \\
\hat{z}_2^1 &= \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T \\
R_2^1 &= \begin{bmatrix} -1 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1 \end{bmatrix}
\end{align*}
\] (3.103)

2. From the definition. Using the general approach,

\[
\begin{align*}
R_2^1 &= \begin{bmatrix} \hat{x}_1 \cdot \hat{x}_2 & \hat{x}_1 \cdot \hat{y}_2 & \hat{x}_1 \cdot \hat{z}_2 \\
\hat{y}_1 \cdot \hat{x}_2 & \hat{y}_1 \cdot \hat{y}_2 & \hat{y}_1 \cdot \hat{z}_2 \\
\hat{z}_1 \cdot \hat{x}_2 & \hat{z}_1 \cdot \hat{y}_2 & \hat{z}_1 \cdot \hat{z}_2 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & -1 \end{bmatrix}.
\end{align*}
\] (3.107)

One may confuse this matrix of dot products with its transpose. Recall that

\[
(R_B^G)^T = (R_B^G)^{-1} = R_B^G, \text{ so one must be careful. To resolve the confusion,}
\]

observe that the column that defines the \(\hat{y}_2\) axis contains \(\hat{y}_2\) at each element.

Having found the orientation of frame \(\{2\}\) in frame \(\{1\}\), we can now combine

it with the translation component \(\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix}^T\) to get

\[
H_2^1 = \begin{bmatrix} -1 & 0 & 0 & \frac{1}{2} \\
0 & -1 & 0 & \frac{1}{2} \\
0 & 0 & -1 & 1 \\
0 & 0 & 0 & 1 \end{bmatrix}.
\] (3.108)

We follow the same procedure to find \(H_3^1\). Using the dot product method,

\[
R_3^1 = \begin{bmatrix} \hat{x}_1 \cdot \hat{x}_3 & \hat{x}_1 \cdot \hat{y}_3 & \hat{x}_1 \cdot \hat{z}_3 \\
\hat{y}_1 \cdot \hat{x}_3 & \hat{y}_1 \cdot \hat{y}_3 & \hat{y}_1 \cdot \hat{z}_3 \\
\hat{z}_1 \cdot \hat{x}_3 & \hat{z}_1 \cdot \hat{y}_3 & \hat{z}_1 \cdot \hat{z}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\
-1 & 0 & 0 \\
0 & 0 & 1 \end{bmatrix}.
\] (3.109)

We can again construct the translation vector \(\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}^T\) by inspection. Thus,

\[
H_3^1 = \begin{bmatrix} 0 & 1 & 0 & \frac{1}{2} \\
-1 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \end{bmatrix}.
\] (3.110)
Finally, we wish to construct an expression for $H_3^2$ using only $H_0^1$, $H_1^2$, $H_2^3$. We can compose two homogeneous transform matrices to introduce an intermediate step, so that

$$H_3^2 = H_1^2 H_2^3,$$ (3.111)

but we do not know $H_1^2$ so we must invert the matrix we do have, yielding

$$H_3^2 = (H_2^1)^{-1} H_3^1.$$ (3.112)

Unlike a rotation matrix, we cannot simply take the transpose of a homogeneous transform to get its inverse. Instead, we can decompose a homogeneous transform into two separate matrices, which accomplish the rotation and the translation separately, as

$$H^{-1} = \begin{bmatrix} R_{3x3} & d \\ 0 & 1 \end{bmatrix}^{-1} = (H_{\text{trans}} H_{\text{rot}})^{-1}$$ (3.113)

$$= H_{\text{rot}}^{-1} H_{\text{trans}}^{-1}$$ (3.114)

$$= H_{\text{rot}}^T ( -H_{\text{trans}} ) = \begin{bmatrix} R_{3x3}^T & -R_{3x3}^T d \\ 0 & 1 \end{bmatrix}. $$ (3.115)

To understand the final step, consider that the translation is being performed in the \textit{current frame} – after the rotation is performed. Finally, we can substitute in the values from the matrices,

$$H_3^2 = (H_2^1)^{-1} H_3^1 = \begin{bmatrix} -1 & 0 & 0 & \frac{1}{2} \\ 0 & -1 & 0 & \frac{1}{2} \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & \frac{1}{2} \\ -1 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ (3.116)

$$= \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$ (3.117)

3.5.3 Relative Motions of Rigid Bodies

When a particle moves with respect to a base coordinate frame \{B\}, that motion is described by a translation in \{B\}. If a rigid body moves with respect to \{B\}, then we use a homogeneous transform $H_B$ to track both its rotation and its translation
3.6. FORWARD KINEMATICS FOR MANIPULATOR ARMS

in \{B\} because in general both are required. We have already seen that the same homogeneous transform \(H^B\) can be re-expressed in any coordinate frame, and we know that its parameters are a function of the choice of frame.

We observe that there are special-case rigid body motions \(H^B\) that are defined by pure rotation about the origin of \{B\}. Interestingly, it is always possible to find a base frame \{C\} such that any rigid body motion \(H^B\) is a pure rotation \(R^C\) in \{C\}. Noting that the choice of orientation of that frame is invariant to the fact that the specified motion is a pure rotation, we refer to the origin of \{C\} as a rotation center.

One common case in which a rotation center is important is the design of joints in a robot arm. Since a revolute joint forces one rigid body to perform a pure rotation with respect to another rigid body, any point on the joint axis is a rotation center of the rigid body when the joint moves.

In general, as a rigid body is moving, it need not travel in a perfect arc. In such case, the rotation center is not fixed. We refer to the rotation center with respect to a continuing rigid body motion at an instant in time as the instantaneous rotation center. This concept will be crucial to analyzing the motion of wheeled robots in Section 3.7.

3.6 Forward Kinematics for Manipulator Arms

This section covers manipulator kinematics and forward kinematic chains.

3.6.1 Configuration Space

The configuration of a robot is the vector of all its degrees of freedom. Each degree of freedom may have upper and lower limits due to the physical mechanism, or it may be unconstrained. The set of possible values of the configuration vector defines the configuration space of a robot.

Internal degrees of freedom are defined by the angles of joints; an additional set of degrees of freedom (up to six) describe the rigid body motion of the robot body as a whole. The configuration vector includes both internal and external degrees of freedom as applicable. For example, the KUKA youBot mobile manipulator robot has a mobile base that moves on the floor with three degrees of freedom as well as an arm with five joints for five internal degrees of freedom. Thus, the KUKA youBot has a total of eight degrees of freedom in its configuration.

Just as the coordinate frame of a rigid body defines the position of every particle within the body, the configuration of a robot similarly defines the position of every particle within the robot.
3.6.2 Link Parameters

A robot manipulator is composed of a series of links connected together by joints. The links are numbered from 0, which is the base of the manipulator, to \( n \), which is the free end of the manipulator. Although the joint angles define the degrees of freedom and thus the configuration of a robot, they alone are insufficient to specify where each of the links is located with respect to the ones before it in a kinematic chain. Since each link is a rigid body, its position in 3D with respect to the previous link is nominally defined by six parameters: one configuration variable and five constants. The Denavit-Hartenberg convention provides only four parameters, called Denavit-Hartenberg parameters (or DH parameters), that fully express the position of consecutive links in a kinematic chain.

The common normal of a link is the shortest distance orthogonal to the two joint axes that the link connects. Each link can be described by four link parameters (see Fig. 3.12):

1. \( a \), the link length, is the length of the common normal of the link,
2. \( \alpha \), the link twist, is the angle that exists between the joint axes when projected onto a plane whose normal is the common normal of the link,
3. \( d \), the link offset, is the distance between the intersections of the common normals of successive links with their common joint axis, and
4. \( \theta \), the joint angle, is the angle between the common normals of successive links.

In general, the joints in a manipulator have just one degree of freedom to simplify the mechanical design and control. Most manipulators have rotating joints called revolute joints or sliding joints called prismatic joints. Since these types of joints have one degree of freedom, three of the four link parameters are fixed for a given link while the fourth is variable and is called the joint variable. The joint variable for a revolute joint is \( \theta \) and for a prismatic joint is \( d \). Determining the cumulative effect the entire set of joint variables is the objective of forward kinematics; that is, we solve the forward kinematics problem to determine the position and orientation of the end effector given the values of the joint variables.

3.6.3 Link Frames

In order to solve the forward kinematics problem, we assign a frame to each link in the following manner:
3.6. FORWARD KINEMATICS FOR MANIPULATOR ARMS

Figure 3.12: Link parameters are used to characterize the relationship between the two joints connected by a link.

1. Assign the origin of the link frame \( \{ i \} \) at the point where the common normal of link \( i \) meets the \( i \)th joint axis.

2. Assign axis \( \hat{z}_i \) to be coincident with the \( i \)th joint axis.

3. Assign axis \( \hat{x}_i \) to be parallel to the common normal of link \( i \).

4. Assign axis \( \hat{y}_i \) to complete a right-hand coordinate system.

Finally, assign the coordinate frame \( \{ 0 \} \), i.e. the coordinate frame of the base, to be parallel to frame \( \{ 1 \} \) when the first joint variable is zero.

**Example 3.7** Determining link frames and parameters

Fig. 3.13 shows a planar arm with three revolute joints. We wish to assign coordinate frames to the links of this manipulator and write down the link parameters.

To do this, we start by identifying the joint axes of the manipulator. Since the arm lies in a two-dimensional plane, the joint axes are all parallel and point out of the paper (and thus all the \( \hat{z} \)-axes point out of the paper as well). The common normal of each link is then directly along the link, so each \( \hat{x} \)-axis is assigned to point along the corresponding link. Finally, the \( \hat{y} \)-axes are added consistent with the right-hand rule to complete each link frame.

Note that the placement of the origin of \( \{ 0 \} \) in Fig. 3.13a was chosen arbitrarily. The origin could be fixed to the very bottom of the base, or it could coincide with the origin of frame \( \{ 1 \} \). The key is that \( \{ 0 \} \) is attached to the non-moving base of the robot and that the axes of \( \{ 0 \} \) and \( \{ 1 \} \) are parallel when \( \theta_1 = 0 \).
CHAPTER 3. KINEMATICS

3.6.4 Link Transformations

Once each link has been assigned a frame, we would like to define a transform that relates frame \( \{i\} \) to frame \( \{i - 1\} \). To do this, we first consider four smaller transformations, one for each link parameter, that transform \( \{i - 1\} \) into \( \{i\} \) when

\[
\begin{array}{cccccc}
  i & \alpha_i & \alpha_i & d_i & \theta_i \\
  1 & \ell_1 & 0 & 0 & \theta_1 \\
  2 & \ell_2 & 0 & 0 & \theta_2 \\
  3 & \ell_3 & 0 & 0 & \theta_3 \\
\end{array}
\]

Figure 3.14: The link parameters of the manipulator.
chained together. We will label the three intermediate frames between \( \{i-1\} \) and \( \{i\} \) as \( \{R\}, \{Q\}, \) and \( \{P\} \). The transformation steps are as follows:

1. Rotate \( \{i-1\} \) about the common normal of link \( i-1 \) by an angle of \( \alpha_{i-1} \) to get frame \( \{R\} \).
2. Translate \( \{R\} \) along the common normal of link \( i-1 \) by a distance of \( a_{i-1} \) to get frame \( \{Q\} \).
3. Rotate \( \{Q\} \) about the \( i \)th joint axis by an angle of \( \theta_i \) to get frame \( \{P\} \).
4. Translate \( \{P\} \) along the \( i \)th joint axis by a distance of \( d_i \) to get frame \( \{i\} \).

Next, we write down the homogeneous transform for each step respectively:

\[
T_{i-1}^R = \begin{bmatrix} R_\hat{x}(\alpha_{i-1}) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c(\alpha_{i-1}) & -s(\alpha_{i-1}) & 0 \\ 0 & s(\alpha_{i-1}) & c(\alpha_{i-1}) & 0 \end{bmatrix}, \quad (3.118)
\]

\[
T_{Q}^R = \begin{bmatrix} I_3 & [a_{i-1} & 0 & 0]^T \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad (3.119)
\]

\[
T_{Q}^P = \begin{bmatrix} R_\hat{z}(\theta_i) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c(\theta_i) & -s(\theta_i) & 0 & 0 \\ s(\theta_i) & c(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad (3.120)
\]

\[
T_i^P = \begin{bmatrix} I_3 & [0 & 0 & d_i]^T \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \end{bmatrix}, \quad (3.121)
\]

where \( c(x) = \cos x \) and \( s(x) = \sin x \). Now we can express the transform from \( \{i\} \) to \( \{i-1\} \) as

\[
T_{i}^{i-1} = T_{i-1}^{i-1} T_{Q}^{R} T_{P}^{Q} T_{i}^{P}, \quad (3.122)
\]

which multiplies out to the following general form:

\[
T_{i}^{i-1} = \begin{bmatrix} c(\theta_i) & -s(\theta_i) & 0 & a_{i-1} \\ s(\theta_i)c(\alpha_{i-1}) & s(\theta_i)c(\alpha_{i-1}) & -s(\alpha_{i-1}) & -s(\alpha_{i-1})d_i \\ s(\theta_i)s(\alpha_{i-1}) & c(\theta_i)s(\alpha_{i-1}) & c(\alpha_{i-1}) & c(\alpha_{i-1})d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (3.123)
\]
Since we now have a transform that relates the frames attached to neighboring links, we can multiply together \( n \) of these transformations to relate frame \( \{n\} \) to frame \( \{0\} \). Hence, a single transformation that relates \( \{n\} \) to \( \{0\} \) is
\[
T^0_n = T^0_1 T^1_2 T^2_3 \ldots T^{n-1}_n,
\]
which is a function of all \( n \) joint variables. Thus, if we obtain the values of the entire set of joint variables from the robot’s sensors, we can solve for the position and orientation of the end effector relative to the base using \( T^0_n \).

### 3.7 Mobile Robot Kinematics

A kinematic model of a wheeled mobile robot allows us to determine how the robot moves based on the geometry of constraints imposed by its wheels. To find tractable equations for the model, we make several assumptions:

1. The robot is a single rigid body.
2. The robot moves on a smooth, flat surface.
3. No translational *slip* occurs between the robot’s wheels and the floor; that is, the torque applied to the wheels does not exceed the available traction.
4. Wheels only spin about their central axis and – if they are steering wheels – about the \( \hat{z} \)-axis, which is perpendicular to the floor.

We write the configuration of a wheeled mobile robot as \( q = [x \ y \ \theta]^T \). The *freedoms* of the robot describe allowed motion, whereas the *constraints* describe unallowed motion. We express the freedoms as
\[
\dot{q} = \sum_i g_i(q) u_i,
\]
where the functions \( g_i(q) \in \mathbb{R}^3 \) are the velocity vector fields over \( q \) and the \( u_i \) terms are the control inputs. The constraints are written as
\[
\forall i, w_i(q) \cdot \dot{q} = 0,
\]
thus prohibiting all motion in each of the directions \( w_i \).

Note that the freedoms of a mobile robot are collective, meaning that any linear combination of the \( u_i \) terms is an allowable motion, whereas the constraints are
3.7. MOBILE ROBOT KINEMATICS

A constraint is called holonomic (or integrable) if it depends only on the configuration \( q \) and not its derivative \( \dot{q} \); that is, it must be expressible as a function

\[
f(q) = 0. \tag{3.127}
\]

A constraint that does not meet the above criteria is called nonholonomic. If all of the constraints of a system are holonomic, it is considered a holonomic system; any other system is a nonholonomic system.

Let us consider two simple cases of mobile robots.

3.7.1 Unicycle model

The most basic mobile robot is a unicycle (see Fig. 3.15). The point of contact between the unicycle and the ground is \((x, y)\), and its heading direction is \(\theta\).

There are two control inputs to the unicycle: \(u_1\), which is the forward-backward driving speed, and \(u_2\), which is the heading direction turning speed. The freedoms associated with these controls are

\[
g_1(q) = \begin{bmatrix} \cos \theta \\ \sin \theta \\ 0 \end{bmatrix} \quad \text{and} \quad g_2(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{3.128}
\]

The combined freedoms give

\[
\dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 & u_1 \\ \sin \theta & 0 & u_2 \end{bmatrix} = \begin{bmatrix} u_1 \cos \theta \\ u_1 \sin \theta \\ u_2 \end{bmatrix}. \tag{3.129}
\]
The unicycle’s single motion constraint is

\[ w_1(q) = \begin{bmatrix} \sin \theta \\ -\cos \theta \\ 0 \end{bmatrix}, \tag{3.130} \]

meaning that the unicycle cannot move sideways. Written out,

\[ \dot{x} \sin \theta - \dot{y} \cos \theta = 0. \tag{3.131} \]

Since this constraint cannot be integrated (it has no expression in terms of only \( q \)), it is nonholonomic, and therefore the unicycle is a nonholonomic system.

Another way to check whether a system is holonomic or nonholonomic is to compare the number of controllable degrees of freedom to the total degrees of freedom. If the number of controllable degrees of freedom is less than the total degrees of freedom, the system is nonholonomic. If the number of controllable degrees of freedom is equal to the total degrees of freedom, the system is holonomic.

In the case of the unicycle, it has two controls but three degrees of freedom (its \( x \)-position, its \( y \)-position, and its heading direction) and thus is nonholonomic.

### 3.7.2 Unsteered cart

The cart shown in Fig. 3.16 has four wheels, all of which are fixed to the body of the cart. It therefore has no ability to steer.

![Unsteered cart](image)

Figure 3.16: An unsteered cart cannot change its heading direction.

Since the cart cannot turn, its only control input is the forward-backward driving speed. Hence, its freedoms are

\[ \dot{q} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \\ 0 \end{bmatrix} u_1 = \begin{bmatrix} u_1 \cos \theta \\ u_1 \sin \theta \\ 0 \end{bmatrix}, \tag{3.132} \]
and its constraints are

\[
    w_1(q) = \begin{bmatrix} \sin \theta \\ - \cos \theta \\ 0 \end{bmatrix} \quad \text{and} \quad w_2(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.
\] (3.133)

Thus, the constraint expressions are

\[
    \dot{x} \sin \theta - \dot{y} \cos \theta = 0 \tag{3.134}
\]
\[
    \dot{\theta} = 0, \tag{3.135}
\]

which can be integrated to a derivative-free form:

\[
    (x - x_0) \sin \theta_0 - (y - y_0) \cos \theta_0 = 0 \tag{3.136}
\]
\[
    \theta = \theta_0. \tag{3.137}
\]

Since these constraints can be integrated, they are both holonomic, and thus the unsteered cart is a holonomic system. Each holonomic constraint reduces the system’s total degrees of freedom by one, so although the cart has an \(x\)-position, a \(y\)-position, and a heading direction \(\theta\), it only has one degree of freedom. Intuitively, we can see that the cart must always remain on a single line in the configuration space because it cannot turn, therefore restricting its configuration space to a single degree of freedom. Finally, note that the number of controls (one) is equal to the total degrees of freedom (one), which confirms that unsteered cart is holonomic.

### 3.7.3 Mobile Robot Motion

We can analyze the motion of a wheel by observing the pattern of freedoms and constraints. A wheel can move forward or backward, but it resists side-to-side motion. It is illuminating to look at the situation in terms of possible instantaneous rotation centers of the wheel, as in Fig. 3.17. The nonholonomic constraint of the wheel permits an instantaneous rotation center at any point along the line perpendicular to the direction of travel. A rotation center at the center of the wheel would correspond to turning in place. Any other point would result in a curved path. Rotation about the point at infinity (in either direction) corresponds to a straight motion of the wheel with no turning.

Unicycles are hard to control because there is no kinematic means to select which instantaneous rotation center to follow. In a bicycle (Fig. 3.17b), the wheels’ perpendiculars always meet at exactly one point that kinematically defines the steering direction of the bicycle. In this analysis, we consider two parallel lines to meet at infinity. Thus, when the wheels are parallel, the bicycle travels straight.
The analysis by instantaneous rotation centers gives us new insight into why the unsteered cart in Fig. 3.16 can only move in a straight line, since the four lines of instantaneous rotation centers meet only at infinity (Fig. 3.17c). Note that a translation is equivalent to a rotation about a point at infinity. There is a class of mobile robots called *skid-steered vehicles* that are built with four fixed wheels, like the unsteered cart, for ruggedness. They steer by violating the nonholonomic constraint of one or more wheels. Such vehicles generally traverse rough terrain like mud and rocks where there is not always good contact with the ground.

One can imagine that a two-wheeled variant of the cart would be steerable without slip provided that the two wheels share a common perpendicular axis (Fig. 3.17d). This layout is the differential drive robot steering design.

### 3.8 Inverse Kinematics

#### 3.8.1 Manipulator Inverse Kinematics

In the previous section, we discussed how to solve for the position and orientation of a manipulator’s end effector given the values of its joint variables. We will now consider the inverse of this problem, which is to find values of the joint variables that place the end effector at a desired position and orientation in space. In general, this problem of *inverse kinematics* is more difficult than forward kinematics since the equations we must solve are often complicated and nonlinear, and there may exist many solutions or no solutions.

**Solution Existence and Workspace**

In order for at least one solution to exist, the desired goal point must lie within the manipulator’s *reachable workspace*, which is the volume of space that the robot can reach in at least one orientation. Depending on the manipulator, there may also be a subset of this space called the *dexterous workspace*, which is all the points that the robot can reach from any arbitrary orientation of its end effector.

Consider the two-link manipulator shown in Fig. 3.18. Assuming that the manipulator can rotate its joints a full 360 degrees, the reachable workspace of this manipulator is a disk where the outer radius is the sum of its two link lengths and the inner radius is the difference between its two link lengths. Does this manipulator have any dexterous workspace? As shown in Fig. 3.18a, the manipulator can reach a point on the boundary of its workspace from only one orientation, so the boundaries of the disk cannot constitute dexterous workspace. Fig. 3.18b shows that a point inside of the manipulator’s workspace can be reached from exactly two orientations. Thus, no point in this manipulator’s workspace can be
3.8. **INVERSE KINEMATICS**

(a) A single wheel is capable of rotating and rolling forward or back. It is constrained against sliding sideways by friction. The set of instantaneous rotation centers consistent with the two wheel freedoms is a line perpendicular to the direction of rolling through its center.

(b) A bicycle is two connected wheels, one of which is fixed whereas the other can steer. They are rigidly joined together (red line). The bicycle steers by controlling the instantaneous rotation center that is consistent with both wheels; the two perpendiculars intersect at exactly one point.

(c) The perpendiculars of the unsteered four-wheeled cart form parallel lines that meet at infinity. Thus, the four-wheeled cart can only move in a straight line (a translation can be regarded as a rotation about a point at infinity).

(d) The two-wheeled cart permits steering by turning the wheels at different rates. Thus, it is generally referred to as differential drive.

Figure 3.17: Instantaneous rotation centers of wheels can be any point along the perpendicular.
reached from any arbitrary orientation of the end effector, so the manipulator has no dexterous workspace. Note, however, that if the two links were the same length, the workspace would be a circle, and there would be a single point of dexterous workspace: the origin of the circle.

Figure 3.18: On the boundary of its reachable workspace, this two-link manipulator can reach a point from only one orientation, as shown in (a). Inside of its reachable workspace, two orientations per point are possible: one with the manipulator’s elbow up, the other with its elbow down, as shown in (b). The “elbow down” orientation is given in dashed lines.

Multiple Solutions

Although the two-link manipulator above can reach points on the interior of its workspace from two orientations, there is no point it can reach from multiple joint configurations with the same position and orientation of its end effector. Adding a third link would allow for multiple solutions of this type (see Fig. 3.19).

Figure 3.19: A three-link manipulator that can position and orient its end effector the same way with different joint parameters values. An alternate solution is shown in dashed lines.
3.8. INVERSE KINEMATICS

The number of solutions to an inverse kinematics problem is a function of the manipulator’s link parameters and the range of motion of each joint. In general, the more nonzero link parameters, the more solutions there will be. While having multiple solutions (and thus multiple ways to reach a goal) may be beneficial to the task at hand, it may also be problematic since the system must be able to choose one solution. The selection criteria vary, but reasonable factors to take into account include potential obstacle collision and how far each joint will need to move.

Solution Methods

Whereas the forward kinematics problem can always be solved with the method described in the previous section, there are no general algorithms for the inverse kinematics problem that guarantee a solution if one exists. The strategies for solving the inverse kinematics problem can be divided into closed-form solutions, which are exact, and numerical solutions, which are approximate. Closed-form solutions are typically preferred since they are often much faster than iterative numerical searches, and they also ease the process of developing of rules to choose among multiple solutions. We therefore will focus our attention on two classes of closed-form solutions: algebraic solutions and geometric solutions.

Example 3.8 Solving the inverse kinematics problem algebraically

Let us return to the three-link planar manipulator from the previous section for which we gave the link parameters (see Fig. 3.20). We can use our method for solving the forward kinematics problem to obtain the transforms

\[
T_1^0 = \begin{bmatrix}
c_1 & -s_1 & 0 & 0 \\
 s_1 & c_1 & 0 & 0 \\
 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 \\
\end{bmatrix}
, \quad T_2^1 = \begin{bmatrix}
c_2 & -s_2 & 0 & \ell_1 \\
 s_2 & c_2 & 0 & 0 \\
 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 \\
\end{bmatrix}
, \quad T_3^2 = \begin{bmatrix}
c_3 & -s_3 & 0 & \ell_2 \\
 s_3 & c_3 & 0 & 0 \\
 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 \\
\end{bmatrix}
, \quad (3.138)
\]

which we then multiply together to get the kinematic equations

\[
T_1^0 T_2^1 T_3^2 = T_3^0 = \begin{bmatrix}
c_{123} & -s_{123} & 0 & \ell_1 c_1 + \ell_2 c_{12} \\
 s_{123} & c_{123} & 0 & \ell_1 s_1 + \ell_2 s_{12} \\
 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 \\
\end{bmatrix}
, \quad (3.138)
\]

where \(c_1 = \cos(\theta_1), s_{12} = \sin(\theta_1 + \theta_2),\) and so forth.

Since the manipulator is planar, goal points for the end effector can most easily be specified with an \((x, y)\)-position and an orientation angle \(\phi\). Thus, all attainable
goals must lie within the subspace implied by the transformation

\[
T_3^0 = \begin{bmatrix}
c_\phi & -s_\phi & 0 & x \\
s_\phi & c_\phi & 0 & y \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]  \hspace{1cm} (3.139)

Equating (3.138) and (3.139) gives us the nonlinear equations which must be solved for the joint parameters \(\theta_1\), \(\theta_2\), and \(\theta_3\):

\[
c_\phi = c_{123}, \hspace{2cm} (3.140)
\]
\[
s_\phi = s_{123}, \hspace{2cm} (3.141)
\]
\[
x = \ell_1 c_1 + \ell_2 c_{12}, \hspace{2cm} (3.142)
\]
\[
y = \ell_1 s_1 + \ell_2 s_{12}. \hspace{2cm} (3.143)
\]
To solve these equations, we first square (3.142) and (3.143) and add them to get
\[
x^2 + y^2 = (\ell_1 c_1 + \ell_2 c_{12})^2 + \ell_1 s_1 + \ell_2 s_{12}
\]
\[
= \ell_1^2 (c_1^2 + s_{12}^2) + \ell_2^2 (c_{12}^2 + s_{12}^2) + 2 \ell_1 \ell_2 (c_1 c_{12} + s_1 s_{12})
\]
\[
= \ell_1^2 (1) + \ell_2^2 (1) + 2 \ell_1 \ell_2 (c_1 c_{12} + s_1 s_{12})
\]
\[
= \ell_1^2 + \ell_2^2 + 2 \ell_1 \ell_2 (c_1 c_{12} + s_1 s_{12})
\]
\[
= \ell_1^2 + \ell_2^2 + 2 \ell_1 \ell_2 (c_1 c_{12} + s_1 s_{12})
\]
\[
= \ell_1^2 + \ell_2^2 + 2 \ell_1 \ell_2 (c_1 c_{12} + s_1 s_{12})
\]
\[
= \ell_1^2 + \ell_2^2 + 2 \ell_1 \ell_2 (c_1 c_{12} + s_1 s_{12})
\]
\[
(3.144)
\]
using the trigonometric Pythagorean identity and angle sum identities.

Next, we solve (3.144) for \(c_2\), which yields
\[
c_2 = \frac{x^2 + y^2 - \ell_1^2 - \ell_2^2}{2 \ell_1 \ell_2}.
\]
(3.145)
The right-hand side of this equation must have a value between \(-1\) and \(1\) in order for a solution to exist; otherwise, the goal point is out of the manipulator’s reach. Assuming that the goal falls within the workspace, we use the Pythagorean identity to write an expression for \(s_2\),
\[
s_2 = \pm \sqrt{1 - c_2^2},
\]
(3.146)
where the choice of sign corresponds to the elbow-up and elbow-down solutions. Lastly, we use the two-argument arctangent to write an expression for \(\theta_2\),
\[
\theta_2 = \text{atan2}(s_2, c_2).
\]
(3.147)

Now that we have obtained \(\theta_2\), we can solve (3.142) and (3.143) for \(\theta_1\). Since \(c_1\) and \(c_2\) are the only unknowns, we rewrite the equations to isolate them, as
\[
x = \ell_1 c_1 + \ell_2 (c_1 c_2 - s_1 s_2) = c_1 (\ell_1 + \ell_2 c_2) - s_1 (\ell_2 s_2) = k_1 c_1 - k_2 s_1
\]
(3.148)
\[
y = \ell_1 s_1 + \ell_2 (c_1 s_2 + s_1 c_2) = s_1 (\ell_1 + \ell_2 c_2) + c_1 (\ell_2 s_2) = k_1 s_1 + k_2 c_1
\]
(3.149)
where
\[
k_1 = \ell_1 + \ell_2 c_2,
\]
\[
k_2 = \ell_2 s_2,
\]
(3.150)
are constants. If we then define
\[
r = \sqrt{k_1^2 + k_2^2}
\]
(3.151)
and
\[ \gamma = \atan2(k_2, k_1), \] (3.152)
we can rewrite the constants as
\[ k_1 = r \cos \gamma, \]
\[ k_2 = r \sin \gamma, \] (3.153)
which allows us to write (3.148) and (3.149) as
\[ \frac{x}{r} = \cos \gamma \cos \theta_1 - \sin \gamma \sin \theta_1, \] (3.154)
\[ \frac{y}{r} = \cos \gamma \sin \theta_1 + \sin \gamma \cos \theta_1, \] (3.155)
or, using the angle sum identities,
\[ \cos(\gamma + \theta_1) = \frac{x}{r}, \] (3.156)
\[ \sin(\gamma + \theta_1) = \frac{y}{r}. \] (3.157)
Finally, we use the two-argument arctangent to write
\[ \gamma + \theta_1 = \atan2\left(\frac{y}{r}, \frac{x}{r}\right) = \atan2(y, x), \] (3.158)
and so we subtract \( \gamma \) from both sides to get
\[ \theta_1 = \atan2(y, x) - \atan2(k_2, k_1). \] (3.159)

If \( x \) and \( y \) are both zero, then (3.159) is undefined, which makes \( \theta_1 \) arbitrary. Whenever \( \theta_1 \) is defined, its sign is determined by the choice of \( \theta_2 \) above since the sign of \( \theta_2 \) impacts the sign of \( k_2 \), therefore affecting the sign of \( \theta_1 \).

Lastly, we need to solve for \( \theta_3 \). Given (3.140) and (3.141), we can solve for the sum \( \theta_1 + \theta_2 + \theta_3 \) as
\[ \theta_1 + \theta_2 + \theta_3 = \atan2(s_\phi, c_\phi) = \phi, \] (3.160)
from which we can obtain \( \theta_3 \) since the first two angles are known.

**Example 3.9** Solving the inverse kinematics problem geometrically

Consider the three-link planar manipulator of Fig. 3.21. As before, the dashed lines indicate an alternate solution that would place the manipulator’s end effector in the same position and orientation as the one shown. The solid diagonal connects the origin of frame \( \{0\} \) to the origin of frame \( \{3\} \) and creates a triangle with the first
two links of the arm. We will solve for the joint variables using the rules of planar geometry. Note that in the more general case where the arm is non-planar, we can still use planar geometry to solve the inverse kinematics problem by dividing the spatial geometry of the arm into smaller planar geometry problems.

To begin, observe that we know all three side lengths of the triangle formed by the solid line and the arm: \( \ell_1, \ell_2 \), and \( \sqrt{x^2 + y^2} \) by the Pythagorean theorem. Planar geometry tells us that in order for this triangle to exist, \( \sqrt{x^2 + y^2} \) must be less than or equal to \( \ell_1 + \ell_2 \), the sum of the link lengths. If this constraint is not met, then the goal point \((x, y)\) is not within the manipulator’s workspace.

Assuming that the goal point is reachable, we use the law of cosines to determine the angle opposite the solid diagonal and solve for \( \theta_2 \). Recall that a joint angle \( \theta \) is defined as the angle between the common normals of successive links; thus, \( \theta_2 \) is the angle between the common normals of links 1 and 2. If \( \theta_2 \) were 0, then the angle formed by links 1 and 2 would be 180° (see Fig. 3.22a). Hence, in general, the angle formed by links 1 and 2 is 180° + \( \theta_2 \), as shown in Fig. 3.22b.

The law of cosines tells us that

\[
x^2 + y^2 = \ell_1^2 + \ell_2^2 - 2\ell_1\ell_2 \cos(180^\circ + \theta_2),
\]

which we may rewrite as

\[
\cos(\theta_2) = \frac{x^2 + y^2 - \ell_1^2 - \ell_2^2}{2\ell_1\ell_2}
\]

using the fact that \( \cos(180^\circ + \theta_2) = -\cos(\theta_2) \). We solve (3.162) only for values of \( \theta_2 \) between 0 and \(-180^\circ \) since the triangle would not exist otherwise. Note that we may find the alternate solution of Fig. 3.21 by multiplying \( \theta_2 \) by \(-1\).

Now we solve for \( \theta_1 \) by finding expressions for the angles \( \beta \) and \( \psi \) shown in Fig. 3.21. Since \( \beta \) could be in any quadrant (depending on the signs of \( x \) and \( y \)),
we must use the two-argument arctangent:

$$\beta = \text{atan2}(y, x).$$  \hfill (3.163)

Another application of the law of cosines gives us an expression for $\psi$,

$$\cos \psi = \frac{x^2 + y^2 + \ell_1^2 - \ell_2^2}{2\ell_1 \sqrt{x^2 + y^2}},$$  \hfill (3.164)

which we solve only for values of $\psi$ between 0 and $180^\circ$ to preserve the given geometry. Then we have

$$\theta_1 = \begin{cases} 
\beta + \psi & \text{if } \theta_2 < 0, \\
\beta - \psi & \text{if } \theta_2 > 0, \\
\beta & \text{otherwise}.
\end{cases}$$  \hfill (3.165)

Since angles in a plane add, we can compute the orientation of the last link as

$$\theta_1 + \theta_2 + \theta_3 = \phi,$$  \hfill (3.166)

which we can then solve for $\theta_3$ to complete the solution.

### 3.8.2 Mobile Robot Inverse Kinematics

For mobile robots, the inverse kinematics problem is the boundary value problem. Given a start and end pose, how does the robot find a path from one to the other?
In its most general form, this is the motion planning problem because it includes obstacle avoidance. That problem is highly complex and is not addressed in detail here. An interesting special-case is the Dubins car, which is a bounded turning-radius car that moves forward at a constant velocity in a plane without obstacles.

Dubins found that to compute shortest paths from start to goal pose, it is sufficient to only let the car to trace out path segments that are straight lines (S) or maximum curvature to the right (R) or left (L). It is assumed that the car instantaneously changes curvature between path segments (perhaps by stopping while steering). Since only three different curvatures are ever needed, it is possible to consider combinatorial sequences of segments.

If we consider the categories of possible path sequences, we find that there are only six of them: RLR, LRL, LSL, RSR, LSR, RSL. Any other sequence of curves is not optimal. The selection of a sequence depends primarily on the distance between the start and goal poses, with the straight segments arising only when the distance is greater than some threshold value. One can compute the shortest path complying with each path sequence fairly readily, and then the shortest legal path among them can be returned.

3.9 Velocity Kinematics

In the previous two sections, we discussed how to relate the position and orientation of a manipulator’s end effector to the values of its joint variables. In this section, we go a step further to investigate the relationship of the velocity of an end effector to the velocities of its joints. We define linear and angular velocity and show how to study the velocity of a rigid body as the motion of coordinate frames relative to each other. We describe skew-symmetric matrices as well as a matrix fundamental to velocity kinematics called the Jacobian.

3.9.1 Linear and Angular Velocity

Recall that the position of a particle $A$ with respect to a frame $\{B\}$ is given by the position vector $P_A^B$. We define the derivative of this vector relative to $\{B\}$ as

$$V_{P_A^B}^B = \frac{d}{dt} P_A^B = \lim_{\Delta t \to 0} \frac{P_A^B(t + \Delta t) - P_A^B(t)}{\Delta t},$$

which can be thought of as the linear velocity of $A$. It is important to specify the frame in which the differentiation is done since the position vector may vary in time differently with respect to different frames. For example, if $\{B\}$ is fixed to $A$, then $V_{P_A^B}^B$ will be zero since $P_A^B$ never varies in time with respect to $\{B\}$. 


Chapter 3. Kinematics

Whereas linear velocity is the property of a single point, angular velocity is a property of the attached coordinate frame. The angular velocity of a frame \( \{ B \} \) rotating with respect to frame \( \{ G \} \) is denoted as

\[
\Omega_{\text{G} \to \text{B}} = u \frac{d\theta}{dt}.
\]  
(3.168)

Here, \( u \) is a unit vector in the direction of the axis of rotation, and \( \theta \) is the angle between \( u \) and a perpendicular from any point of the body. Thus, at any moment in time, the direction of \( \Omega_{\text{G} \to \text{B}} \) gives the instantaneous axis of rotation of \( \{ B \} \) relative to \( \{ G \} \), and the magnitude of \( \Omega_{\text{G} \to \text{B}} \) gives the instantaneous speed of rotation.

3.9.2 Linear and Angular Velocity of Rigid Bodies

Through the analysis of moving coordinate frames, we will use our definitions from above to study the motion of a rigid body induced by its velocity. We will first look at linear velocity and angular velocity independently, and then we will combine them to describe the simultaneous linear and angular velocity of a rigid body.

Linear velocity

Consider a frame \( \{ B \} \) attached at point \( O_B \) to a rigid body. The position of a point \( A \) is given by the vector \( P_{A}^{B} \). We would like to describe the motion of \( A \) with respect to the global-fixed frame \( \{ G \} \) (see Fig. 3.23).

![Figure 3.23](image)

Figure 3.23: Frame \( \{ B \} \) is translating with respect to frame \( \{ G \} \).

If we assume the relative orientation of \( \{ G \} \) and \( \{ B \} \) remains constant, then all we need to do is take the derivatives of the two vectors in Fig. 3.23 with respect to \( \{ G \} \) and sum them. We carry out the differentiation as follows:
1. The first vector is $P_G^B$, which gives the position of the origin of \{B\} relative to \{G\}. Its derivative is therefore $V_G^B$, but to reduce the number of subscripts we must write, we will use the simplified notation $V_G^B$ to describe the velocity of the origin of frame \{B\} relative to frame \{G\}.

2. The second vector is $P_B^A$. Since this vector is expressed relative to \{B\}, we need to include the rotation matrix that accomplishes the change to frame \{G\}, yielding the derivative $R_B^G V_B^P_B$.

Hence, the linear velocity of $A$ is

$$V_{PA}^G = V_B^G + R_B^G V_B^P_B,$$

under the assumption that $R_B^G$ does not change with time.

**Angular velocity**

Suppose we have a body-fixed frame \{B\} with origin coincident with the origin of the global-fixed frame \{G\} and with zero linear velocity (so that its origin always stays coincident). Like before, the position of a point $A$ is given by the vector $P_A^B$. As Fig. 3.24 shows, \{B\} is rotating relative to \{G\} with angular velocity $\Omega_B^G$. We wish to determine how point $A$ moves with respect to \{G\}.

![Diagram](image)

Figure 3.24: The orientation of frame \{B\} relative to frame \{G\} is varying over time.

To do this, we can make use of the fact that the linear velocity of $A$ relative to \{G\} is the cross product of the angular velocity $\Omega_B^G$ and the position vector $P_A^G$ (see Fig. 3.25). That is,

$$V_{PA}^G = \Omega_B^G \times P_A^G.$$  \hspace{1cm} (3.170)

Since we need a change in frame, we use the appropriate rotation matrix to write

$$V_{PA}^G = \Omega_B^G \times R_B^G P_A^B.$$  \hspace{1cm} (3.171)
which holds when we assume that $P_B^A$ does not vary in time.

Figure 3.25: The linear velocity of point $A$ relative to frame $\{G\}$ is induced by the angular velocity of frame $\{B\}$ relative to $\{G\}$ and is computed as a cross product.

**Simultaneous linear and angular velocity**

Now we will remove our previous assumptions to address simultaneous linear and angular velocity. If frame $\{B\}$ is both translating (as in Fig. 3.23) and rotating (as in Fig. 3.24) with respect to frame $\{G\}$, then by adding (3.169) and (3.171) we can derive the general formula for the velocity of a point $A$ located with respect to $\{B\}$ as seen from $\{G\}$:

$$V_{P_A}^G = V_B^G + R_B^G V_{P_B}^G + \Omega_B^G \times R_B^G P_A^B.$$  \hspace{1cm} (3.172)

Another way to interpret (3.172) is that it is the derivative of a vector fixed in a moving frame as seen from a stationary frame.

**3.9.3 Skew-symmetric matrices**

Before we discuss how to compute velocity transformations for manipulator arms, we will first introduce a type of matrix that will allow us to simplify the transformation calculations. The type of matrix in question is called a **skew-symmetric matrix** and is defined as a square matrix with the property $S^T + S = 0$; that is, the transpose of the matrix equals its negative. For example, the following $4 \times 4$ matrix is skew-symmetric:

$$A = \begin{bmatrix} 0 & -3 & -1 & 7 \\ 3 & 0 & 4 & -8 \\ 1 & -4 & 0 & -2 \\ -7 & 8 & 2 & 0 \end{bmatrix}.$$
3.9. VELOCITY KINEMATICS

The skew-symmetric matrix has a very important relationship with the derivative of a rotation matrix. Suppose that a rotation matrix $R$ is a function of a single variable $\theta$. Since $R$ is an orthogonal matrix, we have

$$RR^T = I. \quad (3.173)$$

If we differentiate both sides of (3.173) with respect to $\theta$, we get

$$\left[ \frac{d}{d\theta} R \right] R^T + R \left[ \frac{d}{d\theta} R^T \right] = 0, \quad (3.174)$$

which we may rewrite as

$$\left[ \frac{d}{d\theta} R \right] R^T + \left( \left[ \frac{d}{d\theta} R \right] R^T \right)^T = 0. \quad (3.175)$$

Now let us define the matrix $S$ as

$$S = \left[ \frac{d}{d\theta} R \right] R^T. \quad (3.176)$$

It follows from (3.175) that

$$S + S^T = 0, \quad (3.177)$$

i.e. $S$ is a skew-symmetric matrix. Postmultiplying both sides of (3.176) by $R$ and simplifying with the fact that $R^T R = I$ yields

$$SR = \frac{d}{d\theta} R, \quad (3.178)$$

which tells us that taking the derivative of a rotation matrix $R$ is equivalent to multiplying the skew-symmetric matrix $S$ by $R$.

We will now explore how (3.178) helps us compute the derivative of a rotation matrix. Recall that the rotation matrix $R_z(\theta)$, which represents the rotation about the $z$-axis for an angle $\theta$, is defined as

$$R_z(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (3.179)$$

Further, let us define the skew-symmetric matrix $S(a)$, where $a = [a_x \ a_y \ a_z]^T$, to be

$$S(a) = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}. \quad (3.180)$$
Direct computation of (3.176) yields

\[
S = \left[ \frac{d}{d\theta} R \right] R^T = \begin{bmatrix}
-\sin \theta & -\cos \theta & 0 \\
\cos \theta & -\sin \theta & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\cos \theta & \sin \theta & 0 \\
-\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

\[
= \begin{bmatrix}
0 & 0 & -\cos^2(\theta) - \sin^2(\theta) \\
\cos^2(\theta) + \sin^2(\theta) & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\]

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

\[
= S(\hat{k})
\]

(3.181)

where \(\hat{k} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T\); that is, the unit vector in the direction of the \(z\)-axis. We have therefore shown that

\[
\frac{d}{d\theta} R_z(\theta) = S(\hat{k}) R_z(\theta). 
\]

(3.182)

In a similar manner, we can derive the relationships

\[
\frac{d}{d\theta} R_x(\theta) = S(\hat{i}) R_x(\theta) \quad \text{and} \quad \frac{d}{d\theta} R_y(\theta) = S(\hat{j}) R_y(\theta), 
\]

(3.183)

where \(\hat{i}\) is the unit vector in the direction of the \(x\)-axis and \(\hat{j}\) is the unit vector in the direction of the \(y\)-axis. In fact, in the general case where the axis of rotation is some arbitrary unit vector \(\hat{u}\), it is possible to show that

\[
\frac{d}{d\theta} R_{\hat{u}}(\theta) = S(\hat{u}) R_{\hat{u}}(\theta). 
\]

(3.184)

Thus, the skew-symmetric matrix allows us to simplify our computation of the derivative of a rotation matrix. We will make use of this property shortly.

### 3.9.4 The Jacobian

We will now introduce the Jacobian matrix, a quantity essential to the planning, analysis, and control of robot motion. Suppose we have \(m\) equations in \(n\) variables,

\[
y_1 = f_1(x_1, x_2, \ldots, x_n), \\
y_2 = f_2(x_1, x_2, \ldots, x_n), \\
\vdots \\
y_m = f_m(x_1, x_2, \ldots, x_n).
\]
3.9. VELOCITY KINEMATICS

which we may write as a vector function,

\[ y = f(x). \]  \hspace{1cm} (3.185)

Then the **Jacobian matrix** of \( y \) is the matrix of the partial derivatives of \( y \):

\[
J(x_1, x_2, \ldots, x_n) = \begin{bmatrix}
\frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_n} \\
\frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_n} \\
\vdots & \ddots & \ddots & \vdots \\
\frac{\partial y_m}{\partial x_1} & \frac{\partial y_m}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_n}
\end{bmatrix}. \hspace{1cm} (3.186)

Thus, the Jacobian matrix is a multidimensional form of the derivative. Note that we also refer to the Jacobian matrix simply as “the Jacobian,” which in some literature means the determinant of a square Jacobian matrix. In these notes, “the Jacobian” will always mean the Jacobian matrix.

We can use the Jacobian matrix to relate joint velocities to Cartesian velocities of the end effector of a manipulator arm. Consider an \( n \)-link manipulator with joint variables \( q_1, q_2, \ldots, q_n \). Let

\[
T_0^n(q) = \begin{bmatrix}
R_0^n(q) & d_0^n(q) \\
0 & 1
\end{bmatrix} \hspace{1cm} (3.187)
\]

be the homogeneous transformation describing the motion of frame \( \{n\} \) relative to frame \( \{0\} \), where \( q = [q_1 \ q_2 \ \cdots \ q_n]^T \) is the vector of joint variables. As the robot moves, its joint variables as well as the position and orientation of its end effector are functions of time. If we take the derivative of the translation and rotation described by this homogeneous transform with respect to time, we get

\[
\dot{d}_0^n(q) = V_0^n(q) \quad \text{and} \quad \dot{R}_0^n(q) = S(\Omega_0^n)R_0^n(q). \hspace{1cm} (3.188)
\]

We can then relate the linear velocity \( V_0^n \) and angular velocity \( \Omega_0^n \) of the end effector to the joint variables as

\[
\begin{bmatrix}
V_0^n(q) \\
\Omega_0^n(q)
\end{bmatrix} = \begin{bmatrix}
J_V(q) \\
J_\Omega(q)
\end{bmatrix} \dot{q}. \hspace{1cm} (3.189)
\]

The Jacobian matrix formed by stacking the two smaller Jacobians \( J_V \) and \( J_\Omega \) will have as many rows as there are degrees of freedom in the Cartesian space being considered and as many columns as there are joints of the arm. Understanding the dimensionality of the Jacobian is important for solving the inverse kinematics problem. If the Jacobian is not square or does not have full rank, then it is not
invertible, and we must use other techniques (such as computing a pseudoinverse) to solve for joint velocities that will produce a desired end-effector velocity.

But the dimensionality of the Jacobian tells us more than just whether the matrix is invertible or not. In the case where there are more columns than there are rows, we know that the robot must be redundant; that is, it has more degrees of freedom than is necessary for performing a specified task. The human arm is an example of a redundant manipulator: while only six degrees of freedom are required to position and orient the hand in any arbitrary way, the human arm has seven degrees of freedom. This additional degree of freedom allows us to move our arm — specifically, reposition our elbow — while keeping our hand in the exact same position and orientation. (Prove this to yourself: plant your hand on the surface of a table and then move your elbow around.)

When a robot is redundant, the columns of its Jacobian are linearly dependent. (To see why, consider for example that there cannot be more than six linearly independent vectors in \( \mathbb{R}^6 \), so the seven columns of the Jacobian matrix for a human arm must be linearly dependent.) It follows that the Jacobian matrix has a nontrivial null space. Velocities drawn from the null space generate internal joint motions but do not cause any motion at the end effector. Therefore, when solving the inverse kinematics problem, the null space of the Jacobian can be used to optimize a solution or find an alternate trajectory. Thus we see how fundamental the Jacobian is for both forward and inverse velocity kinematics of manipulator arms.

### 3.9.5 Long-Form Examples

**Example 3.10** Forward Velocity Kinematics.

**Concepts reviewed:** homogeneous transforms, forward kinematics, the Jacobian

**Problem:** Suppose you have a two-link robot arm equipped with a billiards cue, as shown in Fig. 3.26. When the cue strikes the ball, the joints are instantaneously moving at \( \dot{\theta}_1 = -1 \) rad/sec and \( \dot{\theta}_2 = 1 \) rad/sec. The instantaneous joint angles are \( \theta_1 = \frac{8\pi}{15} \) rad and \( \theta_2 = -\frac{\pi}{2} \) rad. The cue strikes the ball with a line of force that passes through its center (i.e. there is no spin, and it goes in a straight line in the direction the cue tip was moving). The cue ball starts exactly centered on a table of dimensions 2 m long by 1 m wide, and the robot link lengths are \( \ell_1 = 0.5 \) m and \( \ell_2 = 0.5 \) m. Where will the ball strike the rail at the edge of the table?

**Solution:** During the collision between cue and ball, the impulse instantly imparts the same velocity on the ball as the cue had at that moment. Therefore, we must find the linear (Cartesian) velocity of the cue tip at that moment.

We can compute the forward kinematics of the tip with respect to the base of
3.9. VELOCITY KINEMATICS

Figure 3.26: A pool table equipped with a robot cue stick. The robot always strikes the ball at its center to avoid spin.

We may combine these into a Jacobian, which gives the rate of change of the Cartesian (end-effector) coordinates as a function of the rates of the internal joint angles, as

\[
J = \begin{bmatrix}
-\ell_1 s_1 - \ell_2 s_{12} & -\ell_2 s_{12} \\
\ell_1 c_1 + \ell_2 c_{12} & \ell_2 c_{12}
\end{bmatrix}
\] (3.195)
The Jacobian gives us almost exactly what we need to know:

\[
\begin{bmatrix}
\dot{x}_2 \\
\dot{y}_2
\end{bmatrix} = J \begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}
\]

(3.196)

\[
= \begin{bmatrix}
-\ell_1 s_1 - \ell_2 s_{12} \\
\ell_1 c_1 + \ell_2 c_{12}
\end{bmatrix} \begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}
\]

(3.197)

\[
= \begin{bmatrix}
-\ell_1 \sin(\theta_1) - (\ell_2 \dot{\theta}_1 - \ell_2 \dot{\theta}_2) \sin(\theta_1 + \theta_2) \\
\ell_1 \cos(\theta_1) + (\ell_2 \dot{\theta}_1 + \ell_2 \dot{\theta}_2) \cos(\theta_1 + \theta_2)
\end{bmatrix}
\]

(3.198)

\[
\approx \begin{bmatrix}
0.497261 \\
0.052264
\end{bmatrix}
\]

(3.199)

Computing the slope \(m = \frac{\dot{y}_2}{\dot{x}_2} \approx 0.10510\), we can predict that the cue ball will hit the side rail about 10.5 cm above the center.

Example 3.11 Inverse Velocity Kinematics.

Concepts reviewed: inverse kinematics, the Jacobian

Problem: Again considering the two-link robot arm with the billiards cue in Fig. 3.26, what arm velocity parameters does the robot need to set in order to strike the ball in the top-right corner pocket with the cue ball? Assume that the cue ball will travel at 10 m/s and that the joint angles at the moment of impact remain \(\theta_1 = \frac{8\pi}{15}\) and \(\theta_2 = -\frac{\pi}{2}\).

Solution: The desired direction of the ball can be derived from the diagram as \(\phi = \text{atan2}(0.5, 1) \approx 0.46365\). Scaled by the desired magnitude, we have

\[
\begin{bmatrix}
\dot{x}_2 \\
\dot{y}_2
\end{bmatrix} = \begin{bmatrix}
10 \cos(\phi) \\
10 \sin(\phi)
\end{bmatrix} \approx \begin{bmatrix}
8.9443 \\
4.4721
\end{bmatrix}
\]

(3.200)

The values for \(\theta_1 = \frac{8\pi}{15}\) and \(\theta_2 = -\frac{\pi}{2}\) are taken from Example 3.10 because they solve the inverse kinematics problem. At this configuration, the Jacobian is as reported in (3.195). In general,

\[
\begin{bmatrix}
\dot{x}_2 \\
\dot{y}_2
\end{bmatrix} = J \begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix}
\]

(3.201)

which can be solved for the \(\theta\) rates in this case because the Jacobian is square and has full rank. That is, we compute the inverse of the Jacobian,

\[
J^{-1} = \frac{1}{\ell_1 \ell_2 \sin(\theta_2)} \begin{bmatrix}
\ell_2 \cos(\theta_1 + \theta_2) & \ell_2 \sin(\theta_1 + \theta_2) \\
-\ell_1 \cos(\theta_1) - \ell_2 \cos(\theta_1 + \theta_2) - \ell_1 \sin(\theta_1) - \ell_2 \sin(\theta_1 + \theta_2)
\end{bmatrix}
\]

(3.202)

\[
\approx \begin{bmatrix}
-1.9890 & -0.2091 \\
1.7800 & 2.1981
\end{bmatrix}
\]

(3.203)
and then solve for the $\theta$ rates, yielding

$$
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2
\end{bmatrix} = J^{-1} \begin{bmatrix}
\dot{x}_2 \\
\dot{y}_2
\end{bmatrix}
$$

(3.204)

\[\approx \begin{bmatrix}
-1.9890 & -0.2091 \\
1.7800 & 2.1981
\end{bmatrix} \begin{bmatrix}
8.9443 \\
4.4721
\end{bmatrix}
\] (3.205)

\[\approx \begin{bmatrix}
-18.725 \\
25.751
\end{bmatrix}.
\] (3.206)

3.10 Further Reading

For more discussion of kinematics, see Lynch and Park [2] (available online), Craig [1], Murray et al. [3] and Spong et al. [4].

Bibliography


