Kevin D. Wise
Adrian Bowyer

Department of Mechanical Engineering
Faculty of Engineering and Design
University of Bath
Bath, BA2 7AY, UK
a.bowyer@bath.ac.uk

A Survey of Global Configuration-Space Mapping Techniques for a Single Robot in a Static Environment

Abstract

The mapping from workspace to configuration space (C-space) plays a major role in the field of kinematics, with applications including robotics path planning, packing and nesting, automated assembly, and mechanism analysis. Over the past 20 years, research into the problem has resulted in many techniques that can be combined to suit a specific application. This survey aims to provide the developer of a C-space-based system with an overview of those techniques that map the global C-space of a single robot in a static environment. We discuss issues concerning how the robot and its environment are modeled (including how approximations can be used to make C-space mapping easier) and describe a range of schemes used to represent a C-space map. We then discuss the key techniques used to generate a C-space map for mobile robots and manipulators. The survey of literature is summarized by tables that list some 50 individual mapmaking papers, classifying each according to criteria identified in earlier sections. Finally, we draw conclusions from the findings of the survey. Note that, although reference is made throughout to robots, the controlled objects may equivalently be components or assemblies. In particular, results for mobile robots are fundamental to all C-space mapmaking problems.

KEY WORDS—configuration-space map, path planning, survey, kinematics, mechanism analysis

1. Introduction

1.1. The Configuration Space Approach to Spatial Planning

No concept has played a greater role in the study of rigid-body kinematics than the *configuration space* (or *C-space*) approach to spatial planning, as formally defined by Lozano-Pérez (1983). A configuration of an object or group of objects is a set of parameters sufficient to position every point in the object(s) in space. Thus, for a rigid body with 6 degrees of freedom, a configuration is a 6-tuple, which specifies the position and orientation of the body relative to (arbitrary) reference values. Similarly, the configuration of a system of rigid bodies with a total of n degrees of freedom may be defined by an n-tuple. The configuration space for a system of rigid objects is the space of all its possible configurations and is therefore an n-dimensional space bounded by upper and lower limits on each of the degrees of freedom. Since two solid objects cannot overlap, configurations of such a system can fall into three categories: *prohibited* (or *forbidden*) configurations, in which objects would overlap; *safe* (or *free*) configurations, in which no overlap occurs; and *contact* configurations, in which two or more objects touch each other in one or more places. The *C-space map* for a particular problem is a classification of every configuration of the system as one of those three categories. The prohibited regions in such a map are commonly referred to as *C-space obstacles*, the safe regions as *free space*,[1] and the boundary between the two as the *contact surface*.[2]

The relationship between workspace and C-space is illustrated in Figure 1, which considers the simple case of a two-dimensional translating object. Figure 2 shows an example of C-space mapping for a manipulator, after Siméon (1988).

Generating a C-space map—that is, transforming a problem from the *workspace*[3] (or *problem-space*) into the configuration space—is a far from trivial task for all but the most elementary problems. Indeed, in the worst case, the amount of storage needed for a C-space map may increase exponentially

The International Journal of Robotics Research
Vol. 19, No. 8, August 2000, pp. 762-779,

1. Confusingly, this term is also sometimes used to refer to the complement of the obstacles in the problem space.
2. For some problems, the contact surface may also exhibit whiskers, cracks, and other nonmanifold geometry in addition to that boundary (see Latombe 1993).
3. In C-space literature, this term is commonly used to mean the two- or three-dimensional space in which the robot exists, although elsewhere it sometimes refers to the subset of that space which is reachable by the end effector of a manipulator.

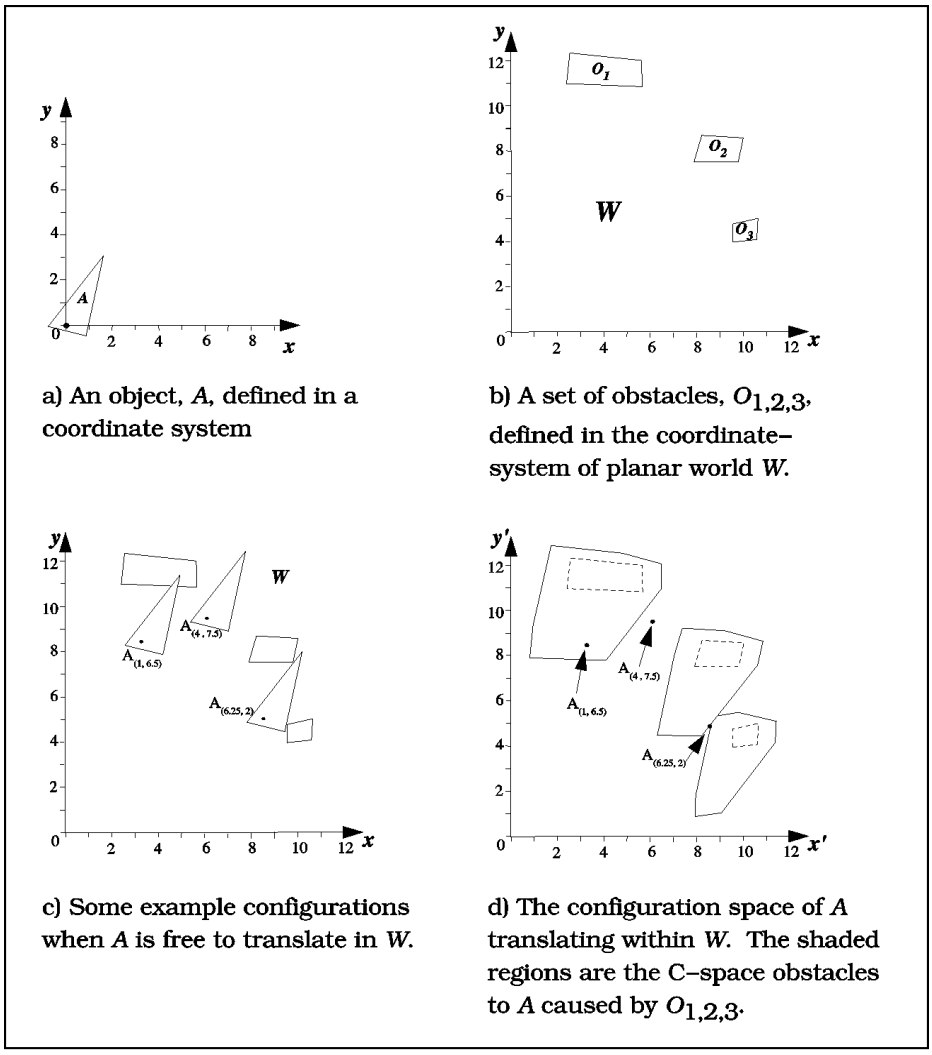
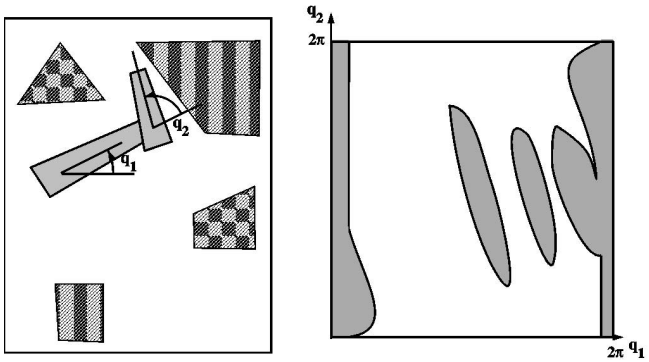


Fig. 1. An illustration of C-space mapping.

with the number of degrees of freedom (for an in-depth study of the complexity of C-space mapping and related issues, see, for example, Reif 1979; Hopcroft, Schwartz, and Sharir 1984; Canny 1988; Cesati and Wareham 1995). A second disadvantage of the C-space approach is that it relies on an a priori knowledge of the geometry of the objects involved. However, there is a strong motivation for performing the C-space mapping step whenever possible: a continuous path of the system of rigid objects through the 2- or 3-dimensional world corresponds to a path for a 0-dimensional point through the C-space map. This means that the need for 2- or 3-dimensional collision detection has been removed and the path-planning problem is reduced to finding a line that connects the initial and goal configurations without entering a prohibited region.

For this reason, C-space maps are most commonly used in robotics path planning where optimal global paths are achieved by arranging a decomposition of the free space and/or contact surface into a graph and then applying a graph-search algorithm such as the A* (Hart, Nilsson, and Raphael 1968). This method is not restricted to the case of a single robot in a static environment: C-space techniques have been developed to model both mobile robots and manipulators in environments that change predictably (for example, Fujimura 1993) and systems where multiple robots share an environment (for example, Parsons and Canny 1990). Beyond robotics, C-space maps have also been applied successfully in the areas of packing and nesting (for example, Adamowicz and Albano 1976), automated assembly planning (for example, Schweikard and Wilson 1995), and mechanism analysis. In the latter area, researchers have exploited the effectiveness of C-space maps to capture the kinematic constraints imposed on a set of rigid objects by their geometry and to describe every change of contact between the components of a mechanism. C-space maps have thus been



(a) A polygonal model of a 2–DOF manipulator in an environment with obstacles.

(b) The obstacles in the 2–D configuration space.

Fig. 2. An illustration of C-space mapping for a manipulator, after Siméon (1988).

seized upon as a key link between shape and behavior and have been used successfully for reasoning about kinematic behavior (for example, Faltings 1987), kinematic simulation (for example, Sacks and Joskowicz 1993), design classification and retrieval (for example, Murakami and Gossard 1992), and automated innovative design (for example, Subramanian and Wang 1995).

1.2. Motivation for and Scope of This Survey

Over the past 20 years, an enormous amount of research has gone into the mapping from workspace to C-space, resulting in a raft of techniques that can be combined to suit specific applications. Some of these are described in textbooks on robotics (for example, Latombe 1993) and in surveys on the wide field of motion planning (for example, Hwang and Ahuja 1992; Hwang 1995). However, we are not aware of a single survey specializing on C-space mapping, so there does not appear to be a comprehensive examination of C-space techniques available to those interested in using a C-space method for one of the applications mentioned above.[4]

This paper provides an overview of the techniques developed to date to map the *global* C-space of a single robot in a static environment. Papers concerned with the *use* of C-space maps (for example, Cheng and Cheng 1996) are not mentioned hereafter, and likewise neither are papers that map either a *small subset* (such as a *roadmap*, Canny 1988) of the C-space (for example, Kavraki, Kolountzakis, and Latombe 1996) or a space other than the configuration space (for example, Schwartz and Sharir 1983). For an overview of techniques *and applications* for C-space mapping for more than one moving object, see another survey of ours (Wise and Bowyer 1998). For a treatise on the mathematical structure of C-space (including topological and differential properties) and a detailed discussion of algebraic and geometric properties relating to the mapping of C-space obstacles, see Latombe (1993).

The rest of the paper is organized as follows: Section 2 examines issues concerning how the robot and its environment are modeled, including a look at how approximations can be used to ease the C-space mapping, while Section 3 describes the range of schemes used to represent a C-space map. Sections 4 and 5 discuss the key techniques used to generate a C-space map for mobile robots and manipulators, respectively, focusing on the range of techniques rather than on individual papers. In contrast, Section 6 contains tables that list some 50 mapmaking papers, classifying each according to the criteria identified in Sections 2 and 3. Finally, Section 7 draws some conclusions from the findings of the survey.

Note that although reference is made throughout to robots, the controlled object may equivalently be a component or an assembly.

2. Modeling the Robot and Its Environment

2.1. Geometry of the Objects Involved

Three key aspects of the geometry of the robot and its environment affect the difficulty of the C-space mapping:

Dimensionality. Clearly, three-dimensional problems are more difficult than two-dimensional problems. In particular, new types of contact between objects are introduced—for example, edge-edge contacts must be considered between polyhedra while between polygons such contacts are singularities that do not need attention. Also, increasing the dimensionality of the problem affects the potential dimensionality of the C-space, but the latter depends far more on the mechanics of the problem: a planar robot arm with 10 joints has a 10-D C-space, while an arm in 3-D workspace with 3 joints has a 3-D C-space.

Convexity. Nonconvex objects tend to be much more difficult to handle than convex objects.[5] Analytical representation of the contact surface is harder to formulate because contact between two nonconvex objects may occur simultaneously at multiple discrete points. Also, distance computations (commonly used by divide-and-classify approaches) are far less complicated between convex objects. Now, the general problem of decomposing an arbitrary solid object into convex parts is very difficult (see Bajaj and Kim 1988; Requicha and Voelcker 1983) but is possible for polyhedra and other practical cases. When this can be done, the C-space obstacles can be

4. As a result of the specialization and later date of this survey, only about a quarter of the papers we describe are covered by the surveys mentioned above.

5. As will be seen in Section 4.3, there are exceptions—some grid-based algorithms are independent of the shapes involved.

generated piecewise by exploiting the fact that C-space mapping is distributive over set union—using the notation $CSO_X(Y)$ to mean the C-space obstacle caused to object X by obstacle Y,

$$CSO_C(A \cup B) \equiv CSO_C(A) \cup CSO_C(B).$$

It is also worth noting that if two objects are convex, then the C-space obstacle that one causes to the other will itself be convex.

Surface algebraic degree. Like introducing nonconvexity, increasing the algebraic degree of the surfaces of the input models increases the difficulty of the C-space mapping for both analytical methods and most divide-and-classify approaches. The result, as will be seen in Section 6, is that the majority of mapmaking systems restrict their interest to polygonal or polyhedral input models. For most robotics applications, this is practical since objects should not get close enough for their precise geometry to be important, but this is not true in other applications such as mechanism analysis.

As a compromise between polytopes and arbitrary geometry, some mapmakers handle *generalized polytopes*, which, in addition to flat surfaces, contain simple curved shapes—circular arcs in two dimensions and parts of spheres or cylinders in three.

2.2. Representation Schemes

By far the most common representation scheme for the robot and its environment is the *boundary representation* (B-rep), whereby the surfaces of the objects are represented by lines or patches, which may be parametric (for a description of this and other solid-modeling representations, see, for example, Woodwark 1986). Such a representation can be used as input for boundary representation, divide-and-classify, or hybrid algorithms (see Section 3).

The other common representation scheme is discretization into axially aligned rectangloid cells (pixels in two dimensions or voxels in three), which may employ a tree-structure such as the binary- or 2^n-tree. This has the disadvantage that it limits the mapmaker to discrete methods but has the advantage that the objects' representations can sometimes be obtained in real-time directly from sensor data.

Our system (Wise and Bowyer 1996) is the only surveyed mapmaker to exploit the constructive solid geometry (CSG) representation, even though it has the strength that its structure is notationally independent of the dimensionality, and recent research (Wise 1998) demonstrates that it is suitable for exact as well as divide-and-classify techniques (Wise and Bowyer 1996).

Divide-and-classify mapmakers commonly employ a hierarchical approximation method that could be implemented using any of the above methods. Objects are represented by a tree in which each level is a successively simple and more conservative approximation of the real shape (as illustrated in Fig. 3); tests for intersection can then begin by doing fast tests on the simple representations and only progress up to the slower more accurate ones if necessary. Lozano-Pérez (1983) credits Marr and Nishihara (1977) with this idea.

2.3. Introduction of Approximation

Performing the C-space mapping precisely is so difficult that a key question is whether to ease the burden by introducing approximation at the object-modeling or C-space mapping stages. The classification tables in Section 6 use the following definitions for the four possible strategies:

Precise object models, precise C-space map; e.g., Bajaj and Kim (1990).

Precise object models, approximate C-space map; e.g., Bellier et al. (1992).

Approximate object models, precise C-space map; either analytical representations of the C-space obstacles are obtained (e. g., Avnaim, Boissonnat, and Faverjon 1988) or discrete methods are used without significant approximations (e.g., Kavraki 1993).

Approximate object models, Approximate C-space map; e.g., Lozano-Pérez and Wesley (1979) and Tso and Liu (1993).

3. C-Space Map Representation

Central to any C-space mapping system is the method it uses to represent the map. The various choices of representation scheme can be arranged in a taxonomy as shown in Figure 4. As illustrated, the methods—any of which can be used to decompose the C-space into a set of connected regions that can subsequently be arranged in a graph—can be split

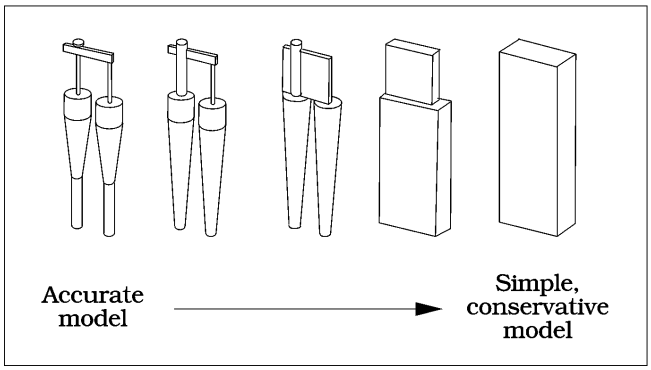


Fig. 3. An illustration of hierarchical approximation, after Faverjon and Tournassoud (1988).

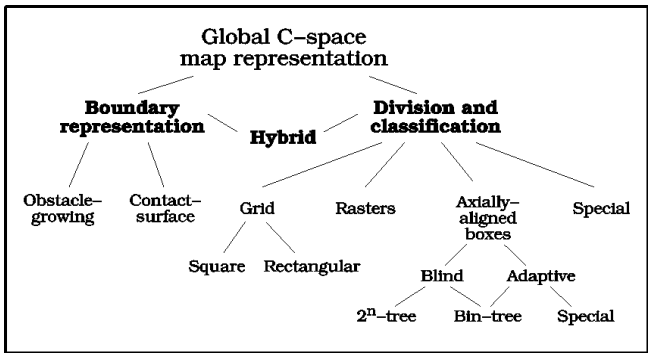


Fig. 4. A taxonomy of C-space representation schemes.

into three main subgroups: *boundary representation* methods, *division-and-classification* methods, and *hybrid* methods. Every scheme has strengths and weaknesses, and the choice of scheme will depend on a number of application-specific factors such as the available data, the number and types of degrees of freedom, the relative importance of speed or accuracy, and the way in which the final map will be used.

3.1. Boundary Representation

3.1.1. Obstacle-Growing

In his seminal 1983 paper, Lozano-Pérez demonstrated how, for a mobile robot translating without rotation, the C-space obstacle boundary could be generated by growing the problem-space obstacles using Minkowski point-set operations. Minkowski sum, monadic minus, and difference are defined on sets of points (equivalently vectors) in $\Re^n$, A, B as follows:

$$A \oplus B = \{a + b | a \in A, b \in B\}$$

$$\ominus B = \{-b | b \in B\}$$

$$A \ominus B = \{a - b | a \in A, b \in B\} \equiv A \oplus (\ominus B).$$

These operations are illustrated in Figure 5.

Lozano-Pérez observed that if B is a mobile robot that translates and is obstructed by obstacle A, the C-space obstacle to B caused by A is given by $A \ominus B$. Thus, a C-space obstacle for a translating robot can be generated by reflecting the robot in the origin and swelling the obstacles by the result. This can be achieved for convex polytopes by reflecting each robot vertex in the origin and adding the result to each vertex of the obstacle; the C-space obstacle is the convex hull of the resulting point set.

On the subject of an n-dimensional convex polytope, Gouzènes (1984, 59) observed the following:

- It is bounded by at least $n+1$ hyperplanes, each defined by $n + 1$ coefficients.

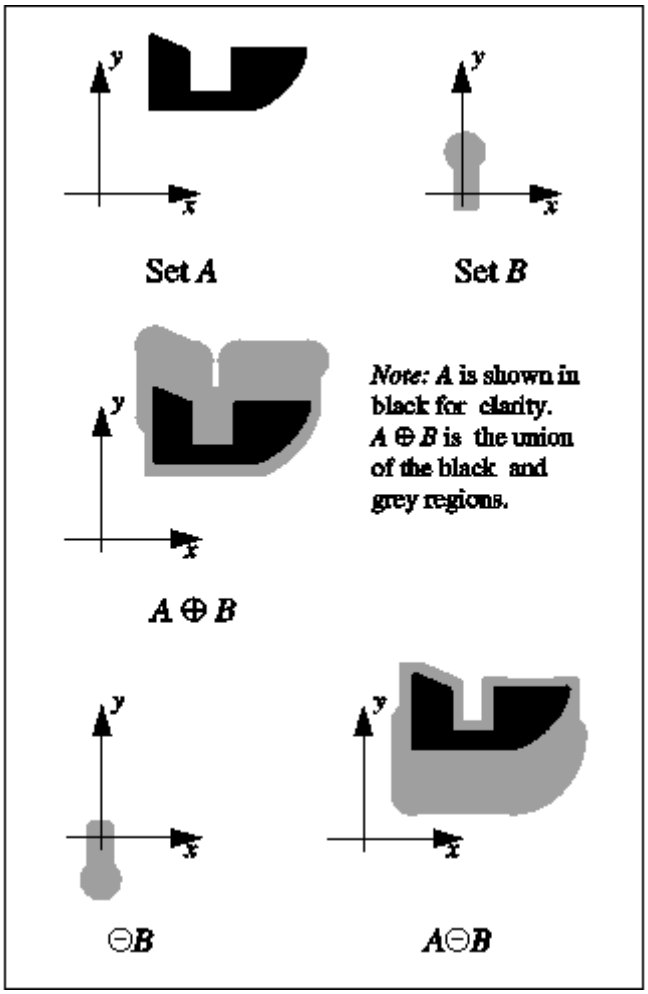


Fig. 5. Minkowski operations.

- Inclusion requires an operation that is $O(n^3)$.
- The generalization of 3-D B-rep modeling (the faces-edges-vertices structure) involves $O(n!)$ pointers for a simple convex object.

Gouzenès also points out that a quadratic boundary representation of an n-dimensional C-space obstacle requires $O(n^2)$ coefficients per hypersurface and does not easily support the operations needed for collision avoidance: point membership testing, inclusion, and intersection.

3.1.2. Contact-Surface

Obstacle growing by Minkowski difference works for both two- and three-dimensional cases and can be implemented for objects of arbitrary shape, but it does not extend intuitively to rotations. However, a boundary representation of the C-space obstacles is still achievable since the boundary of any C-space obstacle is a patchwork of surfaces, each of which corresponds to a *contact condition* (Brost 1989)—that is, a

specific interaction between an element of the moving object and an element of an obstacle. For example, in the case of a polygonal robot translating and rotating amid polygonal obstacles, the contact-surface will be a patchwork of two types of robot-obstacle contact—edge-vertex and vertex-edge; the corresponding polyhedral case gives rise to a five-dimensional surface (embedded in the six-dimensional C-space), which is a patchwork of contact-conditions of three types—vertex-face, face-vertex, and edge-edge. Contact conditions are illustrated for a polygonal case in Figure 6.

A common method for obtaining a boundary representation of a C-space obstacle is to calculate the contact-surface for *every possible* contact condition and patch the results together. However, unless the objects are convex, some contact-conditions never actually occur so the patches generated for them lie redundant, contained within the others. As a result, systems that use that approach either restrict themselves to convex objects (for example, Bajaj and Kim 1990) or have a stage that sorts the useful surfaces from the redundant ones (for example, Faltings 1987; Brost 1989), which can be computationally expensive.

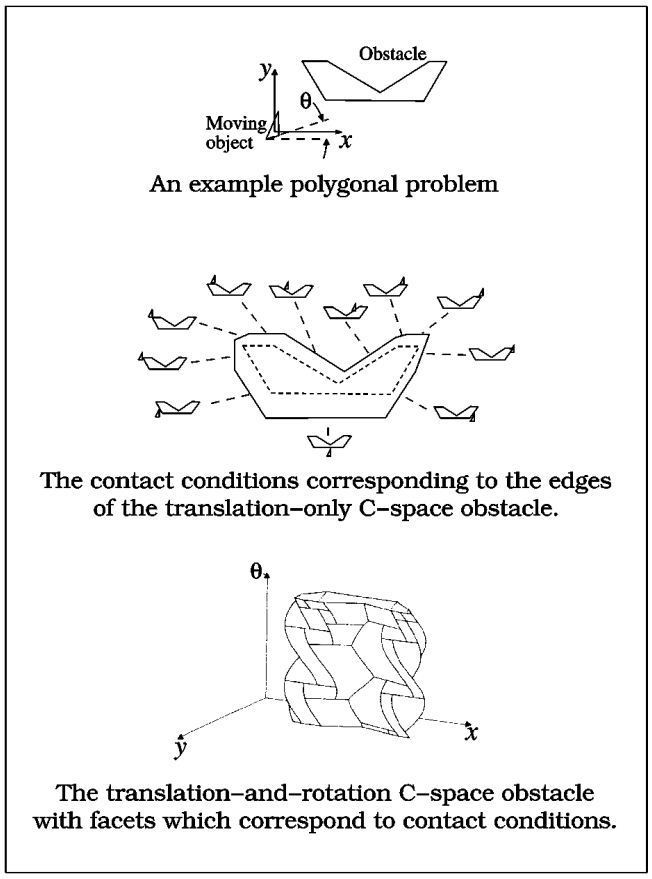


Fig. 6. Illustration of contact conditions for a polygonal case after Brost (1989).

3.2. Division and Classification

An alternative to boundary representation is to discretize the C-space into a number of cells and then to use some test to classify each one as safe, prohibited, or contact.

3.2.1. Division Strategies

A number of different division strategies have been employed:

A grid. The grid cells are often approximated by a configuration point (typically the centroid) during the cell classification stage. The resolution may be equal in each degree of freedom (an *isotropic grid* in our taxonomy of Fig. 4) or not—for example, the rotational freedom of the base link of a revolute manipulator may be divided more finely than those higher up.

Grid-based mapmaking systems tend to handle more general geometry than other systems. They are also often good for parallelization.

Clearly, the drawback to grids is their exponential growth in the memory requirement with the number of dimensions; this currently limits them to three degrees of freedom.

Rasters. A one-dimensional C-space map may be represented by a ray that is divided into safe and prohibited segments. A two-dimensional C-space map can be approximated by a stack of *rasters*—rays with a finite thickness—divided in the same way. A three-dimensional C-space map can be built by stacking resolution-thick 2-D maps, and so on. This was demonstrated by Lozano-Pérez (1987); it increases the smallest cell from being an approximation to a (0-D) point to being an approximation to a (1-D) line. Rasters are especially appropriate for revolute manipulators (see Section 5.3). Figure 2 shows an example by Red and Truong-Cao (1985) of this.

Axially aligned boxes. An n-dimensional configuration space can be recursively chopped into axially aligned boxes. In *blind* division (Zhu and Latombe 1991), this works like a quad- or oct-tree. *Adaptive* schemes decide the position of the chopping hyperplanes using the box's contents. This will tend to bound the C-space obstacles tighter with fewer cells—but the system must work out where to chop. A bin-tree, which divides into two at each recursion regardless of dimensionality, can be either blind or adaptive.

A tree of axially aligned boxes has the following characteristics:

Memory requirement. Memory requirement for a box tree may be orders of magnitude less than that of a grid of the same resolution.

Isotropy. All degrees of freedom are treated equally (cf. Lozano-Pérez 1981).

Known efficient algorithms. As Gouzènes observed (1984, 59), a box supports $O(n)$ algorithms for membership, inclusion, and intersection.

Refinability. Resolution can be increased locally by adding to (not recomputing) the existing representation (Paden, Mees, and Fisher 1989).

Regularity. A regular blind division has the advantage that a cell's neighbors can be visited without maintaining a connectivity graph (Faverjon and Tournassoud 1988, 100).

A clue to relative safety of cells. A motion that remains far from the obstacles can be obtained quickly by searching only large cells (Faverjon and Tournassoud 1988, 100).

A safety margin. A divided C-space is always a conservative approximation.

Special. Some schemes divide the C-space into cells that are not axially aligned. For example, see Lozano-Pérez (1981) or Shiller and Gwo (1993).

3.2.2. Cell Classification Schemes

Finding if a C-space cell is safe, prohibited, or a mixture of both can be done in several ways:

Swelling the robot. A fast method sometimes used as the first cell-classification step: the robot is swollen by the maximum distance any part of it can move (this is not always easy to determine—see Lozano-Pérez 1987). This swollen robot is then tested against the obstacles.

Swept volumes. The volume swept by the robot as it moves is calculated. This can be very effective in conjunction with raster-division (see Section 5.3) because, as observed by Gouzènes (1984), the shape of the swept volume of a link exercising its full range of motion is constant. Verwer (1990) uses a swept volume scheme that employs *bubble hierarchies* whereby volumes are approximated by increasingly accurate unions of spheres. Intersection testing between spheres is trivial.

Inverse kinematics. Warren, Danos, and Mooring (1989) and Adolphs and Nafziger (1990) divide C-space into a uniform grid and then map discrete configurations of an end-effector into that grid via inverse kinematics. Adolphs and Nafziger store the mapping from workspace to joint-space in a look-up table so that changes can be handled quickly.

Bounded Jacobians. Avoiding obstacles places a bound on the Jacobians of points on the robot—this can be used to classify cells (Faverjon and Tournassoud 1988; Paden, Mees, and Fisher 1989).

Contact conditions. Analysis of contact conditions can be used to classify a C-space cell (see Red and Truong-Cao 1985; Lozano-Pérez 1987).

Distance calculations. The distance-to-contact when the robot is in the cell's centroidal configuration (negative if it is prohibited) can be found. If the modulus of the result is greater than half of the length of a diagonal of the cell, then the cell can be classified as safe or prohibited by sign. The Minkowski difference operation can be used to do this (Siméon 1988).

Ralli and Hirzinger (1996) use a pre-mapmaking step in which the workspace is gridded. Wavefront propagation labels each cell with the distance from it to the nearest obstacle. A map of robot C-space is then plotted into a grid by using forward kinematics to position each of the links and then only using collision detection if the distance value of the midpoint of each link is less than half the link's length.

3.2.3. Other Useful Techniques

Several other methods have been developed for use with divide-and-classify representations:

Classification propagation. If one cell enables nearby cells to be classified, this can be done. Siméon (1988) and Laumond et al. (1988) calculate the distance-to-contact for the centroid of a cell and hence classify it: this information is propagated to other cells within a ball with a radius of the returned distance.

Restructuring the division. Sometimes a postdivision merging of cells can be used. This is good if the original cells are long thin rasters (see Lozano-Pérez 1987; Siméon 1988; Faverjon and Tournassoud 1988). Smoother paths can be planned through a grid C-space map after Kohonen map–based reorganization (Ralli and Hirzinger 1997) has increased the resolution of the free space (Ralli and Hirzinger 1996, 1997).

Selected refinement. The C-space map is only refined where a path is likely to go. This was used by Lozano-Pérez (1981) and is fundamental to many later planners (Hasegawa and Terasaki 1988; Verwer 1990; Duelen and Willnow 1991; Bellier et al. 1992; Chen and Hwang 1992).

C-space obstacle labeling. Siméon (1988) labels each obstacle cell with the obstacle(s) that caused it. Then the map does not need to be recomputed if an obstacle is removed—all C-space obstacle cells caused only by the removed obstacle are then safe.

3.3. Hybrid

A third and final approach to representing a global C-space map is to create a *hybrid* representation that combines elements of those described above. As an example of this, during the mapmaking process described by Lozano-Pérez (1987), obstacles are grown using a Minkowski sum, the C-space is recursively divided into a series of rasters, and cells are merged to result in an adaptively divided, axially aligned box representation.

4. Mapmaking Techniques for a Mobile Robot in a Static Environment

Dealing with a single moving object (such as a mobile robot) in a static environment is not only the most simple C-space problem, it is the most fundamental—techniques developed here provide the building blocks for all other C-space mapmakers.

4.1. Founding Work

One of the characteristics of the C-space approach to spatial planning is that it isolates the kinematic constraints caused by the shapes of objects, leaving other constraints to be dealt with later. As a result, a C-space mapping algorithm may ignore the nonholonomic constraints imposed by the steering mechanism of a mobile robot and treat the robot as a free-floating body. Moreover, since most mobile robots are limited to moving on a flat floor and the shapes of the obstacles are approximated as having a constant horizontal cross section, mobile robot problems are commonly modeled as two-dimensional.

An unconstrained two-dimensional object has three degrees of freedom (two translational and one rotational), resulting in a C-space, which is the three-dimensional manifold $\Re^2 \times S^1$ (where S^1 is the unit circle) and which may be represented by the Cartesian space $\Re^2 \times \Re/(2\pi Z)$. A configuration is parameterized by (x', y', θ), where $(x', y') \in \Re^2$ and $\theta \in [0, 2\pi)$. An unconstrained three-dimensional object (which might be a robot in space or a component of a mechanism) has three translational and three rotational degrees of freedom, resulting in a six-dimensional C-space.

The Minkowski obstacle-growing technique implemented by Lozano-Pérez (1983) was able to handle both two- and three-dimensional polytopes, but it was unable to handle either rotations or more general geometry. Examining how these two limitations have been overcome provides a convenient way of examining C-space mapmaking techniques for mobile robots.

4.2. Handling Rotations

Early attempts to handle rotations (for example, Lozano-Pérez 1983; Jarvis 1983) worked with polygons and treated the rotational degree of freedom differently from the others. The range of rotational values was divided into slices, and for each slice a polygonal approximation to the area swept by the rotating polygon was calculated (called a *slice projection*); the obstacles were then grown by each slice projection. This resulted in sets of grown obstacles (called *θ-slices*), which were stacked together to approximate the three-dimensional C-space map for the problem. Figure 7 shows this. It has the drawback that the nonisotropic map only allows motions that alternate between translations and rotations.

The technique of recursive subdivision-and-classification of C-space was introduced by Brooks and Lozano-Pérez (1985), who describe a scheme that sits on top of a precise boundary representation calculated using contact conditions. Laumond et al. (1988) also demonstrate such a scheme but classify cells by calculating distances to contact. A third technique was developed by Zhu and Latombe (1991), which, like that in Brooks and Lozano-Pérez (1985), sits above an accurate C-space-obstacle representation. Zhu and Latombe

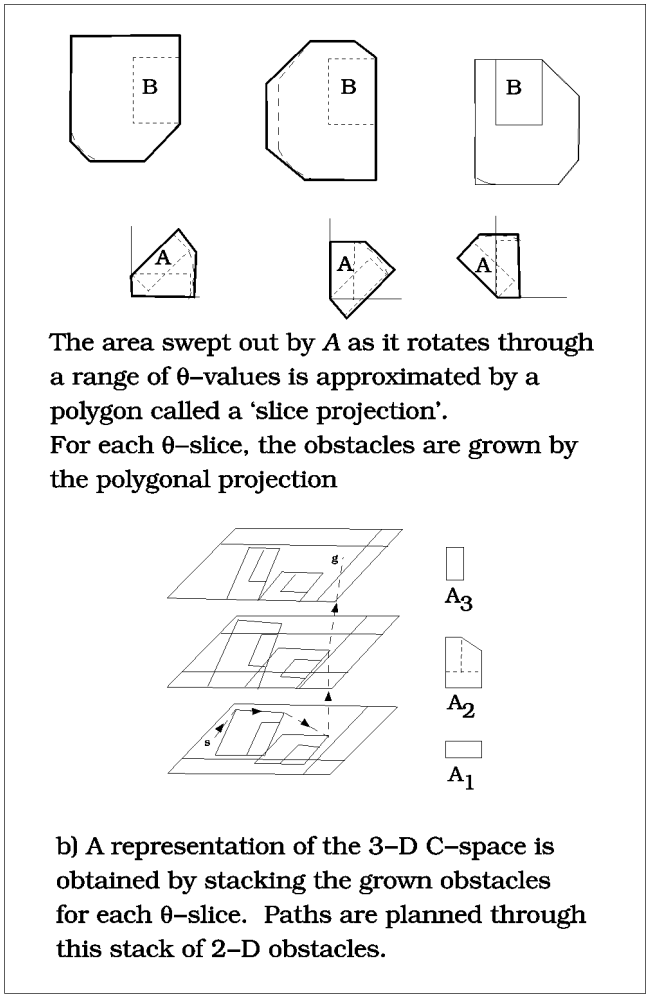


Fig. 7. Illustration of the slice-projection technique, after Lozano-Pérez (1983).

introduce *bounded* and *bounding* approximations of C-space obstacles (an approach taken from computer graphics) and show that they get a tight decomposition of the C-space obstacles faster than with quad-trees.

The majority of C-space mapping research has developed boundary representations of C-space obstacles using the contact condition mentioned in Section 3. Donald (1985, 1987), for example, increased the dimensionality of mappable C-space so that he could make a map for a three-dimensional polyhedral object with three translational and three rotational degrees of freedom. He did this by characterizing the five-dimensional surfaces that bound the C-space obstacles and by establishing operators that move along and between these surfaces. Meanwhile, Avnaim and Boissonnat (on their own, 1988; and with Faverjon, 1988) concentrated on a polygon translating and rotating, developing an efficient and exact path planner (based on the mapping of contact-condition patches) with which they obtained very impressive results. Brost (1989) did similar work and attached *contact information* to facets defining C-space obstacles—an important step toward kinematic analysis of physical interaction. Liu and Onda (1993) analyzed local contact conditions for interacting polygons and plotted the result into a grid over the global C-space.

In contrast to all these new analytical methods, Lengyel et al. (1990) attacked the 3-DoF polygon problem using exactly the same method as Lozano-Pérez (1983) but were able to get finer resolution in rotation using parallel graphics hardware that represented each θ-slice of the C-space obstacles by a rasterized bitmap. More recently, Solano González and Jones (1996) also used slice projection to investigate parallel computation for C-space mapping.

4.3. Handling More General Geometry

The first paper to map the C-space of a mobile robot more complicated than a polygon appears to be that of Laumond (1987), which extended the Minkowski difference obstacle-growing technique from polygons to generalized polygons. Bajaj and Kim also took up the challenge of generalizing obstacle-growing, but they focused on three-dimensional objects: in Bajaj and Kim (1988), they grow arbitrarily shaped three-dimensional obstacles by a moving sphere, and then in Bajaj and Kim (1990), they grow convex three-dimensional obstacles of arbitrary complexity by an object in that same class. A more recent paper by Kohler and Spreng (1995, 590) also grows convex B-rep objects of (near) arbitrary complexity by each other, only this time back in the two-dimensional domain. Kohler and Spreng's contribution is the development of an efficient obstacle-growing method, which, by subdividing the boundary segments of the interacting objects, obtains a precise representation of the grown obstacle without relying on a computer algebra system (unlike the methods of Bajaj and Kim).

In 1996, Heegaard suggests that if the objects in the workspace are represented by convex parametric curves, the contact surface can be obtained more efficiently by plotting it in to an alternative kind of C-space. If two interacting planar bodies are represented by parametric curves in terms of parameters ϵ_1 and ϵ_2, Heegaard observes that the contact surface for both translating and rotating motion can be easily formulated in terms of ϵ_1, ϵ_2, and d_n, where d_n is the distance between the closest points on the two objects. Note that although the space $\epsilon_1\epsilon_2 d_n$ is not defined in terms of degrees of freedom, it is still a valid configuration space for the specific domain of objects defined by convex parametric curves, since any point within it is sufficient to specify the location of every point in the system.

At the other end of the C-space representation spectrum, a number of C-space mapping systems have emerged that deal with arbitrary geometry by using a grid division:

- Dehne, Hassenklover, and Sack (1989) use a parallel computer architecture of a type usually used for image processing.

- Kavraki (1993) uses the fast Fourier transform. This technique is based on the observation that the Minkowski difference of two point-sets that are discretized into two-dimensional arrays is a convolution that can be achieved by taking the Fourier transform of the two arrays, multiplying the two transforms pointwise, and then taking the inverse Fourier transform of the result. As a result, the computational cost of this method depends only on the resolution of the grid—the geometry of the objects involved has no effect. Curto and Moreno (1997) have since applied the same technique to map the C-space of a planar revolute manipulator.

- Lin and Chang (1993) create a grid representation of a C-space map using *mathematical morphology* (which is similar but not identical to Minkowski operations) and a *shape decomposition* step, which breaks arbitrarily complicated shapes down into simpler elements, as illustrated in Figure 8.

- Chan, Tam, and Leung (1994) attack the problem using neural networks.

Last of all in this survey of techniques for mobile-robot mapmakers comes our own work (see Wise and Bowyer 1996; Wise, Eisenthal, and Bowyer 1997). Our method of C-space mapping, originally based on Woodwark's novel method of feature recognition (Parry-Barwick and Bowyer 1995), exploits the dimension-independent characteristic of constructive solid geometry. Starting with a CSG model of a d-dimensional object that is free to move in its world with n

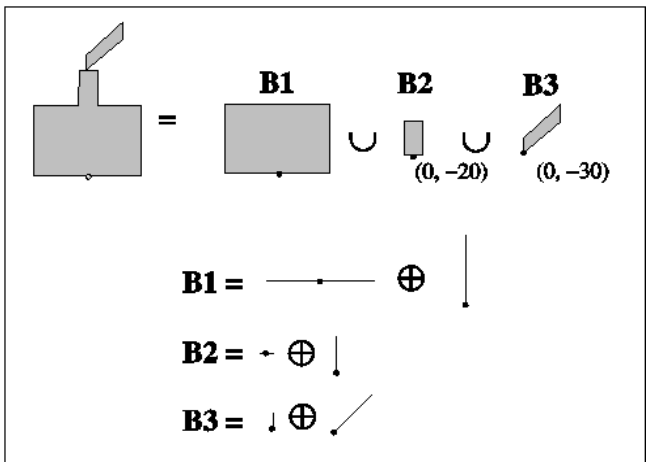


Fig. 8. Illustration of the shape decomposition technique of Lin and Chang (1993).

degrees of freedom, we construct a $(d+n)$-dimensional *omnimodel*, which represents the object at every conceivable position and orientation. Since this omnimodel contains information regarding every interaction that can happen between any part of the moving object and any part of its world, it contains enough information to build a global C-space map. The information is extracted by recursive subdivision in a bin-tree of axially aligned hyperboxes, which projects the $(d+n)$-dimensional omnimodel down into the n-dimensional C-space. In its most basic form, our mapmaking technique generates an ordinary bin-tree C-space map; it is slow but will handle any geometry that can be represented by semi-algebraic sets. From this starting point, we are developing a smarter system that uses a boundary representation whenever possible, generating a hybrid C-space map. We are also developing schemes that exploit the fact that in practical C-space mapping many regions are geometric transformations of other regions of the map.

5. Mapmaking Techniques for a Single Manipulator in a Static Environment

5.1. Founding Work

The links of a manipulator are modeled as a series of connected rigid moving objects. The degree of freedom of a particular link is specified by the controlled joint—which is usually prismatic or revolute—and the degrees of freedom of the whole manipulator (which define the configuration space) are the sum of the degrees of freedom of the parts. The C-space of a manipulator with n revolute joints and m prismatic ones is the $(n+m)$-dimensional manifold $\Re^m \times (S^1)^n$ (or a subset of that space when the motion of each joint it limited by mechanical stops; Latombe 1993).

The first path planner of any kind to use the configuration space approach is widely regarded as that of Udupa (1977), which maps the C-space for a manipulator, even though the author did not refer to it in those words.[6] Udupa plans the path of a polar-coordinate manipulator by mapping the problem into a space where the manipulator is reduced to a point, beginning by approximating the links of the manipulator as cylinders and growing the obstacles by their radius. This quick and easy method of reducing the links to line segments has been adopted by many other systems (for example, Faverjon 1984; Hasegawa and Terasaki 1988; Warren, Danos, and Mooring 1989; Hwang 1990). Udupa's work is also notable for its use of two C-space mappings—one of which considers the orientation of the highest link and one of which does not; this idea has since been returned to by others (for example, Faverjon 1984; Hasegawa and Terasaki 1988).

A second founding work for manipulator C-space mapping is that of Lozano-Pérez (1981), in which he outlines several ideas that have played major roles in C-space mapping research ever since, including hierarchical approximations (described in Section 2.2), selective refinement (Section 3.2.2), and the idea of representing a manipulator as a tree structure where each level restricts one degree of freedom. Lozano-Pérez goes on to describe an application of the θ-slice scheme outlined in Section 4 to the case of a Cartesian manipulator. The system still uses projected swept volumes but introduces a hybrid representation of C-space consisting of a tree where some nodes are axially aligned boxes and others are arbitrary convex polyhedra.

The rest of this section on C-space mapping for a manipulator is divided according to the representation scheme used for the C-space map.

5.2. Boundary Representation

The combination of multiple rotational degrees of freedom and the fact that moving a link moves every higher link makes it difficult to represent the boundary of the C-space obstacles for a revolute manipulator. Ge and McCarthy (1989, 1990), however, go some way to achieving this by representing the configuration of the end effector using *dual quaternions* in the *image space*. The set of possible configurations for the end-effector is bound by two constraints: the reachability constraint, determined by the links of the manipulator, and the contact constraint, determined by contact between the effector and the obstacles. Ge and McCarthy exploit the fact that both types of constraint can be represented by algebraic and parametric forms in the image space that can then be intersected easily. When this intersection is parameterized piecewise in terms of the joint angles by using inverse-kinematics, the result is an explicit representation of the C-space obstacles to the end-effector. This method is noteworthy, but the collisions between the links and the obstacles are not

6. Instead, Udupa used a delightfully Trekesque vocabulary of his own including "sectoroids," "pascs," and, our personal favorite, "pgram motion."

considered, so the C-space obstacles do not represent a true C-space map for the manipulator.

A true mapmaker that uses a boundary representation is that of Hwang (1990), who approximates a three-dimensional revolute manipulator as linked cylinders, and convex polyhedral obstacles as a collection of planar triangular facets. Hwang reduces the manipulator links to lines (like Udupa 1977) and calculates the intersection conditions between the first link and each triangular patch, unioning the results to derive boundary equations for the first joint variable. He goes on to show how the boundary equation of each higher joint variable can be solved explicitly in terms of the lower joint variables. Interestingly, Hwang confesses that although a precise representation of the C-space obstacles can be obtained in this way, the computational expense of intersecting the boundary equations means that any practical path planner must first convert the map into a representation that is easier to work with.

Discretization of the obstacles is taken one step further by Zhao, Farooq, and Bayoumi (1995), and Ma et al. (1995), who treat the obstacles as a discrete set of points. Zhao, Farooq, and Bayoumi (1995) discretize the workspace into a grid and then, for each cell that contains the surface of an obstacle, use a variation on inverse-kinematics to obtain parametric equations for the C-space obstacles caused to the manipulator by a point-obstacle at the cell's centroid. Obviously, the C-space obstacle for the complete obstacle set is obtained by unioning the results. Ma et al. (1995) use a similar approach but, dealing with obstacles composed of axially aligned boxes, reduce the amount of contact-surfaces generated by isolating the "fundamental points"—the small number of points on the surface of the obstacle set whose C-space obstacles contribute to the final result.

Finally in this section, recent research in computational geometry by authors such as Efrat and Sharir (1997) has shown that restricting one's attention to *fat* objects can give a significant reduction in the computational complexity of some problems. The literature differs slightly in its definitions of fatness, but the essence of the idea is captured by the quotient of the radius of the largest sphere completely enclosed within an object by the radius of the smallest sphere that completely surrounds it. Small fatness values mean that objects are either long and spindly like pencils or very convoluted like starfish. The largest (and fattest) possible quotient is 1, which is only achieved by a sphere itself.

van der Stappen, Halperin, and Overmars (1993) show that the combinatorial complexity of the free space in a C-space map is linear in the number of obstacles as long as they are fat and satisfy certain other assumptions such as not being too small in comparison to the robot. van der Stappen and Overmars (1994) go on to give a good[7] paradigm for motion planning avoiding fat obstacles that uses a decomposition of the workspace made efficient by the fatness of those obstacles, rather than a direct decomposition of the higher-dimensional C-space free space.

5.3. Division and Classification

Most of the divide-and-classify techniques that have been used to generate a C-space map for a manipulator have been covered in previous sections. However, one domain-specific technique exploits the tree nature of a manipulator to obtain a raster representation (see, for example, Lozano-Pérez 1987):

Starting at the base joint,

1. Determine what ranges of joint angles (if any) are completely safe from interference regardless of what joint angles the higher links take. One method of achieving this is by calculating the volume swept out by the higher links.

2. Discretize the complement of those safe ranges into equal-sized intervals of the desired resolution.

3. For each of these intervals, approximate the joint angle range as a single joint angle, then consider the effect of the next joint by recursively entering this process at step (1).

5.4. Hybrid

Shiller and Gwo (1993) present a mapmaking system that uses a hybrid representation. The authors establish an analytical representation of two-dimensional C-space obstacles, then divide the C-space using an adaptive division strategy. The safe leaf cells that result are bounded by axially aligned lines on some sides but the precise C-space obstacle surface on the others (as illustrated in Fig. 9). Significantly, the entire boundaries of the C-space obstacles are not evaluated—the only evaluations are of intersections and tangency points with axially aligned rays.

5.5. Related Techniques

Finally in this section, a brief mention of some techniques that have emerged for nonexhaustive mapping, which may be applicable to the global C-space mapping problem:

Buck passing. The SANDROS planner described by Chen and Hwang (1992) plans a path by mapping the C-space only until a more efficient local planner can handle the situation adequately. The result is a system that the authors claim gives a performance time proportional to task difficulty.

7. $O(n \log n)$ for a two-dimensional problem, $O(n^2 \log n)$ for three-dimensional polyhedral obstacles, and $O(n^3)$ for arbitrary obstacles.

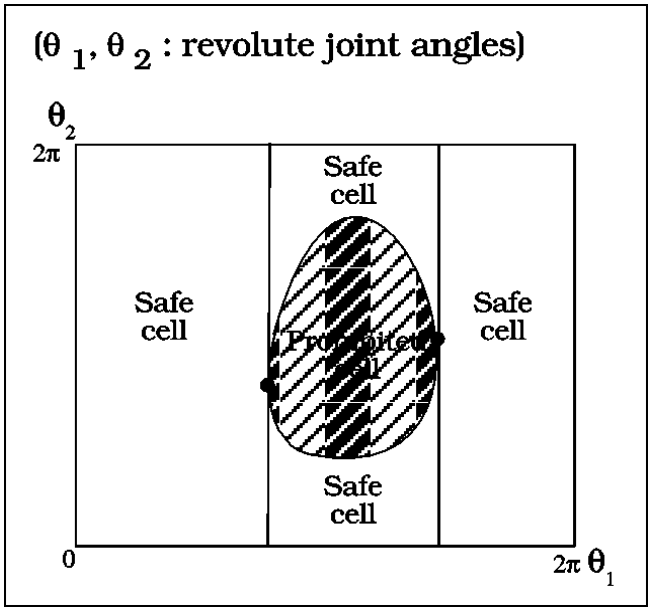


Fig. 9. An illustration of the hybrid division scheme of Shiller and Gwo (1993).

Topology analysis. Maciejewski and Fox (1993) borrow techniques from the analysis of kinematically redundant manipulators to extract information about the *topology* of the configuration space obstacles for a 3-DoF manipulator.

Contour tracing. Tso and Liu (1993) map the boundary of a two-dimensional C-space obstacle by finding a configuration on the surface and then using contour information to trace the boundary around the perimeter of the obstacle.

6. Classification Tables

Table 1 explains the abbreviations used in the following tabular summary of the C-space mapmaking literature:

Table 2. C-space mapmakers for a mobile robot in a static environment.

Table 3. C-space mapmakers for a manipulator in a static environment.

7. Summary and Conclusions

In summary, the key decisions to be made regarding how the robot and its environment are modeled are

What limits are placed on the shapes of the object models. Two-dimensional problems are much easier than three-dimensional ones, and polytopes are easier to handle than generalized polytopes, which are easier than more arbitrary geometry. Convex objects are much easier to handle than nonconvex objects, but the distributive nature of the C-space mapping makes it possible to handle nonconvex problems if the objects can be broken into convex components.

Which representation scheme to use. The most common representation schemes for the robot and environment are B-rep and pixel/voxel decomposition. B-rep is more flexible since it can be used in both boundary representation and discrete C-space mapping algorithms, but the choice will depend on the available data—for example, a discrete representation may be available directly from sensors.

Whether to introduce approximation. The computational expense of C-space mapping may be eased by introducing approximation at the object modeling stage and/or the C-space mapping stage.

The most central decision regarding the C-space mapmaking algorithm is which representation scheme to use for the map itself. The most common options are

Boundary representation. Representing a C-space obstacle by its boundary is achievable by performing Minkowski operations if the robot is a translating object, or otherwise by calculating patches of the contact surface corresponding to specific contact conditions. Such representations tend to be both compact and fast to compute. However, operations such as membership testing, inclusion, and Boolean operations may be expensive. A greater restriction is that precise solutions have only been implemented for arbitrary-shaped objects for the case of a single object translating in a fixed orientation, although they are available for unconstrained objects and manipulators if the objects are modeled as polytopes.

Divide-and-classify. The approach of dividing the C-space into discrete cells and classifying each as safe, prohibited, or contact is applicable to a wide range of problems, and there are a host of different division strategies, cell-classification schemes, and other useful techniques. Discretizing the C-space into a grid enables algorithms such as the fast Fourier transform to be used, which are independent of the shape of the object represented, but the memory requirement grows exponentially with dimensionality of C-space. In practical cases, hierarchical tree structures such as the bintree or 2^n-tree use orders of magnitude less space and are suitable for selective refinement, which makes it possible to map higher dimensional C-spaces.

Hybrid schemes. A common approach is to develop a hybrid representation. For example, some mapmakers use a

Table 1. Abbreviations Used in Tables 2 and 3

Criterion	Abbreviation	Meaning
Any	(. . .)	The researcher(s) suggest that the method could be applied to the bracketed case, but details of the application are not given—e.g., 2D(3D) indicates that the paper states that 3-D problems can be solved, but only details a 2-D solution.
Degrees of freedom	(*n*)	The researcher(s) suggest that the method could be extended to an arbitrary number of degrees of freedom.
C-space degrees of freedoms	PM	Only a *partial map* is made
Allowable geometry	conv-	convex
	cyl	cylindrical
	poly	polytopes
	gen-poly	generalized polytopes
	arb	more general than generalized polytopes
Approximation strategy (see Section 2.3)	P-P	Precise models, precise C-space mapping
	P-A	Precise models, approximate C-space mapping
	A-P	Approximate models, precise C-space mapping
	A-A	Approximate models, approximate C-space mapping
C-space representation	grown-obs	A B-rep of the translation-only C-space is used.
	c-surface	A B-rep of a C-space containing rotations is used.
	adapt-div	An adaptive division strategy is used.
	grid	A uniform grid division is used.
	raster	A raster representation is used.
	rep1/*rep2*	Both *rep1* and *rep2* are used.

boundary representation in the translation dimensions but discretize the rotation dimensions.

The state of the art of C-space mapping for a mobile robot is that a precise C-space map can be obtained for precise object models for both two- and three-dimensional cases if the robot translates in a fixed orientation and can be broken into convex pieces (Bajaj and Kim 1990). Rotations make a boundary representation much more difficult, but a precise C-space map can be made for the two-dimensional case as long as the objects are approximated by polytopes (for example, Avnaim and Boissonnat 1988) or as a collection of discrete cells (for example, Kavraki 1993). Six-dimensional C-space obstacles are so difficult to represent that only one paper (Donald 1985) obtains a global C-space map for a three-dimensional floating object with full degrees of freedom, and that map only enables paths to be planned that remain on the contact surface.

C-space mapmaking for manipulators has not been concerned with accurate object models since in practical applications the robot should not get close enough to the obstacles for precise geometry to play a role—instead, the focus has been on increasing the dimensionality of the C-space maps. This has mainly been achieved by using increasingly intelligent adaptive divide-and-classify techniques and selective refinement of the map, culminating in C-space maps of up to six dimensions (for example, Chen and Hwang 1992). Precise C-space maps have also been obtained for manipulators by representing the links as cylinders and treating obstacles as a collection of facets (Hwang 1990) or points (Ma et al. 1995; Zhao, Farooq, and Bayoumi 1995)—this has also enabled six-dimensional C-space maps.

In conclusion, the literature demonstrates that C-space mapmaking is an ongoing area of research, driven by the wide range of applications and the challenge of increasing the dimensionality of the maps and range of allowable geometry taken as input. Perhaps the most significant goal is to produce a precise six-dimensional C-space map for object models more general than polytopes, which supports efficient operations for membership testing, ray tracing, and path planning through safe as well as contact regions.

Acknowledgments

The authors would like to thank the United Kingdom's Engineering and Physical Sciences Research Council, Southco Ltd, and Information Geometers Ltd for their support for the work of which this review forms a part.

They would also like to thank their colleague David Eisenthal, who has developed half the software for that work and is applying it to areas other than C-space mapmaking.

Finally, they would like to thank one of the anonymous reviewers for suggesting an example that they have used in Section 2.1.

Table 2. C-Space Mapmakers for a Mobile Robot in a Static Environment

Paper	Dimen-sionality	Rotations Allowed?	Allowable Geometry		Approximate Strategy	C-space Rep.
			Robot	Obstacle		
Lozano-Pérez and Wesley, 1979	2D(3D)	yes for 2D	conv-poly	conv-poly	A-A	grown-obs /grid
Jarvis, 1983	2D(3D)	yes for 2D	poly	conv-poly	A-A	grown-obs /grid
Lozano-Pérez, 1983	2D(3D)	yes for 2D	poly	poly	A-A	grown-obs /grid
Gouzènes, 1984	2D(3D)	yes	poly (arb)	poly (arb)	A-A	adapt-div /raster
Brooks and Lozano-Pérez, 1985	2D	yes	poly	poly	A-A	adapt-div /c-surface
Donald, 1985	3D	yes	poly	poly	A-P	c-surface
Laumond, 1987	(2D)	(no)	(gen-poly)	(gen-poly)	A-P	(grown-obs)
Avnaim, Boissonnat, and Faverjon, 1988	2D	yes	poly	poly	A-P	c-surface
Avnaim and Boissonnat, 1988	2D(3D)	yes for 2D	poly	poly	A-P	c-surface
Bajaj and Kim, 1988	3D	no	sphere	arb	A-P	grown-obs
Laumond et al., 1988	2D	yes	poly	poly	A-A	adapt-div
Brost, 1989	2D	yes	poly	poly	A-P	c-surface
Dehne, Hassenklover, and Sack, 1989	2D	no	conv-arb	arb	A-P	grid
Paden, Mees, and Fisher, 1989	(2D)	(no)	(gen-poly)	(gen-poly)	A-A	(2^n-tree)
Bajaj and Kim, 1990	3D	no	conv-arb	conv-arb	P-P	grown-obs
Lengyel et al., 1990	$2\frac{1}{2}$D (3D)	about 1 axis	poly	poly	A-A	grown-obs /strips/grid
Verwer, 1990	(3D)	(yes)	(poly)	(poly)	(A-A)	(2^n-tree)
Zhu and Latombe, 1991	2D	yes	poly	poly	A-A	adapt-div /c-surface
Halperin, Overmars, and Sharir, 1992	2D	yes	L-shape	poly	A-P	c-surface
Kavraki, 1993	2D (3D)	yes for 2D	arb	arb	A-P	grid
Lin and Chang, 1993	2D (3D)	no	arb	arb	A-P	grid
Liu and Onda, 1993	2D (3D)	yes for 2D	poly	poly	A-A	grid
Chan, Tam, and Leung, 1994	2D	yes	conv-arb	arb	A-A	grid
Kohler and Spreng, 1995	2D	no	conv-arb	conv-arb	P-P	grown-obs
Heegaard, 1996	2D	yes	conv-arb	conv-arb	P-P	c-surface
Wise and Bowyer, 1996	2D(3D)	no (yes)	gen-poly (arb)	gen-poly (arb)	A-A	bin-tree (/c-surface)
Curto and Moreno, 1997	2D	yes	arb	arb	A-P	grid

Table 3. C-Space Map-Makers for a Manipulator in a Static Environment

Paper	Dimen-sionality	DoF		Allowable Geometry		Approximate Strategy	CS-Rep.
		Robot	CS-Map	Robot	Obstacle		
Udupa, 1977	2D (3D)	3	3	cyl	poly	A-A	grown-obs
Lozano-Pérez, 1981	3D	4(7)	4(7)PM	conv-poly	conv-poly	A-A	grown-obs /adapt-div
Faverjon, 1984	3D	6	3	conv-gen-poly	conv-gen-poly	A-A	grown-obs /grid /2^n-tree
Gouzènes, 1984	2D (3D)	3	3	poly (arb)	poly (arb)	A-A	adapt-div /raster
Laugier and Germain, 1985	3D	6	4	conv-poly	conv-poly	A-A	adapt-div
Red and Truong-Cao, 1985	2D	2	2	poly	poly	A-A	raster
Lozano-Pérez, 1987	3D	6 (n)	4 (n)	conv-poly	poly	A-A	grown-obs /raster /adapt-div
Faverjon and Tournassoud, 1988	3D	6	3	gen-poly (arb)	gen-poly (arb)	A-A	raster /2^n-tree
Hasegawa and Terasaki, 1988	3D	6	3	gen-poly	gen-poly	A-A	grid
Siméon, 1988	3D	4	4	poly	poly	A-A	raster /adapt-div
Paden, Mees, and Fisher, 1989	2D	2	2	gen-poly	gen-poly	A-A	2^n-tree
Warren, Danos, and Mooring, 1989	3D	2	2	cyl	conv-poly	A-A	grown-obs /grid
Adolphs and Nafziger, 1990	3D	6	3	poly?	poly?	A-A	grid
Branicky and Newman, 1990	3D	3	3	conv-poly	poly	A-A	grid
Ge and McCarthy, 1990	3D	6	6 PM	conv-poly	conv-poly	A-P	c-surface
Hwang, 1990	3D	2 (n)	2 (n)	cyl	poly	A-A	c-surface
Verwer, 1990	3D	5	5 PM	poly	poly	A-A	2^n-tree
Newman and Branicky, 1991	3D	3	3	gen-poly	gen-poly	A-A	grid
Bellier et al., 1992	3D	6	3	arb	arb	P-A	2^n-tree
Chen and Hwang, 1992	3D	6	6 PM	poly	poly	A-A	grid /adapt-div
De Pedro and Rosa, 1992	2D	2	2	line	points	A-A	grid
Shiller and Gwo, 1993	3D	2 (n)	2 (n)	poly	poly	A-P	c-surface /adapt-div
Tso and Liu, 1993	3D	3	3 PM	conv-poly	conv-poly	A-P	c-surface
Ma et al., 1995	3D	6 (n)	6 (n)	cyl	axially-aligned boxes	A-P	c-surface
Zhao, Farooq, and Bayoumi, 1995	3D	3 (n)	3 (n)	cyl	poly	A-P	c-surface
Ralli and Hirzinger, 1996	3D	≤ 6	≤ 6	cyl	arb	A-A	grid (with Kohonen reorganisation)
Duelen and Willnow, 1991	3D	5 (n)	5 (n) PM	gen-poly	gen-poly	A-A	2^n-tree

References

Adamowicz, M., and Albano, A. 1976. Nesting two-dimensional shapes in rectangular modules. *Computer Aided Design* 8(1):27.

Adolphs, P., and Nafziger, D. 1990. A method for fast computation of collision-free robot movements in configuration space. *IEEE International Workshop on Intelligent Robots and Systems*, Tokyo, Japan, July 4–6, pp. 5–12.

Avnaim, F., and Boissonnat, J. D. 1988. Polygon placement under translation and rotation. *Springer's Lecture Notes in Computer Science Series* 294:323–333.

Avnaim, F., Boissonnat, J. D., and Faverjon, B. 1988. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. *Springer's Lecture Notes in Computer Science Series* 391:67–86.

Bajaj, C., and Kim, M. S. 1988. Generation of configuration space obstacles—the case of a moving sphere. *IEEE Journal of Robotics and Automation* 4(1):94–99.

Bajaj, C., and Kim, M. S. 1990. Generation of configuration space obstacles: Moving algebraic surfaces. *International Journal of Robotics Research* 9(1):92–112.

Bellier, C., Laugier, C., Mazer, E., and Troccaz, J. 1992. Planning/executing six DOF robot motions in complex environments. *Proc. 91 IEEE RSJ International Workshop Intelligence Robots Systems IROS 91*, Osaka, Japan, pp. 91–96.

Branicky, M. S., and Newman, W. S. 1990. Rapid computation of configuration space obstacles. *Proc. 1990 IEEE International Conference Robotics Automation*, Cincinnati, OH, pp. 304–310.

Brooks, R. A., and Lozano-Pérez, T. 1985. A subdivision algorithm in configuration space for findpath with rotation. *IEEE Transactions on Systems Man and Cybernetics* 15(2):224–233.

Brost, R. C. 1989. Computing metric and topological properties of configuration-space obstacles. *International Conference Robotics Automation 1989* 1:170–176.

Canny, J. F. 1988. *The Complexity of Robot Motion Planning*. Cambridge, MA: MIT Press.

Cesati, M., and Wareham, H. T. 1995. Parameterized complexity analysis in robot motion planning. *Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics. Part 1 (Of 5)*, Vancouver, B.C., October 22–25 (Conf. Code 44222), vol. 1, pp. 880–885. IEEE, Piscataway, NJ.

Chan, R.H.T., Tam, P.K.S., and Leung, D.N.K. 1994. Solving the motion planning problem by using neural networks. *Robotica* 12(4):323–333.

Chen, P. C., and Hwang, Y. K. 1992. SANDROS: A motion planner with performance proportional to task difficulty. *Proceedings—IEEE International Conference on Robotics and Automation* 3:2346–2353.

Cheng, H., and Cheng, H. D. 1996. Feasible map algorithm for path planning. *Robotics and Autonomous Systems* 17(3):149–170.

Curto, B., and Moreno, V. 1997. Mathematical formalism for the fast evaluation of the configuration space. *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Cira, Monterey, CA, July 10–11 (Conf. Code 46973), Record - 85, pp. 194–199. IEEE, Piscataway, NJ.

Dehne, F., Hassenklover, A. L., and Sack, J. R. 1989. Computing the configuration space for a robot on a mesh-of-processors. *Parallel Computing* 12(2):221–231.

De Pedro, M. T., and Rosa, R. G. 1992. Robot path planning in the configuration space with automatic obstacle transformation. *Cybernetics and Systems* 23(3–4):367–378.

Donald, B. R. 1985. On motion planning with 6 degrees of freedom: Solving the intersection problems in configuration space. *Proc. IEEE International Conference on Robotics and Automation*, St. Louis, MO, pp. 536–541.

Donald, B. R. 1987. A search algorithm for motion planning with 6–degrees of freedom. *Artificial Intelligence* 31(3):295–353.

Duelen, G., and Willnow, C. 1991. Path planning of transfer motions for industrial robots by heuristically controlled decomposition of the configuration space. In *Robotic Systems: Proceedings of the 1991 EURISCON Conference, Korfu*, ed. S. G. Tzafestas, 217–224. Boston: Kluwer Academic.

Efrat, A., and Sharir, M. 1997. On the complexity of the union of fat objects in the plane. *Proceedings of the ACM Symposium on Computational Geometry '97*, Nice, France, pp. 104–112.

Faltings, B. 1987. Qualitative kinematics in mechanisms. *Proceedings of IJCAI-87*, Milan, Italy, pp. 436–442.

Faverjon, B. 1984. Obstacle avoidance using an octree in the configuration space of a manipulator. *Proceedings of the IEEE Int. Conf. Robotics*, Atlanta, GA, March, pp. 504–512.

Faverjon, B., and Tournassoud, P. 1988. Motion planning for manipulators in complex environments. *Geometry & Robotics, in Springers' Lecture Notes in Computer Science Series* 391:87.

Fujimura, K. 1993. Motion planning amidst dynamic obstacles in three dimensions. *1993 International Conference on Intelligent Robots and Systems*, Yokohama, Japan, p. 1387.

Ge, Q., and McCarthy, J. M. 1989. Equations for boundaries of joint obstacles for planar robots. *International Conference Robotics Automation* 1:164–169.

Ge, Q. J., and McCarthy, J. M. 1990. An algebraic formulation of configuration-space obstacles for spatial robots. *Proc. 1990 IEEE International Conference Robotics Automation*, Cincinnati, OH, pp. 1542–1547.

Gouzènes, L. 1984. Strategies for solving collision-free trajectories problems for mobile and manipulator robots. *International Journal of Robotics Research* 3(4):51.

Halperin, D., Overmars, M. H., and Sharir, M. 1992. Efficient motion planning for an L-shaped object. *SIAM Journal on Computing* 21(1):1–23.

Hart, P. E., Nilsson, N. J., and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions Systems, Science and Cybernetics* SSC-4(2):100.

Hasegawa, T., and Terasaki, H. 1988. Collision avoidance—A divide-and-conquer approach by space characterization and intermediate goals. *IEEE Transactions on Systems Man and Cybernetics* 18(3):337–347.

Heegaard, J. H. 1996. Efficient parameterization of the configuration space for rigid multi-body contact and impact. *Proceedings of the 1996 ASME International Mechanical Engineering Congress and Exposition*, Atlanta, GA, November 17–22 (Conf. Code 45867), vol. 33, pp. 431–432. ASME, New York.

Hopcroft, J. E., Schwartz, J. T., and Sharir, M. 1984. On the complexity of motion planning for multiple independent objects: P-space-hardness of the "warehouseman's problem." *International Journal of Robotics Research* 3(4):76.

Hwang, Y. K. 1990. Boundary equations of configuration obstacles for manipulators. *Proc. 1990 IEEE International Conference Robotics Automation*, Sacramento, CA, p. 298.

Hwang, Y. K. 1995. Current status and future research in motion planning. *Proceedings, IEEE International Symposium on Assembly and Task Planning*, Pittsburgh, PA, August 10–11, pp. 431–432.

Hwang, Y. K., and Ahuja, N. 1992. Gross motion planning—A survey. *ACM Computing Surveys* 24(3):219–291.

Jarvis, R. A. 1983. Growing polyhedral obstacles for planning collision-free paths. *Australian Computer Journal* 15(3):103–110.

Kavraki, L. 1993. Computation of configuration-space obstacles using the fast Fourier transform. *Proceedings IEEE International Conference on Robotics and Automation* 3:255–261.

Kavraki, L. E., Kolountzakis, M. N., and Latombe, J. C. 1996. Analysis of probabilistic roadmaps for path planning. *Proceedings of the 1996 IEEE 13th International Conference on Robotics and Automation. Part 4 (Of 4)*, Minneapolis, MN, April 22–28 (Conf. Code 44943), vol. 4, pp. 3020–3025. IEEE, Piscataway, NJ.

Kohler, M., and Spreng, M. 1995. Fast computation of the C-space of convex 2-d algebraic objects. *International Journal of Robotics Research* 14(6):590–608.

Latombe, J. C. 1993. *Robot Motion Planning*. Boston: Kluwer Academic.

Laugier, C., and Germain, F. 1985. An adaptative collision-free trajectory planner. *International Conference Advanced Robotics*, Tokyo, Japan, pp. 33–41.

Laumond, J. P. 1987. Obstacle growing in a nonpolygonal world. *Information Processing Letters* 25(1):41–50.

Laumond, J. P., Simeon, T., Chatila, R., and Giralt, G. 1988. Trajectory planning and motion control for mobile robots. *Geometry & Robotics, in Springer's Lecture Notes in Computer Science Series* 391:133.

Lengyel, J., Reichert, M., Donald, B. R., and Greenberg, D. P. 1990. Real-time robot motion planning using rasterizing computer graphics hardware. *Computer Graphics (ACM)* 24(4):327–335.

Lin, P. L., and Chang, S. H. 1993. A shortest path algorithm for a nonrotating object among obstacles of arbitrary shapes. *IEEE Transactions Systems, Man and Cybernetics* 23(3):825.

Liu, Y. H., and Onda, H. 1993. Constructing an approximate representation of a configuration space without using an intersection check. *1993 International Conference on Intelligent Robots and Systems*, pp. 644–651.

Lozano-Pérez, T. 1981. Automatic planning of manipulator transfer movements. *IEEE Transactions on Systems, Man and Cybernetics* 11(10):681.

Lozano-Pérez, T. 1983. Spatial planning: A configuration space approach. *IEEE Transactions on Computers* 32(2):108–119.

Lozano-Pérez, T. 1987. A simple motion-planning algorithm for general robot manipulators. *IEEE Journal of Robotics and Automation* 3(3):224–238.

Lozano-Pérez, T., and Wesley, M. A. 1979. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM* 22(10):560.

Ma, C., Li, W., Yang, Y., and Chang, L. 1995. Robot motion planning with many degrees of freedom. *Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics. Part 1 (Of 5)*, Vancouver, B.C., October 22–25 (Conf. Code 44222), vol. 1, pp. 892–897. IEEE, Piscataway, NJ.

Maciejewski, A. A., and Fox, J. J. 1993. Path planning and the topology of configuration-space. *IEEE Transactions on Robotics and Automation* 9(4):444–456.

Marr, D., and Nishihara, H. K. 1977. Representation and recognition of the spatial organization of three-dimensional shapes. Technical Report AIM-416, Artificial Intell. Lab, MIT, Cambridge, MA, May.

Murakami, T., and Gossard, D. C. 1992. Mechanism concept retrieval by behavioral specification using configuration space. *ASME Design Engineering Division (Publication): Design Theory and Methodology* 42:343–350.

Newman, W. S., and Branicky, M. S. 1991. Real-time configuration space transforms for obstacle avoidance. *International Journal of Robotics Research* 10(6):650–667.

Paden, B., Mees, A., and Fisher, M. 1989. Path planning using a Jacobian-based freespace generation algorithm. *Proceedings IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, pp. 1732–1737.

Parry-Barwick, S., and Bowyer, A. 1995. Multidimensional set-theoretic feature recognition. *Computer-Aided Design* 27(10):731–740.

Parsons, D., and Canny, J. 1990. A motion planner for multiple mobile robots. *Proc. 1990 IEEE International Conference Robotics Automation* 1:8.

Ralli, E., and Hirzinger, G. 1996. Global and resolution complete path planner for up to 6dof robot manipulators. *Proceedings of the 1996 IEEE 13th International Conference on Robotics and Automation. Part 4 (Of 4)*, Minneapolis, MN, April 22–28 (Conf. Code 44943), Record - 26, vol. 4, pp. 3295–3302. IEEE, Piscataway, NJ.

Ralli, E., and Hirzinger, G. 1997. Efficient c-space modelling using kohonen maps. *Proceedings of the 1997 8th International Conference on Advanced Robotics, Icar'97*, Monterey, CA, July 7–9 (Conf. Code 46955), Record - 74, pp. 433–438. IEEE, Piscataway, NJ.

Red, W. E., and Truong-Cao, H. V. 1985. Configuration maps for robot path planning in two dimensions. *ASME Journal of Dynamic Systems, Measurement and Control* 107(4):292–298.

Reif, J. H. 1979. Complexity of the mover's problem and generalizations. *Proceedings of the 20th Symposium on the Foundations of Computer Science*, San Juan, Puerto Rico, pp. 421–427.

Requicha, A., and Voelcker, H. 1983. Solid modeling: Current status and research directions. *IEEE Computer Graphics Applications*, 3(7):25–37.

Sacks, E., and Joskowicz, L. 1993. Automated modeling and kinematic simulation of mechanisms. *Computer Aided Design* 25(2):106–118.

Schwartz, J. T., and Sharir, M. 1983. On the piano movers' problem I: The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. *Comm. on Pure & Applied Maths* 36:345.

Schweikard, A., and Wilson, R. H. 1995. Assembly sequences for polyhedra. *Algorithmica* 13:539–552.

Shiller, Z., and Gwo, Y. R. 1993. Collision-free path planning of articulated manipulators. *Journal of Mechanical Design* 115(4):901–908.

Siméon, T. 1988. Planning collision-free trajectories by a configuration space approach. *Geometry & Robotics, in Springers Lecture Notes in Computer Science Series* 391:116–132.

Solano González, J., and Jones, D. I. 1996. Parallel computation of configuration space. *Robotica* 14(2):205–212.

Subramanian, D., and Wang, C.S.E. 1995. Kinematic synthesis with configuration-spaces. *Research in Engineering Design-Theory Applications and Concurrent Engineering* 7(3):193–213.

Tso, S. K., and Liu, K. P. 1993. Fast motion planner based on configuration space. *1993 International Conference Intelligence Robotics Systems*, Yokohama, Japan, pp. 1401–1408.

Udupa, S. M. 1977. Collision detection and avoidance in computer controlled manipulators. *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, Cambridge, MA, pp. 737–748.

van der Stappen, A. F., Halperin, D., and Overmars, M. H. 1993. The complexity of the free-space for a robot moving amidst fat obstacles. *Computational Geometry—Theory and Applications* 3(6):353–373.

van der Stappen, A. F., and Overmars, M. H. 1994. Motion planning amidst fat obstacles. *Proc. 10th ACM Symposium on Computational Geometry*, Stony Brook, NY, pp. 31–40.

Verwer, B.J.H. 1990. A multiresolution work space, multiresolution configuration space approach to solve the path planning problem. *Proc. 1990 IEEE International Conference Robotics Automation*, Cincinnati, OH, pp. 2107–2112.

Warren, C. W., Danos, J. C., and Mooring, B. W. 1989. An approach to manipulator path planning. *International Journal of Robotics Research* 8(5):87–95.

Wise, K. D. 1998. Exact configuration-space maps using constructive solid geometry. Technical Report 08/98, Dept. of Mechanical Engineering, University of Bath, February.

Wise, K. D., and Bowyer, A. 1996. Using CSG models in many dimensions to map where things can and cannot go. *Proceedings from CSG 96 Set-theoretic Solid Modelling: Techniques and Applications*, Winchester, UK, pp. 359–376.

Wise, K. D., and Bowyer, A. 1998. Configuration-space mapping for multiple moving objects: A survey of techniques and applications. Technical Report 09/98, Dept. of Mechanical Engineering, University of Bath, February.

Wise, K. D., Eisenthal, D., and Bowyer, A. 1997. Design optimization of rigid-body mechanisms via multidimensional CSG, 1998. Presented at the *1997 ASME Design for Manufacturing Symposium*, published in the *1998 ASME Design For Manufacturing Symposium*.

Woodwark, J. R. 1986. *Computing Shape*. London, UK: Butterworths.

Zhao, C. S., Farooq, M., and Bayoumi, M. M. 1995. Analytical solution for configuration space obstacle computation and representation. *Proceedings of the 1995 IEEE 21st International Conference on Industrial Electronics, Control, and Instrumentation* 2:1278–1283. IEEE, Los Alamitos, CA.

Zhu, D. J., and Latombe, J. C. 1991. New heuristic algorithms for efficient hierarchical path planning. *IEEE Transactions on Robotics and Automation* 7(1):9–20.