# Chapter 3

# The Configuration Space

The concepts of the previous chapter enable the construction and transformation geometric models, which can be used to formulate path planning problems in the world, $\mathcal{W}$. For the development and analysis of algorithms, however, it will be convenient to view the path planning as a search in the space of possible transformations that can be applied to the robot, $\mathcal{A}$. This space of transformations is termed the configuration space, which represents a powerful representational tool that unifies a broad class of path planning problems in a single mathematical framework. This facilitates the development of path planning techniques that can be applied or adapted to a wide variety of robots and models.

Section 3.1 introduces some preliminary mathematical concepts from topology. Section 3.2 defines the configuration space for a given robot and its set of possible transformations. Section 3.2 characterizes the portion of the configuration space in which the robot is in collision with an obstacle. Section 3.4 presents an overview of the collision detection problem. Section 3.5 represents configuration space concepts for the complicated case in which closed kinematic chains exist.

## 3.1 Basic Topological Concepts

### 3.1.1 Topological Spaces

A set $X$ is called a *topological space* if there is a collection of subsets of $X$ called *open sets* such that the following axioms hold:

1. The union of any number of open sets is an open set.
2. The intersection of a finite number of open set is an open set.
3. Both $X$ and $\emptyset$ are open sets.

Examples of topological spaces:

- $X = \mathbb{R}$
  In this case, the open sets are defined in the familiar way; a topological space is simply a generalization of the open set concept for $\mathbb{R}$. Examples of open sets are open intervals, such as $(0, 2)$ or $(1, 3)$ (i.e., the endpoints are not included). By using the axioms, sets such as $(1, 2)$ and $(0, 3)$ must be open sets. Other examples include a set of intervals, such as $(0, 1) \cup (5, 6)$, and intervals that extend toward infinity, such as $(0, \infty)$.
- $X = \mathbb{R}^n$
  Consider the case of $n = 2$. An example of an open set is $\{(x, y) \mid x^2 + y^2 < 1\}$, the interior of the unit circle.
- $X = \{cat, dog, tree, house\}$. A topological space can be defined for any set, as long as the declared open sets obey the axioms. For this case, suppose that $\{cat\}$ and $\{dog\}$ are open sets. Then, $\{car, dog\}$ must also be an open set. Closed sets and boundary points can be easily derived for this topology once the open sets are defined.

- We could even define a trivial topological space for any set $X$ by simply declaring that $X$ and $\emptyset$ are the only open sets (verify that all of the axioms are satisfied).

**Closed sets** For a given topological space, a collection of closed sets can be inferred. A subset $C$ of a topological space $X$ is a *closed set* if and only if $X \setminus C$ is an open set.

Examples of closed sets for the case of $X = \mathbb{R}$ are closed intervals, such as $[0, 1]$, and countable collections of points, such as $\{-1, 3, 7\}$. Verify that these satisfy the definition.

Here is a question to ponder: are all subsets of $X$ either closed or open? Although it appears that open sets and closed sets are opposites in some sense, the answer to the question is NO. The interval $[0, 2\pi)$ is neither open nor closed if $X = \mathbb{R}$ (the interval $[2\pi, \infty)$ is closed and $(-\infty, 0)$ is open). Note that $X$ and $\emptyset$ are both open and closed!

**Boundary and Closure** Let $U$ be any subset of a topological space $X$. A point $x \in X$ is a *boundary point* of $U$ if neither $U$ nor $X \setminus U$ contains an open subset that contains $x$.

Consider an example for which $X = \mathbb{R}$. If $U = (0, 1)$, then 0 and 1 are boundary points of $U$. If $U = [0, 1]$, 0 and 1 are still boundary points of $U$. Let $X = \mathbb{R}^2$. Any point $(x, y)$ such that $x^2 + y^2 = 1$ is a boundary point of the set $\{(x, y) \mid x^2 + y^2 < 1\}$.

The set of all boundary points of $U$ is called the *boundary* of $U$, denoted by $\partial U$. The *closure* of $U$ is the union of $U$ with its boundary, $\partial U$. Note that the closure of a closed set always yields the original set (because it includes all of its boundary points).

**Homeomorphism** Let $f : X \to Y$ be a function between topological spaces $X$ and $Y$. If $f$ is bijective (1-1 and onto) then the inverse $f^{-1}$ exists. If both $f$ and $f^{-1}$ are continuous, then $f$ is a *homeomorphism.* The technical definition of a continuous function can be expressed in terms of open sets, and hence on any topological space. A function, $f$, is *continuous* if $f^{-1}(O) \subseteq X$ is an open set, for any open set $O \subseteq Y$. The notation $f^{-1}(O)$ denotes $\{x \in X \mid f(x) \in O\}$.

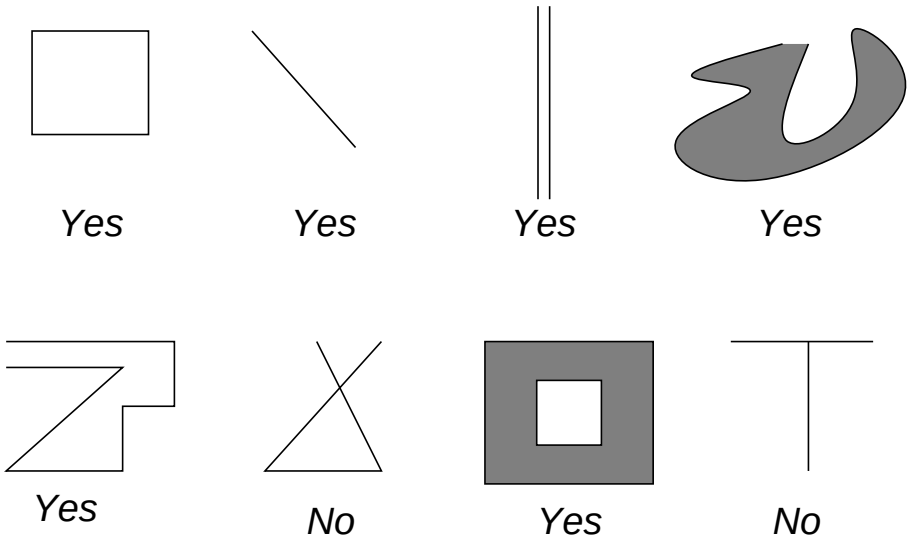
Two topological spaces, $X$ and $Y$, are said to be *homeomorphic* if there exists a homeomorphism between them. Intuitively, the homeomorphism can be considered as a way to declare two spaces to be equivalent (to a topologist, at least). Technically, the homeomorphism concept implies an equivalence relation on the set of topological spaces (verify that the reflexive, symmetric, and transitive properties are implied by the homeomorphism).

Consider some examples. Suppose $X = \{(x, y) \mid x^2 + y^2 = 1\}$ and $Y = \{(x, y) \mid x^2 + y^2 = 2\}$. The spaces $X$ and $Y$ are homeomorphic. The homeomorphism maps each point on the unit circle to a point on a circle with radius two. The space $Z = \{(x, y) \mid x^2 + y^2 < 1\}$ is not homeomorphic to $X$ or $Y$. As a general example, if $f$ represents a nonsingular linear transformation, then $f$ is a homeomorphism. For example, the rigid body transformations of the previous chapter were examples of homeomorphisms applied to the robot.

### 3.1.2 Constructing Manifolds

A topological space $M$ is a *manifold* if for every $x \in M$, an open set $O \subset M$ exists such that: 1) $x \in O$, 2) $O$ is (locally) homeomorphic to $\mathbb{R}^n$, and 3) $n$ is fixed for all $x \in M$. The dimension $n$ is referred to as the dimension of the manifold, $M$.

Intuitively, a manifold can be considered as a "nice" topological space that behaves at every point like our intuitive notion of a surface. Several examples are shown below.

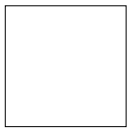
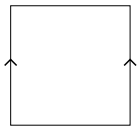
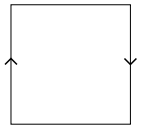
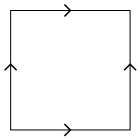
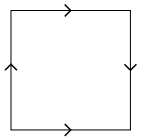
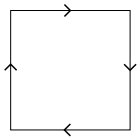



**Cartesian Products** The Cartesian product defines a way to construct new topological spaces from simpler ones. The Cartesian product, $X \times Y$, defines a new topological space. Every point in this space is of the form $\langle x, y \rangle$, in which $x \in X$ and $y \in Y$. The open sets in $X \times Y$ are formed by taking the Cartesian products of all pairs of open sets from $X$ and $Y$. The most common example of a Cartesian product is $\mathbb{R} \times \mathbb{R}$, which is equivalent to $\mathbb{R}^2$. In general, $\mathbb{R}^n$ is equivalent to $\mathbb{R} \times \mathbb{R}^{n-1}$ or $\mathbb{R} \times \mathbb{R} \times \cdots \times \mathbb{R}$.

**One-dimensional manifolds** $\mathbb{R}$ is an example of a one-dimensional manifold. The range can restricted to the unit interval to yield the manifold $(0, 1)$, which is homeomorphic $\mathbb{R}$. Another manifold can be defined, which is not homeomorphic to $\mathbb{R}$. To define this, we introduce a concept known as *identification*, in which pairs of distinct elements in the manifold are considered to be identical for all purposes. Consider the manifold given by $S^1 = [0, 1]$, in which the elements 0 and 1 are identified. Identification has the effect of "gluing" the ends of the interval together, forming a closed loop. Clearly, this is topologically distinct from $\mathbb{R}$. Another way to define $S^1$ is the set of all points on the unit circle, $\{(x, y) \mid x^2 + y^2 = 1\}$. Although these two definitions different geometrically, they are equivalent topologically.

**Two-dimensional manifolds** Several two-dimensional manifolds can be defined by applying the Cartesian product to one-dimensional manifolds. The two-dimensional manifold $\mathbb{R}^2$ is formed by $\mathbb{R} \times \mathbb{R}$. The product $\mathbb{R} \times S^1$ defines a manifold that is equivalent to an infinite cylinder. The product $S^1 \times S^1$ is a manifold that is equivalent to a torus (the surface only).

Can any other topologically distinct two-dimensional manifolds be defined? The identification trick can be applied once again, but in this case, several new manifolds can be generated. Consider identifying opposite points on the unit square in the plane. In the table below, several manifolds are constructed by identifying opposing pairs of points on opposite sides of the square. In each case, either the opposing vertical sides are identified, the opposing horizontal sides are identified, both, or neither. If the arrows for a pair of opposite edges are drawn in opposite directions, then there is a "twist" in the identification.

| Identification | Name | Manifold | Identification | Name | Manifold |
|---|---|---|---|---|---|
|  | Plane | $\mathbb{R}^2$ |  | Cylinder | $\mathbb{R} \times S^1$ |
|  | Möbius band | |  | Torus | $S^1 \times S^1$ |
|  | Klein bottle | |  | Projective plane | $P^2$ |

Through a different form of identification on the unit square, another two-dimensional manifold, $S^2$, can be defined which is the set all points on the surface of a three-dimensional sphere. Thus, $S^2 = \{(x, y, z) \mid x^2 + y^2 + z^2 = 1\}$.

**Other manifolds** Higher dimensional manifolds can be defined by taking Cartesian products of the manifolds shown thus far. For example, $\mathbb{R}^2 \times S^1$ is an important manifold that will correspond to the configuration space of a 2D rigid body. Other manifolds can be defined by using identification on the unit $n$-dimensional cube. There exist higher-dimensional versions of $S^2$ and $P^2$. In general, $S^n$ is the $n$-dimensional surface of a sphere in $\mathbb{R}^{n+1}$. $P^n$ can be defined by identifying antipodal points of $S^n$.

### 3.1.3 Paths in Topological Spaces

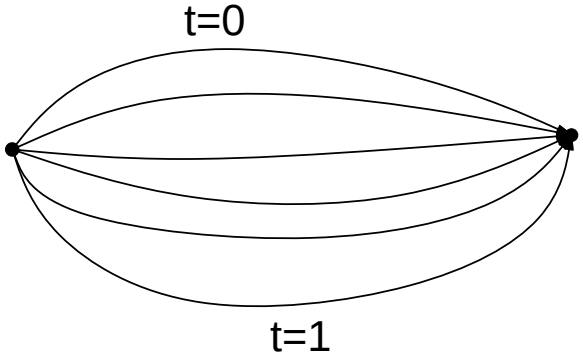
A *path*, $\tau$, is a continuous function, $\tau : [0, 1] \rightarrow X$, in which $X$ is a topological space. Note that a path is a function, not a set of points. Each point along the path is given by $\tau(s)$ for some $s \in [0, 1]$.

**Homotopy** Two paths $\tau_1$ and $\tau_2$ are homotopic (having fixed endpoints) if there exists a continuous function $h : [0, 1] \times [0, 1] \rightarrow X$ such that

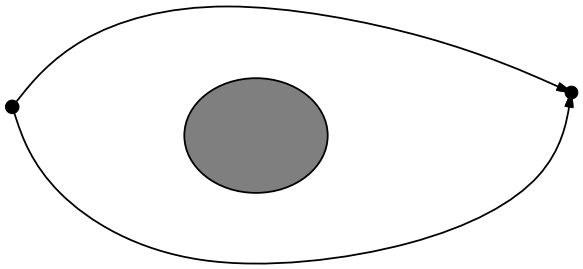
$$h(s, 0) = \tau_1(s) \quad \forall s \in [0, 1]$$
$$h(s, 1) = \tau_2(s) \quad \forall s \in [0, 1]$$
$$h(0, t) = h(0, 0) \quad \forall t \in [0, 1]$$
$$h(1, t) = h(1, 0) \quad \forall t \in [0, 1]$$

The parameter $t$ can intuitively be interpreted as a knob that is turned to gradually deform the path from $\tau_1$ into $\tau_2$. The value $t = 0$ yields $\tau_1$ and $t = 1$ yields $\tau_2$. Homotopy can considered as an equivalence relation on the set of all paths.

Homotopy can be considered as a way to continuously "warp" or "morph" one path into another.



The image of the path cannot, however, be continuously warped over a hole. In this case, the two paths are not homotopic.




**Connectedness** Consider loop paths of the form $\tau(0) = \tau(1)$. If all loop paths are homotopic, then $X$ is defined to be *simply connected*. In a 2D space, this intuitively means that $X$ contains no holes. If $X$ is not simply connected, it is defined to be *multiply-connected*. Note that there are different kinds of holes in higher-dimensional spaces. For example, in a 3D space there could be holes that run entirely through the space, or holes that are like bubbles. Bubbles cannot prevent two paths from being homotopic; however, higher-order homotopy (for functions of multiple variables) can be used to define a notion of simply-connected that prohibits bubbles and higher-dimensional generalizations.

**Fundamental Group** In the study of algebraic topology, a topological space is transformed into a mathematical group. A *group* is a set, $G$, together with an operation, $*$, such that for every $g \in G$, there exists an inverse, $g^{-1}$, such that $g * g^{-1} = 1_G$, and $1_G$ is a unique identity element of the group. The set, $Z$, of all integers is an example of a group. Although the construction of a fundamental group is beyond the scope of this course, there are some nice intuitions that follow from fundamental groups. Generally, each element of the fundamental group corresponds to a maximal set of homotopic paths. This set of paths is often called a *path class*. Consider two-dimensional topological spaces. The fundamental group of a simply-connected space is simply the contains the identity element by itself. The group contains one element because all paths are in the same path class. The fundamental group of $S^1$ is $Z$. For every integer $i \in Z$, there is a path class that corresponds to paths that "wind" $i$ times around $S^1$. Positive integers in $Z$ correspond to path classes that wind around clockwise, and negative integers wind corresponds to path classes that wind around counter-clockwise. The fundamental group of a 2D space with $m$ holes is the space of $m$-dimensional vectors, in which all elements are integers. Each component corresponds to windings around a particular hole.

## 3.2 Defining the Configuration Space

One of the most important tools in analyzing and solving a motion strategy problem is the configuration space, or C-space. Assume that the world and a robot (or set of robots) have been defined. The configuration space, $\mathcal{C}$, is a topological space generated by the set of all possible configurations. Each configuration $q \in \mathcal{C}$ corresponds to a transformation that can be applied to the robot, $\mathcal{A}$. A complicated problem such as determining how to move a piano from one room to another in a house can be reduced using C-space concepts to determining a path in $\mathcal{C}$. In other words, the piano (3D rigid body) becomes a moving point in $\mathcal{C}$.

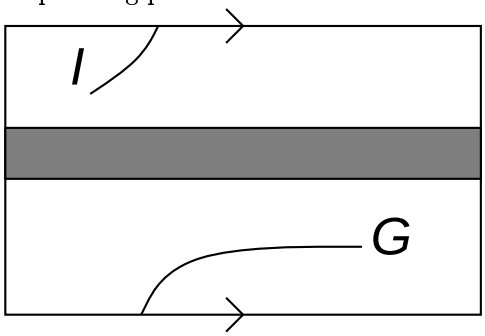
**2D Rigid Bodies** Consider the case in which the world, $\mathcal{W} = \mathbb{R}^2$, and the robot, $\mathcal{A}$, is a rigid body that can both rotate and translate. Recall that three parameters, $x$, $y$, and $\theta$, are necessary to place $\mathcal{A}$ at any position and orientation. For the translation parameters, $x \in \mathbb{R}$ and $y \in \mathbb{R}$; however, for the rotation parameter, $\theta \in [0, 2\pi)$ with the understanding that 0 and $2\pi$ would produce the same rotation. This is equivalent to identification, and the set of all 2D rotations can be considered as $S^1$. Every point in $S^1$ can be mapped to a unique rotation (for example, use the angle of a point on the unit circle in polar coordinates). Using Cartesian products, the configuration space can be constructed by "gluing" the spaces for each parameter

together to yield

$$\mathcal{C} = \mathbb{R} \times \mathbb{R} \times S^1 = \mathbb{R}^2 \times S^1$$

In practice, there are usually limits on the range of $x$ and $y$, but this makes no difference topologically. An interval in $\mathbb{R}$ is topologically equivalent to $\mathbb{R}$ by a homeomorphism.

It is important to consider the topological implications of $\mathcal{C}$. Since $S^1$ is multiply connected, $\mathbb{R} \times S^1$ and $\mathbb{R}^2 \times S^1$ are multiply connected. In fact, all three have the same fundamental group, $Z$, which is the set of integers. It is difficult to visualize $\mathcal{C}$ because it is a three-dimensional manifold. (Be careful not to confuse the dimension of a manifold with the dimension of the space that contains it.) Consider $\mathbb{R} \times S^1$ because it is easy to visualize. Imagine a path planning problem in $\mathbb{R} \times S^1$ that looks like the following:




The top and bottom are identified to make a cylinder, and there is an obstacle across the middle. Suppose the task is to find a path from $I$ to $G$. If the top and bottom were not identified, then it would not be possible to connect $I$ to $G$; however, once we know that we are looking at a cylinder, the task is straightforward. In general, it is very important to understand the topology of $\mathcal{C}$; otherwise, potential solutions will be lost.

**3D Rigid Bodies** One might expect that defining $\mathcal{C}$ for a 3D rigid body is an obvious extension of the 2D case; however, 3D rotations are significantly more complicated. The resulting C-space will be $\mathcal{C} = \mathbb{R}^3 \times P^3$, in which $P^3$ is a three-dimensional projective space. The $\mathbb{R}^3$ arises from possible choices of translation parameters, $x$, $y$, and $z$.

The set of all rotations in 2D could be nicely represented by a single angle, $\theta$, but what parameters should be used for the set of all 3D rotations? Yaw, pitch, and roll angles, $\alpha$, $\beta$, and $\gamma$, might appear to be a reasonable choice; however, there are several problems. There are some cases in which nonzero angles yield the identity rotation matrix, which is equivalent to $\alpha = \beta = \gamma = 0$. In general, there are many cases in which distinct values for yaw, pitch, and roll angles yield identical rotation matrices. A solution to this problem can be obtained by representing rotations with quaternions, which are similar to complex numbers, but have one real part and three imaginary parts.

Before introducing quaternions to represent 3D rotations, consider using complex numbers to represent 2D rotations. Any complex number, $a + bi$ is considered a *unit* complex number if and only if $a^2 + b^2 = 1$. Recall that complex numbers can be represented in polar form as $re^{i\theta}$; a unit complex number is simply $e^{i\theta}$. A one-to-one correspondence can be made between 2D rotations and unit complex numbers by letting $e^{i\theta}$ correspond to a 2D rotation by $\theta$. If complex numbers are used to represent rotations, it is important that they behave algebraically in the same way. If two rotations are combined, can the corresponding complex numbers be combined? This can be achieved by simply multiplying the complex numbers. Suppose that a 2D robot is rotated by $\theta_1$, followed by $\theta_2$. In polar form, the complex numbers are multiplied to yield $e^{i\theta_1} e^{i\theta_2} = e^{i(\theta_1 + \theta_2)}$, which clearly represents a rotation of $\theta_1 + \theta_2$. If the unit complex number is represented in Cartesian form, then the rotations corresponding to $a_1 + b_1 i$ and $a_2 + b_2 i$ are combined to obtain $a_1 a_2 - b_1 b_2 + a_1 b_2 i + a_2 b_1 i$. Note that we did not use complex number numbers to provide the solution to a polynomial equation, we simply borrowed their nice algebraic properties. Also, note that only one independent parameter was specified. Even though complex numbers have two components, the second component can be determined from the first using the constraint $a^2 + b^2 = 1$.

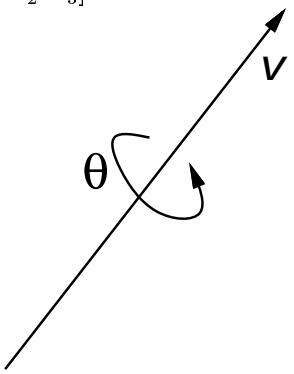
The manner in which complex numbers were used to represent 2D rotations will now be generalized to using quaternions to represent 3D rotations. Let $H$ represent the space of *quaternions*, in which each quaternion, $h \in H$ is represented as $h = a + bi + cj + dk$. A quaternion can be considered as a four-dimensional vector. The symbols $i$, $j$, and $k$, are used to denote three distinct imaginary components of the quaternion.

The following relationships are defined: $i^2 = j^2 = k^2 = -1$, $ij = k$, $jk = i$, and $ki = j$. Using these relationships, multiplication of two quaternions, $h_1 = a_1 + b_1 i + c_1 j + d_1 k$ and $h_2 = a_2 + b_2 i + c_2 j + d_2 k$, can be defined as $h_1 \cdot h_2 =$

$$(a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2) + (a_1b_2 + a_2b_1 + c_1d_2 - c_2d_1)i + (a_1c_2 + a_2c_1 + b_2d_1 - b_1d_2)j + (a_1d_2 + a_2d_1 + b_1c_2 - b_2c_1)k.$$

For convenience, this multiplication can be expressed in terms of vector multiplications, a dot product, and a cross product. Let $v = [b \;\; c \;\; d]$, a three dimensional vector that represents the final three quaternion components. The first component of $h_1 \cdot h_2$ is $a_1a_2 - v_1 \cdot v_2$. The final three components are given by the vector $a_1v_2 + a_2v_1 - v_1 \times v_2$.

A quaternion is called a *unit* quaternion if and only if $a^2 + b^2 + c^2 + d^2 = 1$. Three-dimensional rotations will be represented as follows. Any 3D rotation can be considered as a rotation by an angle $\theta$ about the axis given by the unit direction vector $v = [v_1 \;\; v_2 \;\; v_3]$:




Let the following unit quaternion represent this rotation:

$$h = \cos\frac{\theta}{2} + v_1 \sin\frac{\theta}{2}\, i + v_2 \sin\frac{\theta}{2}\, j + v_3 \sin\frac{\theta}{2}\, k.$$

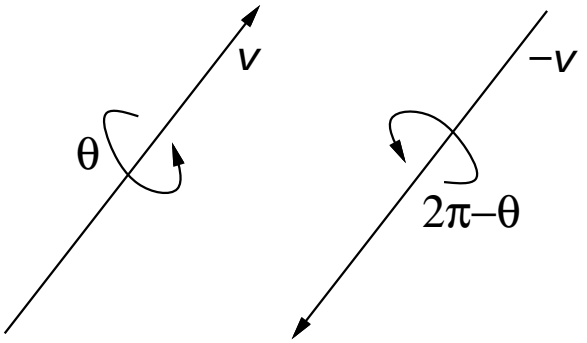
It can be shown that the rotation matrix represented by the unit quaternion $h = a + bi + cj + dk$ is

$$R(h) = \begin{pmatrix} 2(a^2 + b^2) - 1 & 2(bc - ad) & 2(bd - ac) \\ 2(bc + ad) & 2(a^2 + c^2) - 1 & 2(cd - ab) \\ 2(bd - ac) & 2(bc + ad) & 2(a^2 + d^2) - 1 \end{pmatrix}$$

Another way to transform a point, $p \in \mathbb{R}^3$, using the quaternion $h$ is to construct a quaterion of the form $p' = (0, p)$, and perform the quaternion product $hp'h^*$. In this product, $h^*$ denotes the conjugate of $h$, which is given by $[a \;\; -b \;\; -c \;\; -d]$.

Note that the unit quaternion can be considered as a point in $S^3$ because of the constraint $a^2+b^2+c^2+d^2 = 1$. The topology of the space of rotations, however, is more complicated than $S^3$ because $h$ and $-h$ represent the same rotation. This implies that opposite (antipodal) points of $S^3$ are equivalent through identification. It follows that the projective space $P^3$ is the topological space that corresponds to the set of all 3D rotations.

To see that $h$ and $-h$ are the same rotation, it should first be observed that a rotation of $\theta$ about the direction $v$ is equivalent to a rotation of $2\pi - \theta$ about the direction $-v$:



Using the first representation, a quaternion $h = a + bi + cj + dk$ can be formed. Consider the quaternion representation of the second expression of rotation with respect to the first. The real part will be

$$\cos(\frac{2\pi - \theta}{2}) = \cos(\pi - \frac{\theta}{2}) = -\cos(\frac{\theta}{2}) = -a.$$

The $i$, $j$, and $k$ components will be

$$-v \sin(\frac{2\pi - \theta}{2}) = -v \sin(\pi - \frac{\theta}{2}) = -v \sin(\frac{\theta}{2}) = [-b \quad -c \quad -d].$$

The quaternion $-h$ has been constructed. Thus, $h$ and $-h$ represent the same rotation.

**Chains of Bodies and Other Structures** For a 2D chain of bodies attached by revolute joints, with the first link attached, but able to rotate:

$$\mathcal{C} = S^1 \times S^1 \times \cdots \times S^1$$

If the first link can translate, then

$$\mathcal{C} = \mathbb{R}^2 \times S^1 \times S^1 \times \cdots \times S^1$$

For a 3D chain of bodies attached by revolute joints, with the first link attached, but unable to rotate:

$$\mathcal{C} = S^1 \times S^1 \times \cdots \times S^1$$

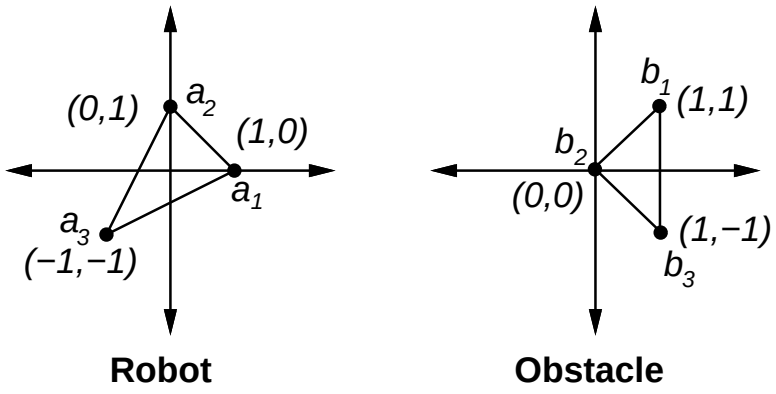
If the first link can translate and rotate, then

$$\mathcal{C} = \mathbb{R}^3 \times P^3 \times S^1 \times S^1 \times \cdots \times S^1$$

If any of the links are not revolute, then the configuration space components must be carefully assigned. Each prismatic joint yields an $\mathbb{R}$ component. A spherical joint can yield a $P^3$ component. Limits on the joint variables might alter the topology. For example, if a revolute joint has minimum and maximum angular limits, then its corresponding component will be $\mathbb{R}$, as opposed to $S^1$ (the human elbow joint is such an example).

## 3.3 The Obstacle Region in $\mathcal{C}$

The obstacle region in $\mathcal{C}$ identifies the set of all configurations that lead to collision. Suppose that the world, $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$, contains an obstacle region, $\mathcal{O}$, and a rigid robot $\mathcal{A}$. By considering the space of possible transformations of $\mathcal{A}$, a configuration space, $\mathcal{C}$ can be defined. The next task is to characterize configurations at which the robot will collide with obstacles in the world. Conceptually, the transformation appears as:



Formally, the obstacle region, $\mathcal{C}_{obs}$, is defined as

$$\mathcal{C}_{obs} = \{q \in \mathcal{C} \mid \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset\}, \tag{3.1}$$

which is the set of all configurations, $q$, at which $\mathcal{A}(q)$ (the transformed robot) intersects the obstacle region, $\mathcal{O}$.

Several important terms can be defined. The set of configurations not in $\mathcal{C}_{obs}$ is referred to as the *free space*, $\mathcal{C}_{free}$. Thus, $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obs}$. It is assumed that $\mathcal{O}$ and $\mathcal{A}$ are closed sets in $\mathcal{W}$. This implies (see Latombe) that $\mathcal{C}_{obs}$ is closed, and that $\mathcal{C}_{free}$ is open. Let $\mathcal{C}_{contact} = \partial\mathcal{C}_{free}$ be the boundary of $\mathcal{C}_{free}$. Let $\mathcal{C}_{valid} = cl(\mathcal{C}free)$, which is the closure of $\mathcal{C}_{free}$. The difference between $\mathcal{C}_{free}$ and $\mathcal{C}_{valid}$, is that configurations in $\mathcal{C}_{valid}$ allow the robot to "touch" or "graze" the obstacles, while configurations in $\mathcal{C}_{free}$ prohibit any form of contact.

**Robots that consist of multiple bodies** If the robot consists of multiple bodies, the situation is more complicated. The definition in (3.1) only implies that the robot does not collide with the obstacles; however, if the robot consists of multiple bodies, then it might also be appropriate to avoid collisions between different parts of the robot. Let the robot be modeled as a collection, $\{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_m\}$, of $m$ bodies, which could be attached by joints, or could be unattached.

Let $P$ be called that *collision pairs*, and denote a set of two-dimensional integer vectors, each of the form $[i, j]$ such that $1 \leq i, j \leq m$. Each collision pair, $[i, j] \in P$, indicates that collision between $\mathcal{A}_i(q)$ and $\mathcal{A}_j(q)$ should be avoided. Note that one typically does not want to check for collisions between *all* pairs of bodies. For example, for a chain of bodies, it is customary to allow contact between consecutive bodies in the chain, but to avoid collisions between nonconsecutive bodies.

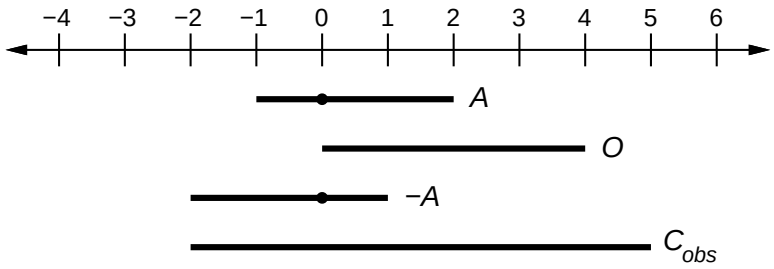
The formal definition below for $\mathcal{C}_{obs}$ includes any configurations that cause a body to collide with an obstacle (the left term) or that cause a designated collision pair to be in collision (the right term):

$$\mathcal{C}_{obs} = \left\{ \bigcup_{i=1}^{m} \{q \in \mathcal{C} \mid \mathcal{A}_i(q) \cap \mathcal{O} \neq \emptyset\} \right\} \bigcup \left\{ \bigcup_{[i,j] \in P} \{q \in \mathcal{C} \mid \mathcal{A}_i(q) \cap \mathcal{A}_j(q) \neq \emptyset\} \right\}. \tag{3.2}$$

**The basic path planning problem (Piano Moving)** Using the concepts of this chapter, the basic path planning problem can be defined succinctly. Given the world, $\mathcal{W}$, an algebraic obstacle region, $\mathcal{O}$, an algebraic robot, $\mathcal{A}$, a configuration space, $\mathcal{C}$, and two configurations, $q_{init} \in \mathcal{C}_{free}$ and $q_{goal} \in \mathcal{C}_{free}$, the goal is to find a path, $\tau$, such that $\tau[0, 1] \rightarrow \mathcal{C}_{free}$, $\tau(0) = q_{init}$, and $\tau(1) = q_{goal}$. If contact between the robot and obstacles is allowed, then $\mathcal{C}_{free}$ can be replaced with $\mathcal{C}_{valid}$. The key difficulty is to characterize $\mathcal{C}_{free}$ (or equivalently, $\mathcal{C}_{obs}$).

**The translational case** The simplest case for characterizing $\mathcal{C}_{obs}$ is when $\mathcal{C} = \mathbb{R}^n$ for $n = 1$, 2, and 3, and the robot is a rigid body that is restricted to translation only. Under these conditions, $\mathcal{C}_{obs}$ can be expressed as a type of convolution. Let any two subsets of $\mathbb{R}^n$, $X$ and $Y$, let their Minkowski difference $\ominus$ be the set $Z = X \ominus Y$. A point, $z$ belongs to $Z$ if any only if there exists an $x \in X$ and there exists a $y \in Y$ such that $z = x - y$. The subtraction $x - y$ is applied in the vector sense.

In terms of the Minkowski difference, $\mathcal{C}_{obs} = \mathcal{O} \ominus \mathcal{A}(0)$. To see this, it is helpful to consider a one-dimensional example. The Minkowski difference between $X$ and $Y$ can also be considered as the Minkowski sum of $X$ and $-Y$. The Minkowski sum, $\oplus$, is obtained by simply adding elements of $X$ and $Y$, as opposed to subtracting them. The set $-Y$ is obtained by replacing each $y \in Y$ by $-y$. In the example below, both the robot, $\mathcal{A} = [-1, 2]$ and obstacle region, $\mathcal{O} = [0, 4]$ are intervals in a one-dimensional world, $\mathcal{W} = \mathbb{R}$.




The negation, $-\mathcal{A}$, of the robot is shown as the interval $[-2, 1]$. Finally, by applying the Minkowski sum to $\mathcal{O}$ and $-\mathcal{A}$, the $\mathcal{C}_{obs}$ is obtained as the interval $[-2, 4]$.

One interesting interpretation of the Minkowski difference is as a convolution. For our previous example, let $f : \mathbb{R} \rightarrow \{0, 1\}$ be a function such that $f(x) = 1$ if and only if $x \in \mathcal{O}$. Similarly, let $g : \mathbb{R} \rightarrow \{0, 1\}$ be a function such that $g(x) = 1$ if and only if $x \in \mathcal{A}$. The following convolution,

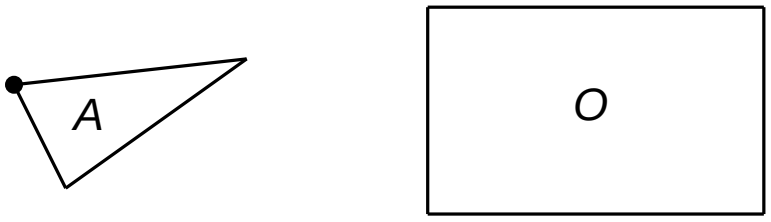
$$h(x) = \int_{-\infty}^{\infty} f(\tau) g(x - \tau) d\tau,$$

will yield a function $h$ of $x$ that is 1 if $x \in \mathcal{C}_{obs}$, and 0 otherwise.

**A polygonal C-space obstacle** An efficient method of computing $\mathcal{C}_{obs}$ exists in the case of a 2D world that contains a convex polygonal obstacle, $\mathcal{O}$, and a convex polygonal robot, $\mathcal{A}$. For this problem, $\mathcal{C}_{obs}$ is also a convex polygon. Recall that nonconvex obstacles and robots can be modeled as the union of convex parts. The concepts discussed below can also be applied in the nonconvex case by considering $\mathcal{C}_{obs}$ as the union of convex components, each of which corresponds to a convex component of $\mathcal{A}$ colliding with a convex component of $\mathcal{O}$.

The method is based on sorting normals to the edges of the polygons on the basis of angles. The key observation is that every edge of $\mathcal{C}_{obs}$ is a translated edge from either $\mathcal{A}$ or $\mathcal{O}$. In fact, every edge from $\mathcal{O}$ and $\mathcal{A}$ is used exactly once in the construction of $\mathcal{C}_{obs}$. The only problem is to determine the ordering of these edges of $\mathcal{C}_{obs}$. Let $\alpha_1$, $\alpha_2$, ..., $\alpha_n$ denote the angles of the inward edge normals in counter-clockwise order around $\mathcal{A}$. Let $\beta_1$, $\beta_2$, ..., $\beta_n$ denote the outward edge normals to $\mathcal{O}$. After sorting both sets of angles in circular order around $S^1$, $\mathcal{C}_{obs}$ can be constructed incrementally by adding the edges that correspond to the sorted normals, in the order in which they are encountered.

To gain an understanding of the method, consider the case of a triangular robot and a rectangular obstacle:




The black dot on $\mathcal{A}$ denotes the origin of its coordinate frame. Consider sliding the robot around the obstacle in such as way that they are always in contact:

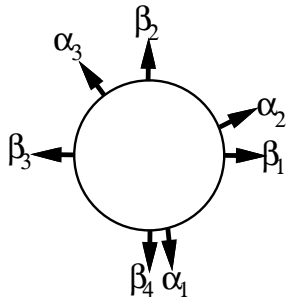
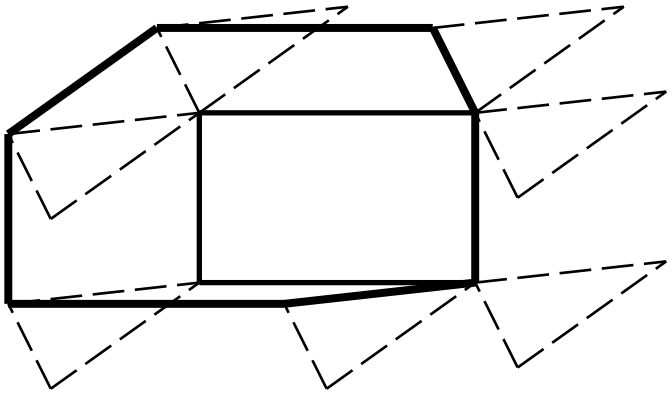
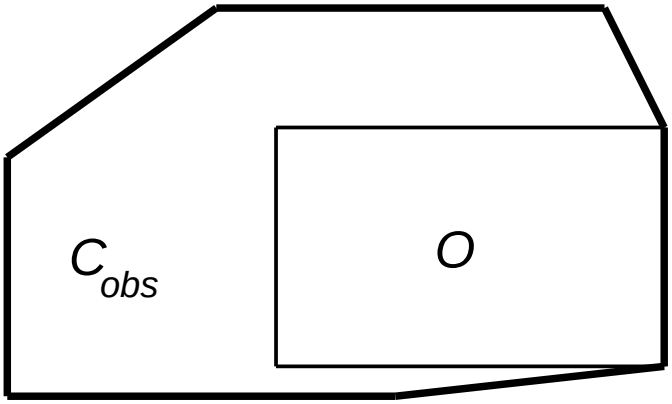


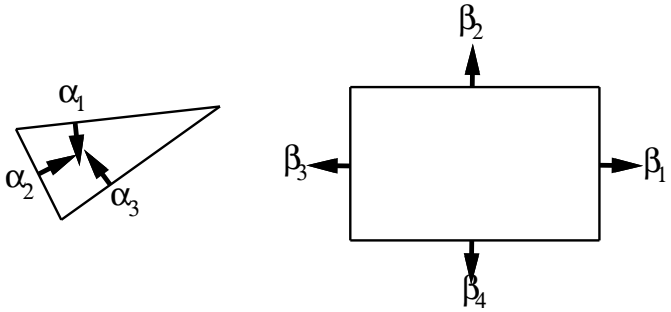

Figure 3.1: The edge normals are sorted by orientation.



This corresponds to the traversal of all of the configurations in $\mathcal{C}_{contact}$. The origin of $\mathcal{A}$, will trace out the edges of $\mathcal{C}_{obs}$:



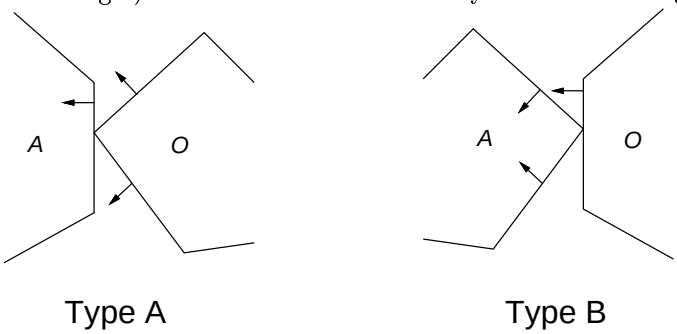

Note that there are 7 edges, and that each edge corresponds to either an edge of $\mathcal{A}$ or an edge of $\mathcal{O}$. The directions of the normals are defined as shown:




When sorted as shown in Figure 3.1, the edges of $\mathcal{C}_{obs}$ can be incrementally constructed.

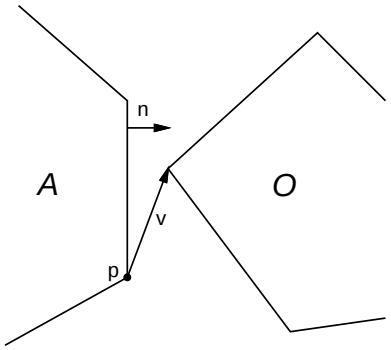
The running time of the algorithm is $O(n+m)$, in which $n$ is the number of edges defining $\mathcal{A}$, and $m$ is the number of edges defining $\mathcal{O}$. Note that the angles can be sorted in linear time because they already appear in counterclockwise order around $\mathcal{A}$ and $\mathcal{O}$; a simple merging is performed. If two edges are collinear, then they can be placed end-to-end as a single edge of $\mathcal{C}_{obs}$.

The previous method quickly identifies each edge that contributes to $\mathcal{C}_{obs}$. Suppose that we would like to construct a geometric model of $\mathcal{C}_{obs}$ in terms of half planes. This requires defining $n+m$ linear equations (assume there are no collinear edges). There are two different ways in which an edge of $\mathcal{C}_{obs}$ is generated:




Type A contact refers to the case in which an edge of $\mathcal{A}$ is in contact with a vertex of $\mathcal{O}$. Type A contacts contribute to $n$ edges of $\mathcal{C}_{obs}$, once for each edge of $\mathcal{A}$. Type B contact refers to the case in which an edge of $\mathcal{A}$ is in contact with a vertex of $\mathcal{O}$. This contributes to $m$ edges of $\mathcal{C}_{obs}$. Notice the relationships between the edge normals shown in the figure. For Type A, the inward edge normal lies between the outward edge normals of the obstacle edges that share the contact vertex. Likewise, for Type B, the outward edge normal of $\mathcal{O}$ lies between the inward edge normals of $\mathcal{A}$.

Using the ordering shown in Figure 3.1, Type A contacts occur precisely when an edge normal of $\mathcal{A}$ is encountered, and Type B contacts occur precisely when an edge normal of $\mathcal{O}$ is encountered. The task is to determine a line equation at each of these instances. Consider the case of a Type A contact; the Type B contact can be handled in a similar manner. The key condition that must be satisfied, in addition to the directions of the edge normals, is that the contact vertex of $\mathcal{O}$ lies on the contact edge of $\mathcal{A}$. Recall that convex obstacles were constructed by the intersection of half planes. Each edge of $\mathcal{C}_{obs}$ can be defined in terms of a supporting half plane; hence, it is only necessary to determine whether the vertex of $\mathcal{O}$ lies on the line through the contact edge of $\mathcal{A}$. This condition occurs precisely when the vectors $n$ and $v$, shown below are perpendicular.




This occurs precisely when their inner product (dot product) is zero, $n \cdot v = 0$. Note that $n$ does not depend on the configuration of $\mathcal{A}$ (because rotation is not allowed). The vector $v$, however, depends on the translation, $q = (x_0, y_0)$ of the point $p$. Therefore, it is more appropriate to write the condition as $n \cdot v(x_0, y_0) = 0$. The transformation equations are linear for translation; hence, $n \cdot v = 0$ is the equation of a line in $\mathcal{C}$. For example, if the coordinates of $p$ are $(1,2)$ when $\mathcal{A}$ is at the origin, then the expression for $p$ at configuration $(x_0, y_0)$ is simply $(1 + x_0, 2 + y_0)$. Clearly, the resulting inner product $n \cdot v = 0$ is a linear equation. Let $f(x_0, y_0) = n \cdot v$. Let $H = \{(x_0, y_0) \in \mathcal{C} \mid f(x_0, y_0) \leq 0\}$. Observe that configurations not in $H$ lie in $\mathcal{C}_{free}$. The half plane $H$ is used to define one edge of $\mathcal{C}_{obs}$. The obstacle region $\mathcal{C}_{obs}$ can be completely characterized by intersecting the resulting half planes for each of the Type A and Type B contacts. This yields a convex polygon in $\mathcal{C}$ that has $n+m$ sides, as expected.

**A polyhedral C-space obstacle** Most of the previous ideas generalize nicely for the case of a polyhedral robot that is capable of translation only in a 3D world that contains polyhedral obstacles. If $\mathcal{A}$ and $\mathcal{O}$ are convex polyhedra, the resulting $\mathcal{C}_{obs}$ is a convex polyhedron.

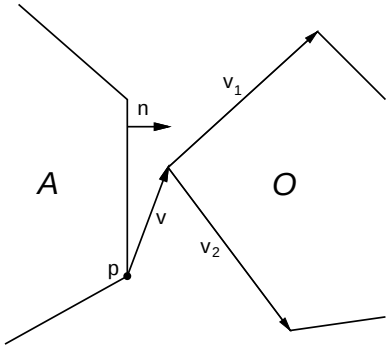
There are three different kinds of contacts that lead to half spaces:

- Type A: A face of $\mathcal{A}$ and a vertex of $\mathcal{O}$
- Type B: A vertex of $\mathcal{A}$ and a face of $\mathcal{O}$
- Type C: An edge of $\mathcal{A}$ and an edge of $\mathcal{O}$

Each half space defines a face of the polyhedron. The resulting polyhedron can be constructed in $O(n+m+k)$ time, in which $n$ is the number of faces of $\mathcal{A}$, $m$ is the numfer of faces of $\mathcal{O}$, and $k$ is the number of faces of $\mathcal{C}_{obs}$, which is at most $nm$.

**A semi-algebraic C-space obstacle** Unfortunately, the cases in which $\mathcal{C}_{obs}$ is polygonal or polyhedral are quite limited. Most problems yield far more complicated C-space obstacles. With a few tricks, however, $\mathcal{C}_{obs}$ can almost always be expressed using algebraic models. This will hold true for any of the robots and transformations presented in Chapter 2. It might not be true for other kinds of transformations, such as parameters that warp a flexible material.

Consider the case of a convex polygonal robot and a convex polygonal obstacle in a 2D world. The robot is allowed to rotate and translate; thus, $\mathcal{C} = \mathbb{R}^2 \times S^1$ and $q = (x_0, y_0, \theta)$. The task is to define a set of algebraic primitives that can be combined to define $\mathcal{C}_{obs}$. Once again, it is important to distinguish between Type A and Type B contacts. We will describe how to construct the algebraic primitives for the Type A contacts; Type B can be handled in a similar manner. Confider the following figure for a Type A contact:




For the translation-only case, we were able to determine all of the Type A conditions by sorting the edge normals. With rotation, the ordering of edge normals depends on $\theta$. This implies that the applicability of a Type A contact depends on $\theta$. Recall the constraint that the inward normal of $\mathcal{A}$ must lie between the outward normals of the edges of $\mathcal{O}$ that contain the vertex of contact. This constraint can be expressed in terms of inner products using the vectors $v_1$ and $v_2$. The statement regarding the directions of the normals can equivalently be formulated as the statement that the angle between $n$ and $v_1$, and between $n$ and $v_2$, must each be less than $\frac{\pi}{2}$. Using inner products, this implies that $n \cdot v_1 \geq 0$ and $n \cdot v_2 \geq 0$. As in the translation case, the condition $n \cdot v = 0$ is required for contact. Note that $n$ depends on $q$. A given configuration, $q$, lies in $\mathcal{C}_{free}$ if $n(q) \cdot v_1 \geq 0$, $n(q) \cdot v_2 \geq 0$, and $n(q) \cdot v > 0$. Note that for any $q \in \mathcal{C}$, if $n(q) \cdot v_1 \geq 0$, $n(q) \cdot v_2 \geq 0$, and $n(q) \cdot v(q) > 0$, then $q \in \mathcal{C}_{free}$. Let $H_f$ denote the set of configurations that satisfy these conditions. These conditions can be used to determine whether a point is in $\mathcal{C}_{free}$; however, it is not a complete characterization of $\mathcal{C}_{free}$ because all other Type A and Type B contacts could possibly imply that other points are in $\mathcal{C}_{free}$. Thus, $H_f$ is s strict subset, $H_f \subset \mathcal{C}_{free}$. This implies that the complement, $\mathcal{C} \setminus H_f$, is a superset of $\mathcal{C}_{obs}$, i.e., $\mathcal{C}_{obs} \subset \mathcal{C} \setminus H_f$. Let $H_A = \mathcal{C} \setminus H_f$. The following primitives can be used to define $H_A$:

$$H_1 = \{q \in \mathcal{C} \mid n(q) \cdot v_1 < 0\}$$
$$H_2 = \{q \in \mathcal{C} \mid n(q) \cdot v_2 < 0\}$$
$$H_3 = \{q \in \mathcal{C} \mid n(q) \cdot v(q) \leq 0\}$$

The set of configurations in $H_A = H_1 \cup H_2 \cup H_3$ possibly lie in $\mathcal{C}_{obs}$, although some may lie in $\mathcal{C}_{free}$. It is known that any configurations outside of $H_A$ must be in $\mathcal{C}_{free}$, and should therefore be eliminated from consideration. If the set $H_A$ is intersected with all other corresponding sets for each possible Type A and Type B contact, the result will be $\mathcal{C}_{obs}$. Each contact has the opportunity to remove a portion of $\mathcal{C}_{free}$ from consideration. Eventually, enough pieces of $\mathcal{C}_{free}$ are removed so that the only configurations remaining lie in $\mathcal{C}_{obs}$. For any Type A constraint, $(H_1 \cup H_2) \setminus H_3 \subseteq \mathcal{C}_{free}$. A similar statement can be made for Type B constraints. A logical predicate could also be defined, which returns TRUE if $q \in \mathcal{C}_{obs}$ and FALSE otherwise.

One important issue remains. The expression $n(q)$ is not a polynomial because of the $\cos\theta$ and $\sin\theta$ terms due to rotation. If polynomals could be substituted for these expressions, then everything would be fixed because the expression of the normal vector (not a unit normal) and the inner product are both linear functions, thus converting polynomials into polynomials. Such a substitution can be made using stereographic projection (see Latombe).

A simpler substitution can be made using complex numbers to represent rotation. Recall that $a + bi$ can be used to represent rotation, assuming that $a^2 + b^2 = 1$. The conversion from polar form to rectangular form yields $a = \cos\theta$ and $b = \sin\theta$. This implies that the $2 \times 2$ rotation matrix can be written as

$$\begin{pmatrix} a & -b \\ b & a \end{pmatrix}$$

and the $3 \times 3$ homogeneous transformation matrix becomes

$$\begin{pmatrix} a & -b & x_0 \\ b & a & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

.

Clearly, any transformed point on $\mathcal{A}$ will be a linear function of $a$, $b$, $x_0$, and $y_0$. This was a simple trick to make a nice, lienar function, but what was the cost? The dependency is now on $a$ and $b$, instead of $\theta$. This appears to increase the dimension of $\mathcal{C}$ from 3 to 4, and $\mathcal{C} = \mathbb{R}^4$. However, the special algabraic primitive $H_s = \{(x_0, y_0, a, b) \in \mathcal{C} \mid a^2 + b^2 = 1\}$ must be added. This constrains the angles to lie in $S^1$, as opposed to $\mathbb{R}^2$, which would correspond to any possible assignment of $a$ and $b$. By using complex numbers, algebraic primitives in $\mathbb{R}^4$ are obtained for each Type A and Type B contact. By defining $\mathcal{C} = \mathbb{R}^4$, then the following algebraic primitives are defined for a Type A contact:

$$H_1 = \{(x_0, y_0, a, b) \in \mathcal{C} \mid n(x_0, y_0, a, b) \cdot v_1 < 0\}$$

$$H_2 = \{(x_0, y_0, a, b) \in \mathcal{C} \mid n(x_0, y_0, a, b) \cdot v_2 < 0\}$$

$$H_3 = \{(x_0, y_0, a, b) \in \mathcal{C} \mid n(x_0, y_0, a, b) \cdot v(x_0, y_0, a, b) \leq 0\}$$

The following set must be intersected with $H_A$ to ensure that unit complex numbers are used:

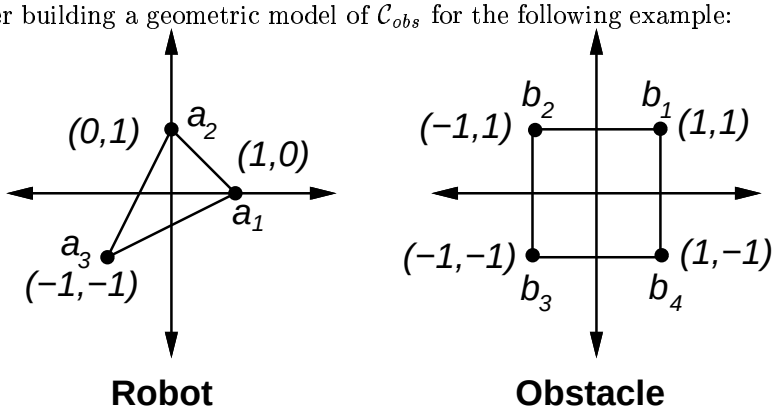
$$H_s = \{(x_0, y_0, a, b) \in \mathcal{C} \mid a^2 + b^2 - 1 = 0\}$$

This constraint preserves the topology of the original configuration space. The set $H_s$ remains fixed over all Type A and Type B contacts; therefore, it only needs to be considered once.

Consider other cases beyond 2D translation and rotation. For a chain of bodies in a 2D world, the rotation component of each homogeneous transformation matrix can be expressed using complex numbers to represent rotation. This will increase the dimension of the configuration space by the number of revolute joints, but the resulting primitives will be algebraic. Alternatively, the stereographic projection can be used (see Latombe) to keep the dimension fixed. For a chain of bodies, it one might want to also add expressions that check for self-collision. Self-collision occurs when one component of the robot intersects another.

For the case of a 3D rigid body that can translate or rotate, quaternions can be used to generate a rotation matrix in which each component is a polynomial. Thus, any configuration space for 3D rigid bodies or a chain of bodies can be expressed using polynomials.

**Example** Consider building a geometric model of $\mathcal{C}_{obs}$ for the following example:




Suppose that the orientation of $\mathcal{A}$ is fixed shown, and $\mathcal{C} = \mathbb{R}^2$. In this case, $\mathcal{C}_{obs}$ will be a convex polygon with seven sides. The following contact conditions occur (the ordering is given as normals appear as shown in Figure 3.1).

| Type | Vertex | $\mathcal{O}$ Edge | $n$ | $v$ | Half Plane |
|---|---|---|---|---|---|
| B | $a_3$ | $b_4$-$b_1$ | $[1,0]$ | $[x_0-2, y_0]$ | $H_B = \{q \in \mathcal{C} \mid x_0 - 2 \leq 0\}$ |
| B | $a_3$ | $b_1$-$b_2$ | $[0,1]$ | $[x_0-2, y_0-2]$ | $H_B = \{q \in \mathcal{C} \mid y_0 - 2 \leq 0\}$ |
| A | $b_2$ | $a_3$-$a_1$ | [1,-2] | $[-x_0, 2-y_0]$ | $H_A = \{q \in \mathcal{C} \mid -x_0 + 2y_0 - 4 \leq 0\}$ |
| B | $a_1$ | $b_2$-$b_3$ | $[-1,0]$ | $[2+x_0, y_0-1]$ | $H_B = \{q \in \mathcal{C} \mid -x_0 - 2 \leq 0\}$ |
| A | $b_3$ | $a_1$-$a_2$ | $[1,1]$ | $[-1-x_0, -y_0]$ | $H_A = \{q \in \mathcal{C} \mid -x_0 - y_0 - 1 \leq 0\}$ |
| B | $a_2$ | $b_3$-$b_4$ | $[0,-1]$ | $[x_0+1, y_0+2]$ | $H_B = \{q \in \mathcal{C} \mid -y_0 - 2 \leq 0\}$ |
| A | $b_4$ | $a_2$-$a_3$ | $[-2,1]$ | $[2-x_0, -y_0]$ | $H_A = \{q \in \mathcal{C} \mid 2x_0 - y_0 - 4 \leq 0\}$ |

For the rotation case, all possible contacts must be considered. For this example, there are 12 Type A contacts and 12 Type B contacts. Each contact produces 3 algebraic primitives. With the inclusion of $H_s$, this simple example produces 73 primitives! Rather than construct all of these, we derive the primitives for a single contact. Consider the Type B contact between $a_3$ and $b_4$-$b_1$. The outward edge normal, $n$, remains fixed at $n = [1,0]$. The vectors $v_1$ and $v_2$ are derived from the edges that share $a_3$, which are $a_3$-$a_2$ and $a_3$-$a_1$. Note that each of $a_1$, $a_2$, and $a_3$ depend on the configuration. Using the 2D homogeneous transformation, $a_1$ at configuration $(x_0, y_0, \theta)$ is $(\cos\theta + x_0, sin\theta + y_0)$. Suppose that the unit complex number $a + bi$ is used to represent the rotation $\theta$. This implies that $a = \cos\theta$ and $b = \sin\theta$. The representation of $a_1$ becomes $(a + x_0, b + y_0)$. The representations of $a_2$ and $a_3$ are $(-b + x_0, a + y_0)$ and $(-a + b + x_0, -b - a + y_0)$, respectively. It follows that $v_1 = a_2 - a_3 = [a - 2b, 2a + b]$ and $v_2 = a_1 - a_3 = [2a - b, a + 2b]$. Note that $v_1$ and $v_2$ depend only on the orientation of $\mathcal{A}$, as expected. Assume that $v$ is drawn from $b_4$ to $a_3$. This yields $v = a_3 - b_4 = [-a + b + x_0 - 1, -a - b + y_0 + 1]$. The inner products $v_1 \cdot n$, $v_2 \cdot n$, and $v \cdot n$ can easily be computed to form $H_1$, $H_2$, and $H_3$ as algebraic primitives.

## 3.4 Collision Detection

The previous section was concerned with constructing an explicit representation of the obstacle region, $\mathcal{C}_{obs}$, in C-space. For most problems, however, an explicit representation is not required. It is often preferable to simply build a logical predicate that serves as a probe that tests whether a configuration lies in $\mathcal{C}_{obs}$. This is referred to as *collision detection*, and it is expected in general that collision detection is much more efficient than constructing a complete representation of $\mathcal{C}_{obs}$. The tradeoff is that a motion strategy algorithm has better information with a complete characterization of $\mathcal{C}_{obs}$.

A variety of collision detection algorithms exist, ranging from theoretical algorithms that have excellent computational complexity to heuristic, practical algorithms whose performance is tailored to a particular application. The techniques from Section **??** can, of course, be used to develop a collision detection algorithm by defining a logical predicate using the geometric model of $\mathcal{C}_{obs}$. For the case of a 2D world, with a convex robot and obstacle, this leads to an $O(n + m)$ time collision detection algorithm.

### 3.4.1 Basic Concepts

*Collision detection* may be viewed abtractly as the construction of a logical predicate (boolean-valued function), $c : \mathcal{C} \rightarrow \{true, false\}$ which yields $c(q) = true$ if and only if $q \in \mathcal{C}_{obs}$. A collision detector may be viewed as a kind of probe that can indicate whether or not some $q \in \mathcal{C}$ is in collision.

In the boolean-valued function there is no notion of how far robot is from hitting obstacles. For many path planning algorithms, this additional information is quite useful. A *distance function* serves this purpose, and is defined as $d : \mathcal{C} \rightarrow [0, \infty)$, in which the real-value in the range of $f$ indicates the distance in the world, $\mathcal{W}$, between the closest pair of points over all pairs from $\mathcal{A}(q)$ and $\mathcal{O}$. In general, for two closed, bounded subsets, $E$,$F$, of $\mathbb{R}^n$, the *(Hausdorff) distance* is defined as

$$\rho(E, F) = \min_{e \in E} \min_{f \in F} \|e - f\|,$$

in which $\|E - F\|$ denotes the Euclidean distance between two points in $\mathbb{R}^n$. Clearly, if $E \cap F \neq \emptyset$, then $\rho(E, F) = 0$.

The methods described below may be used to either compute distances, or simply collision.

**Two-phase collision detection** If the robot, $\mathcal{A}$ is a single, rigid body, and the obstacle region, $\mathcal{O}$, is a single connected region, then a collision checking or distance computation algorithm is applied to $\mathcal{A}$ and $\mathcal{O}$. Suppose, however, that the robot is a collection of $m$ attached bodies, $\mathcal{A}_1$, $\mathcal{A}_2$, ..., $\mathcal{A}_m$, and that $\mathcal{O}$ is comprised of $k$ components. Two questions become important: 1) Is there a way to avoid checking each body against each component? 2) Do the robot bodies have to be checked against each other for collision?

For this complicated situation, collision detection can be viewing as a two-phase process. In the *broad phase*, the task is to avoid performing expensive computations for bodies that are far enough away from each other. In the *narrow phase*, individual pairs of bodies are each checked carefully for collision. In the broad phase, simple bounding boxes can be placed around each of the bodies, and the expensive collision checking can be performed only of the boxes overlap. Hashing schemes can be employed in some cases to greatly reduce the number of pairs of boxes that have to be tested for overlap [**?**].

For a robot that consists of multiple bodies, the pairs of bodies that should be considered for collision must be specified in advance. Recall from Section 3.3 that this is the set, $P$, of collision pairs. This information is utilized in the broad phase. For many robots, it might make sense to never consider checking certain pairs of bodies for collision. For example, consider a 2D chain of three rigid bodies, $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$, which are attached by revolute joints. If $\mathcal{A}_1$ and $\mathcal{A}_2$ are attached, they might always be in collision in some sense, but this is not useful for motion strategy problems. However, we might want to avoid the collision of the first link, $\mathcal{A}_1$, with a link that is not adjacent, such as $\mathcal{A}_3$.

Two appraoches to the narrow phase are described below, for a generic pair, $E$ and $F$, of bodies. Each body could be either a robot body or part of the obstacle region.

### 3.4.2 Hierarchical Methods

In this section, suppose that two complicated, nonconvex bodies are checked for collision. These bodies could be defined using any kind of geometric primitives, such as polyhedra or triangles in space. Hierarchical methods generally represent each body as a tree in which each nodes represent a simple shape that bounds all of the points in one portion of the body. The root node of the tree bounds the entire body.

There are generally two opposing criteria that guide the selection of a bounding volume:

1. The volume should fit the actual data as tightly as possible.

2. The intersection test for two volumes should be as efficient as possible.

Several popular choices are shown in Figure 3.2 for an L-shaped body.

The tree is constructed recursively as follows. For each node, consider the set, $X$, of all points that are contained in the bounding volume. Two child nodes are constructed by defining two smaller bounding volumes whose union covers $X$. The split is made so that the portion covered by each child is of similar size. If the geometric model consists of primitives such as triangles, then a split could be made separate the triangles into two sets of roughly equal size. A bounding volume is then computed for each of the children.

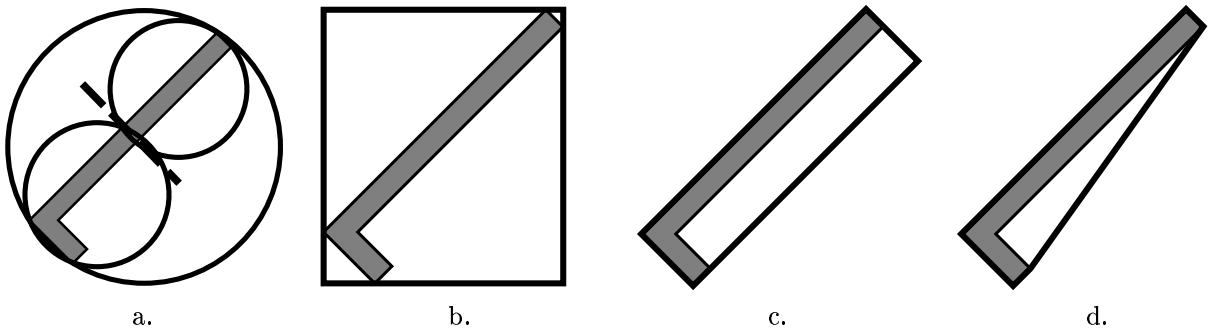



Figure 3.2: Four different kinds of bounding volumes: a) sphere, b) axis-aligned bounding box (AABB), c) oriented bounding box (OBB), d) convex hull. Each one usually provides a tighter approximation than the previous one, but is more expensive to test for overlapping pairs.

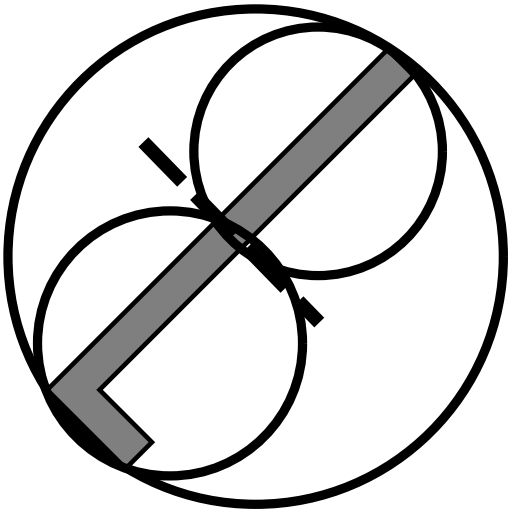


Figure 3.3: The large circle shows the bounding volume for a node that covers an L-shaped body. After performing a split along the dashed line, two smaller circles are used to cover the two halves of the body. Each circle corresponds to a child node.

Figure 3.3 shows an example of a split for the case of an L-shaped body. Children are generated recursively by making splits until geometric primitives are reached. For example, in the case of triangles in space, splits are made until a node is reached that bounds only a single triangle.

Consider the problem of determining whether bodies $E$ and $F$ are in collision. Suppose that a trees, $T_e$ and $T_f$, have been constructed for $E$ and $F$, respectively. If the bounding volumes of the root nodes of $T_e$ and $T_f$ do not intersect, then it is known that $T_e$ and $T_f$ are not in collision without performing any additional computation. If the volumes do intersect, then the bounding volumes of the children of $T_e$ are compared to the bounding volume of $T_f$. If either of these intersect, then the bounding volume of $T_f$ is replaced with the bounding volumes of its children, and the process continues recursively. As long as the bounding volumes overlap, lower levels of the trees will be traversed, until eventually the leaves are reached. If triangle primitives are used for the geoemtric models, then at the leaves, the algorithm will test the individual triangles for collision, instead of bouding volumes. Note that as the trees are traversed, if a bounding volume from the node, $n_1$, of $T_e$ does not intersect the bounding volume from a node, $n_2$, of $T_f$, then no children of $n_1$ have to be compared to children of $n_1$. This can generally result in dramamtic reduction in comparison to the amount of comparisons needed in a naive approach that, for example, tests all pairs of triangles for intersection.

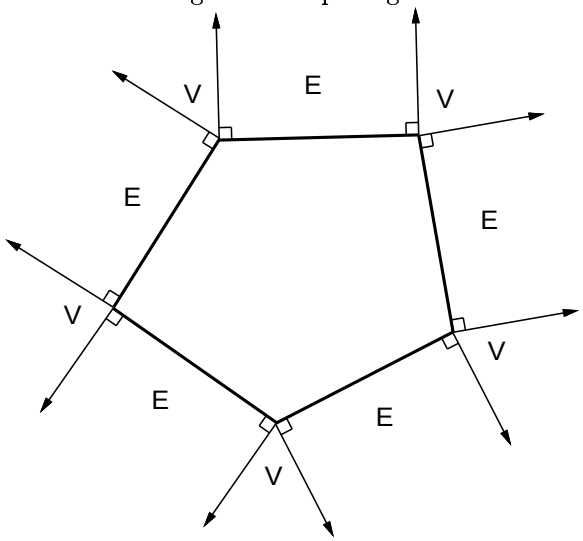
It is possible to extend the hierarchical collision detection scheme to also compute distance. If at any time, a pair of bounding volumes have a distance greater then the smallest distance computed so far, then their children do not have to be considered.

### 3.4.3 Incremental Methods

In this section we focus on a particular approach to collision detection that is based on *incremental distance computation*. The technique assumes that each time the collision detection algorithm is called, the bodies do not move very much. Under this assumption, the algorithm achieves "almost constant time" performance for the case of convex polyhedral bodies.

We first define *Voronoi regions* of a convex polygon, in terms of *features*. Each edge and each vertex are considered as features. A polygon with $n$ edges has $2n$ features. Any point that is outside of the polygon has a closest feature on the polygon in terms of Euclidean distance. In some instances, the closest feature is an edge; in the remaining instances, the closest feature is a vertex. For a given feature, $g$, the set of all points from which $g$ is the closest feature is known as the *Voronoi region* of $g$, denoted $Vor(g)$.

The figure below shows all ten Voronoi regions for a pentagon.




The Voronoi regions of vertices are labeled with a "V", and the Voronoi regions of edges are labeled with an "E". Note that the Voronoi regions alternate between "V" and "E" (no two Voronoi regions of the same kind are adjacent). The adjacencies between these Voronoi regions follow the same pattern as the adjacencies between vertices and edges in the polygon (a vertex is always between two edges, etc.).

For any two convex polygons that do not intersect, the closest point will be determined by a pair of points, one on each polygon (usually the points are unique, but this is not true in general). Consider the feature for each point in this pair. There are only three possible combinations:

- **Edge-Edge** Each point of the closest pair each lies on an edge. In this case, the edges must be parallel.
- **Edge-Vertex** One point of the closest pair lies on an edge, and the other lies on a vertex.
- **Vertex-Vertex** Each point of the closest pair is a vertex of a polygon.

Let $g_e$ and $g_f$ represent any feature pair of $E$ and $F$. Let $(x_e, y_e) \in g_e$ and $(x_f, y_f) \in g_f$ denote the closest pair of points, among all points in $g_e$ and $g_f$. The following condition can be used to determine whether the distance between $(x_e, y_e)$ and $(x_f, y_f)$ is the distance between $E$ and $F$:

$$(x_f, y_f) \in Vor(g_e) \text{ and } (a_e, y_e) \in Vor(g_f)$$

If this condition is satisfied for a given feature pair, then the distance between $E$ and $F$ equal to the distance between $g_e$ and $g_f$. This imples that the distance between $E$ and $F$ can be determined in constant time. The assumption that $E$ moves only a small amount is made to increase the likelihood that the closest

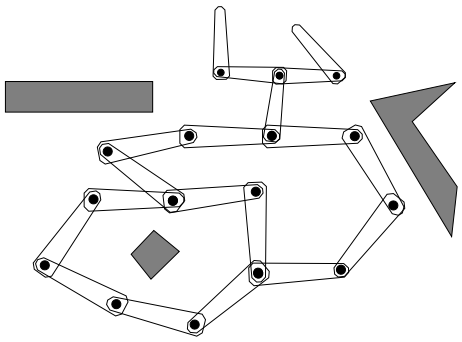


Figure 3.4: The set of all configurations for a linkages that maintains a closed kinematic chain is generally not a manifold. It can, however, be formulated as an algebraic variety which can be stratified into a collection of manifolds.

feature pair will remain the same. This is why the phrase "almost constant time" is used to describe the performance of the algorithm. Of course, it is possible that the closest feature pair will change. In this case, neighboring features can be tested using the condition above, until the new closest pair of features is found.

The same ideas can be applied for the case in which the bodies are convex polyhedra. The primary difference is that three kinds of features are considered: faces, edges, and vertices. The cases become more complicated, but the idea is the same. Once again, the condition regarding mutual Voronoi regions holds, and the algorithm has nearly constant time performance.

## 3.5 Kinematic Closure and Varieties

Our problem will be defined in a bounded 2D or 3D world, $\mathcal{W} \subset \mathbb{R}^N$, such that $N = 2$ or $N = 3$. Let a *link*, $L_i$, be a rigid body in the world, which is considered as a closed, bounded point set. Let $\mathcal{L}$ be a finite collection of $n_l$ links, $\{L_1, L_2, \ldots, L_{n_l}\}$. Let $J(L_i, L_j)$ (or simply $J$) define a *joint*, which indicates that the pair of links $L_i$ and $L_j$ are attached. Additional information is associated with a joint: the point of attachment for $L_i$, the point of attachment for $L_j$, the type of joint (revolute, spherical, etc.), and the range of allowable motions. Let $\mathcal{J}$ be a collection of $n_j$ joints that connect various links in $\mathcal{L}$. Let $\mathcal{M} = (\mathcal{L}, \mathcal{J})$ define a *structure*. An example is shown in Figure 3.4. It will be sometimes convenient to consider $\mathcal{M}$ as a graph in which the joints correspond to vertices and the links correspond to edges. Therefore, let $G_M$ denote the underlying graph of $\mathcal{M}$.

Now consider the kinematics of $\mathcal{M}$. Using a standard parameterization technique, such as the Denavit-Hartenburg representation [2, 3], the *configuration* of $\mathcal{M}$ can be expressed as a vector, $q$, of real-valued parameters. Let $\mathcal{C}$ denote the configuration space or $C$-space. Let $\mathcal{M}(q)$ denote the transformation of $\mathcal{M}$ to the configuration given by $q$. Note that the graph $G_M$ is a tree if and only if there are no closed kinematic chains. In $G_M$ is a tree, then any configuration $q$ yields an acceptable position and orientation for each of the links if we ignore collision issues. In this case, path planning can be performed directly in the $C$-space.

We are primarily concerned with the case in which $\mathcal{M}$ contains closed kinematic chains, which implies that $G_M$ contains cycles. In this case, there will generally exist configurations that do not satisfy closure constraints of the form $f(q) = 0$. Constraints can be defined by breaking each cycle in $G_M$ at a vertex, v, and writing the kinematic equation that forces the pose of the corresponding joint to be the same, regardless of which of the two paths were chosen to $v$. Let $\mathcal{F}$ represent the set, $\{f_1(q) = 0, f_2(q) = 0, \ldots, f_m(q) = 0\}$ of $m$ closure constraints. In general, if $n$ is the dimension of $\mathcal{C}$, then $m < n$. Let $\mathcal{C}_{cons} \subset \mathcal{C}$ denote the set of all configurations that satisfy the constraints in $\mathcal{F}$.

In addition to the usual complications of path planning for articulated bodies with many degrees of freedom, we are faced with the challenge of keeping the configuration in $\mathcal{C}_{cons}$. Although $\mathcal{C}$ is usually a manifold, $\mathcal{C}_{cons}$ will be more complicated. It can be expressed as an algebraic variety (the zero set of a system of polynomial equations), and it is assumed that a parameterization of $\mathcal{C}_{cons}$ is not available.

We next describe how a collision is defined for $\mathcal{M}(q)$. Let $\mathcal{O} \subset \mathcal{W}$ denote a static *obstacle region* in

the world. The structure $\mathcal{M}(q)$ is in *obstacle-collision* if $L_i(q) \cup \mathcal{O} \neq \emptyset$ for some $L_i \in \mathcal{L}$ ($L_i(q)$ denotes the transformed link $L_i$ when $\mathcal{M}$ is at configuration $q$). Let $int(X)$ denote the open interior of a set $X$. The structure, $\mathcal{M}$, is in *self-collision* if $int(L_i) \cap int(L_j) \neq \emptyset$ for some $L_i, L_j \in \mathcal{L}$. We allow the boundaries of links to intersect without causing collision because a pair of links can "touch" at a joint. Let $\mathcal{M}$ be in *collision* if it is in obstacle-collision or self-collision. Using standard terminology, let $\mathcal{C}_{free}$ denote the set of all configurations such that $\mathcal{M}(q)$ is not in collision.

The task can now be defined as follows. Let $\mathcal{C}_{sat} = \mathcal{C}_{cons} \cap \mathcal{C}_{free}$, which defines the set of configurations that satisfy both kinematic and collision constraints. Let $q_{init} \in \mathcal{C}_{sat}$ and $q_{goal} \in \mathcal{C}_{sat}$ be the *initial configuration* and *goal configuration*, respectively. The task is to find a continuous path $\tau : [0,1] \rightarrow \mathcal{C}_{sat}$ such that $\tau(0) = q_{init}$ and $\tau(1) = q_{goal}$.

## Literature

Hierarchical collision detection is covered in [**?**, 5, **?**]. The incremental collision detection ideas are borrowed from the Lin-Canny algorithm [4] and V-Clip [6].

A good book on basic topology is [1].

## Exercises

1. Consider the set $X = \{1, 2, 3, 4, 5\}$. Let $X$, $\emptyset$, $\{1, 3\}$, $\{1, 2\}$, $\{2, 3\}$, $\{1\}$, $\{2\}$, and $\{3\}$ be the collection of all subsets of $X$ that are designated as *open sets*. Is $X$ a topological space? Is it a topological space if $\{1, 2, 3\}$ is added to the collection of open sets? Explain. What are the closed sets (assuming $\{1, 2, 3\}$ is included as an open set)? Are any subsets of $X$ neither open nor closed?

2. What is the dimension of the configuration space for a cylindrical rod that can translate and rotate in $\mathbb{R}^3$? If the rod is rotated about its central axis, it is assumed that the rod's position and orientation is not changed in any detectable way. Express the configuration space of the rod in terms of a Cartesian product of simpler spaces (such as $S^1$, $S^2$, $\mathbb{R}^n$, $P^2$, etc.). What is your reasoning?

3. Let $\tau_1 : [0,1] \rightarrow \mathbb{R}^2$ be a loop path in the plane, defined as follows: $\tau_1(s) = (\cos(2\pi s), \sin(2\pi s))$. This path traverses a unit circle. Let $\tau_2 : [0,1] \rightarrow \mathbb{R}^2$ be another loop path, defined as follows: $\tau_1(s) = (-2 + 3\cos(2\pi s), \frac{1}{2}\sin(2\pi s))$. This path traverses an ellipse that is centered at $(-2, 0)$. Show that $\tau_1$ and $\tau_2$ are homotopic (by constructing a continuous function with an additional parameter that "morphs" $\tau_1$ into $\tau_2$).

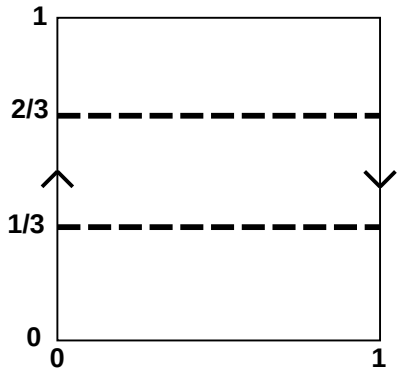
4. a) Define a unit quaternion, $h_1$, that expresses a rotation of $-\frac{\pi}{2}$ (-90 degrees) around the axis given by the vector $[\frac{1}{\sqrt{3}} \ \frac{1}{\sqrt{3}} \ \frac{1}{\sqrt{3}}]$.

   b) Define a unit quaternion, $h_2$, that expresses a rotation of $\pi$ around the axis given by the vector $[0 \ 1 \ 0]$.

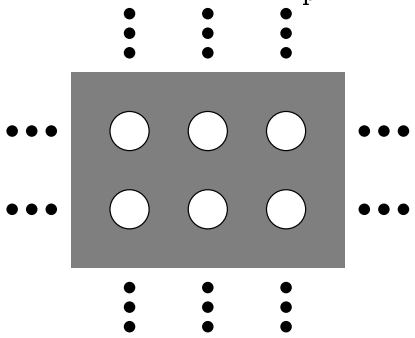
   c) Suppose the rotation represented by $h_1$ is performed, followed by the rotation represented by $h_2$. This combination of rotations can be represented as a single rotation around an axis given by a vector. Find this axis and the angle of rotation about this axis. Please convert the trig functions whenever possible (for example $\sin\frac{\pi}{6} = \frac{1}{2}$, $\sin\frac{\pi}{4} = \frac{1}{\sqrt{2}}$, and $\sin\frac{\pi}{3} = \frac{\sqrt{3}}{2}$).

5. Suppose there are five polyhedral bodies that can float freely in a 3D world. They are each capable of rotating and translating. If these are treated as "one" composite robot, what would be the topology of the resulting configuration space (assume that the bodies are NOT attached to each other)? What is its dimension?

6. The figure below shows the Möbius band defined by identification of sides of the unit square. Imagine that scissors are used to cut the band along the two dashed lines. Describe the resulting topological space. Is it a manifold? (Hint: You might want to play with papers and scissors.)




7. Consider the set of points in $\mathbb{R}^2$ that are remaining after a closed disk of radius $\frac{1}{4}$ with center $(x, y)$ is removed for every value of $(x, y)$ such that $x$ and $y$ are both integers. This forms an open set that looks like "infinite swiss cheese." Is this a manifold? Explain.



## Projects

# Bibliography


[1] M. A. Armstrong. *Basic Topology.* Springer-Verlag, New York, NY, 1983.

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

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

[4] M. C. Lin and J. F. Canny. Efficient algorithms for incremental distance computation. In *IEEE Int. Conf. Robot. & Autom.*, 1991.

[5] M. C. Lin, D. Manocha, J. Cohen, and S. Gottschalk. Collision detection: Algorithms and applications. In J.-P. Laumond and M. Overmars, editors, *Algorithms for Robotic Motion and Manipulation*, pages 129–142. A K Peters, Wellesley, MA, 1997.

[6] B. Mirtich. V-Clip: Fast and robust polyhedral collision detection. Technical Report TR97-05, Mitsubishi Electronics Research Laboratory, 1997.