



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

 Previous

Next 

Chapter 3: Configuration Space

To create motion plans for robots, we must be able to specify the position of the robot. More specifically, we must be able to give a specification of the location of every point on the robot, since we need to ensure that no point on the robot collides with an obstacle. This raises some fundamental questions: How much information is required to completely specify the position of every point on the robot? How should this information be represented? What are the mathematical properties of these representations? How can obstacles in the robot's world be taken into consideration while planning the path of a robot?

In this chapter, we begin to address these questions. We first discuss exactly what is meant by a configuration of a robot and introduce the concept of the configuration space, one of the most important concepts in robot motion planning. We then briefly discuss how obstacles in the robot's environment restrict the set of admissible paths. We then begin a more rigorous investigation of the properties of the configuration space, including its dimension, how it sometimes can be represented by a differentiable manifold, and how manifolds can be represented by embeddings and parameterizations. We conclude the chapter by discussing mappings between different representations of the configuration, and the *Jacobian* of these mappings, which relates velocities in the different representations.

3.1 Specifying a Robot's Configuration

To make our discussion more precise, we introduce the following definitions. The *configuration* of a robot system is a complete specification of the position of every point of that system. The *configuration space*, or *C-space*, of the robot system is the space of all possible configurations of the system. Thus a configuration is simply a point in this abstract configuration space. Throughout the text, we use q to denote a configuration and \mathcal{Q} to denote the configuration space. ^[1] The number of *degrees of freedom* of a robot system is the dimension of the configuration space, or the minimum number of parameters needed to specify the configuration.

To illustrate these definitions, consider a circular mobile robot that can translate without rotating in the plane. A simple way to represent the robot's configuration is to specify the location of its center, (x, y) , relative to some fixed coordinate frame. If we know the radius r of the robot, we can easily determine from the configuration $q = (x, y)$ the set of points occupied by the robot. We will use the notation $R(q)$ to denote this set. When we define the configuration as $q = (x, y)$, we have

$$R(x, y) = \{(x', y') \mid (x - x')^2 + (y - y')^2 \leq r^2\},$$

and we see that these two parameters, x and y , are sufficient to completely determine the configuration of the circular robot. Therefore, for the circular mobile robot, we can represent the configuration space by \mathbb{R}^2 once we have chosen a coordinate frame in the plane.

Robots move in a two- or three-dimensional Euclidean ambient space, represented by \mathbb{R}^2 or \mathbb{R}^3 , respectively. We sometimes refer to this ambient space as the *workspace*. Other times we have a more specific meaning for "workspace." For example, for a robot arm, often we call the workspace the set of points of the ambient space reachable by a point on the hand or *end effector* (see [figure 3.3](#)). For the translating mobile robot described above, the workspace and the configuration space are both two-dimensional Euclidean spaces, but it is important to keep in mind that these are different spaces. This becomes clear when we consider even slightly more complicated robots, as we see next.

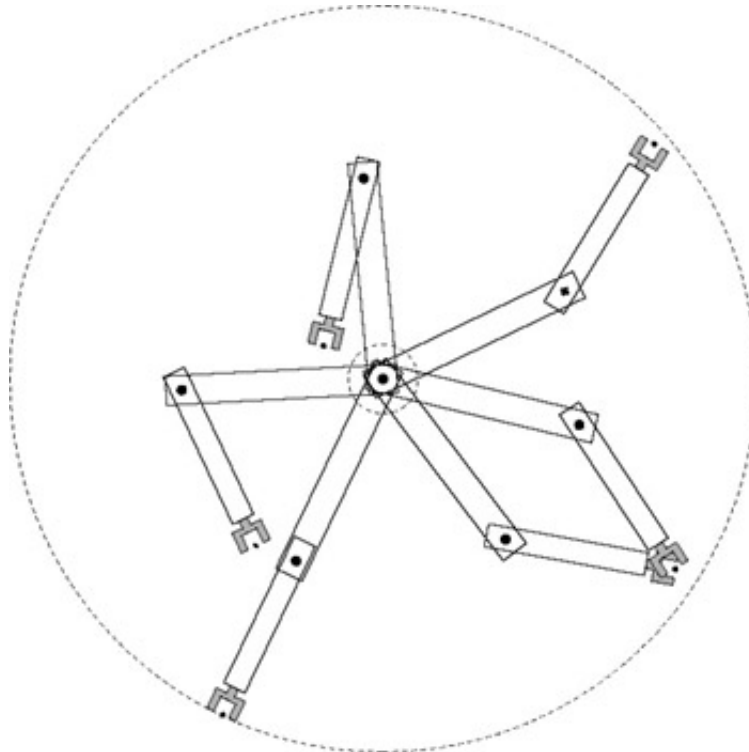


Figure 3.3: The workspace for this two-joint manipulator is an annulus, a disk with a smaller disk removed from it. Note that all points in the interior of the annulus are reachable with a right-arm configuration and a left-arm configuration.

Consider a two-joint planar robot arm, as shown in [figure 3.1](#). A point on the first link of the arm is pinned, so that the only possible motion of the first link is a rotation about this joint. Likewise, the base of the second link is pinned to a point at the end of the first link, and the only possible motion of the second link is a rotation about this joint. Therefore, if

we specify the parameters θ_1 and θ_2 , as shown in figure 3.1, we have specified the configuration of the arm. For now we will assume no joint limits, so the two links can move over each other.

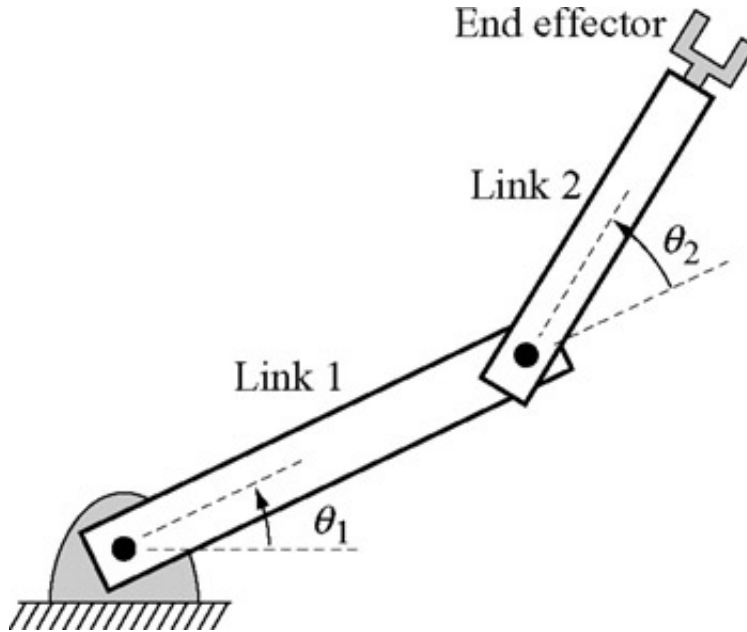


Figure 3.1: The angles θ_1 and θ_2 specify the configuration of the two-joint robot.

Each joint angle θ_i corresponds to a point on the unit circle S^1 , and the configuration space is $S^1 \times S^1 = T^2$, the two-dimensional torus. It is common to picture a torus as the surface of a doughnut because a torus has a natural embedding in \mathbb{R}^3 , just as a circle S^1 has a natural embedding in \mathbb{R}^2 . By cutting the torus along the $\theta_1 = 0$ and $\theta_2 = 0$ curves, we can flatten the torus onto the plane, as shown in figure 3.2. With this planar representation, we are identifying points on S^1 by points in the interval $[0, 2\pi) \subset \mathbb{R}$. While this representation covers all points in S^1 , the interval $[0, 2\pi)$, being a subset of the real line, does not naturally wrap around like S^1 , so there is a discontinuity in the representation. As we discuss in section 3.4, this is because S^1 is topologically different from any interval of \mathbb{R} .

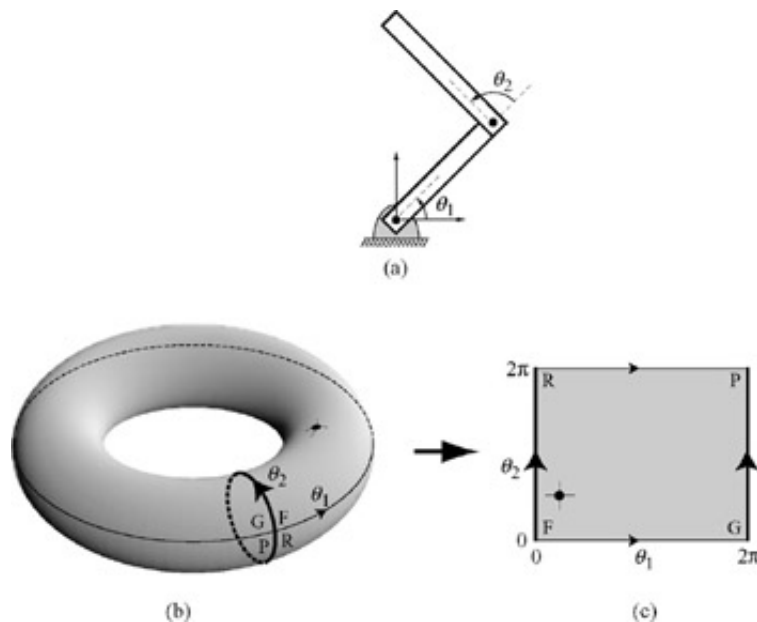


Figure 3.2: (a) A two-joint manipulator. (b) The configuration of the robot is represented as a point on the toral configuration space. (c) The torus can be cut and flattened onto the plane. This planar representation has "wraparound" features where the edge FR is connected to GP , etc.

We define the workspace of the two-joint manipulator to be the reachable points by the end effector. The workspace for our two-joint manipulator is an annulus (figure 3.3), which is a subset of \mathbb{R}^2 . All points in the interior of the annulus are reachable in two ways, with the arm in a *right-arm* and a *left-arm* configuration, sometimes called *elbow-up* and *elbow-down*. Therefore, the position of the end effector is *not* a valid configuration (not a complete description of the location of all points of the robot), so the annulus is not a configuration space for this robot.

So we have seen that the configuration spaces of both the translating mobile robot and the two-joint manipulator are two-dimensional, but they are quite different. The torus T^2 is doughnut-shaped with finite area, while \mathbb{R}^2 is flat with infinite area. We delve further into these sorts of differences when we discuss topology in section 3.4.

^[1]While q is used almost universally to denote a configuration, the configuration space is sometimes denoted by \mathcal{C} , particularly in the path-planning community.



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

◀ Previous

Next ▶

3.2 Obstacles and the Configuration Space

Equipped with our understanding of configurations and of configuration spaces, we can define the path-planning problem to be that of determining a continuous mapping, $c : [0, 1] \rightarrow \mathcal{Q}$, such that no configuration in the path causes a collision between the robot and an obstacle. It is useful to define explicitly the set of configurations for which such a collision occurs. We define a *configuration space obstacle* \mathcal{QO}_i to be the set of configurations at which the robot intersects an obstacle WO_i in the workspace, i.e.,

$$\mathcal{QO}_i = \{q \in \mathcal{Q} \mid R(q) \cap WO_i \neq \emptyset\}.$$

The *free space* or *free configuration space* $\mathcal{Q}_{\text{free}}$ is the set of configurations at which the robot does not intersect any obstacle, i.e.,

$$\mathcal{Q}_{\text{free}} = \mathcal{Q} \setminus \left(\bigcup_i \mathcal{QO}_i \right).$$

With this notation, we define a *free path* to be a continuous mapping

$c : [0, 1] \rightarrow \mathcal{Q}_{\text{free}}$, and a *semifree path* to be a continuous mapping $c : [0, 1] \rightarrow \text{cl}(\mathcal{Q}_{\text{free}})$, in which $\text{cl}(\mathcal{Q}_{\text{free}})$ denotes the closure of $\mathcal{Q}_{\text{free}}$. A free path does not allow contact between the robot and obstacles, while a semifree path allows the robot to contact the boundary of an obstacle. We assume that $\mathcal{Q}_{\text{free}}$ is open unless otherwise noted.

We now examine how obstacles in the workspace can be mapped into the configuration space for the robots that we discussed above.

3.2.1 Circular Mobile Robot

Consider the circular mobile robot in an environment with a single polygonal obstacle in the workspace, as shown in figure 3.4. In figure 3.4(b), we slide the robot around the obstacle to find the constraints the obstacle places on the configuration of the robot, i.e., the possible locations of the robot's reference point. We have chosen to use the center of the robot, but could easily choose another point. Figure 3.4(c) shows the resulting obstacle in the configuration space. Motion planning for the circular robot in figure 3.4(a)

is now equivalent to motion planning for a point in the configuration space, as shown in figure 3.4(c).

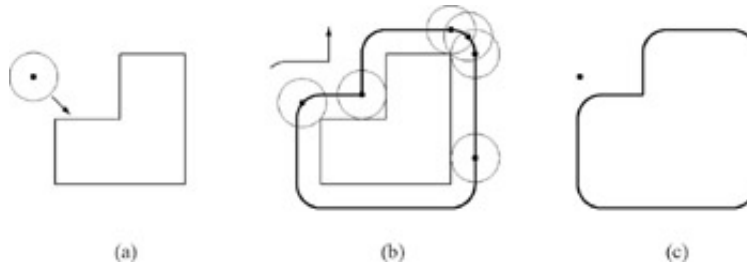


Figure 3.4: (a) The circular mobile robot approaches the workspace obstacle. (b) By sliding the mobile robot around the obstacle and keeping track of the curve traced out by the reference point, we construct the configuration space obstacle. (c) Motion planning for the robot in the workspace representation in (a) has been transformed into motion planning for a point robot in the configuration space.

Figure 3.5 shows three mobile robots of different radii in the same environment. In each case, the robot is trying to find a path from one configuration to another. To transform the workspace obstacles into configuration obstacles, we "grow" the polygon outward and the walls inward. The problem is now to find a path for the point robot in the configuration space. We see that the growing process has disconnected the free configuration space Q_{free} for the largest robot, showing that there is no solution for this robot.

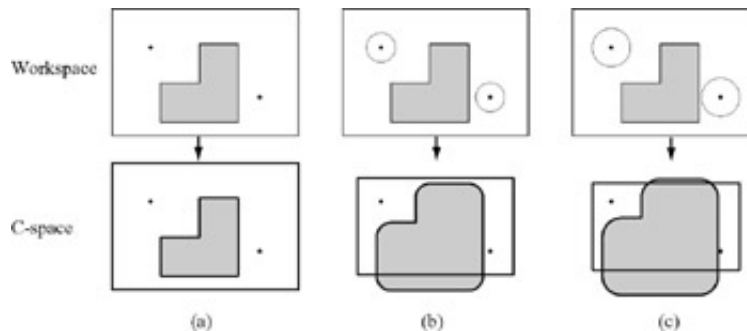


Figure 3.5: The top row shows the workspace and the bottom row shows the configuration space for (a) a point mobile robot, (b) a circular mobile robot, and (c) a larger circular mobile robot.

Although the example in figure 3.5 is quite simple, the main point is that it is easier to think about points moving around than bodies with volume. Keep in mind that although both the workspace and the configuration space for this system can be represented by \mathbb{R}^2 , and the obstacles appear to simply "grow" in this example, the configuration space and workspace are different spaces, and the transformation from workspace obstacles to configuration space obstacles is not always so simple.

For example, appendix F discusses how to generate configuration space obstacles for a polygon that translates and rotates among polygonal obstacles in the plane. The two-joint arm example is examined next.

3.2.2 Two-Joint Planar Arm

For the case of the circular mobile robot in a world populated with polygonal obstacles, it is easy to compute configuration space obstacles. When the robot is even slightly more complex, it becomes much more difficult to do so. For this reason, grid-based representations of the configuration space are sometimes used. Consider the case of the two-joint planar arm, for which $Q = T^2$. We can define a grid on the surface of the torus, and for each point on this grid we can perform a fairly simple test to see if the corresponding configuration causes a collision between the arm and any obstacle in the workspace. If we let each grid point be represented by a pixel, we can visualize the configuration space obstacle by "coloring" pixels appropriately. This method was used to obtain figures 3.6, 3.7, and 3.8.^[2] In each of the figures, the image on the left depicts a two-joint arm in a planar workspace, while the image on the right depicts the configuration space. In each case, the arm on the left is depicted in several configurations, and these are indicated in the configuration spaces on the right.

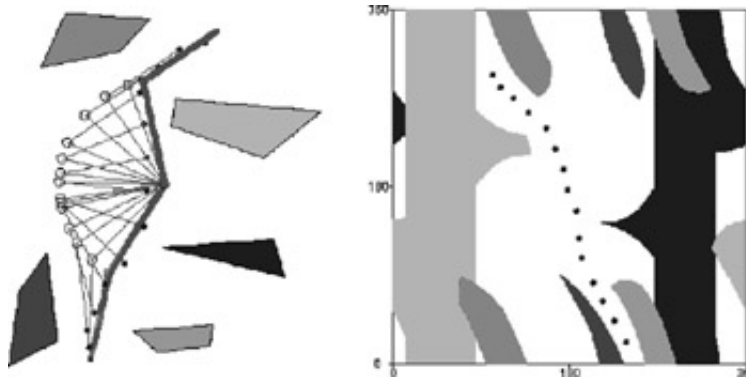


Figure 3.6: (Left) A path for the two-joint manipulator through its workspace, where the start and goal configurations are darkened. (Right) The path in the configuration space.

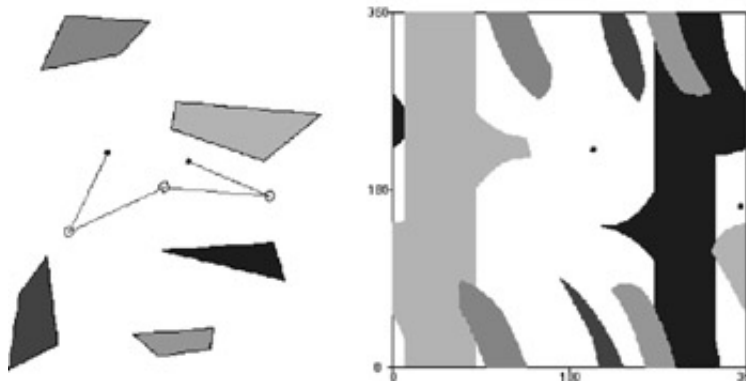


Figure 3.7: (Left) The workspace for a two-joint manipulator where the start and goal configurations are shown. (Right) The configuration space shows there is no free path between the start and goal configurations.

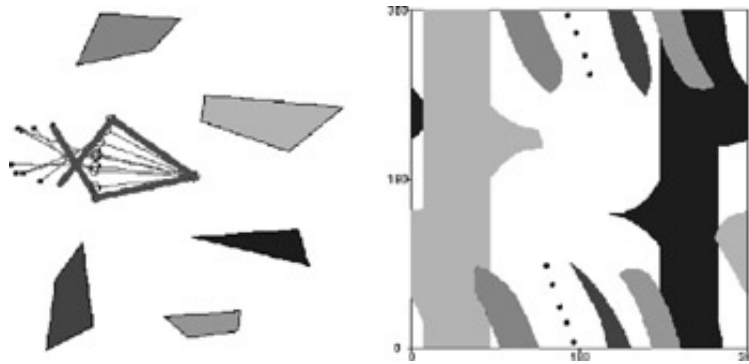


Figure 3.8: (Left) The workspace for a two-joint manipulator where the start and goal configurations are darkened. (Right) The path shows the wraparound of the planar representation of the configuration space.

While pictures such as those in figures 3.6, 3.7, and 3.8 are useful for visualizing configuration space obstacles, they are *not* sufficient for planning collision-free motions. The reason for this is that the grid only encodes collision information for the discrete set of points lying on the grid. A path includes not only grid points, but also the points on the curves that connect the grid points. One possible remedy for this problem is to "thicken" the robot when we test it at a grid point, so that if the thickened robot is collision-free, then paths to adjacent grid points are also collision-free.^[3] We could also choose to ignore this problem by choosing a grid resolution that is "high enough."

^[2]These figures were obtained using the applet at <http://ford.ieor.berkeley.edu/cspace/>

^[3]This approach is called *conservative*, as a motion planner using this approach will never find an incorrect solution, but it might miss solutions when they exist. As a result, the planner can only be resolution complete, not complete.

◀ Previous

Next ▶



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

[Previous](#)

[Next](#)

3.3 The Dimension of the Configuration Space

In our introduction to configuration space above, we restricted our attention to two-dimensional configuration spaces that are easy to visualize. For each example, it was fairly straightforward to conclude that there were only two degrees of freedom: for a translating robot, the configuration was specified by a point in the familiar Euclidean plane, while for the two-joint arm the two joint angles gave a complete specification of the arm's position. In this section we determine the number of degrees of freedom of more complex systems by considering constraints on the motions of individual points of the systems.

As a first example, suppose the robot is a point that can move in the plane. The configuration can be given by two coordinates, typically

$q = (x, y) \in Q = \mathbb{R}^2$, once we have chosen a reference coordinate frame fixed somewhere in space. Thus the robot has two degrees of freedom; the configuration space is two-dimensional.

Now consider a system consisting of three point robots, A , B , and C , that are free to move in the plane. Since the points can move independently, in order to specify the configuration of the system we need to specify the configuration of each of A , B , and C . By simply generalizing the case for a single point, we see that a configuration for this system can be given by the six coordinates x_A , y_A , x_B , y_B , x_C , and y_C (assuming that the points can overlap). The system has six degrees of freedom, and in this case we have

$$Q = \mathbb{R}^6.$$

Real robots are typically modeled as a set of rigid bodies connected by joints (or a single rigid body for the case of most mobile robots), not a set of points that are free to move independently. So, suppose now that the robot is a planar rigid body that can both translate and rotate in the plane. Define A , B , and C to be three distinct points that are fixed to the body. To place the body in the plane, we are first free to choose the position of A by choosing its coordinates (x_A, y_A) . Now we wish to choose the coordinates of B , (x_B, y_B) , but the rigidity of the body requires that this point maintain a constant distance $d(A, B)$ from A :

$$d(A, B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}.$$

This equation constrains B to lie somewhere on a circle of radius $d(A, B)$ centered at $(x_A,$

y_A), and our only freedom in placing B is the angle θ from A to B .

Now when we try to choose coordinates (x_C, y_C) for C , we see that our choice is subject to two constraints:

$$d(A, C) = \sqrt{(x_A - x_C)^2 + (y_A - y_C)^2}$$

$$d(B, C) = \sqrt{(x_B - x_C)^2 + (y_B - y_C)^2}$$

In other words, C has already been placed for us. In fact, every point on the body has been placed once we have chosen (x_A, y_A, θ) , making this a good representation of the configuration. The body has three degrees of freedom, and its configuration space is $\mathbb{R}^2 \times S^1$.

Each of the distance constraints above is an example of a *holonomic constraint*. A holonomic constraint is one that can be expressed purely as a function g of the configuration variables (and possibly time), i.e., of the form

$$g(q, t) = 0.$$

Each linearly independent holonomic constraint on a system reduces the dimension of the system's configuration space by one. Thus a system described by n coordinates subject to m independent holonomic constraints has an $(n - m)$ -dimensional configuration space. In this case, we normally attempt to represent the configuration space by a smaller set of $n - m$ coordinates subject to no constraints, e.g., the coordinates (x_A, y_A, θ) for the planar body above.

Nonholonomic constraints are velocity constraints of the form

$$g(q, \dot{q}, t) = 0$$

which do not reduce the dimension of the configuration space. Nonholonomic constraints are left to [chapter 12](#).

We can apply the counting method above to determine the number of degrees of freedom of a three-dimensional rigid body. Choose three noncollinear points on the body, A, B, C . The coordinates (x_A, y_A, z_A) of point A are first chosen arbitrarily. After fixing A , the distance constraint from A forces B to lie on the two-dimensional surface of a sphere centered at A . After both A and B are fixed, the distance constraints from A and B force C to lie on a one-dimensional circle about the axis formed by A and B . Once this point is chosen, all other points on the body are fixed. Thus the configuration of a rigid body in space can be described by nine coordinates subject to three constraints, yielding a six-dimensional configuration space.

We have already seen that a rigid body moving in a plane has three degrees of freedom, but we can arrive at this same conclusion if we imagine a spatial rigid body with six degrees of freedom with a set of constraints that restricts it to a plane. Choose this book as an example, using three corners of the back cover as points A, B and C . The book can be confined to a plane (e.g., the plane of a tabletop) using the three holonomic constraints

$$z_A = z_B = z_C = 0.$$

The two-joint planar arm can be shown to have two degrees of freedom by this (somewhat indirect) counting method. Each of the two links can be thought of as a rigid spatial body with six degrees of freedom. Six constraints restrict the bodies to a plane (three for each link), two constraints restrict a point on the first link (the first joint) to be at a fixed location in the plane, and once the angle of the first link is chosen, two constraints restrict a point on the second link (the second joint) to be at a fixed location. Therefore we have $(12 \text{ coordinates}) - (10 \text{ constraints}) = 2 \text{ degrees of freedom}$.

Of course we usually count the degrees of freedom of an open-chain jointed robot, also known as a *serial mechanism*, by adding the degrees of freedom at each joint. Common joints with one degree of freedom are *revolute* (R) joints, joints which rotate about an axis, and *prismatic* (P) joints, joints which allow translational motion along an axis. Our two-joint robot is sometimes called an RR or 2R robot, indicating that both joints are revolute. An RP robot, on the other hand, has a revolute joint followed by a prismatic joint. Another common joint is a *spherical* (ball-and-socket) joint, which has three degrees of freedom.

A *closed-chain* robot, also known as a *parallel mechanism*, is one where the links form one or more closed loops. If the mechanism has k links, then one is designated as a stationary "ground" link, and $k - 1$ links are movable. To determine the number of degrees of freedom, note that each movable link has N degrees of freedom, where $N = 6$ for a spatial mechanism and $N = 3$ for a planar mechanism. Therefore the system has $N(k - 1)$ degrees of freedom before the joints are taken into account. Now each of the n joints between the links places $N - f_i$ constraints on the feasible motions of the links, where f_i is the number of degrees of freedom at joint i (e.g., $f_i = 1$ for a revolute joint, and $f_i = 3$ for a spherical joint). Therefore, the *mobility* M of the mechanism, or its number of degrees of freedom, is given by

$$\begin{aligned} (3.1) \quad M &= N(k - 1) - \sum_{i=1}^n (N - f_i) \\ (3.1) \quad &= N(k - n - 1) + \sum_{i=1}^n f_i. \end{aligned}$$

This is known as *Grübler's formula* for closed chains, and it is only valid if the constraints due to the joints are independent. In the planar mechanism of [figure 3.9](#), there are seven joints, each with one degree of freedom, and six links, yielding a mobility of $3(6 - 7 - 1) + 7 = 1$.

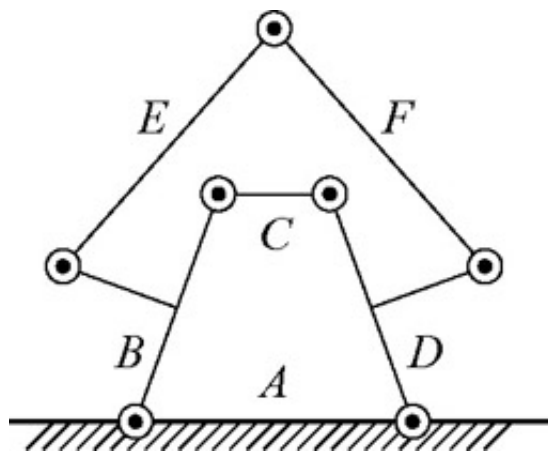


Figure 3.9: A planar mechanism with six links (*A* through *F*), seven revolute joints, and one degree of freedom.

As an application of the ideas in this section, determine the number of degrees of freedom of your arm by adding the degrees of freedom at your shoulder, elbow, and wrist. To test your answer, place your palm flat down on a table with your elbow bent. Without moving your torso or your palm, you should find that it is still possible to move your arm. This internal freedom means that your arm is *redundant* with respect to the task of positioning your hand (or a rigid body grasped by your hand) in space; an infinity of arm configurations puts your hand in the same place.^[4] How many internal degrees of freedom do you have? How many holonomic constraints were placed on your arm's configuration when you fixed your hand's position and orientation? The sum of the number of constraints and internal degrees of freedom is the number of degrees of freedom of your (unconstrained) arm, and you should find that your arm has more than six degrees of freedom. A robot is said to be *hyper-redundant* with respect to a task if it has many more degrees of freedom than required for the task. (There is no strict definition of "many" here.)

^[4]Provided your arm is away from its joint limits.

◀ Previous

Next ▶



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

[Previous](#)

[Next](#)

3.4 The Topology of the Configuration Space

Now that we understand how to determine the dimension of a configuration space, we can begin to explore its topology and geometry, each of which plays a vital role in developing and analyzing motion-planning algorithms. Some basic concepts from topology are discussed in [appendixes B and C](#).

Topology is a branch of mathematics that considers properties of objects that do not change when the objects are subjected to arbitrary continuous transformations, such as stretching or bending. For this reason, topology is sometimes referred to as "rubber sheet geometry." Imagine a polygon drawn on a rubber sheet. As the sheet is stretched in various directions, the polygon's shape changes; however, certain properties of the polygon do not change. For example, points that are inside the polygon do not move to the outside of the polygon simply because the sheet is stretched.

Two spaces are topologically different if cutting or pasting is required to turn one into the other, as cutting and pasting are not continuous transformations. For example, the configuration spaces of the circular mobile robot (\mathbb{R}^2) and the two-joint planar arm (T^2) are topologically different. If we imagine \mathbb{R}^2 as the surface of a rubber doughnut, we see that no matter how we stretch or deform the doughnut (without tearing it), the doughnut will always have a hole in it. Also, if we imagine \mathbb{R}^2 as an infinite rubber sheet, there is no way to stretch it (without tearing it) such that a hole will appear in the sheet. To a topologist, all rubber doughnuts are the same, regardless of how they are stretched or deformed ([figure 3.10](#)). Likewise, all rubber sheet versions of \mathbb{R}^2 are the same.

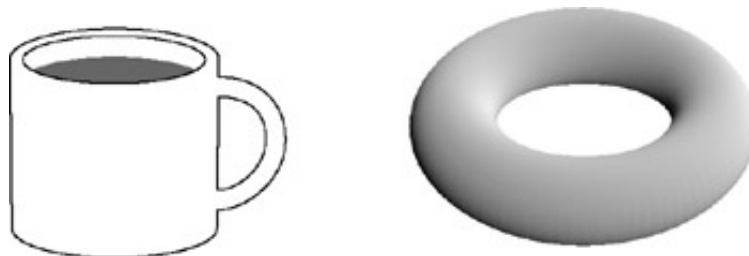


Figure 3.10: The surfaces of the coffee mug and the torus are topologically equivalent.

One reason that we care about the topology of configuration space is that it will affect our representation of the space. Another reason is that if we can derive a path-planning

algorithm for one kind of topological space, then that algorithm may carry over to other spaces that are topologically equivalent (see, e.g., [chapter 4, section 4.6](#)).

Since topology is concerned with properties that are preserved under continuous transformations, we begin our study of the topology of configuration spaces by describing two types of continuous transformations: homeomorphisms and diffeomorphisms. [Appendix C](#) provides an introduction to differentiable transformations.

3.4.1 Homeomorphisms and Diffeomorphisms

A mapping $\phi: S \rightarrow T$ is a rule that places elements of S into correspondence with elements of T . We respectively define the *image* of S under ϕ and the *preimage* of T by

$$\phi(S) = \{\phi(s) \mid s \in S\} \text{ and } \phi^{-1}(T) = \{s \mid \phi(s) \in T\}.$$

If $\phi(S) = T$ (i.e., every element of T is covered by the mapping) then we say that ϕ is *surjective* or *onto*. If ϕ puts each element of T into correspondence with at most one element of S , i.e., for any $t \in T$, $\phi^{-1}(t)$ consists of at most one element in S , then we say that ϕ is *injective* (*one-to-one*). If ϕ is injective, then when $s_1 \neq s_2$ we have $\phi(s_1) \neq \phi(s_2)$ for $s_1, s_2 \in S$. Maps that are both surjective and injective are said to be *bijective*. [Figure 3.11](#) illustrates these definitions. As another example, the map $\sin: (-\frac{\pi}{2}, \frac{\pi}{2}) \rightarrow (-1, 1)$ is bijective, whereas $\sin: \mathbb{R} \rightarrow [-1, 1]$ is only surjective.

Bijective maps have the property that their inverse exists at all points in the range T , and thus they allow us to move easily back and forth between the two spaces S and T . In our case, we will use bijective maps to move back and forth between configuration spaces (whose geometry can be quite complicated) and Euclidean spaces.

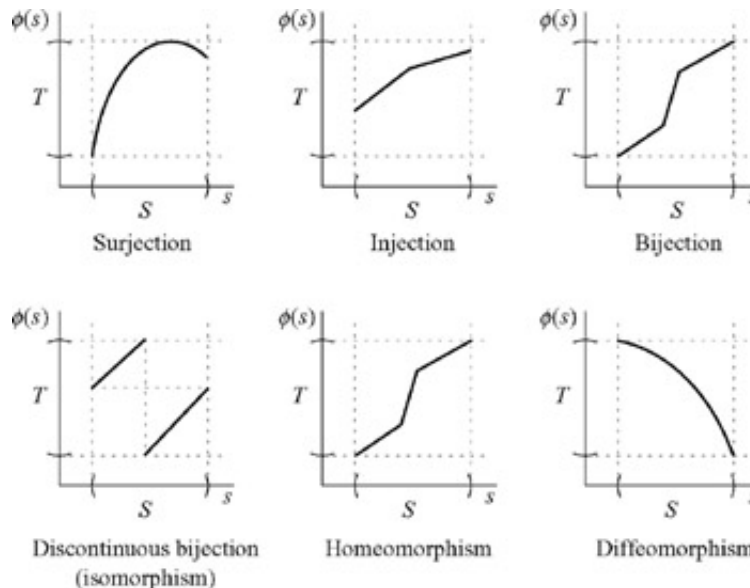


Figure 3.11: Representative ways of looking at surjective, injective, and bijective mappings. Bijections may become homeomorphisms or diffeomorphisms if they are sufficiently differentiable.

We will consider two important classes of bijective mappings.

DEFINITION 3.4.1

If $\varphi: S \rightarrow T$ is a bijection, and both φ and φ^{-1} are continuous, then φ is a *homeomorphism*. When such a φ exists, S and T are said to be *homeomorphic*.

A mapping $\varphi: U \rightarrow V$ is said to be *smooth* if all partial derivatives of φ , of all orders, are well defined (i.e., φ is of class C^∞). With the notion of smoothness, we define a second type of bijection.

DEFINITION 3.4.2

A smooth map $\varphi: U \rightarrow V$ is a *diffeomorphism* if φ is bijective and φ^{-1} is smooth. When such a φ exists, U and V are said to be *diffeomorphic*.

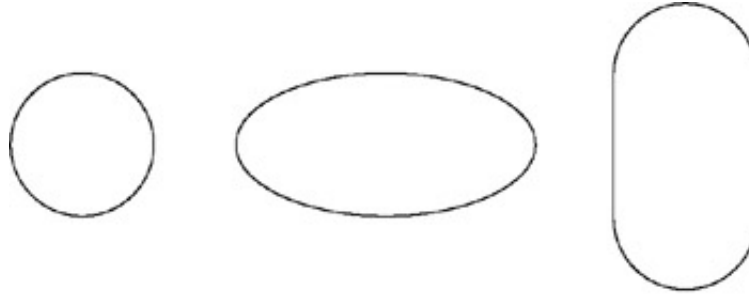
The condition for diffeomorphisms (smoothness) is stronger than the condition for homeomorphisms (continuity), and thus all diffeomorphisms are homeomorphisms.

To illustrate these ideas, consider three one-dimensional surfaces: a circle, denoted by \mathcal{M}_c ; an ellipse, denoted by \mathcal{M}_e ; and a "racetrack," denoted by \mathcal{M}_r . The racetrack consists of two half-circles connected by straight lines (figure 3.12). We define these shapes mathematically as

$$(3.2) \quad \mathcal{M}_c = \{(x, y) \mid f_c(x, y) = x^2 + y^2 - 1 = 0\}$$

$$(3.3) \quad \mathcal{M}_e = \{(x, y) \mid f_e(x, y) = \frac{x^2}{4} + y^2 - 1 = 0\}$$

$$(3.4) \quad \mathcal{M}_r = \{(x, y) \mid f_r(x, y) = 0\}$$



Circle

Ellipse

Racetrack

Figure 3.12: A circle, an ellipse, and a racetrack.

with

$$(3.5)$$

$$(3.5) \quad f_r(x, y) = \begin{cases} x - 1 & : -1 \leq y \leq 1, x > 0 \\ (y + 1)^2 + x^2 - 1 & : y < -1 \\ (y - 1)^2 + x^2 - 1 & : y > 1 \\ x + 1 & : -1 \leq y \leq 1, x < 0 \end{cases}.$$

Note that these surfaces are implicitly defined as being the set of points that satisfy some equation $f(x, y) = 0$.

In some ways, these three surfaces are similar. For example, they are all simple, closed curves in the plane; all of $f_c(x, y)$, $f_e(x, y)$, and $f_r(x, y)$ are continuous. In other ways, they seem quite different. For example, both $f_c(x, y)$ and $f_e(x, y)$ are continuously differentiable, while $f_r(x, y)$ is not. We can more precisely state the similarities and differences between these surfaces using the concepts of homeomorphism and diffeomorphism. In particular, it can be shown that \mathcal{M}_c , \mathcal{M}_e , and \mathcal{M}_r are all homeomorphic to each other. For example, the map $\phi : \mathcal{M}_e \rightarrow \mathcal{M}_c$ given by

$$\phi(x, y) = \left[\frac{x}{\sqrt{x^2 + y^2}}, \frac{y}{\sqrt{x^2 + y^2}} \right]^T$$

is a homeomorphism.

For this choice of ϕ , both ϕ and ϕ^{-1} are smooth, and therefore, \mathcal{M}_c is diffeomorphic to \mathcal{M}_e . Neither \mathcal{M}_c nor \mathcal{M}_e is diffeomorphic to \mathcal{M}_r , however. This is because $f_r(x, y)$ is not continuously differentiable, while both $f_c(x, y)$ and $f_e(x, y)$ are. This can be seen by examining the curvatures of the circle, ellipse, and racetrack. For the circle, the curvature is constant (and thus continuous), and for the ellipse, curvature is continuous. For the racetrack, there are discontinuities in curvature (at the points $(-1, 1)$, $(-1, -1)$, $(1, 1)$, $(1, -1)$), and therefore there is no smooth mapping from either the circle or the ellipse to the racetrack.

We are often concerned only with the local properties of configuration spaces. Local properties are defined on *neighborhoods*. For metric spaces^[5], neighborhoods are most easily defined in terms of open balls. For a point p of some manifold \mathcal{M} , we define an open ball of radius ϵ by

$$B_\epsilon(p) = \{p' \in \mathcal{M} \mid d(p, p') < \epsilon\},$$

where d is a metric on \mathcal{M} .^[6] A neighborhood of a point $p \in \mathcal{M}$ is any subset $\mathcal{U} \subseteq \mathcal{M}$ with $p \in \mathcal{U}$ such that for every $p' \in \mathcal{U}$, there exists an open ball $B_\epsilon(p') \subset \mathcal{U}$. Any open ball is itself a neighborhood. The open disk in the plane is an example. For the point (x_0, y_0) in the plane, an open ball defined by the Euclidean metric is

$$B_\epsilon(x_0, y_0) = \{(x, y) \mid (x - x_0)^2 + (y - y_0)^2 < \epsilon^2\}.$$

We say that S is *locally diffeomorphic* (resp. *locally homeomorphic*) to T if for each $p \in S$ there exists a diffeomorphism (resp. homeomorphism) f from S to T on some neighborhood \mathcal{U} with $p \in \mathcal{U}$.

The sphere presents a familiar example of these concepts. At any point on the sphere, there exists a neighborhood of that point that is diffeomorphic to the plane. It is not surprising that people once believed the world was flat - they were only looking at their neighborhoods!

Let us now reflect on the two examples from the beginning of this chapter. For the circular mobile robot, the workspace and the configuration space are diffeomorphic. This is easy to see, since both are copies of \mathbb{R}^2 . In this case, the identity map $\phi(x) = x$ is a perfectly fine global diffeomorphism between the workspace and configuration space. In contrast, the two-joint manipulator has a configuration space that is T^2 , the torus. The torus T^2 is not diffeomorphic to \mathbb{R}^2 , but it is locally diffeomorphic. If the revolute joints in the two-joint manipulator have lower and upper limits, $\theta_j^l < \theta_j < \theta_j^u$, so that they cannot perform a complete revolution, however, then the configuration space of the two-joint manipulator becomes an open subset of the torus, which is diffeomorphic to \mathbb{R}^2 (globally). This follows from the fact that each joint angle lies in an *open* interval of \mathbb{R}^1 , and we can "stretch" that open interval to cover the line. An example of such a stretching function is $\tan : (-\frac{\pi}{2}, \frac{\pi}{2}) \rightarrow \mathbb{R}$.

3.4.2 Differentiable Manifolds

For all of the configuration spaces that we have seen so far, we have been able to uniquely specify a configuration by n parameters, where n is the dimension of the configuration space (two for the planar two-joint arm, three for a polygon in the plane, etc.). The reason that we could do so was that these configuration spaces were all "locally like" n -dimensional Euclidean spaces. Such spaces, called *manifolds*, are a central topic of topology.

DEFINITION 3.4.3

(Manifold) A set S is a k -dimensional manifold if it is locally homeomorphic to \mathbb{R}^k , meaning that each point in S possesses a neighborhood that is homeomorphic to an open set in \mathbb{R}^k .

While a general k -dimensional manifold is locally homeomorphic to \mathbb{R}^k , the configuration spaces that we will consider are locally diffeomorphic to \mathbb{R}^k , an even stronger relationship. In fact, when we parameterized configurations in [section 3.1](#), we were merely constructing diffeomorphisms from configuration spaces to \mathbb{R}^2 . If we construct

enough of these diffeomorphisms (so that every configuration in \mathcal{Q} is in the domain of at least one of them), and if these diffeomorphisms are compatible with one another (an idea that we will formalize shortly), then this set of diffeomorphisms together with the configuration space define a *differentiable manifold*. We now make these ideas more precise.

DEFINITION 3.4.4

(Chart) A pair (U, ϕ) , such that U is an open set in a k -dimensional manifold and ϕ is a diffeomorphism from U to some open set in \mathbb{R}^k , is called a *chart*.

The use of the term *chart* is analogous to its use in cartography, since the subset U is "charted" onto \mathbb{R}^k in much the same way that cartographers chart subsets of the globe onto a plane when creating maps. Charts are sometimes referred to as *coordinate systems* because each point in the set U is assigned a set of coordinates in a Euclidean space [410]. The inverse diffeomorphism, $\phi^{-1} : \mathbb{R}^k \rightarrow U$, is referred to as a *parameterization* of the manifold.

As an example, consider the one-dimensional manifold

$S^1 = \{x = (x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 = 1\}$. For any point $x \in S^1$ we can define a neighborhood that is diffeomorphic to \mathbb{R} . For example, consider the upper portion of the circle, $U_1 = \{x \in S^1 \mid x_2 > 0\}$. The chart $\phi_1 : U_1 \rightarrow (0, 1)$ is given by $\phi_1(x) = x_1$, and thus x_1 can be used to define a local coordinate system for the upper semicircle. In the other direction, the upper portion of the circle can be parameterized by $z \in (0, 1) \subset \mathbb{R}$ with $\phi_1^{-1}(z) = (z, (1 - z^2)^{1/2})$, which maps the open unit interval to the upper semicircle. But S^1 is not globally diffeomorphic to \mathbb{R}^1 ; we cannot find a single chart whose domain includes all of S^1 .

We have already used this terminology in [section 3.1](#), when we referred to θ_1, θ_2 as parameters that represent a configuration of the two-joint arm. Recall that $(\theta_1, \theta_2) \in \mathbb{R}^2$, and that when considered as a representation of the configuration, they define a point in T^2 , the configuration space, which is a manifold. We now see that in [section 3.1](#), when we represented a configuration of the planar arm by the pair (θ_1, θ_2) , we were in fact creating a chart from a subset of the configuration space to a subset of \mathbb{R}^2 .

A single mapping from T^2 to \mathbb{R}^2 shown in [figure 3.2](#) encounters continuity problems at $\theta_i = \{0, 2\pi\}$. For many interesting configuration spaces, it will be the case that we cannot construct a single chart whose domain contains the entire configuration space. In these cases, we construct a collection of charts that cover the configuration space. We are not free to choose these charts arbitrarily; any two charts in this collection must be compatible for parts of the manifold on which their domains overlap. Two charts with such compatibility are said to be C^∞ -related ([figure 3.13](#)).

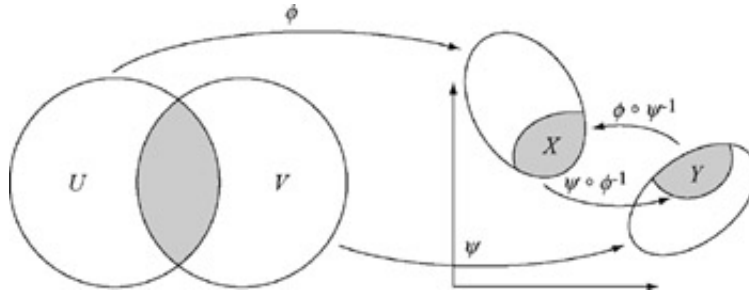


Figure 3.13: The charts (U, ϕ) and (V, ψ) map open sets on the k -dimensional manifold to open sets in \mathbb{R}^k .

DEFINITION 3.4.5: (C^∞ -related)

Let (U, ϕ) and (V, ψ) be two charts on a k -dimensional manifold. Let X be the image of $U \cap V$ under ϕ , and Y be the image of $U \cap V$ under ψ , i.e.,

$$X = \{\phi(x) \in \mathbb{R}^k \mid x \in U \cap V\}$$

$$Y = \{\psi(y) \in \mathbb{R}^k \mid y \in U \cap V\}.$$

These two charts are said to be C^∞ -related if both of the composite functions

$$\psi \circ \phi^{-1} : X \rightarrow Y,$$

$$\phi \circ \psi^{-1} : Y \rightarrow X$$

are C^∞ .

If two charts are C^∞ -related, we can switch back and forth between them in a smooth way when their domains overlap. This idea will be made more concrete in the example of S^1 below.

A set of charts that are C^∞ -related, and whose domains cover the entire configuration space \mathcal{Q} , form an *atlas* for \mathcal{Q} . An atlas is sometimes referred to as the differentiable structure for \mathcal{Q} . Together, the atlas and \mathcal{Q} comprise a *differentiable manifold*. There are other ways to define differentiable manifolds, as we will see in [section 3.5](#).

As an example, consider again the one-dimensional manifold S^1 . Above, we defined a single chart, (U_1, ϕ_1) . If we define three more charts, we can construct an atlas for S^1 . These four charts are given by

$$U_1 = \{x \in S^1 \mid x_2 > 0\}, \quad \phi_1(x) = x_1$$

$$U_2 = \{x \in S^1 \mid x_2 < 0\}, \quad \phi_2(x) = x_1$$

$$U_3 = \{x \in S^1 \mid x_1 > 0\}, \quad \phi_3(x) = x_2$$

$$U_4 = \{x \in S^1 \mid x_1 < 0\}, \quad \phi_4(x) = x_2.$$

The corresponding parameterizations are given by $\phi_i^{-1} : (-1, 1) \rightarrow U_i$, with

$$\phi_1^{-1}(z) = (z, 1 - z^2)$$

$$\phi_2^{-1}(z) = (z, z^2 - 1)$$

$$\phi_3^{-1}(z) = (1 - z^2, z)$$

$$\phi_4^{-1}(z) = (z^2 - 1, z).$$

It is clear that the U_i cover S^1 , so to verify that these charts form an atlas it is only necessary to show that they are C^∞ -related ([figure 3.14](#)). Note that

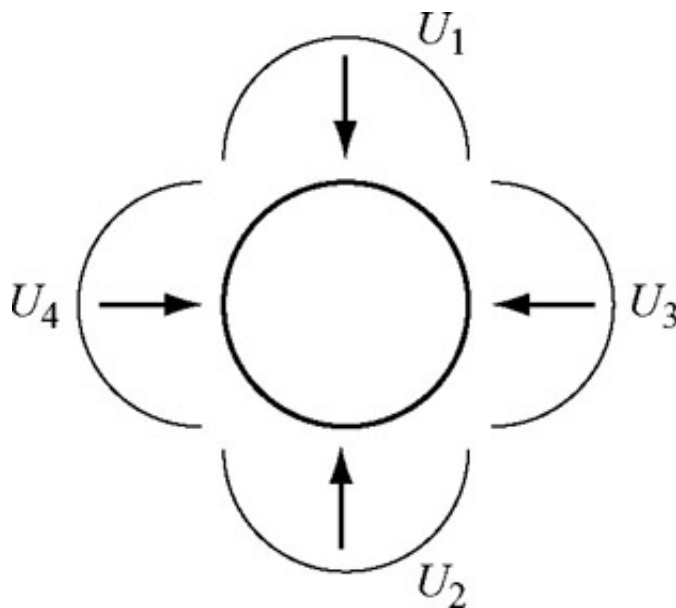


Figure 3.14: Four charts covering the circle S^1 .

$U_1 \cap U_2 = U_3 \cap U_4 = \emptyset$, so we need only check the four pairs of composite maps:

$$\begin{aligned}
\phi_1 \circ \phi_3^{-1} &: (0, 1) \rightarrow (0, 1), & \phi_3 \circ \phi_1^{-1} &: (0, 1) \rightarrow (0, 1) \\
\phi_1 \circ \phi_4^{-1} &: (0, 1) \rightarrow (-1, 0), & \phi_4 \circ \phi_1^{-1} &: (-1, 0) \rightarrow (0, 1) \\
\phi_2 \circ \phi_3^{-1} &: (-1, 0) \rightarrow (0, 1), & \phi_3 \circ \phi_2^{-1} &: (0, 1) \rightarrow (-1, 0) \\
\phi_2 \circ \phi_4^{-1} &: (-1, 0) \rightarrow (-1, 0), & \phi_4 \circ \phi_2^{-1} &: (-1, 0) \rightarrow (-1, 0).
\end{aligned}$$

In each case, $\phi_i \circ \phi_j^{-1}(z)$ is smooth on each of the open unit intervals that define the domains for the composite mappings given above. For example, $\phi_1 \circ \phi_3^{-1}(z) = 1 - z^2$.

This collection of four charts is not minimal; it is straightforward to find two charts to cover S^1 (see problem 9).

3.4.3 Connectedness and Compactness

We say that a manifold is *path-connected*, or just *connected*, if there exists a path between any two points of the manifold.^[7] All of the obstacle-free configuration spaces that we will consider in this text, e.g., \mathbb{R}^n , S^n , and T^n , are connected. The presence of obstacles, however, can disconnect the free configuration space $\mathcal{Q}_{\text{free}}$. In this case, the free configuration space is broken into a set of *connected components*, the maximal connected subspaces. In figure 3.5(c), for example, obstacles break the mobile robot's free configuration space into two connected components. There can be no solution to a motion-planning problem if q_{start} and q_{goal} do not lie in the same connected component of $\mathcal{Q}_{\text{free}}$.

A space is *compact*^[8] if it resembles a closed, bounded subset of \mathbb{R}^n . A space is closed if it includes all of its limit points. As examples, the half-open interval $[0, 1) \subset \mathbb{R}$ is bounded but not compact, while the closed interval $[0, 1]$ is bounded and compact. The space \mathbb{R}^n is not bounded and therefore not compact. The spaces S^n and T^n are both compact, as they can be expressed as closed and bounded subsets of Euclidean spaces. The unit circle S^1 , e.g., can be expressed as a closed and bounded subset of \mathbb{R}^2 .

In configuration spaces with obstacles or joint limits, the modeling of the obstacles may affect whether the space is compact or not. For a revolute joint subject to joint limits, the set of joint configurations is compact if the joint is allowed to hit the limits, but not compact if the joint can only approach the limits arbitrarily closely.

The product of compact configuration spaces is also compact. For a noncompact space $\mathcal{M}_1 \times \mathcal{M}_2$, if \mathcal{M}_1 is compact, then it is called the *compact factor* of the space. Compact and noncompact spaces cannot be diffeomorphic.

3.4.4 Not All Configuration Spaces Are Manifolds

We are focusing on configuration spaces that are manifolds, and more specifically differentiable manifolds, but it is important to keep in mind that not all configuration spaces are manifolds. As a simple example, the closed unit square $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ is not a manifold, but a *manifold with boundary* obtained by

pasting the one-dimensional boundary on the two-dimensional open set $(0, 1) \times (0, 1)$. Also, some parallel mechanisms with one degree of freedom have configurations from which there are two distinct possible motion directions (i.e., the configuration space is a self-intersecting figure eight). It is beyond the scope of this chapter to discuss other types of configuration spaces, but be aware: if you cannot show it to be a manifold, it may not be!


[5] A metric space is a space equipped with a distance metric. See [appendix C](#).

[6] One can define all topological properties, including neighborhoods, without resorting to the use of metrics, but for our purposes, it will be easier to assume a metric on the configuration space and exploit the metric properties.

[7] For more general spaces, the concepts of path-connectedness and connectedness are not equivalent, but for a manifold they are the same. More generally, a space is connected if there is no continuous mapping from the space to a discrete set of more than one element.

[8] In topology, a space is defined to be compact if every open cover of the space admits a finite subcover, but we will not use these concepts here.

 Previous

Next 

TeamUnknown Release



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

◀ Previous

Next ▶

3.5 Embeddings of Manifolds in \mathbb{R}^n

Although a k -dimensional manifold can be represented using as few as k parameters, we have seen above that doing so may require multiple charts. An alternative is to use a representation with "extra" numbers, subject to constraints, to achieve a single global representation. As an example, S^1 is a one-dimensional manifold that we can represent as $S^1 = \{(x, y) \mid x^2 + y^2 = 1\}$; we "embed" S^1 in \mathbb{R}^2 . The fact that we cannot find a single chart for all of S^1 tells us that we cannot embed S^1 in \mathbb{R}^1 . Likewise, although the torus T^2 is a two-dimensional manifold, it is not possible to embed the torus in \mathbb{R}^2 , which is why we typically illustrate the torus as a doughnut shape in \mathbb{R}^3 .

The manifolds S^1 and T^2 can be viewed as *submanifolds* of \mathbb{R}^2 and \mathbb{R}^3 , respectively. Submanifolds are smooth subsets of an ambient space that inherit the differentiability properties of the ambient space. Submanifolds are often created by a smooth set of equality constraints on \mathbb{R}^n , as we see in the example of the circle S^1 above. Any differentiable manifold can be viewed as an embedded submanifold of \mathbb{R}^n for large enough n .

When we are confronted with a configuration space that does not permit a single global coordinate chart, we are faced with a choice. We can either use a single set of parameters and suffer the consequences of singularities and discontinuities in the representation, use multiple charts to construct an atlas, or use a single global representation by embedding the configuration space in a higher-dimensional space. One advantage of the last approach is that the representation can facilitate other operations. Important examples are representations of orientation using complex numbers and quaternions (see [appendix E](#)) and matrix representations for the configuration of a rigid body in the plane or in space, as discussed next.

3.5.1 Matrix Representations of Rigid-Body Configuration

It is often convenient to represent the position and orientation of a rigid body using an $m \times m$ matrix of real numbers. The m^2 entries of this matrix must satisfy a number of smooth equality constraints, making the manifold of such matrices a submanifold of \mathbb{R}^{m^2} . One advantage of such a representation is that these matrices can be multiplied to get another matrix in the manifold. More precisely, these matrices form a *group* with the group

operation of matrix multiplication.^[9] Simple matrix multiplication can be used to change the reference frame of a representation or to rotate and translate a configuration.

We describe the orientation of a rigid body in n -dimensional space ($n = 2 \text{ or } 3$) by the matrix groups $SO(n)$, and the position and orientation by the matrix groups $SE(n)$. After describing these representations abstractly, we look in detail at examples of the use of $SE(n)$ for representing the configuration of a body, for changing the reference frame of the representation, and for translating and rotating a configuration.

Orientation: $SO(2)$ and $SO(3)$

Figure 3.15 shows a rigid body with a frame x - y - z attached to it. Our representation of the orientation of the body will be as a 3×3 matrix

$$R = \begin{bmatrix} \tilde{x}_1 & \tilde{y}_1 & \tilde{z}_1 \\ \tilde{x}_2 & \tilde{y}_2 & \tilde{z}_2 \\ \tilde{x}_3 & \tilde{y}_3 & \tilde{z}_3 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \in SO(3),$$

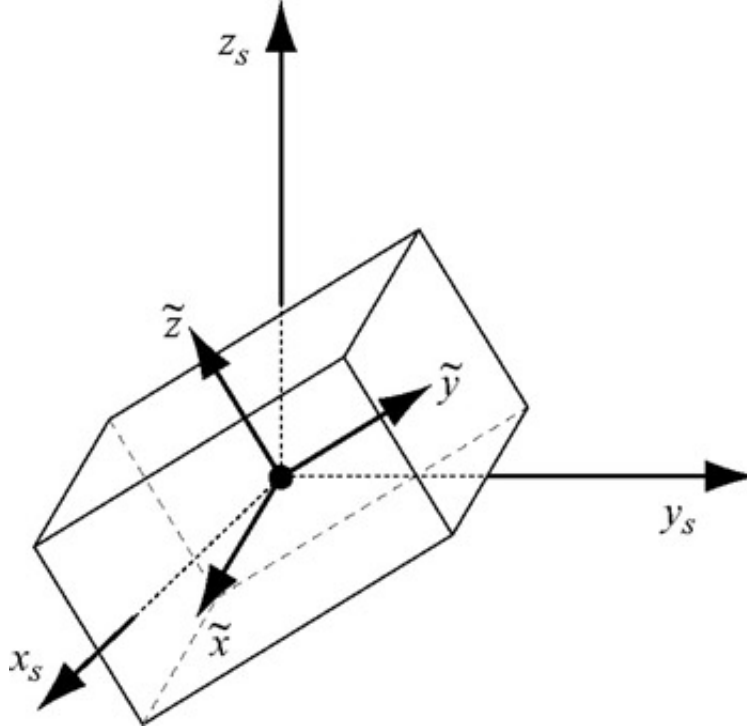


Figure 3.15: The rotation matrix for a body is obtained by expressing the unit vectors \tilde{x} , \tilde{y} , and \tilde{z} of the body x - y - z frame in a stationary frame x_s - y_s - z_s .

where $\tilde{x} = [\tilde{x}_1, \tilde{x}_2, \tilde{x}_3]^T$ is the unit vector in the body x -direction expressed in a stationary coordinate frame x_s - y_s - z_s . The vectors \tilde{y} and \tilde{z} are defined similarly (see figure 3.15).

The matrix R is often called the *rotation matrix* representation of the orientation. This

representation uses nine numbers to represent the three angular degrees of freedom, so there are six independent constraints on the matrix entries: each column (and row) is a unit vector,

$$||\tilde{x}|| = ||\tilde{y}|| = ||\tilde{z}|| = 1,$$

yielding three constraints, and the columns (and rows) are orthogonal to each other,

$$\tilde{x}^T \tilde{y} = \tilde{y}^T \tilde{z} = \tilde{z}^T \tilde{x} = 0,$$

yielding three more constraints. Because we are assuming right-handed frames,^[10] the determinant of R is +1. Matrices satisfying these conditions belong to the *special orthogonal group* of 3×3 matrices $SO(3)$. "Special" refers to the fact that the determinant is +1, not -1.

In the planar case, R is the 2×2 matrix

$$R = \begin{bmatrix} \tilde{x}_1 & \tilde{y}_1 \\ \tilde{x}_2 & \tilde{y}_2 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \in SO(2),$$

where θ is the orientation of the x - y frame relative to the x_s - y_s frame.

Generalizing, orientations in n -dimensional space can be written

$$SO(n) = \{R \in \mathbb{R}^{n \times n} \mid RR^T = \mathcal{I}, \det(R) = 1\},$$

where \mathcal{I} is the identity matrix. This implies the relation

$$R^T = R^{-1}.$$

Position and Orientation: $SE(2)$ and $SE(3)$

Figure 3.16 shows a rigid body with an attached x - y - z coordinate frame relative to a stationary frame x_s - y_s - z_s . Let $p = [p_1, p_2, p_3]^T \in \mathbb{R}^3$ be the vector from the origin of the stationary frame to the body frame, as measured in the stationary frame, and let $R \in SO(3)$ be the rotation matrix as described above, as if the body frame were translated back to the stationary frame. Then we represent the position and orientation of the body frame relative to the stationary frame as the 4×4 transform matrix

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \in SE(3),$$

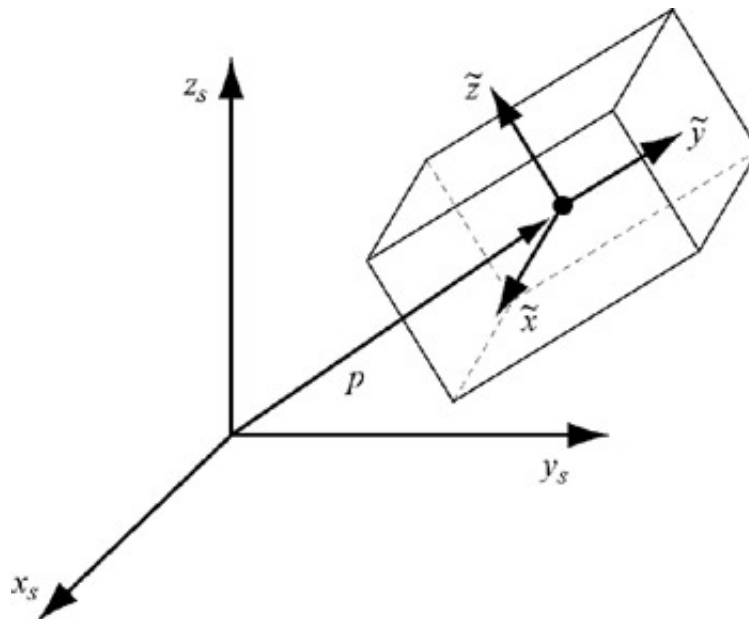


Figure 3.16: The body frame x - y - z relative to a stationary world frame x_s - y_s - z_s .

where the bottom row consists of three zeros and a one. (These "extra" numbers will be needed to allow us to perform matrix multiplications, as we will see shortly.) Since R and p both have three degrees of freedom, the configuration of a rigid body in three-space has six degrees of freedom, as we discovered earlier in the chapter.

Generalizing, the position and orientation of a rigid body in n -dimensional space can be written as an element of the *special Euclidean group* $SE(n)$:

$$SE(n) \equiv \begin{bmatrix} SO(n) & \mathbb{R}^n \\ 0 & 1 \end{bmatrix},$$

where the bottom row consists of n zeros and a one.

Uses of the Matrix Representations

The matrix groups $SO(n)$ and $SE(n)$ can be used to

1. represent rigid-body configurations,
2. change the reference frame for the representation of a configuration or a point, and
3. displace (move) a configuration or a point.

When the matrix is used for representing a configuration, we often call it a frame. When it is used for displacement or coordinate change, we often call it a transform. The various uses are best demonstrated by example.

Figure 3.17 shows three coordinate frames on a regular grid of unit spacing. These frames are confined to a plane with their z-axes pointing out of the page. Let T_{AB} be the configuration of frame B relative to frame A, and let T_{BC} be the configuration of frame C relative to frame B. It is clear from the figure that

$$T_{AB} = \begin{bmatrix} R_{AB} & p_{AB} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & -2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T_{BC} = \begin{bmatrix} 0 & 1 & 0 & -4 \\ -1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

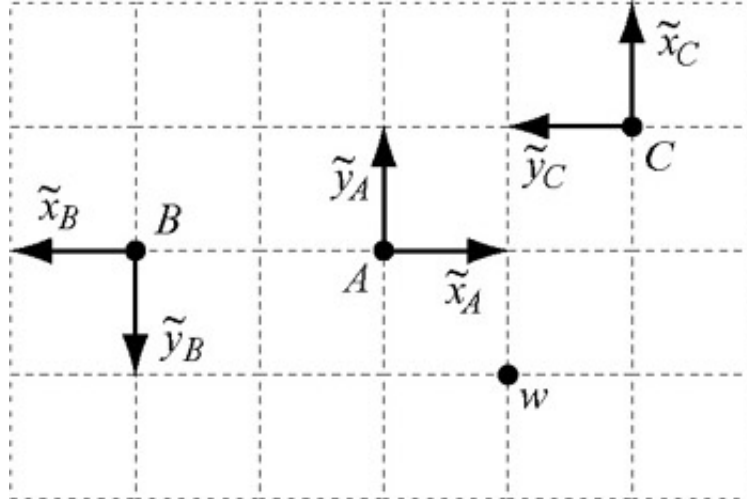


Figure 3.17: Three frames in a plane with their z-axes pointing out of the page.

From these, we can find T_{AC} , the frame C relative to the frame A, by performing a change of reference frame on T_{BC} . This involves premultiplying by T_{AB} , based on the rule for coordinate transformations that the second subscript of the matrix on the left cancels with the first subscript of the matrix on the right, if they are the same subscript. In other words,

$$T_{AB}T_{BC} = T_{AB}T_{BC} = T_{AC}.$$

We find that

$$T_{AC} = \begin{bmatrix} -1 & 0 & 0 & -2 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & -4 \\ -1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 & 2 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

which we can verify by inspection.

The representation of the point w in the coordinates of frame C is written w_C . From figure 3.17, we can see that the coordinates of w in C are $[-2, 1, 0]^T$. To facilitate matrix multiplications, however, we will express points in *homogeneous coordinates* by appending a 1 to the end of the vector, i.e.,

$$w_C = [-2, 1, 0, 1]^T.$$

To find the representation of the point w in other frames, we use a modification of the subscript canceling rule to get

$$T_{BC}w_C = T_{BQ}w_Q = w_B = [-3, 1, 0, 1]^T$$

and

$$T_{AB}T_{BC}w_C = T_{AQ}w_Q = w_A = [1, -1, 0, 1]^T,$$

which can be verified by inspection.

Elements of $SE(n)$ can also be used to displace a point. For example, $T_{AB}w_A$ does not satisfy the subscript canceling rule, and the result is not simply a representation of the point w in a new frame. Instead, the point w is rotated about the origin of the frame A by R_{AB} (expressed in the A frame), and then translated by p_{AB} in the A frame. This is the same motion required to take frame A to frame B . The result is

$$w'_A = T_{AB}w_A = [-3, 1, 0, 1]^T,$$

the location of the transformed point in the frame A . This transformation is shown graphically in [figure 3.18](#).

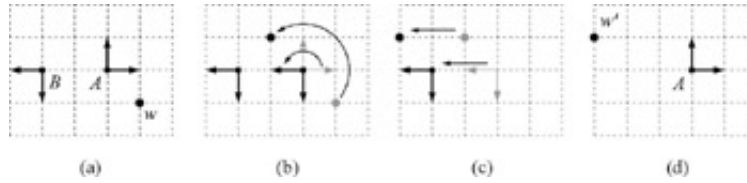


Figure 3.18: Displacing a point by the transformation T_{AB} . (a) The frames A and B and the point w . (b) Rotating frame A to the orientation of frame B , carrying the point w along with it. (c) Translating frame A to the location of frame B , carrying w along with it. (d) The final point $w'_A = T^{AB}w^A$.

Finally, we can use elements of $SE(3)$ to displace frames, not just points. For example, given a frame B represented by T_{AB} relative to frame A , and a transform $T_1 \in SE(3)$, then

$$T_{AB'} = T_{AB}T_1 = \begin{bmatrix} R_{AB}R_1 & R_{AB}p_1 + p_{AB} \\ 0 & 1 \end{bmatrix}$$

is the representation of the transformed frame B' relative to A after rotating B about its origin by R_1 (expressed in the B frame) and then translating by p_1 in the original B frame (before it was rotated). On the other hand,

$$T_{AB''} = T_1 T_{AB} = \begin{bmatrix} R_1 R_{AB} & R_1 p_{AB} + p_1 \\ 0 & 1 \end{bmatrix}$$

is the representation of the transformed frame B'' relative to A after rotating B about the origin of A by R_1 (expressed in the A frame) and then translating by p_1 in the A frame. Note that $T_{AB'}$ and $T_{AB''}$ are generally different, as matrix multiplication is not commutative.

If we consider frame B to be attached to a moving body, we call T_1 a *body-frame* transformation if it is multiplied on the right, as the rotation and translation are expressed relative to the body frame B . If A is a stationary world frame, we call T_1 a *world-frame* transformation if it is multiplied on the left, as the rotation and translation are expressed relative to the fixed A frame. An example is shown in [figure 3.19](#) for

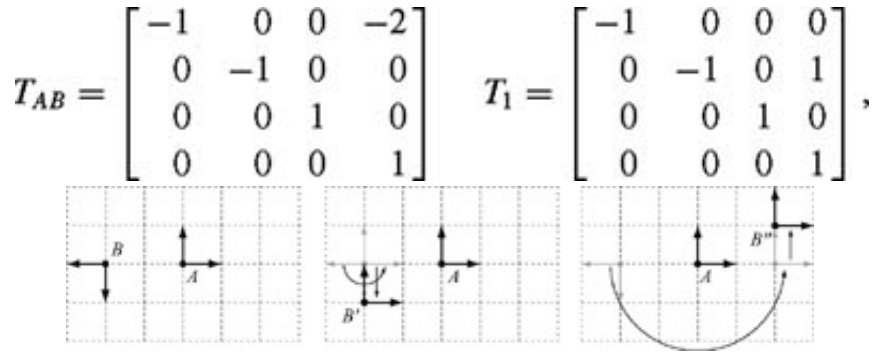


Figure 3.19: (a) The initial frame B relative to A . (b) B' is obtained by rotating about B and then translating in the original y_B -direction. (c) B'' is obtained by rotating about A and then translating in the y_A -direction.


giving


$$T_{AB'} = T_{AB} T_1 = \begin{bmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_{AB''} = T_1 T_{AB} = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Applying n world-frame transformations yields $T_{AB} = T^n \dots T_2 T_1 T_{AB}$, while n body-frame transformations yields $T_{AB'} = T_{AB} T_1 T_2 \dots T_n$.

[9] In fact, the matrix representations in this section are *Lie groups*, as (1) they are differentiable manifolds which are also groups, (2) the group operation is C^∞ , and (3) the mapping from an element of the group to its inverse is C^∞ .

[10] To make a right-handed frame, point straight ahead with your right index finger, point your middle finger 90 degrees to the left, and stick your thumb straight up. Your index finger is pointing in the $+x$ direction, your middle finger is pointing in the $+y$ direction, and your thumb is pointing in the $+z$ direction.

 Previous

Next 

TeamUnknown Release



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

◀ Previous

Next ▶

3.6 Parameterizations of $SO(3)$

We have seen that the nine elements R_{ij} of a rotation matrix $R \in SO(3)$ are subject to six constraints, leaving three rotational degrees of freedom. Thus, we expect that $SO(3)$ can be locally parameterized using three variables. Euler angles are a common parameterization. However, just as we see we cannot find a global parameterization for a circle with a single variable, we cannot build a global parameterization of $SO(3)$ with Euler angles.

Given two coordinate frames \mathcal{F}_0 and \mathcal{F}_1 , we can specify the orientation of frame \mathcal{F}_1 relative to frame \mathcal{F}_0 by three angles (ϕ, θ, ψ) , known as Z-Y-Z Euler angles. These Euler angles are defined by three successive rotations as follows. Initially, the two frames are coincident. Rotate \mathcal{F}_0 about the z-axis by the angle ϕ to obtain frame \mathcal{F}_a . Next, rotate frame \mathcal{F}_a about its y-axis by the angle θ to obtain frame \mathcal{F}_b . Finally, rotate frame \mathcal{F}_b about its z-axis by the angle ψ to obtain frame \mathcal{F}_1 . This is illustrated in figure 3.20.

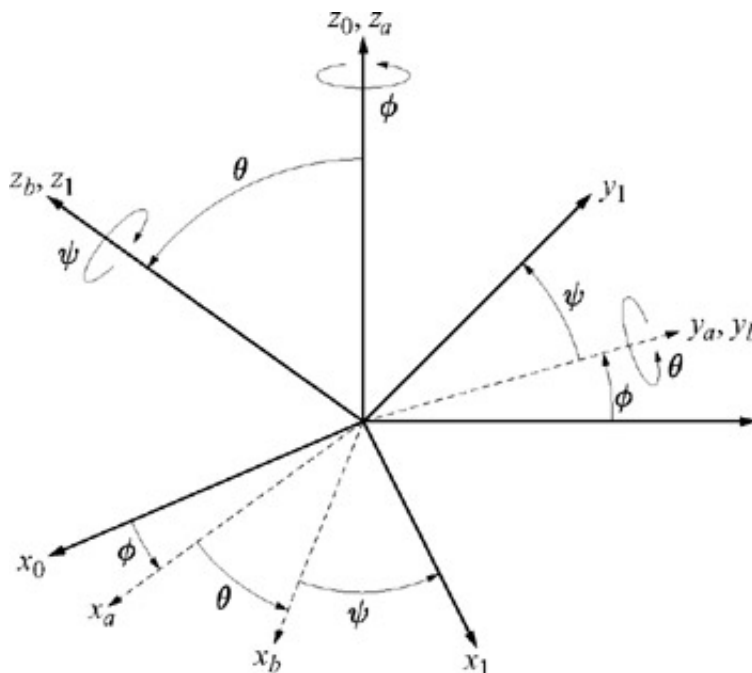


Figure 3.20: Euler angle representation.


The corresponding rotation matrix R can thus be generated by successive multiplication of rotation matrices that define rotations about coordinate axes,

$$\begin{aligned}
 (3.6) \quad R &= R_{z,\phi} R_{y,\theta} R_{z,\psi} \\
 (3.7) \quad &= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
 (3.8) \quad &= \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}.
 \end{aligned}$$

Note that successive rotation matrices are multiplied on the right, as successive rotations are defined about axes in the changing "body" frame.

Parameterization of $SO(3)$ using Euler angles, along with some other representations of $SO(3)$, are described in detail in [appendix E](#).

 Previous

Next 



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

◀ Previous

Next ▶

3.7 Example Configuration Spaces

In most cases, we can model robots as rigid bodies, articulated chains, or combinations of these two. Some common robots and representations of their configuration spaces are given in [table 3.1](#).

Table 3.1: Some common robots and their configuration spaces.

Type of robot	Representation of Q
Mobile robot translating in the plane	\mathbb{R}^2
Mobile robot translating and rotating in the plane	$SE(2)$ or $\mathbb{R}^2 \times S^1$
Rigid body translating in the three-space	\mathbb{R}^3
A spacecraft	$SE(3)$ or $\mathbb{R}^3 \times SO(3)$
An n -joint revolute arm	T^n
A planar mobile robot with an attached n -joint arm	$SE(2) \times T^n$

When designing a motion planner, it is often important to understand the underlying structure of the robot's configuration space. In particular, we note the following.

- $S^1 \times S^1 \times \dots \times S^1$ (n times) $= T^n$, the n -dimensional torus
- $S^1 \times S^1 \times \dots \times S^1$ (n times) $\neq S^n$, the n -dimensional sphere in \mathbb{R}^{n+1}
- $S^1 \times S^1 \times S^1 \neq SO(3)$
- $SE(2) \neq \mathbb{R}^3$
- $SE(3) \neq \mathbb{R}^6$


It is sometimes important to know whether a manifold is compact. The manifolds S^n , T^n , and $SO(n)$ are all compact, as are all of their direct products. The manifolds \mathbb{R}^n and $SE(n)$ are not compact, and therefore $\mathbb{R}^n \times \mathcal{M}$ is not compact, regardless of whether or not the manifold \mathcal{M} is compact.

Despite their differences, all of these configuration spaces have an important similarity. When equipped with an atlas, each is a differentiable manifold. In particular,

- \mathbb{R}^1 and $SO(2)$ are one-dimensional manifolds;
- \mathbb{R}^2 , S^2 and T^2 are two-dimensional manifolds;
- \mathbb{R}^3 , $SE(2)$ and $SO(3)$ are three-dimensional manifolds;
- \mathbb{R}^6 , T^6 and $SE(3)$ are six-dimensional manifolds.

Thus, for example, all of \mathbb{R}^3 , $SE(2)$, and $SO(3)$ can be represented locally by a set of three coordinates.

 Previous

Next 

TeamUnknown Release



Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

◀ Previous

Next ▶

3.8 Transforming Configuration and Velocity Representations

We often need to transform from one representation of the configuration of a robot $q \in \mathcal{Q}$ to some other representation $x \in \mathcal{M}$. A common example occurs when q represents the joint angles of a robot arm and x represents the configuration of the end effector as a rigid body in the ambient space. The representation x is more convenient when planning manipulation tasks in the world, but control of the robot arm is more easily expressed in q variables, so we need an easy way of switching back and forth. It is often the case that \mathcal{Q} and \mathcal{M} are not homeomorphic; the dimension of the two spaces may not even be equal.

Using the robot arm as inspiration, we define the *forward kinematics map* $\phi : \mathcal{Q} \rightarrow \mathcal{M}$ and the *inverse kinematics map* $\phi^{-1} : \mathcal{M} \rightarrow \mathcal{Q}$. These maps may not be homeomorphisms even if the dimensions of $\dot{x} = \frac{dx}{dt}$ is related to the time derivative $\dot{q} = \frac{dq}{dt}$ by

$$\dot{x} = \frac{\partial \phi}{\partial q} \dot{q} = J(q) \dot{q},$$

where J is the *Jacobian* of the map ϕ , also known as the *differential* $D\phi$ (see [appendix C](#)). The Jacobian is also useful for transforming forces expressed in one set of coordinates to another (see [chapter 4](#), [section 4.7](#), and [chapter 10](#)).

EXAMPLE 3.8.1

The 2R robot arm of [figure 3.21](#) has link lengths L_1 and L_2 . Its configuration space is $\mathcal{Q} = T^2$, and we represent the configuration by the two joint angles $q = [\theta_1, \theta_2]^T$. The endpoint of the hand in the Cartesian space is $x = [x_1, x_2]^T \in \mathcal{M} \subset \mathbb{R}^2$. In this case, the dimensions of \mathcal{Q} and \mathcal{M} are equal, but they are not homeomorphic. The forward kinematics map $\phi : \mathcal{Q} \rightarrow \mathcal{M}$ is

$$\phi(q) = \begin{bmatrix} \phi_1(q) \\ \phi_2(q) \end{bmatrix} = \begin{bmatrix} L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) \\ L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) \end{bmatrix}.$$

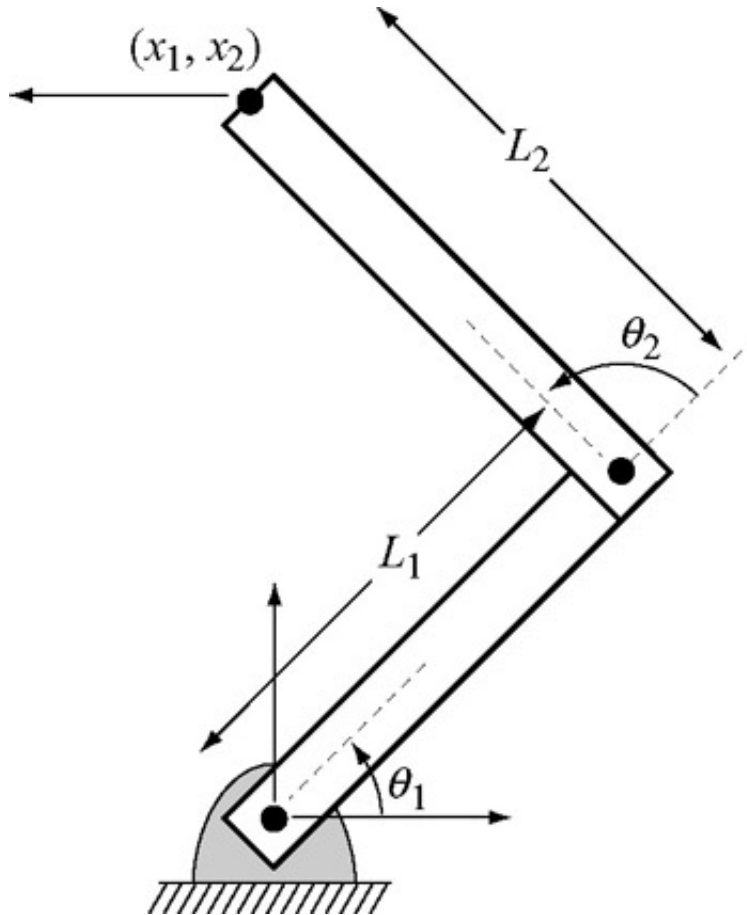


Figure 3.21: The 2R robot arm and the velocity at its endpoint.

The inverse kinematics map ϕ^{-1} is one-to-two at most points of \mathcal{M} , meaning that the robot can be chosen to be in either the right-arm or left-arm configuration. The inverse kinematics of the 2R arm is most easily found geometrically using the law of cosines and is left for problem 20.

The Jacobian of the forward kinematics map is

$$\begin{aligned} J(q) &= \frac{\partial \phi}{\partial q} = \begin{bmatrix} \frac{\partial \phi_1}{\partial \theta_1} & \frac{\partial \phi_1}{\partial \theta_2} \\ \frac{\partial \phi_2}{\partial \theta_1} & \frac{\partial \phi_2}{\partial \theta_2} \end{bmatrix} \\ &= \begin{bmatrix} -L_1 \sin \theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}. \end{aligned}$$

Plugging in $L_1 = L_2 = 1$, $\theta_1 = \pi/4$, $\theta_2 = \pi/2$, and $\dot{q} = [1, 0]^T$ as shown in figure 3.21, we find that

$$\dot{x} = J(q)\dot{q} = \begin{bmatrix} -\sqrt{2} & -\sqrt{2}/2 \\ 0 & -\sqrt{2}/2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -\sqrt{2} \\ 0 \end{bmatrix},$$

matching the motion seen in the figure.

When $\sin \theta_2 = 0$, the Jacobian $J(q)$ loses rank, and the robot is said to be in a *singular* configuration. In this case, the two-dimensional set of joint velocities \dot{q} maps to a one-dimensional set of endpoint velocities \dot{x} - instantaneous endpoint motion is impossible in one direction.

EXAMPLE 3.8.2

A polygon moving in the plane is represented by the configuration

$q = [q_1, q_2, q_3]^T \in \mathcal{Q} = \mathbb{R}^2 \times S^1$, where (q_1, q_2) gives the position of a reference frame \mathcal{F}_p attached to the polygon relative to a world frame \mathcal{F} , and q_3 gives the

orientation of \mathcal{F}_p relative to \mathcal{F} (see figure 3.22). A point is fixed on the polygon at $r = [r_1, r_2]^T$ in the polygon frame \mathcal{F}_p , and let $x = [x_1, x_2]^T \in \mathcal{M} = \mathbb{R}^2$ position of this point in the plane. Then the forward kinematics mapping is

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \phi(q) = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} + \begin{bmatrix} \cos q_3 & -\sin q_3 \\ \sin q_3 & \cos q_3 \end{bmatrix} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix},$$

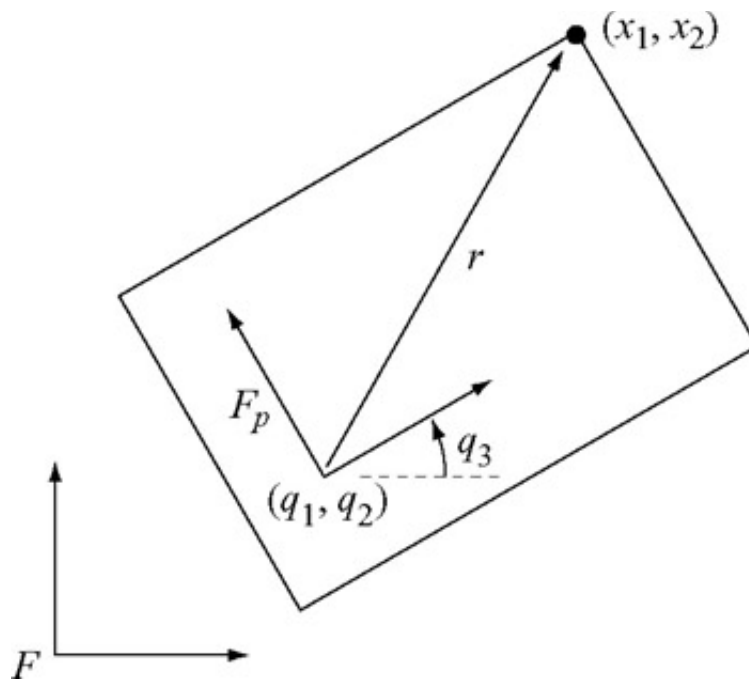


Figure 3.22: The point on the polygon is at r in the polygon frame \mathcal{F}_p and x in the world frame \mathcal{F} .

where we recognize the 2×2 rotation matrix. The inverse map ϕ^{-1} in this example is one-to-many, as the dimension of \mathcal{Q} is greater than the dimension of \mathcal{M} . The velocities \dot{x} and \dot{q} are related by the Jacobian

$$J(q) = \frac{\partial \phi}{\partial q} = \begin{bmatrix} \frac{\partial \phi_1}{\partial q_1} & \frac{\partial \phi_1}{\partial q_2} & \frac{\partial \phi_1}{\partial q_3} \\ \frac{\partial \phi_2}{\partial q_1} & \frac{\partial \phi_2}{\partial q_2} & \frac{\partial \phi_2}{\partial q_3} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -r_1 \sin q_3 - r_2 \cos q_3 \\ 0 & 1 & r_1 \cos q_3 - r_2 \sin q_3 \end{bmatrix}.$$

◀ Previous

Next ▶




Chapter 3 - Configuration Space

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

 Previous

Next 

Problems

1. Invent your own nontrivial robot system. It could consist of one or more robot arms, mobile platforms, conveyor belts, fixed obstacles, movable objects, etc. Describe the configuration space mathematically. Explain whether or not the configuration space is compact, and if not, describe the compact factors. Describe the connected components of the configuration space. Draw a rough picture of your robot system.
2. Give the dimension of the configuration spaces of the following systems. Explain your answers.
 - a. Two mobile robots rotating and translating in the plane.
 - b. Two translating and rotating planar mobile robots tied together by a rope.
 - c. Two translating and rotating planar mobile robots connected rigidly by a bar.
 - d. The two arms of a single person (with torso stationary) holding on firmly to a car's steering wheel.
 - e. A train on train tracks. What if we include the wheel angles? (The wheels roll without slipping.)
 - f. A spacecraft with a 6R robot arm.
 - g. The end effector of the 6R robot arm of a spacecraft.
 - h. Your legs as you pedal a bicycle (remaining seated with feet fixed to the pedals).
 - i. A sheet of paper.
3. Describe the Bug2 algorithm for a two-joint manipulator. What are the critical differences between the Bug2 algorithm for a mobile base and the two-joint manipulator? What does a straight line mean in the arm's configuration space? Can Bug2 be made to work for the two-joint arm?
4. Prove the configuration space obstacle of a convex mobile robot translating in a

plane with a convex obstacle is convex.

5. Prove the union operator propagates from the workspace to the configuration space. That is, the union of two configuration space obstacles is the configuration space obstacle of the union of two workspace obstacles. In other words, assuming Q is a configuration space operator, show that

$$Q(WO_i \cup WO_j) = QO_i \cup QO_j.$$

6. How many degrees of freedom does a rigid body in n -space have? How many of them are rotational? Prove these two ways: (a) using the method of choosing a number of points on the body and sequentially adding their independent degrees of freedom until each point on the body is fixed, and (b) using the definitions of $SE(n)$ and $SO(n)$.
7. Use cardboard and pushpins to create a closed chain with four links, a four-bar mechanism. One of these bars is considered stationary, or fixed to the ground. Going around the loop, the link lengths between joints are 6.5 (the ground link), 3.0, 1.5, and 3.0 inches (or centimeters) in length. Poke a hole at the midpoint of the 1.5 inch link and trace the path that the hole traces. Describe a good representation of the configuration space of the linkage.
8. Give a homeomorphism from the racetrack to the ellipse in [figure 3.12](#).
9. Find two charts for the unit circle S^1 and prove they form an atlas.
10. Explain why the latitude-longitude chart we often place on the Earth is not a global parameterization. Find two charts for the sphere and prove that they form an atlas.
11. The set of right-arm and left-arm configurations of the 2R manipulator in [figure 3.3](#) each give an annulus of reachable positions by the end effector, neither of which is diffeomorphic to the robot's configuration space. Consider the right-arm and left-arm workspaces as two separate annuluses, and describe how they can be glued together to make a single space that is a valid representation of the configuration space. Comment on the topology of this glued space.
12. For the 2R manipulator of [figure 3.7](#), how many connected components of free configuration space are there? Copy the figure, color each of the connected components a different color, and give a drawing of the robot in each of these connected components.
13. Show that compact and noncompact spaces are never diffeomorphic.
14. Find a diffeomorphism from any open interval $(a, b) \in \mathbb{R}$ to the whole real line \mathbb{R} .
15. Give an implicit constraint equation $f(x, y, z) = 0$ that embeds a torus in \mathbb{R}^3 .
16. For $T \in SE(3)$ consisting of the rotation matrix $R \in SO(3)$ and the translation $p \in \mathbb{R}^3$, find the inverse transform T^{-1} , so that $TT^{-1} = T^{-1}T = I$. Your answer

should not contain any matrix inverses.

17. Consider two three-dimensional frames aligned with each other, called A and B . Rotate B 90 degrees about the x -axis of A , then rotate again by 90 degrees about the y -axis of A , then move the origin of B by three units in the z -direction of A . (Make sure you rotate in the right direction! Use the right-hand rule: thumb points along the positive axis, fingers curl in the direction of positive rotation.) Give the matrix $T_{AB} \in SE(3)$ describing the frame B relative to the frame A . Consider the point $x_B = [4, 3, 1, 1]^T$ in homogeneous coordinates in frame B . What is the point x_A (expressed in the frame A)? Consider the point $y_A = [1, 2, 3, 1]^T$ in homogeneous coordinates in frame A , and perform the transformation T_{AB} . Where is the new point y'_A ?
18. Write a program to calculate the configuration space for a convex polygonal robot translating in an environment with convex obstacles. The program should read in from a file a counterclockwise list of vertices representing the robot, where $(0, 0)$ is the robot reference point. From a second file, the program should read in a set of obstacles in the workspace. The user enters an orientation for the robot and the program calculates the configuration space obstacles (see, e.g., [appendix F](#)). Display the configuration space for different orientations of the robot to show that translating paths between two points may exist for some orientations of the robot, but not for others.
19. Write a program to display the configuration space for a 2R manipulator in a polygonal environment. The program should read in a file containing the location of the base of the robot, the length of the links, and the lists of vertices representing the obstacles. The program should check for collision at 1 degree increments for each joint and create a plot similar to that shown in [figure 3.7](#).
20. Find the inverse kinematics ϕ^{-1} mapping the end-effector coordinates x to the joint coordinates q for the 2R robot arm in [example 3.8.1](#). Note that for most reachable points x , there will be two solutions, corresponding to the right- and left-arm configurations. Your solution will likely make use of the two-argument arctangent $\text{atan2}(x_2, x_1)$, which returns the unique angle in $[-\pi, \pi)$ to the point (x_1, x_2) in the plane, as well as the law of cosines $a^2 = b^2 + c^2 - 2bc \cos A$, where a , b , and c are the lengths of the three edges of a triangle and A is the angle opposite to edge a . Solve for general L_1, L_2 (do not plug in numbers).
21. Give the forward kinematics ϕ for the planar 3R arm shown in [figure 3.23](#), from joint angles q to the position and orientation of the end effector frame in the plane. Find the manipulator Jacobian.

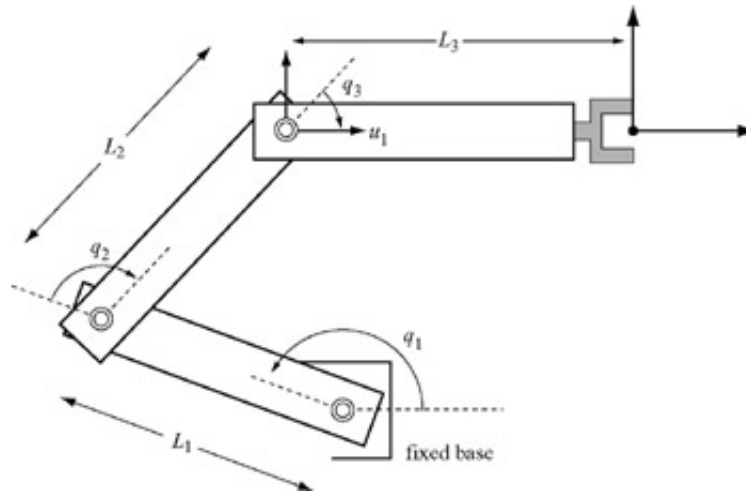


Figure 3.23: A3R planar robot with a frame attached to the end effector.

22. For the problem above, show that the forward kinematics mapping is injective, surjective, bijective, or none of these, when viewed as a mapping from T^3 to $\mathbb{R}^2 \times S^1$. Find a "large" set of joint angle ranges $U \subset T^3$ and a set of end-effector configurations $V \subset \mathbb{R}^2 \times S^1$ for which the mapping is a diffeomorphism.
23. The topology of $SE(2)$ is equivalent to $\mathbb{R}^2 \times SO(2)$. Let's represent an element of $\mathbb{R}^2 \times SO(2)$ by (x, R) , where $x \in \mathbb{R}^2$, $R \in SO(2)$. As we have seen, we can make $SE(2)$ a group by giving it a group operation, namely, matrix multiplication. We can also make $\mathbb{R}^2 \times SO(2)$ a group by using the direct product structure to define composition of two elements:

$$(x_1, R_1)(x_2, R_2) = (x_1 + x_2, R_1 R_2) \in \mathbb{R}^2 \times SO(2)$$

We are using vector addition as the group operation on \mathbb{R}^2 and matrix multiplication on $SO(2)$. With this group operation, is $\mathbb{R}^2 \times SO(2)$ commutative? Is $SE(2)$ commutative? The spaces $SE(2)$ and $\mathbb{R}^2 \times SO(2)$ are topologically equivalent, but are they equivalent as groups?



Appendix E - Representations of Orientation

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

[Previous](#)

[Next](#)

Appendix E: Representations of Orientation

In [Chapter 3](#), we represent orientation by matrices in $SO(3)$, which can be parameterized using three parameters. In this appendix, we describe some of the most popular methods of doing so, including Euler angles and angles with respect to a fixed frame. We also describe how orientation can be described as rotation about an arbitrary axis and by quaternions.

E.1 Euler Angles

Recall that the Euler angles ϕ , θ , ψ in [chapter 3](#) correspond to successive rotations about body Z-Y-Z axes, and that the corresponding rotation matrix is obtained as

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

in which s_θ and c_θ denote $\sin \theta$ and $\cos \theta$ respectively.

Consider now the problem of using Euler angles to define a chart on some open set $U \subset SO(3)$. It is easy to see that a single chart cannot cover all of $SO(3)$. For example, if $R_{33} = 1$, it must be the case that $\theta = 0$, and the rotation matrix is given by

$$(E.2) \quad \begin{bmatrix} R_{11} & R_{12} & 0 \\ R_{21} & R_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_{\phi+\psi} & -s_{\phi+\psi} & 0 \\ s_{\phi+\psi} & c_{\phi+\psi} & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

In this case, it is not possible to uniquely define ϕ and ψ , since only their sum is represented in R . A similar case occurs when $R_{33} = -1$.

To define a chart using Euler angles, we begin by defining the open set

$$U = \{R \in SO(3) \mid R_{33} \notin \{-1, 1\}\},$$

and defining the chart Φ such that

$$\Phi(R) \mapsto [\phi(R), \theta(R), \psi(R)]^T \in \mathbb{R}^3.$$

For any $R \in U$, not both of R_{13}, R_{23} are zero. Then the above equations show that $s_\theta \neq 0$. Since not both R_{13} and R_{23} are zero, then $R_{33} \neq \pm 1$, and we have $c_\theta = R_{33}$,

$$s_\theta = \pm \sqrt{1 - R_{33}^2} \text{ so so}$$

$$(E.3) \quad \theta = \text{atan2} \left(\sqrt{1 - R_{33}^2}, R_{33} \right)$$

or

$$(E.4) \quad \theta = \text{atan2} \left(-\sqrt{1 - R_{33}^2}, R_{33} \right).$$

The function $\theta = \text{atan2}(y, x)$ computes the arc tangent function, where x and y are the cosine and sine, respectively, of the angle θ . This function uses the signs of x and y to select the appropriate quadrant for the angle θ . Note that if both x and y are zero, atan2 is undefined.

If we choose the value for θ given by (E.3), then $s_\theta > 0$, and

$$(E.5) \quad \phi = \text{atan2}(R_{23}, R_{13})$$

$$(E.6) \quad \psi = \text{atan2}(R_{32}, -R_{31}).$$

If we choose the value for θ given by (E.4), then $s_\theta < 0$, and

$$(E.7) \quad \phi = \text{atan2}(-R_{23}, -R_{13})$$

$$(E.8) \quad \psi = \text{atan2}(-R_{32}, R_{31}).$$

Thus there are two solutions depending on the sign chosen for θ .

As described above, when $R_{33} = \pm 1$, only the sum $\phi \pm \psi$ can be determined. For $R_{33} = 1$

$$(E.9) \quad \phi + \psi = \text{atan2}(R_{21}, R_{11}) \\ = \text{atan2}(-R_{12}, R_{11}).$$

In this case there are infinitely many solutions. We may take $\phi = 0$ by convention. If $R_{33} = -1$, then $c_\theta = -1$ and $s_\theta = 0$, so that $\theta = \pi$. In this case (E.1) becomes

$$(E.10) \quad \begin{bmatrix} -c_{\phi-\psi} & -s_{\phi-\psi} & 0 \\ s_{\phi-\psi} & c_{\phi-\psi} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & 0 \\ R_{21} & R_{22} & 0 \\ 0 & 0 & -1 \end{bmatrix}.$$


The solution is thus

$$(E.11) \quad \phi - \psi = \text{atan2}(-R_{12}, -R_{11}) = \text{atan2}(-R_{22}, -R_{21}).$$

As before there are infinitely many solutions.

There is nothing special about the choice of axes we used to define Euler angles. We could just as easily have used successive rotations about, say, the x , y , and z axes. In fact, it is easy to see that there are twelve possible ways to define Euler angles: any sequence of three axes, such that no two successive axes are the same, generates a set of Euler angles.

 Previous

Next 

TeamUnknown Release



Appendix E - Representations of Orientation

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

[Previous](#)

[Next](#)

E.2 Roll, Pitch, and Yaw Angles

A rotation matrix R can also be described as a product of successive rotations about the world coordinate axes. These rotations define the *roll*, *pitch*, and *yaw* angles, and they are illustrated in [figure E.1](#). Typically, the order of rotation is taken to be x-y-z: first a yaw about the world x-axis by an angle ψ , then pitch about the world y-axis by an angle θ , and finally a roll about the world z-axis by an angle ϕ ^[1]. Since the successive rotations are relative to the world coordinate frame, the resulting rotation matrix is given by

$$\begin{aligned}
 (E.12) \quad R &= R_{z,\phi} R_{y,\theta} R_{x,\psi} \\
 &= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\psi & -s_\psi \\ 0 & s_\psi & c_\psi \end{bmatrix} \\
 (E.12) \quad &= \begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}.
 \end{aligned}$$

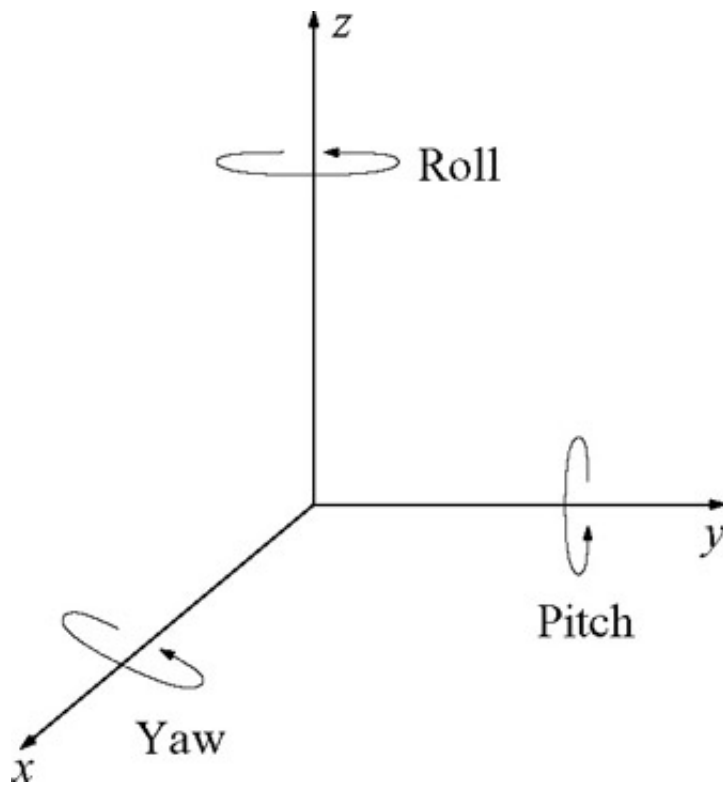


Figure E.1: Roll, pitch, and yaw angles.

The three angles, ϕ , θ , ψ , can be obtained for a given rotation matrix using a method that is similar to that used to derive the Euler angles above.

^[1]As with Euler angles, one can choose a different ordering for the rotations to obtain different *fixed axis* representations of orientation. The term *fixed axis* refers to the fact that successive rotations are taken with respect to axes of the fixed coordinate frame.

◀ Previous

Next ▶



Appendix E - Representations of Orientation

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

[Previous](#)

[Next](#)

E.3 Axis-Angle Parameterization

Above we described a rotation matrix by decomposing a rotation into three successive rotations about the coordinate axes. An alternative to this is to specify a rotation matrix in terms of a rotation about an arbitrary axis in space. This provides both a convenient way to describe rotations, and an alternative parameterization for rotation matrices.

Let $k = [k_x, k_y, k_z]^T$ be a unit vector defining an axis expressed in the world frame. To determine the parameterization, we need to derive the rotation matrix $R_{k,\theta}$ representing a rotation of θ degrees about this axis. A simple way to derive this rotation matrix is to rotate the vector k into one of the coordinate axes, say the z-axis, then rotate about this axis by θ , and finally, rotate k back to its original position. As can be seen in [figure E.2](#) we can rotate k into the world z-axis by first rotating about the world z-axis $-\alpha$, then rotating about the world y-axis by $-\beta$. Since all rotations are performed relative to the world frame, the matrix $R_{k,\theta}$ is obtained as

$$(E.13) \quad R_{k,\theta} = R_{z,\alpha} R_{y,\beta} R_{z,\theta} R_{y,-\beta} R_{z,-\alpha}.$$

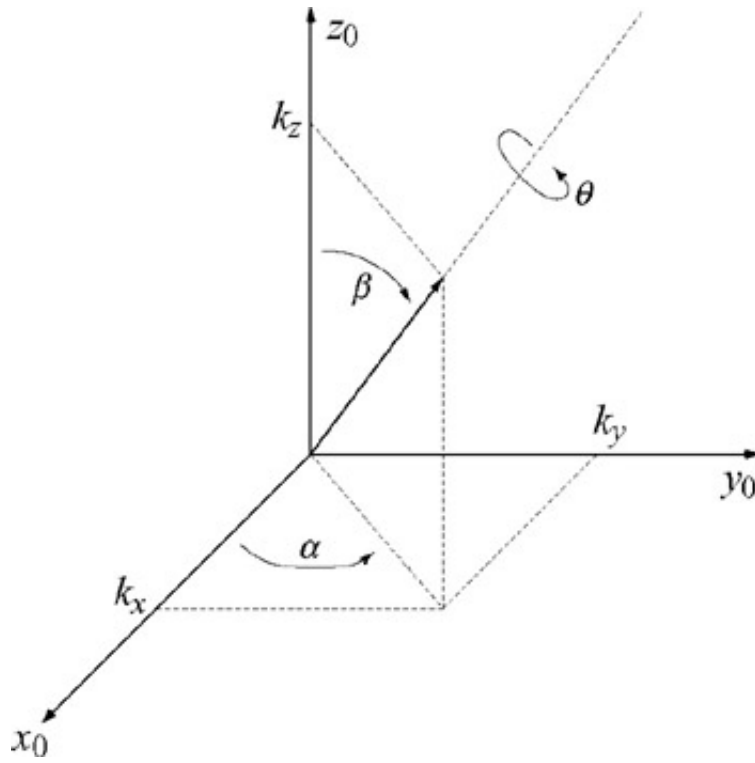


Figure E.2: Rotation about an arbitrary axis.

As can be seen in [figure E.2](#),

$$(E.14) \quad \sin \alpha = \frac{k_y}{\sqrt{k_x^2 + k_y^2}}$$

$$(E.15) \quad \cos \alpha = \frac{k_x}{\sqrt{k_x^2 + k_y^2}}$$

$$(E.16) \quad \sin \beta = \frac{\sqrt{k_x^2 + k_y^2}}{1}$$

$$(E.17) \quad \cos \beta = k_z.$$

The final two equations follow from the fact that k is a unit vector. Substituting [\(E.14\)](#) through [\(E.17\)](#) into [\(E.13\)](#) we can obtain

$$(E.18) \quad R_{k,\theta} = \begin{bmatrix} k_x^2 v_\theta + c_\theta & k_x k_y v_\theta - k_z s_\theta & k_x k_z v_\theta + k_y s_\theta \\ k_x k_y v_\theta + k_z s_\theta & k_y^2 v_\theta + c_\theta & k_y k_z v_\theta - k_x s_\theta \\ k_x k_z v_\theta - k_y s_\theta & k_y k_z v_\theta + k_x s_\theta & k_z^2 v_\theta + c_\theta \end{bmatrix},$$

in which $v_\theta = 1 - c_\theta$.

We can use this parameterization to derive a chart on $SO(3)$ as follows. Let R be an arbitrary rotation matrix with components (R_{ij}) . Let $U = \{R \mid \text{Tr}(R) \neq \pm 1\}$ where

Tr denotes the trace of R . By direct calculation using (E.18) we obtain

$$\begin{aligned}\theta &= \cos^{-1} \left(\frac{R_{11} + R_{22} + R_{33} - 1}{2} \right) \\ &= \cos^{-1} \left(\frac{\text{Tr}(R) - 1}{2} \right), \text{ and} \\ k &= \frac{1}{2 \sin \theta} \begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix}.\end{aligned}$$

This representation is not unique since a rotation of $-\theta$ about $-k$ is the same as a rotation of θ about k , that is,

$$(E.19) \quad R_{k,\theta} = R_{-k,-\theta}.$$

We can now define the mapping ϕ using k and θ . Since the axis k is a unit vector, only two of its components are independent. Therefore, only three independent quantities are required in this representation of a rotation. Thus, we can define ϕ as


$$(E.20) \quad \phi(R) = [\theta k_x, \theta k_y, \theta k_z]^T.$$

Using this convention, we can recover k and θ as

$$(E.21) \quad k = \frac{\phi(R)}{\|\phi(R)\|} \quad \text{and} \quad \theta = \|\phi(R)\|.$$

The angle θ is a good distance measure between two elements of $SO(3)$.

 Previous

Next 



Appendix E - Representations of Orientation

Principles of Robot Motion: Theory, Algorithms, and Implementation

by Howie Choset et al.

The MIT Press © 2005

◀ Previous

Next ▶

E.4 Quaternions

The axis-angle parameterization described above parameterizes a rotation matrix by three parameters (given by (E.21)). Quaternions, which are closely related to the axis-angle parameterization, can be used to define a rotation by four numbers. It is straightforward to use quaternions to define an atlas for $SO(3)$ using only four charts. Furthermore quaternion representations are very convenient for operations such as composition of rotations and coordinate transformations. For these reasons, quaternions are a popular choice for the representation of rotations in three dimensions.

Quaternions are a generalization of the complex numbers to a four-dimensional space. For this reason, we begin with a quick review of how complex numbers can be used to represent orientation in the plane. A first introduction to complex numbers often uses the example of representing orientation in the plane using unit magnitude complex numbers of the form $a + ib$, in which $i = \sqrt{-1}$. In this case, the angle θ from the real axis to the vector $(a + ib) \in \mathbb{C}$ is given by $\text{atan2}(b, a)$, and it is easy to see that $\cos \theta = a$ and $\sin \theta = b$. Since $a, b \in \mathbb{R}$, we can consider this as an embedding of S^1 in the plane.

Using this representation, multiplication of two complex numbers corresponds to addition of the corresponding angles. This can be verified by direct calculation as

$$\begin{aligned} (a_1 + ib_1)(a_2 + ib_2) &= a_1a_2 + ib_1a_2 + ia_1b_2 - b_1b_2 \\ &= (a_1a_2 - b_1b_2) + i(b_1a_2 + a_1b_2) \\ &= \cos \theta_1 \cos \theta_2 - \sin \theta_1 \sin \theta_2 \\ &\quad + i(\sin \theta_1 \cos \theta_2 + \cos \theta_1 \sin \theta_2) \\ &= \cos(\theta_1 + \theta_2) + i \sin(\theta_1 + \theta_2). \end{aligned}$$

While a complex number $a + ib$ defines a point in the complex plane, a quaternion defines a point in a four-dimensional complex space, $q_0 + iq_1 + jq_2 + kq_3$. Here, i, j , and k represent independent square roots of negative one. They are independent in the sense that they do not combine using the normal rules of scalar multiplication. In particular, we have

$$(E.22) \quad -1 = i^2 = j^2 = k^2,$$

$$(E.23) \quad i = jk = -kj,$$

$$(E.24) \quad j = ki = -ik,$$

$$(E.25) \quad k = ij = -ji.$$

It is not a coincidence that multiplication of i , j , and k is similar to the vector cross product for the orthogonal unit basis vectors, $i = [1, 0, 0]^T$, $j = [0, 1, 0]^T$, and $[0, 0, 1]^T$.

Complex numbers with unit magnitude can be used to represent orientation in the plane simply by using their representation in polar coordinates. Likewise, quaternions can be used to represent rotations in 3D. In particular, for a rotation about an axis $n = [n_x, n_y, n_z]^T$ by angle θ , the corresponding quaternion, Q , is defined as

$$(E.26) \quad Q = \left(\cos \frac{\theta}{2}, n_x \sin \frac{\theta}{2}, n_y \sin \frac{\theta}{2}, n_z \sin \frac{\theta}{2} \right).$$

When we define the axis of rotation to be a unit vector, the corresponding quaternion has unit norm, since

$$(E.27) \quad \begin{aligned} \|Q\|^2 &= \cos^2 \frac{\theta}{2} + n_x^2 \sin^2 \frac{\theta}{2} + n_y^2 \sin^2 \frac{\theta}{2} + n_z^2 \sin^2 \frac{\theta}{2} \\ (E.27) \quad &= \cos^2 \frac{\theta}{2} + \sin^2 \frac{\theta}{2} (n_x^2 + n_y^2 + n_z^2) \\ &= 1. \end{aligned}$$

Quaternions with unit norm are sometimes referred to as rotation quaternions.

It is straightforward to apply the results from [section E.3](#) to determine the rotation matrix $R \in SO(3)$ that corresponds to the rotation represented by a rotation quaternion. For the quaternion $Q = (q_0, q_1, q_2, q_3)$ we have

$$(E.28) \quad R(Q) = \begin{bmatrix} 2(q_0^2 + q_1^2) - 1 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & 2(q_0^2 + q_2^2) - 1 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & 2(q_0^2 + q_3^2) - 1 \end{bmatrix}.$$

Quaternions can be used to define an atlas for $SO(3)$ comprising four charts, (U_i, ϕ_i) , with $\phi_i : U_i \rightarrow \mathbb{R}^3$. This is most easily done by using two steps. First, for a rotation matrix R , we determine the corresponding quaternion Q . Then, we use Q to determine which chart applies (i.e., we implicitly define the neighborhoods U_i in terms of Q), and use the appropriate ϕ_i to define the local coordinates.

Determining the quaternion that corresponds to a rotation matrix amounts to solving the inverse of [\(E.28\)](#), and this can be done by a method similar to that given for the axis-

angle parameterization of [section E.3](#). In particular, for rotation matrices R such that $\text{Tr}(R) \neq \pm 1$ we have

$$(E.29) \quad q_0 = \frac{1}{2} \sqrt{1 + \text{Tr}(R)}$$

$$(E.30) \quad \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \frac{1}{4q_0} \begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix}.$$

To define the four charts, we first define the four neighborhoods

$$U_0 = \{Q = (q_0, q_1, q_2, q_3) \mid q_0 \geq q_1, q_2, q_3\}$$

$$U_1 = \{Q = (q_0, q_1, q_2, q_3) \mid q_1 \geq q_0, q_2, q_3\}$$

$$U_2 = \{Q = (q_0, q_1, q_2, q_3) \mid q_2 \geq q_0, q_1, q_3\}$$

$$U_3 = \{Q = (q_0, q_1, q_2, q_3) \mid q_3 \geq q_0, q_1, q_2\}.$$

These are not actually open sets (due to the nonstrict inequality in the set definitions), but they can be used to define open sets using their interiors. Now we define the coordinate maps ϕ_i as

$$\phi_0(q_0, q_1, q_2, q_3) = \left(\frac{q_1}{|q_0|}, \frac{q_2}{|q_0|}, \frac{q_3}{|q_0|} \right)$$

$$\phi_1(q_0, q_1, q_2, q_3) = \left(\frac{q_0}{|q_1|}, \frac{q_2}{|q_1|}, \frac{q_3}{|q_1|} \right)$$

$$\phi_2(q_0, q_1, q_2, q_3) = \left(\frac{q_0}{|q_2|}, \frac{q_1}{|q_2|}, \frac{q_3}{|q_2|} \right)$$

$$\phi_3(q_0, q_1, q_2, q_3) = \left(\frac{q_0}{|q_3|}, \frac{q_1}{|q_3|}, \frac{q_2}{|q_3|} \right)$$

As we have seen above, $R_i \in \text{SO}(3)$ represents a rotation, and the composition of successive rotations, say R_1 and R_2 , is represented by the rotation matrix $R = R_1 R_2$. Likewise, multiplication of quaternions corresponds to the composition of successive rotations. In particular, if Q_1 and Q_2 are two quaternions representing a rotation by θ_1 about axis n_1 and a rotation by θ_2 about axis n_2 , respectively, then the result of performing these two rotations in succession is represented by the quaternion $Q = Q_1 Q_2$. Using [\(E.22\)](#) through [\(E.25\)](#) it is straightforward to determine the quaternion product. In

particular, for two quaternions, X and Y , we compute their product, $Z = XY$, as

$$\begin{aligned} z_0 + iz_1 + jz_2 + kz_3 &= (x_0 + ix_1 + jx_2 + kx_3)(y_0 + iy_1 + jy_2 + ky_3) \\ &= x_0y_0 - x_1y_1 - x_2y_2 - x_3y_3 \\ &\quad + i(x_0y_1 + x_1y_0 + x_2y_3 - x_3y_2) \\ &\quad + j(x_0y_2 + x_2y_0 + x_3y_1 - x_1y_3) \\ &\quad + k(x_0y_3 + x_3y_0 + x_1y_2 - x_2y_1). \end{aligned}$$

By equating the real parts on both sides of the final equality, and by equating the coefficients of i , j , and k on both sides of the final equality, we obtain

$$z_0 = x_0y_0 - x_1y_1 - x_2y_2 - x_3y_3$$

$$z_1 = x_0y_1 + x_1y_0 + x_2y_3 - x_3y_2$$

$$z_2 = x_0y_2 + x_2y_0 + x_3y_1 - x_1y_3$$

$$z_3 = x_0y_3 + x_3y_0 + x_1y_2 - x_2y_1.$$

The quaternion $Q = (q_0, q_1, q_2, q_3)$ can be thought of as having the scalar component q_0 and the vector component $q = [q_1, q_2, q_3]^T$. Therefore, one often represents a quaternion by a pair, $Q = (q_0, q)$. Using this notation, q_0 represents the real part of Q , and q represents the imaginary part of Q . Using this notation, the quaternion product $Z = XY$ can be represented more compactly as

$$z_0 = x_0y_0 - x^T y$$

$$z = x_0y + y_0x + x \times y,$$

in which \times denotes the vector cross product operator.

For complex numbers, the conjugate of $a + ib$ is defined by $a - ib$. Similarly, for quaternions we denote by Q^* the conjugate of the quaternion Q , and define

$$(E.31) \quad Q^* = (q_0, -q_1, -q_2, -q_3).$$

With regard to rotation, if the quaternion Q represents a rotation by θ about the axis n , then its conjugate Q^* represents a rotation by θ about the axis $-n$. It is easy to see that

$$(E.32) \quad QQ^* = (q_0^2 + \|q\|^2, 0, 0, 0)$$

and that

$$(E.33) \quad \|QQ^*\| = \|(q_0^2 + q_1^2 + q_2^2 + q_3^2, 0, 0, 0)\| = \sum q_i^2 = \|Q\|^2.$$

A quaternion, Q , with its conjugate, Q^* , can be used to perform coordinate

transformations. Let the point p be rigidly attached to a coordinate frame \mathcal{F} , with local coordinates (x, y, z) . If Q specifies the orientation of \mathcal{F} with respect to the base frame, and T is the vector from the world frame to the origin of \mathcal{F} , then the coordinates of p with respect to the world frame are given by

$$(E.34) \quad Q(0, x, y, z)Q^* + T,$$

in which $(0, x, y, z)$ is a quaternion with zero as its real component. Quaternions can also be used to transform vectors. For example, if $n = (n_x, n_y, n_z)$ is the normal vector to the face of a polyhedron, then if the polyhedron is rotated by Q , the new direction of the normal is given by

$$(E.35) \quad Q(0, n_x, n_y, n_z)Q^*.$$

 Previous

Next 