Chapter 2

Geometric Representations and Transformations

In any scientific or engineering discipline, the value of the results depends strongly on the completeness and accuracy of the models. This chapter provides important foundational material that will be needed in the remainder of the book. To formulate and solve a motion strategy problem, we will need to provide and manipulate complicated geometric models of a system of bodies in space. Although additional modeling tools will be needed for some of the problems in this book, the geometric tools presented in this chapter form a basis that is common to nearly all motion strategy problems.

2.1 Geometric Modeling

A wide variety of approaches and techniques for geometric modeling exist, and the particular choice usually depends on the application and the difficulty of the problem. In most cases, there are generally two alternatives: 1) a boundary representation, and 2) a solid representation. Suppose we would like to define a model of a planet. Using a boundary representation, we might write the equation of a sphere that roughly coincides with the planet's surface. Using a solid representation, we would describe the set of all points that are contained in the sphere. Both alternatives will be considered in this section.

The first task is to define a space in which the motion strategy problem "lives." This will be defined as the *world*, $\mathcal{W}$, for which there are two possible choices: 1) a 2D world, in which $\mathcal{W} = \mathbb{R}^2$ ($\mathbb{R}$ denotes the set of real numbers), and 2) a 3D world, in which $\mathcal{W} = \mathbb{R}^3$. These choices should be sufficient for most problems; however, one might also want to allow more complicated worlds, such as the surface of a sphere or even a higher-dimensional space. Such generalities are avoided in this book because their current applications are limited.

Unless otherwise stated, the world generally contains two kinds of entities:

1. *Obstacles:* Portions of the world that are "permanently" occupied, for example, as in the walls of a building.

2. *Robots:* Geometric bodies that behave according to a motion strategy. A robot could model a physical robot, or it could be a virtual entity, such as a character in a computer-generated animation.

We will usually represent the world and any moving bodies in the world as subsets of a 2D or 3D Euclidean space. This section presents a method of systematically constructing representations of these subsets using a collection of primitives. Both obstacles and robots will be considered as (closed) subsets of $\mathcal{W}$. Let the *obstacle region*, $\mathcal{O}$, denote the set of all points in $\mathcal{W}$ that lie in one or more obstacles; hence, $\mathcal{O} \subseteq \mathcal{W}$. The next step is to define a systematic way of representing $\mathcal{O}$ that will have great expressive power and be computationally efficient. Robots will be defined in a similar way; however, this will be deferred until transformations of geometric bodies are defined.

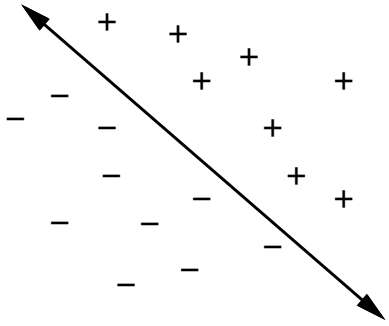


Figure 2.1: The sign of the $f(x, y)$ partitions $\mathbb{R}^2$ into two half planes.

2.1.1 Polygonal and Polyhedral Models

In Sections 2.1.1 and 2.1.2, $\mathcal{O}$ will be characterized in terms of a combination of *primitives*. Each primitive, H_i, represents a subset of $\mathcal{W}$ that is easy to describe and manipulate. A complicated obstacle region will be represented by taking Boolean combinations of primitives. Using set theory, this means that $\mathcal{O}$ is generally defined in terms of unions, intersections, and set differences of primitives.

First, we define $\mathcal{O}$ for the case in which the obstacle region is a convex, polygonal subset of a 2D world, $\mathcal{W} = \mathbb{R}^2$. Recall that a subset, X, of $\mathbb{R}^n$ is *convex* if and only if for any pair of points in X, all points along the line segment that connects them are contained in X. Intuitively, the set contains no pockets or indentations. A set that is not convex is called *nonconvex*.

A boundary representation of $\mathcal{O}$ is an m-sided polygon, which can be described using two kinds of *features*: vertices and edges. Every *vertex* corresponds to a "corner" of the polygon, and every *edge* cooresponds to a line segment between a pair of vertices. The polygon can be specified by a sequence, (x_1, y_1), (x_2, y_2), ..., (x_m, y_m), of m points in $\mathbb{R}^2$, given in counterclockwise order.

A solid representation of $\mathcal{O}$ can be expressed as the intersection of m half-planes. Each half-plane corresponds to the set of all points that lie to one side of a line that is common to a polygon edge. Figure 2.2 shows an example of an octagon that is represented as the intersection of eight half planes.

An edge of the polygon is specified by two points, such as (x_1, y_1) and (x_2, y_2). Consider the equation of a line that passes through (x_1, y_1) and (x_2, y_2). An equation can be determined of the form $ax + by + c = 0$, in which a, b, and c are real-valued constants that determined from x_1, y_1, x_2, and y_2. Let $f(x, y) = ax + by + c$. Note that $f(x, y) < 0$ on one side of the line, and $f(x, y) > 0$ on the other. In other words, the sign of $f(x, y)$ indicates a half plane that is bounded by the line, as depicted in Figure 2.1. Without loss of generality, assume that $f(x, y)$ is defined such that $f(x, y) < 0$ for all points to the left of the edge from (x_1, y_1) to (x_2, y_2) (if it is not, then multiply $f(x, y)$ by -1).

Let $f_i(x, y)$ denote the line equation that corresponds to the edge from (x_i, y_i) to (x_{i+1}, y_{i+1}) for $1 \leq i < m$. Let $f_m(x, y)$ denote the line equation that corresponds to the edge from (x_m, y_m) to (x_1, y_1). Let a *half plane*, H_i, for $1 \leq i \leq m$ be defined as a subset of $\mathcal{W}$:

$$H_i = \{(x, y) \in \mathcal{W} \mid f_i(x, y) \leq 0\}. \tag{2.1}$$

Above, H_i is a primitive that describes the set of all points on one side of the line $f_i(x, y) = 0$ (including the points on the line).

The convex, m-sided polygonal obstacle region, $\mathcal{O}$ is expressed as

$$\mathcal{O} = H_1 \cap H_2 \cap \cdots \cap H_m. \tag{2.2}$$

The assumption that $\mathcal{O}$ is convex is too limiting for most applications. Now suppose that $\mathcal{O}$ is a nonconvex, polygonal subset of $\mathcal{W}$. In this case, $\mathcal{O}$, can be expressed as

$$\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \cdots \cup \mathcal{O}_n, \tag{2.3}$$

in which each $\mathcal{O}_i$ is a convex, polygonal set that is expressed in terms of half spaces using (2.2). Note that $\mathcal{O}_i$ and $\mathcal{O}_j$ for $i \neq j$ need not be disjoint. Using this representation, very complicated obstacle regions in $\mathcal{W}$

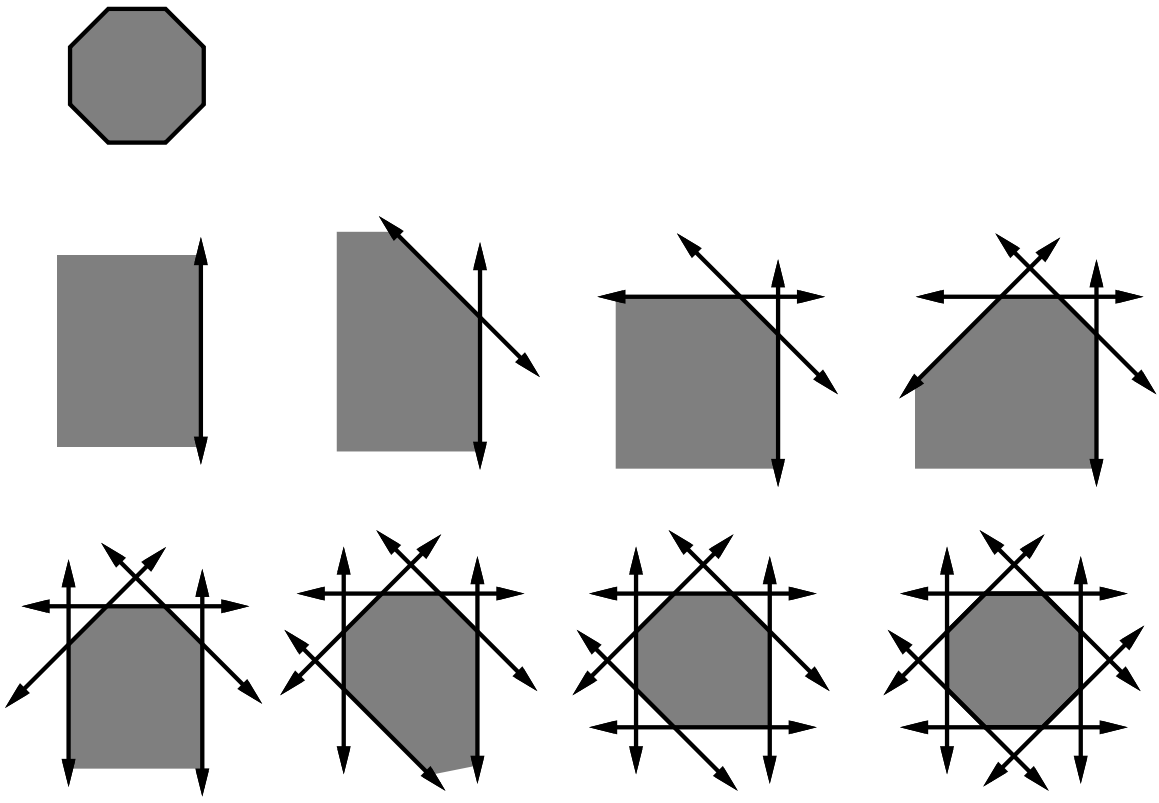


Figure 2.2: A convex polygonal region can be identified by the intersection of half-planes.

can be defined. Although these regions may contain multiple components and holes, if $\mathcal{O}$ is bounded (i.e., $\mathcal{O}$ will fit inside of a rectangular box) its boundary will be comprised of linear segments.

In general, more complicated representations of $\mathcal{O}$ can be defined in terms of any finite combination of unions, intersections, and set differences of primitives; however, it always possible to simplify the representation into the form given by (2.2) and (2.3). A set difference can be avoided by simply redefining the primitive. For a given H_i, a new primitive H_i' can be replacing $f_i(x, y) \leq 0$ with $f_i(x, y) > 0$. This will yield $H_i' = \mathcal{W} \setminus H_i$, which when taken in union with other set is equivalent to the removal of H_i. Given a complicated combination of primitives, once set differences are removed, the expression can be simplified into a finite union of intersections by applying Boolean algebra laws.

Defining a logical predicate What is the value of the previous representation? As a simple example, we can define a logical predicate that serves as a collision detector. The predicate is a Boolean-valued function, $S : \mathcal{W} \rightarrow \{TRUE, FALSE\}$, which returns $TRUE$ for a point in $\mathcal{W}$ that lies in $\mathcal{O}$, and $FALSE$ otherwise. For a line given by $f(x, y) = 0$, let $e(x, y)$ denote a logical predicate that returns $TRUE$ if $f(x, y) \leq 0$, and $FALSE$ otherwise.

A convex polygonal region can be represented by a logical conjunction:

$$\Lambda(x, y) = e_1(x, y) \wedge e_2(x, y) \wedge \cdots \wedge e_m(x, y) \tag{2.4}$$

The predicate $\Lambda(x, y)$ returns TRUE if the point (x, y) lies in the convex polygonal region, and FALSE otherwise. An obstacle region comprised of n convex polygons can be represented by a logical disjunction of conjuncts:

$$S(x, y) = \Lambda_1(x, y) \vee \Lambda_2(x, y) \vee \cdots \vee \Lambda_n(x, y) \tag{2.5}$$

Although more efficient methods exist, the predicate $S(x, y)$ can be used to check whether a point (x_0, y_0) lies inside of $\mathcal{O}$ in time $O(n)$, in which n is the number of primitives that appear in the representation of $\mathcal{O}$ (each primitive is evaluated in constant time).

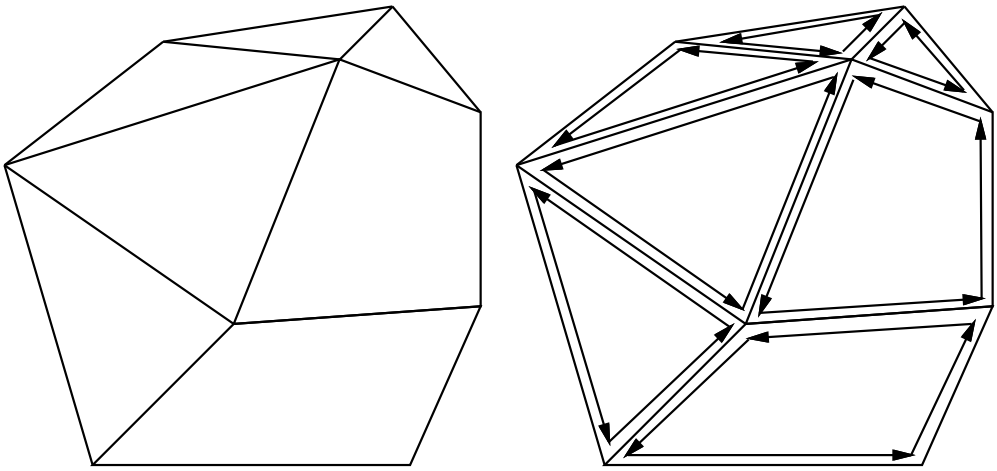


Figure 2.3: a) A polyhedron can be described in terms of faces, edges, and vertices. b) The edges of each face can be stored in a circular list that is traversed in counterclockwise order with respect to the outward normal vector of the face.

For the geometric models considered in this book, there will be a close duality between a logical predicate representation and a set-theoretic representation. Using the logical predicate, the unions and intersections of the set-theoretic representation are replaced by logical OR's and AND's. It is well known from first-order predicate calculus that any complicated logical sentence can be reduced to a logical disjunction of conjunctions (this is often called "sum of products" in computer engineering). This is equivalent to our previous statement that $\mathcal{O}$ can always be represented as a union of intersections of primitives.

Polyhedral models For a 3D world, $\mathcal{W} = \mathbb{R}^3$, and the previous concepts can be nicely generalized from the 2D case by replacing polygons with polyhedra, and replacing half-plane primitives with half-space primitives. Suppose that $\mathcal{O}$ is a convex polyhedron, as shown in Figure 2.3. A boundary representation can be defined in terms of three features: vertices, edges, and faces. Every face is a convex polygon in $\mathbb{R}^3$. Every edge forms a boundary between two faces. Every vertex forms a boundary between three or more edges.

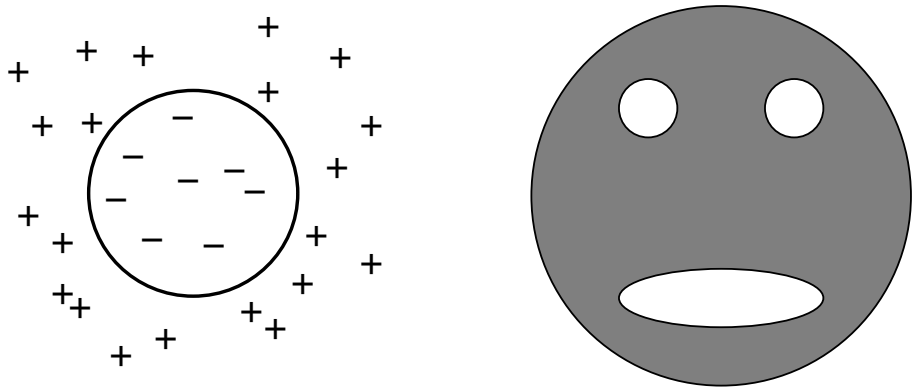
Several data structures have been proposed that allow one to conveniently "walk" around the polyhedral features. For example, the *half edge* data structure contains three types of records: faces, half edges, and vertices. Each vertex record holds the point coordinates, and a pointer to an arbitrary half-edge that touches the vertex. Each face record contains a pointer to an arbitrary half-edge on its boundary. Each face is bounded by a circular list of half-edges. There is a pair of directed half-edge records for each edge of the polyhedon. Each half-edge is shown as an arrow in Figure 2.3.b. Each half-edge record contains pointers to five other records: 1) the vertex from which the half-edge originates, 2) the "twin" half-edge, which bounds the neighboring face, and has the opposite direction, 3) the face that is bounded by the half edge, 4) the next element in the circular list of edges that bound the face, 5) the previous element in the circular list of edges that bound the face. One all of these records have been defined, one can conveniently traverse the structure of the polyhedron.

A solid representation can be constructed from the vertices. Each face of $\mathcal{O}$ has at least three vertices along its boundary. Assuming these vertices are not collinear, an equation of the plane that passes through them can be determined of the form $f(x, y, z) = ax + by + cz + d = 0$, in which a, b, c, and d are real-valued constants.

Let a *half space*, H_i, for $1 \leq i \leq m$, for all m faces of $\mathcal{O}$, be defined as a subset of $\mathcal{W}$:

$$H_i = \{(x, y, z) \in \mathcal{W} \mid f_i(x, y, z) \leq 0\}. \tag{2.6}$$

It is important to choose f_i so that it takes on negative values inside of the polyhedron. In the case of a polygonal model, it was possible to consistently define f_i by proceeding in counterclockwise order around the boundary. In the case of a polyhedron, the half-edge data structure can be used to obtain for each face



a. b.

Figure 2.4: a) Once again, f is used to partition $\mathbb{R}^2$ into two regions. In this case, the algebraic primitive represents a disc-shaped region. b) The shaded "face" can be exactly modeled using only four algebraic primitives.

the list of edges that form its boundary in counterclockwise order. Figure 2.3.b shows the edge ordering for each face. Note that the boundary of each face can be traversed in counterclockwise order. The equation for each face can be consistently determined as follows. Choose three consecutive vertices, p_1, p_2, p_3 (they must not be collinear) in counterclockwise order on the boundary of the face. Let v_{12} denote the vector from p_1 to p_2, and let v_{23} denote the vector from p_2 to p_3. The cross product $v = v_{12} \times v_{23}$ will always yield a vector that points out of the polyhedron and is normal to the face. Recall that the vector $[a \;\; b \;\; c]$ is parallel to the normal to the plane. If these are chosen as $a = v[1]$, $b = v[2]$, and $c = v[3]$, then $f(x, y, z) \leq 0$ for all points in the half space that contains the polyhedron.

As in the case of a polygonal model, a convex polyhedron can be defined as the intersection of a finite number of half spaces, one for each face. A nonconvex polyhedron can be defined as the union of a finite number of convex polyhedra. The predicate $S(x, y, z)$ can be defined in a similar manner, in this case yielding $TRUE$ if $(x, y, z) \in \mathcal{O}$ and $FALSE$ otherwise.

2.1.2 Semi-Algebraic Models

In both the polygonal and polyhedral models, f was a linear function. In the case of a semi-algebraic model for a 2D world, f, can be any polynomial in x and y. For a 3D world, f is a polynomial in x, y, and z. The class of semi-algebraic models includes both polygonal and polyhedral models, which use first-degree polynomials.

Consider the case of a 2D world. A solid representation can be defined using *algebraic primitives* of the form

$$H = \{(x, y) \in \mathcal{W} \mid f(x, y) \leq 0\}. \tag{2.7}$$

As an example, let $f = x^2 + y^2 - 4$. In this case, H, represents a disc of radius 2 that is centered at the origin. This corresponds to the set of points, (x, y), for which $f(x, y) \leq 0$, as depicted in Figure 2.4.a.

Suppose we would like to construct a model of the shaded region shown in Figure 2.4.b. Suppose the center of the outer circle has radius r_1 and is centered at the origin. Suppose that the "eyes" have radius r_2 and r_3, and are centered at (x_2, y_2) and (x_3, y_3), respectively. Suppose the "mouth" is an ellipse with major axis a and minor axis b, and is centered at $(0, y_4)$. The functions are defined as $f_1 = x^2 + y^2 - r_1^2$, $f_2 = -[(x - x_2)^2 + (y - y_2)^2 - r_2^2]$, $f_3 = -[(x - x_3)^2 + (y - y_3)^2 - r_3^2]$, and $f_4 = -[x^2/a^2 + (y - y_4)^2/b^2 - 1]$. For f_2, f_3, and f_4, the familiar circle and ellipse equations were multiplied by -1 to yield algebraic primitives for all points outside of the circle or ellipse. The shaded region, $\mathcal{O}$, can be represented as

$$\mathcal{O} = H_1 \cap H_2 \cap H_3 \cap H_4 \tag{2.8}$$

In the case of semi-algebraic models, the intersection of primitives does not necessarily result in a convex subset of $\mathcal{W}$. In general, however, it might be necessary to form $\mathcal{O}$ be taking unions and intersections of algebraic primitives. A logical predicate, $S(x, y)$ can also be formed. Similar expressions exist for the case of a 3D world. Algebraic primitives can also be defined using $f(x, y) < 0$ or $f(x, y) = 0$, in addition to $f(x, y) \leq 0$.

2.1.3 Other Models

A variety of other models are used in robotics, computer graphics, and engineering design applications. The choice of a model often depends on the types of operations that will be performed. Semi-algebraic models are a good choice for operations such as collision detection and path planning, and are therefore used frequently in motion strategy algorithms.

Other models can be used to represent any object (either robots or obstacles):

- **Nonuniform rational B-splines (NURBS):** A curve can be expressed as

$$C(u) = \frac{\sum_{i=0}^{n} w_i P_i N_{i,k}(u)}{\sum_{i=0}^{n} w_i N_{i,k}(u)}$$

  in which w_i are real-valued *weights*, P_i are control points. The $N_{i,k}$ are normalized basis functions of degree k, which can be expressed recursively as

$$N_{i,k}(u) = \frac{u - t_i}{t_{i+k} - t_i} N_{i,k-1}(u) + \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} N_{i+1,k-1}(u).$$

  The basis of the recursion is $N_{i,0}(u) = 1$ if $t_i \leq u < t_{i+1}$, and $N_{i,0}(u) = 0$ otherwise. A *knot vector* is a nondecreasing sequence of real values, $\{t_0, t_1, \ldots, t_m\}$, that controls that controls the intervals over which certain basic functions take effect.

- **Triangles:** The surface of a 3D object can be represented as a collection triangles in $\mathbb{R}^3$. Nine coordinates are specified for each triangle. A couple of variants of this model include *triangle strips*, which is a sequence of triangles such that each adjacent pair share a common edge, and *triangle fans*, which is triangle strip in which all triangles share a common vertex.

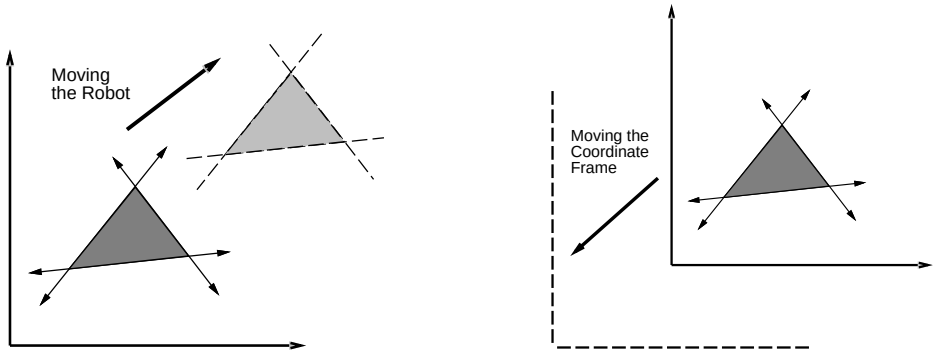
- **Bitmaps:** A 2D or 3D object is represented as a binary image. Each "1" in the image corresponds to a square or cubic cell within the object, and "0" represents the remaining area.

- **Superquadrics:** Instead of using polynomials to define f_i, many generalizations can be constructed. One popular type of model is a *superquadric*, which generalized quadric surfaces. One example is a superellipsoid, given by

$$\left\{ \left| \frac{x}{a} \right|^{n_1} + \left| \frac{y}{b} \right|^{n_2} \right\}^{\frac{n_1}{n_2}} + \left| \frac{z}{c} \right|^{n_1} - 1 \leq 0,$$

  in which $n_1 \geq 2$ and $n_2 \geq 2$. If $n_1 = n_2 = 2$, an ellipse is generated. As n_1 and n_2 increase, the superellipsoid becomes shaped like a box with rounded corners.

- **Generalized cylinders:** A generalized cylinder is a generalization of an ordinary cylinder. Instead of being limited to a line, the center axis is a continuous *spine* curve, $(x(s), y(s), z(s))$ for some parameter $s \in [0, 1]$. Instead of a constant radius, a radius function $r(s)$ along the spine. The value $r(s)$ is the radius of the circle, obtained as the cross section of the generalized cylinder at the point $(x(s), y(s), z(s))$. The normal to the cross section plane is the tangent to the spine curve at s.



a. Translation of the robot b. Translation of the frame

Figure 2.5: For every transformation there are two interpretations.

2.2 Rigid Body Transformations

Any of the techniques from Section 2.1 can be used to define both the obstacle region, $\mathcal{O}$, and the robot, $\mathcal{A}$. Although $\mathcal{O}$ remains fixed in the world, $\mathcal{W}$, motion strategy problems will require us to "move" the robot, $\mathcal{A}$. This will be accomplished by applying transformations to the robot model.

The details of these transformations are covered in the remainder of this chapter. Section 2.2 focuses on the case in which $\mathcal{A}$ is a single, rigid body. Sections 2.3 and 2.4 handle systems of attached bodies and other complications.

2.2.1 2D Transformations

Consider the case of a 2D world, $\mathcal{W} = \mathbb{R}^2$. A point in $(x, y) \in \mathcal{W}$ can be *translated* by some $x_0, y_0 \in \mathbb{R}$ by "replacing" (x, y) with $(x+x_0, y+y_0)$. A set of points, such as the robot, $\mathcal{A}$, can be translated by "replacing" every $(x, y) \in \mathcal{A}$ with $(x + x_0, y + y_0)$. A boundary representation can be transformed by transforming each vertex in the sequence of polygon vertices.

Suppose that a solid representation of $\mathcal{A}$ is defined in terms of primitives. Each primitive of the form

$$H_i = \{(x, y) \in \mathcal{W} \mid f(x, y) \leq 0\}$$

is transformed to

$$H_i = \{(x, y) \in \mathcal{W} \mid f(x - x_0, y - y_0) \leq 0\}.$$

For example, suppose the robot is a disc of unit radius, centered at the origin. It is modeled by a single primitive,

$$\{(x, y) \in \mathcal{W} \mid x^2 + y^2 - 1 \leq 0\}.$$

Suppose $\mathcal{A}$ is translated x_0 units in the x direction, and y_0 units in the y direction. The transformed primitive is

$$\{(x, y) \in \mathcal{W} \mid (x - x_0)^2 + (y - y_0)^2 - 1 \leq 0\},$$

which is the familiar equation for a disc centered at (x_0, y_0).

As shown in Figure 2.5, there are two interpretations of this transformation: 1) the coordinate system remains fixed, and the robot is translated; 2) the robot remains fixed and the coordinate system is translated in the opposite direction. Unless stated otherwise, the first interpretation will be used throughout this book for all transformations.

The translated robot is denoted as $\mathcal{A}(x_0, y_0)$. It will be convenient to use the term *degrees of freedom* to refer to the maximum number of independent parameters that can be selected to completely characterize the robot in the world. If the robot is capable only of translation, then the degrees of freedom is two.

Figure 2.6: Any rotation in 3D can be described as a sequence of yaw, pitch, and roll rotations.

The robot, $\mathcal{A}$, can be *rotated* counterclockwise by some angle $\theta \in [0, 2\pi)$ by "replacing" every $(x, y) \in \mathcal{A}$ by $(x\cos\theta - y\sin\theta, x\sin\theta + y\cos\theta)$. Using a 2×2 rotation matrix,

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

the transformed points can be written as

$$\begin{pmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{pmatrix} = R(\theta) \begin{pmatrix} x \\ y \end{pmatrix}. \tag{2.9}$$

For linear transformations, such as the one defined above, recall that the column vectors represent the basis vectors of the new coordinate frame. The column vectors of $R(\theta)$ are unit vectors, and their inner product (or dot product) is zero, indicating they are orthogonal. Suppose that the XY coordinate axes are "painted" on $\mathcal{A}$. The columns of $R(\theta)$ can be derived by considering the resulting directions of the X and Y axes, respectively, after performing a counterclockwise rotation by the angle θ.

Note that the rotation is performed about the origin. Thus, when defining the model of $\mathcal{A}$, the origin should be placed at the intended axis of rotation. The entire robot model can be rotated by transforming each primitive, yielding $\mathcal{A}(\theta)$. The inverse rotation, $R(-\theta)$, must be applied to each primitive.

Suppose a rotation by θ is performed, followed by a translation by x_0, y_0. These two transformations can be used to place the robot in any desired position and orientation in $\mathcal{W}$. If the operations are applied successively, each $(x, y) \in \mathcal{A}$ is replaced by

$$\begin{pmatrix} x\cos\theta - y\sin\theta + x_0 \\ x\sin\theta + y\cos\theta + y_0 \end{pmatrix}.$$

Notice that the following matrix multiplication will yield the same result for the first two vector components

$$\begin{pmatrix} \cos\theta & -\sin\theta & x_0 \\ \sin\theta & \cos\theta & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x\cos\theta - y\sin\theta + x_0 \\ x\sin\theta + y\cos\theta + y_0 \\ 1 \end{pmatrix}$$

This implies that the following 3×3 matrix, T, may be used to represent both a rotation and a translation:

$$T = \begin{pmatrix} \cos\theta & -\sin\theta & x_0 \\ \sin\theta & \cos\theta & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.10}$$

The matrix T will be referred to as a *homogeneous transform*. It is important to remember that T represents a rotation followed by a translation (not the other way around). Each primitive can be transformed using the inverse of T, resulting in a transformed solid model of the robot, $\mathcal{A}(x_0, y_0, \theta)$. In this case, there are three degrees of freedom. The homogeneous transform is a convenient representation of the combined transformations; therefore, it is frequently used in robotics, computer graphics, and other areas.

2.2.2 3D Transformations

The rigid body transformations for the 3D case are conceptually similar the 2D case; however, the 3D case appears more difficult because 3D rotations are significantly more complicated than 2D rotations.

We can *translate* $\mathcal{A}$ by some $x_0, y_0, z_0 \in \mathbb{R}$ by "replacing" every $(x, y, z) \in \mathcal{A}$ by $(x + x_0, y + y_0, z + z_0)$. Primitives of the form $H_i = \{(x, y, z) \in \mathcal{W} \mid f_i(x, y, z) \leq 0\}$, are transformed to $\{(x, y, z) \in \mathcal{W} \mid f_i(x - x_0, y - y_0, z - z_0) \leq 0\}$. The translated robot is denoted as $\mathcal{A}(x_0, y_0, z_0)$.

Note that a 3D body can be independently rotated about three orthogonal axes, as shown in Figure 2.6. We will borrow aviation terminology, and refer to these rotations as *yaw*, *pitch*, and *roll*:

- **Yaw:** Counterclockwise rotation of α about the Z-axis. The rotation matrix is given by

$$R_Z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

. Note that the upper left entries of $R_Z(\alpha)$ are simply a 2D rotation applied to the XY coordinates.

- **Pitch:** Counterclockwise rotation of β about the Y-axis. The rotation matrix is given by

$$R_Y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}$$

- **Roll:** Counterclockwise rotation of γ about the X-axis. The rotation matrix is given by

$$R_X(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}$$

Each rotation matrix is a simple extension of the 2D rotation matrix that appears in (2.9). For example, the yaw matrix, $R_z(\alpha)$ essentially performs a 2D rotation with respect to the XY plane, while leaving the Z coordinate unchanged. Thus, the third row and third column of $R_z(\alpha)$ look like part of the identity matrix, while the upper right portion of $R_z(\alpha)$ looks like the 2D rotation matrix.

The yaw, pitch, and roll rotations can be used to place a 3D body in any orientation. A single rotation matrix can be formed by multiplying the yaw, pitch, and roll rotation matrices. This yields

$$R(\alpha,\beta,\gamma) = R_Z(\alpha)\, R_Y(\beta)\, R_X(\gamma) = \begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}.$$

It is important to note that $R(\alpha,\beta,\gamma)$ performs the roll first, then the pitch, and finally the yaw. If the order of these operations is changed, a different rotation matrix would result. Note that 3D rotations depend on three parameters, α, β, and γ, whereas 2D rotations depend only on a single parameter, θ. The primitives of the model can be transformed using $R(\alpha,\beta,\gamma)$, resulting in $\mathcal{A}(\alpha,\beta,\gamma)$.

As in the 2D case, a homogeneous transformation matrix can be defined. For the 3D case, a 4×4 matrix is obtained that performs the rotation given by $R(\alpha,\beta,\gamma)$, followed by a translation given by x_0, y_0, z_0. The result is

$$T = \begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma & x_0 \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma & y_0 \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma & z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.11}$$

Once again, the order of operations is critical. The matrix above represents the following sequence of transformations: 1) a roll by γ, 2) a pitch by β, 3) a yaw by α, 4) a translation by (x_0, y_0, z_0). The robot primitives can be transformed, to yield $\mathcal{A}(x_0, y_0, z_0, \alpha, \beta, \gamma)$. We observe that a 3D rigid body that is capable of translation and rotation has six degrees of freedom.

2.3 Transformations of Kinematic Chains of Bodies

The transformations become more complicated for a chain of attached rigid bodies. For convenience, each rigid body is referred to as a *link*. Let $\mathcal{A}_1$, $\mathcal{A}_2$, ..., $\mathcal{A}_m$ denote a set of m links. For each i such that $1 \le i < m$, link $\mathcal{A}_i$ is "attached" to link $\mathcal{A}_{i+1}$ in a way that allows $\mathcal{A}_{i+1}$ some constrained motion with respect to $\mathcal{A}_i$. The motion constraint must be explicitly given, and will be discussed shortly. As an example, imagine a trailer that is attached to the back of a car by a hitch that allows the trailer to rotate with respect to the car. The set of attached bodies will be referred to as a *kinematic chain*.

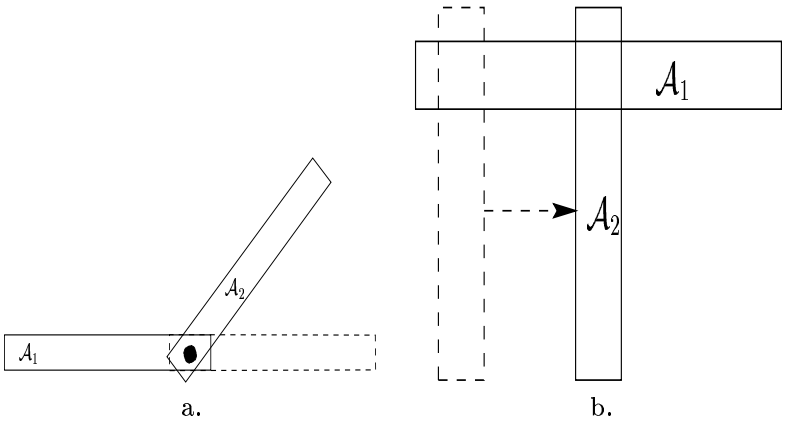


Figure 2.7: Two types of 2D joints: a) a revolute joint allows one link to rotate with respect to the other, b) a prismatic joint allows one link to translate with respect to the other.

2.3.1 A Kinematic Chain in $\mathbb{R}^2$

Before considering a chain, suppose $\mathcal{A}_1$ and $\mathcal{A}_2$ are two rigid bodies, each of which is capable of translating and rotating in $\mathcal{W} = \mathbb{R}^2$. Since each body has three degrees of freedom, there is a combined total of six degrees of freedom, in which the independent parameters are x_1, y_1, θ_1, x_2, y_2, and θ_2. When bodies are attached in a kinematic chain, degrees of freedom are removed.

Figure 2.7 shows two different ways in which a pair of 2D links can be attached. The place at which the links are attached is called a *joint*. In Figure 2.7.a, a *revolute joint* is shown, in which one link is capable only of rotation with respect to the other. In Figure 2.7.b, a *prismatic joint* is shown, in which one link translates along the other. Each type of joint removes two degrees of freedom from the pair of bodies. For example, consider a revolute joint that connects $\mathcal{A}_1$ to $\mathcal{A}_2$. Assume that the point $(0, 0)$ in the model for $\mathcal{A}_2$ is permanently fixed to a point (x_a, y_a) on $\mathcal{A}_1$. This implies that the translation of $\mathcal{A}_2$ will be completely determined once x_a and y_a are given. Note that x_a and y_a are functions of x_1, y_1, and θ_1. This implies that $\mathcal{A}_1$ and $\mathcal{A}_2$ have a total of four degrees of freedom when attached. The independent parameters are x_1, x_2, θ_1, and θ_2. The task in the remainder of this section is to determine exactly how the models of $\mathcal{A}_1$, $\mathcal{A}_2$, ..., $\mathcal{A}_m$ are transformed, and give the expressions in terms of these independent parameters.

Consider the case of a kinematic chain in which each pair of links is attached by a revolute joint. The first task is to specify the geometric model for each link, $\mathcal{A}_i$. Recall that for a single rigid body, the origin of the coordinate frame determines the axis of rotation. When defining the model for a link in a kinematic chain, excessive complications can be avoided by carefully placing the coordinate frame. Since rotation occurs about a revolute joint, a natural choice for the origin is the joint between $\mathcal{A}_i$ and $\mathcal{A}_{i-1}$ for each $i > 1$. For convenience that will soon become evident, the X-axis is defined as the line through both joints that lie in $\mathcal{A}_i$, as shown in Figure 2.7. For the last link, $\mathcal{A}_m$, the X-axis can be placed arbitrarily, assuming that the origin is placed at the joint that connects $\mathcal{A}_m$ to $\mathcal{A}_{m-1}$. The coordinate frame for the first link, $\mathcal{A}_1$, can be placed using the same considerations as for a single rigid body.

We are now prepared to determine the location of each link. The position and orientation of link $\mathcal{A}_1$ is determined by applying the 2D homogeneous transform matrix (2.10),

$$T_1 = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & x_0 \\ \sin\theta_1 & \cos\theta_1 & y_0 \\ 0 & 0 & 1 \end{pmatrix}.$$

As shown in Figure 2.8, let a_{i-1} be the distance between the joints in $\mathcal{A}_{i-1}$. The orientation difference between $\mathcal{A}_i$ and $\mathcal{A}_{i-1}$ is denoted by the angle θ_i. Let T_i represent a 3×3 homogeneous transform matrix

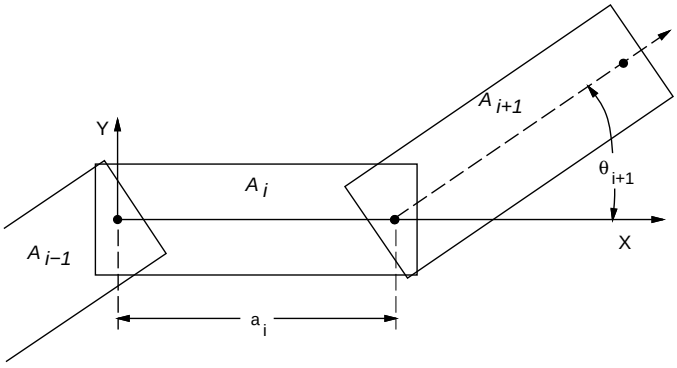


Figure 2.8: The coordinate frame that is used to define the geometric model for each $\mathcal{A}_i$, for $1 < i < m$, is based on the joints that connect $\mathcal{A}_i$ to $\mathcal{A}_{i-1}$ and $\mathcal{A}_{i+1}$.

(2.10), specialized for link $\mathcal{A}_i$ for $1 < i \leq m$, in which

$$T_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & a_{i-1} \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{2.12}$$

which generates the following sequence of transformations:

1. Rotate counterclockwise by θ_i
2. Translate by a_{i-1} along the X-axis

The transformation T_i expresses the difference between the coordinate frame in which $\mathcal{A}_i$ was defined, and the frame in which $\mathcal{A}_{i-1}$ was defined. The application of T_i moves $\mathcal{A}_i$ from its initial frame to the frame in which $\mathcal{A}_{i-1}$ is defined. The application of $T_{i-1}T_i$ moves both $\mathcal{A}_i$ and $\mathcal{A}_{i-1}$ to the frame in which $\mathcal{A}_{i-2}$. By following this procedure, the location of any point (x, y) on $\mathcal{A}_m$ is determined by multiplying the transformation matrices to obtain

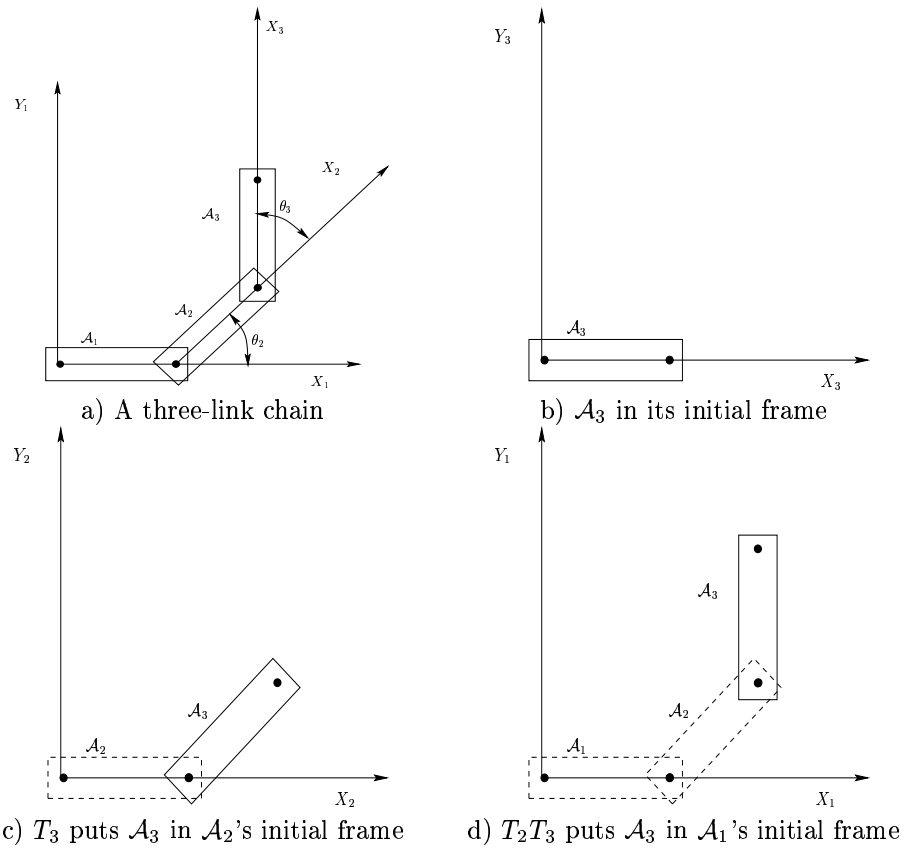
$$T_1 T_2 \cdots T_m \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}. \tag{2.13}$$

To gain an intuitive understanding of these transformations, consider determining the configuration for link $\mathcal{A}_3$, as shown in Figure 2.9. Figure 2.9.a shows a three-link chain, in which $\mathcal{A}_1$ is at its initial configuration, and the other links are each offset by $\frac{\pi}{2}$ from the previous link. Figure 2.9.b shows the frame in which the model for $\mathcal{A}_3$ is initially defined. The application of T_3 causes a rotation of θ_3 and a translation by a_2. As shown in Figure 2.9.c, this places $\mathcal{A}_3$ in its appropriate configuration. Note that $\mathcal{A}_2$ can be placed in its initial configuration, and it will be attached correctly to $\mathcal{A}_3$. The application of T_2 to the previous result places both $\mathcal{A}_3$ and $\mathcal{A}_2$ in their proper configurations, and $\mathcal{A}_1$ can be placed in its initial configuration.

For revolute joints, the parameters a_i are treated as constants, and the θ_i are variables. The transformed m^{th} link is represented as $\mathcal{A}_m(x_0, y_0, \theta_1, \ldots, \theta_m)$. In some cases, the first link might have a fixed location in the world. In this case, the revolute joints account for all degrees of freedom, yielding $\mathcal{A}_m(\theta_1, \ldots, \theta_m)$. For prismatic joints, the a_i are treated as variables, as opposed to the θ_i. Of course, it is possible to include both types of joints in a single kinematic chain.

2.3.2 A Kinematic Chain in $\mathbb{R}^3$

As for a single rigid body, the 3D case is significantly more complicated than 2D due to 3D rotations. Also, several more types of joints are possible, as shown in Figure 2.10. Nevertheless, it is naturally extend the ideas from transformations of 2D kinematic chains to the 3D case. The following steps from Section 2.3.1 will be recycled here: 1) the coordinate frame must be carefully placed to define the model for each $\mathcal{A}_i$; 2) based on joint relationships, several parameters will be defined; 3) the parameters will be used to define



a) A three-link chain

b) $\mathcal{A}_3$ in its initial frame

c) T_3 puts $\mathcal{A}_3$ in $\mathcal{A}_2$'s initial frame

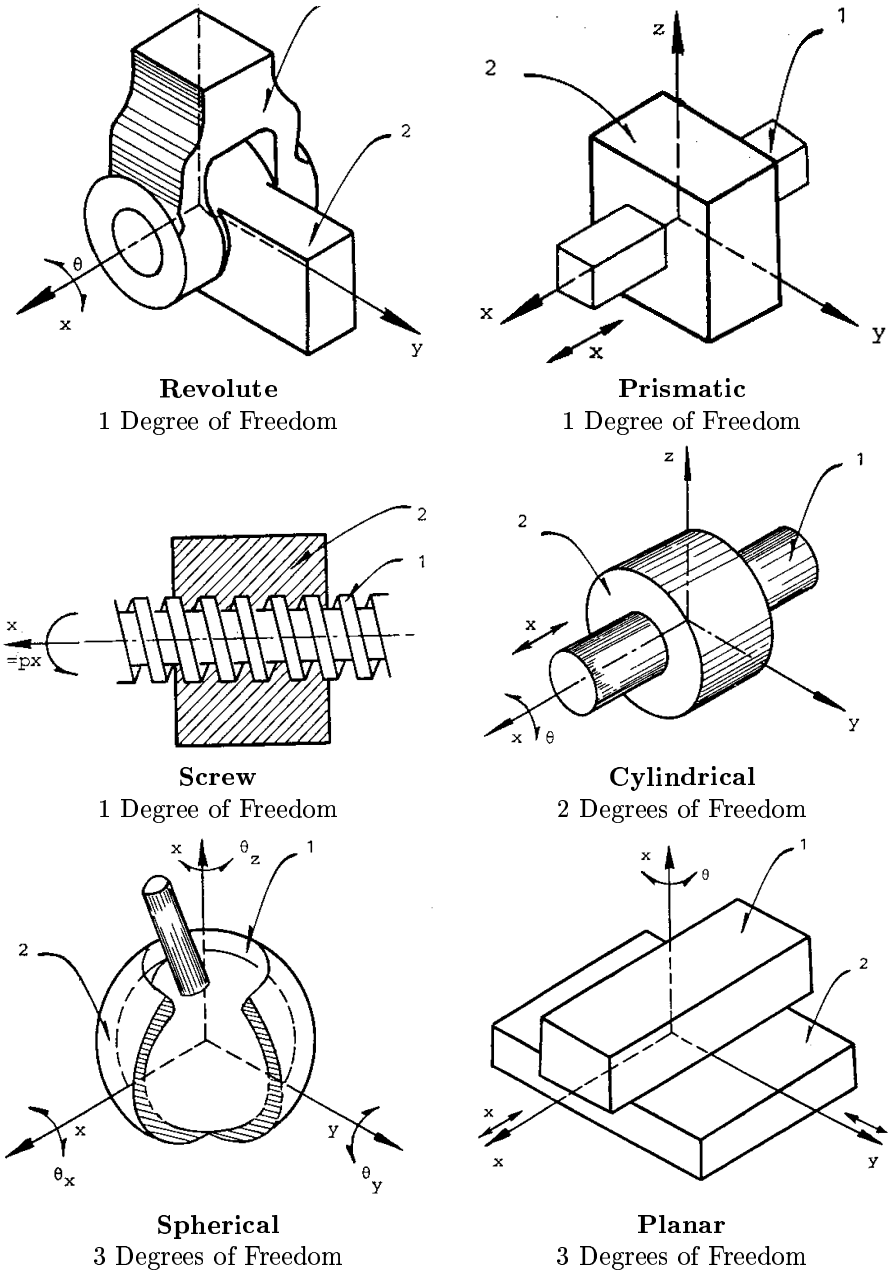
d) T_2T_3 puts $\mathcal{A}_3$ in $\mathcal{A}_1$'s initial frame

Figure 2.9: Applying the transformation T_2T_3 to the model of $\mathcal{A}_3$. In this case, T_1 is the identity matrix.

a homogeneous transformation matrix, T_i; 4) the transformation of points on link $\mathcal{A}_m$ will be given by $T_1T_2 \cdots T_m$.

Consider a kinematic chain of m links in $\mathcal{W} = \mathbb{R}^3$, in which each $\mathcal{A}_i$ for $1 \leq i < m$ is attached to $\mathcal{A}_{i+1}$ by a revolute joint. Each link can be a complicated, rigid body as shown in Figure 2.11.a. For the 2D problem, the coordinate frames were based on the points of attachment. For the 3D problem, it is convenient to use the axis of rotation of each revolute joint (this is equivalent to the point of attachment for the 2D case). The axes of rotation will generally be skew lines in $\mathbb{R}^3$, as shown in Figure 2.11.b. Let Z_i refer to the axis of rotation for the revolute joint that holds $\mathcal{A}_i$ to $\mathcal{A}_{i-1}$. Between each pair of axes in succession, draw a vector, X_i, that joins the two closest pair of points, one from Z_i and the other from Z_{i-1} (this choice is unique if the axes are not parallel). The recommended coordinate frame for defining the geometric model for each $\mathcal{A}_i$ will be given with respect to Z_i and X_i, which are given in Figure 2.11.b. Assuming a right-handed coordinate system, the Y_i axis points away from us in Figure 2.11.b. In the transformations that appear shortly, the coordinate frame given by X_i, Y_i, and Z_i, will be most convenient for defining the model for $\mathcal{A}_i$, even if the origin of the frame lies outside of $\mathcal{A}_i$.

In Section 2.3.1, each T_i was defined in terms of two parameters, a_{i-1} and θ_i. For the 3D case, four parameters will be defined: d_i, θ_i, a_{i-1}, and α_{i-1}. These are referred to as *Denavit-Hartenberg parameters*, or DH parameters for short [2]. The definition of each parameter is indicated in Figure 2.12. Figure 2.12.a shows the definition of d_i. Note that X_{i-1} and X_i contact Z_i at two different places. Let d_i denote signed distance between these points of contact. If X_i is above X_{i-1} along Z_i, then d_i is positive; otherwise, d_i is negative. The parameter θ_i is the angle between X_i and X_{i-1}, which corresponds to the rotation about Z_i that moves X_{i-1} to coincide X_i. In Figure 2.12.b, Z_i is pointing outward. The parameter a_i is the distance between Z_i and Z_{i-1}; recall these are generally skew lines in $\mathbb{R}^3$. The parameter α_{i-1} is the angle between Z_i and Z_{i-1}. In Figure 2.12.d, X_{i-1} is pointing outward.



Revolute
1 Degree of Freedom

Prismatic
1 Degree of Freedom

Screw
1 Degree of Freedom

Cylindrical
2 Degrees of Freedom

Spherical
3 Degrees of Freedom

Planar
3 Degrees of Freedom

Figure 2.10: Types of 3D Joints

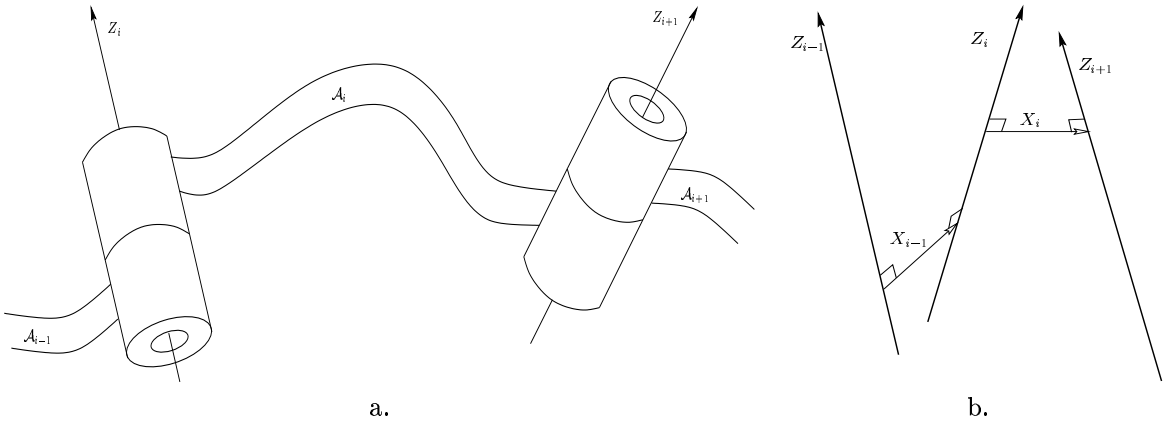


Figure 2.11: a) The diagram of a generic link; b) The rotation axes are skew lines in $\mathbb{R}^3$.

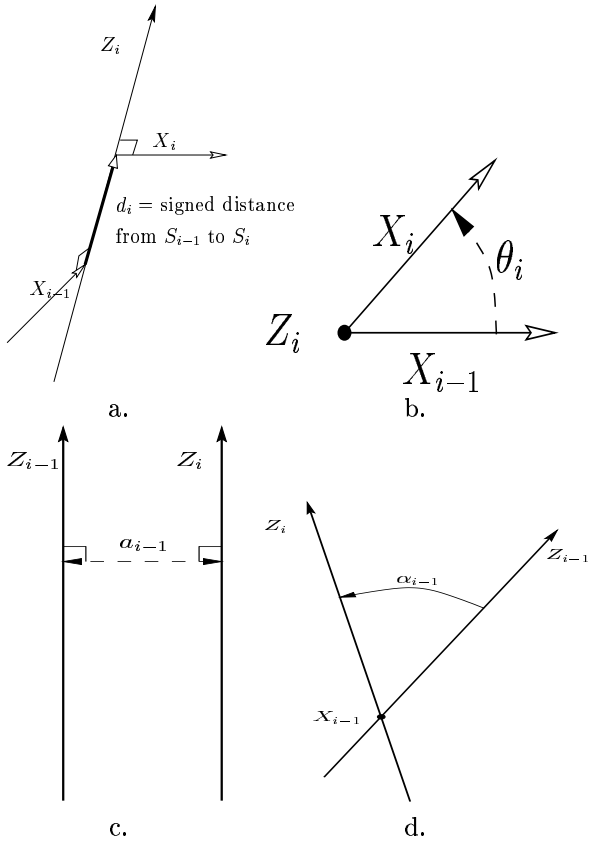


Figure 2.12: Definitions of the four DH parameters: d_i, θ_i, a_{i-1}, α_{i-1}

The homogeneous transformation matrix T_i will be constructed by combining two simpler transformations. The transformation

$$R_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

causes a rotation of θ_i about the Z_i axis, and a translation of d_i along the Z_i axis. Notice that the effect of R_i is independent of the ordering of the rotation by θ_i and the translation by d_i, because both operations occur with respect to the same axis, Z_i. The combined operation of a translation and rotation with respect to the same axis is referred to as a *screw* (as in the motion of a screw through a nut). . The effect of R_i can thus be considered as a screw about Z_i. The second transformation is

$$Q_{i-1} = \begin{pmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & \cos\alpha_{i-1} & -\sin\alpha_{i-1} & 0 \\ 0 & \sin\alpha_{i-1} & \cos\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

which can be considered as a screw about the X_{i-1} axis. A rotation of α_{i-1} about X_{i-1} is followed by a translation of a_{i-1}.

The homogeneous transformation matrix, T_i, for $1 < i \leq m$, is

$$T_i = Q_{i-1}R_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.14}$$

This can be considered as the 3D counterpart to the 2D transformation matrix, (2.10). The following four operations are performed in succession:

1. Translate by d_i along the Z-axis
2. Rotate counterclockwise by θ_i about the Z-axis
3. Translate by a_{i-1} along the X-axis
4. Rotate counterclockwise by α_{i-1} about the X-axis

The transformation T_i gives the relationship of the frame for $\mathcal{A}_i$ to the frame for $\mathcal{A}_{i-1}$. The position of a point (x, y, z) on $\mathcal{A}_m$ is given by

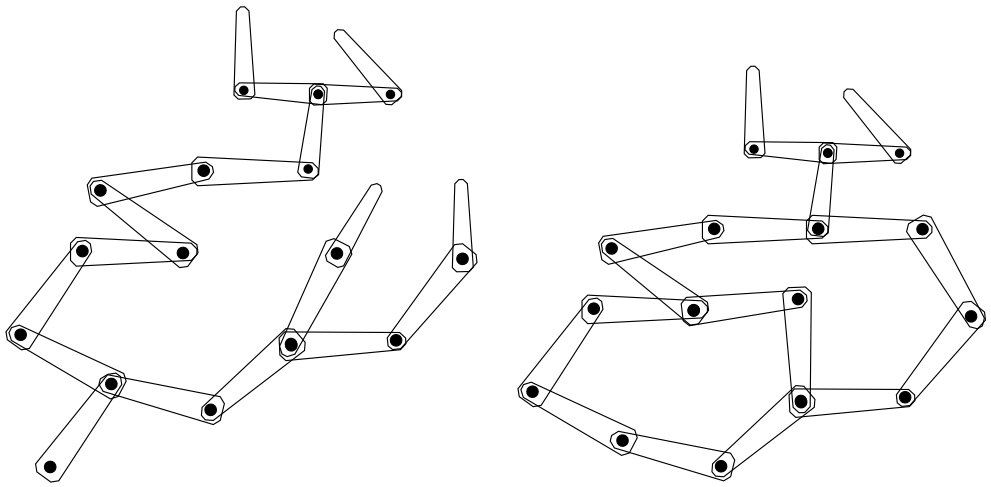
$$T_1T_2\cdots T_m \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}.$$

As in the 2D case, the matrix T_1 is the standard rigid-body homogeneous transformation matrix, which is given by (2.11) for the 3D case.

For each revolute joint, θ_i is treated as the only variable in T_i. A prismatic joints can be simulated by allowing a_i to vary. More complicated joints can be simulated as a sequence of degenerate joints. For example, a spherical joint can be considered as a sequence of three zero-length revolute joints; the joints perform a roll, a pitch, and a yaw. Another option for more complicated joints is to derive a special homogeneous transformation matrix. This might be needed to preserve some of the topological properties that will be discussed in Chapter **??**.

2.4 Transforming Other Structures

Although the concepts from Section 2.3 enable the transformation of a chain of bodies, some realistic motion strategy problems do not fall into this category. This section briefly describes some additional methods for transforming other kinds of structures.



a. b.

Figure 2.13: General linkages: a) Instead of a chain of rigid bodies, a "tree" of rigid bodies can be considered; b) if there are loops, the problem becomes much more difficult because configurations must be assigned in a consistent way.

A Kinematic Tree Many kinematic structures consist of a "tree" of rigid bodies, as opposed to a chain, as shown in Figure 2.13.a. The human body, with its joints and limbs attached to the torso, provides another example that can be modeled as a tree of rigid links. The position and orientation of bodies in a tree can be handled in the same way as those in a chain.

A single link can serve as the root of the tree. The position and orientation of a link, $\mathcal{A}_i$ is determined by applying the product of homogeneous matrices (2.12) or (2.14) along the sequence of links from $\mathcal{A}_i$ to the root link. All other branches in the three should be ignored when assigning the DH parameters.

Suppose that for the case of a 2D tree of bodies, all child links of $\mathcal{A}_i$ are attached the same joint. In this case the a_i parameter may be assigned in the usual way. If the child links are attached in at different joints, then the situation becomes more complicated. For each path in the tree, the chain of links can be handled in the usual way by assigning a distinct a parameter for each joint that connects a child. The required local frame for defining $\mathcal{A}_i$ will be different for each joint because the X axis must pass through the joint that connects the child link. One way to fix this problem is to abandon (2.12), and define a special homogeneous transformation matrix for each case. The situation becomes even more complicated for a 3D tree of bodies.

Closed Kinematic Chains If there are closed loops of rigid bodies, as shown in Figure 2.13.b, the problem becomes much more difficult because a set of constraints must be maintained. For example, suppose we take a 2D chain of rigid bodies and attach $\mathcal{A}_m$ to $\mathcal{A}_1$ with a revolute joint. We are no longer free to choose θ_1, θ_2, ..., θ_m independently. Many choices will cause $\mathcal{A}_m$ to become detached $\mathcal{A}_1$ to become detached. The solution to this problem can be considered as a special case of the *inverse kinematics problem* in robotics.

Flexible Bodies Suppose we would like to model a robotic snake by making a kinematic chain out of hundreds of short links. This would lead to a complicated set of transformations for moving the snake, and a degree of freedom for each link. Imagine another application, in which one would like to warp a thin flexible sheet of material such as aluminum though a small doorway. The sheet could be approximated by a 2D array of links; however, the complexity and degrees of freedom would again be too cumbersome.

For these kinds of problems, it is often advantageous to approximate the entire structure by a smooth curve or surface, given by a parametric function. A family of functions is generally used to encompass possible ways in which the structure can be transformed. The family of functions is characterized by a set of parameters. For each selection of parameters, a particular function in the family is obtained. For example, one set of parameters might place the snake in a straight configuration, and another might place it in an "S"

configuration.

Literature

A vast set of literature covers the topics presented in this chapter. A thorough coverage of solid and boundary representations, including semi-algebraic models, can be found in [3]. A standard geomretric modeling book from a CAD/CAM perspective, including NURBs models is [6]. NURB models are also surveyed in [8]. Theoretical algorithm issues regarding semialgebraic models are covered in [4, 5]. The subject of transformations of rigid bodies and chains of bodies is covered in most robotics texts. Classic references include [1, 7]. The DH parameters were introduced in [2].

Exercises

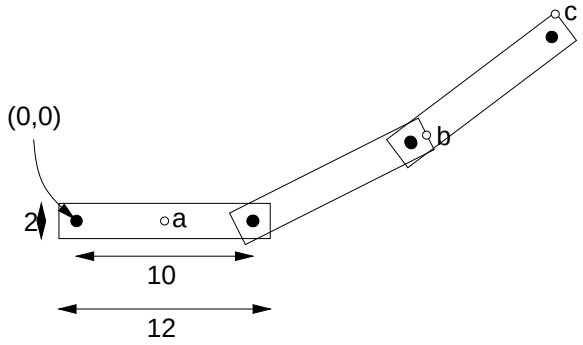
1. How would you define the semi-algebraic model to remove a triangular "nose" from the region shown in Figure 2.4?

2. For distinct values of yaw, pitch, and roll, is it possible to generate the same rotation. In other words, $R(\alpha, \beta, \gamma) = R(\alpha', \beta', \gamma')$, if at least one of the angles is distinct. Characterize the sets of angles for which this occurs.

3. Using rotation matrices, prove that 2D rotation is commutative, but 3D rotation is not.

4. Suppose that $\mathcal{A}$ is a unit disc, centered at the origin and $\mathcal{W} = \mathbb{R}^2$. Assume that $\mathcal{A}$ is represented by a single, semi-algebraic primitive, $H = \{(x, y) \mid x^2 + y^2 \leq 1\}$. Show that the transformed primitive is unchanged after rotation.

5. Consider the articulated chain of bodies shown below. There are three identical rectangular bars in the plane, called $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$. Each bar has width 2 and length 12. The distance between the two points of attachment is 10. The first bar, $\mathcal{A}_1$, is attached to the origin. The second bar $\mathcal{A}_2$ is attached to the $\mathcal{A}_1$, and $\mathcal{A}_3$ is attached to the $\mathcal{A}_2$. Each bar is allowed to rotate about its point of attachment. The configuration of the chain can be expressed with three angles, $(\theta_1, \theta_2, \theta_3)$. The first angle, θ_1 represents the angle between the segment drawn between the two points of attachment of $\mathcal{A}_1$ and the x axis. The second angle, θ_2, represents the angle between $\mathcal{A}_2$ and $\mathcal{A}_1$ ($\theta_2 = 0$ when they are parallel). The third angle, θ_3 represents the angle between $\mathcal{A}_3$ and $\mathcal{A}_2$.



   Suppose the configuration is $(\pi/4, \pi/2, -\pi/4)$. Use the homogeneous transformation matrices to determine the locations of points a, b, and c. Name the set of all configurations such that final point of attachment (near the end of $\mathcal{A}_3$) is at $(0, 0)$ (you should be able to figure this out without using the matrices).

6. Approximate a spherical joint as a chain of three short links that are attached by revolute joints and give the sequence of transformation matrices. If the link lengths approach zero, show that the resulting sequence of transformation matrices can be used to exactly represent the kinematics of a sperical joint.

Projects

1. **Virtual Tinkertoys:** Design and implement a system in which the user can attach various links to make a 3D kinematic tree. Assume that all joints are revolute. The user should be allowed to change parameters and see the resulting positions of all of the links.

2. **Virtual Human Animation:** Construct a model of the human body as a tree of links in a 3D world. For simplicity, the geometric model may be limited to spheres and cylinders. Design and implement a system that displays the virtual human, and allows the user to click on joints of the body to enable them to rotate.

Bibliography

[1] J. J. Craig. *Introduction to Robotics.* Addison-Wesley, Reading, MA, 1989.

[2] R. S. Hartenburg and J. Denavit. A kinematic notation for lower pair mechanisms based on matrices. *J. Applied Mechanics*, 77:215–221, 1955.

[3] C. M. Hoffmann. *Geometric and Solid Modeling.* Morgan Kaufmann, San Mateo, CA, 1989.

[4] B. Mishra. *Algorithmic Algebra.* Springer-Verlag, New York, NY, 1993.

[5] B. Mishra. Computational real algebraic geometry. In J. E. Goodman and J. O'Rourke, editors, *Discrete and Computational Geometry*, pages 537–556. CRC Press, New York, 1997.

[6] M. E. Mortenson. *Geometric Modeling.* Wiley, New York, NY, 1985.

[7] R. P. Paul. *Robot Manipulators: Mathematics, Programming, and Control.* MIT Press, Cambridge, MA, 1981.

[8] L. Piegl. On NURBS: A survey. *IEEE Trans. Comp. Graph. & Appl.*, 11(1):55–71, Jan 1991.