A Simple Characterization of the Infinitesimal Motions Separating General Polyhedra in Contact *

Ernesto Staffetti, Lluís Ros and Federico Thomas
Institut de Robòtica i Informàtica Industrial (UPC-CSIC)
Gran Capità 2-4, 08034 Barcelona, Spain
e-mail: {estaffetti, llros, fthomas}@iri.upc.es

Abstract

We present a simple local geometric characterization of the configuration space of two polyhedra in contact that provides a representation of all infinitesimal motions that separate them. The considered polyhedra are general in the sense that they possibly have non-convex faces and arbitrary number of holes.

The approach presented herein has two main advantages over former ones: (a) it only relies on the classical basic contacts between polyhedra, i.e. the vertex-face and edge-edge contacts; and (b) it does not require the local decomposition of non-convexities into convex parts.

1 Introduction

This paper presents an algorithm for computing the infinitesimal motions separating two polyhedra M and S in contact, seen as a moving body and a static environment, respectively. The algorithm is complete, in the sense that it finds all feasible motions, and conceptually simpler than previous approaches.

Our algorithm mainly operates in the 6-dimensional configuration space of M, i.e. $SO(3) \times \Re^3$. First, it considers the C-surfaces associated with the basic contacts at a configuration X_0. Then, it computes the arrangement of tangent hyperplanes to these surfaces in X_0, obtaining all 6-dimensional convex cones it induces in $\Re^6$. Finally, the output is a classification of each of these cones into *occupied* or *free*, depending on whether the configurations it contains correspond to intersecting or separated polyhedra. This

classification is efficiently attained by evaluating simple boolean expressions that depend on the geometric configurations of the basic contacts.

The problem at hand arises in the *assembly partitioning problem*: given a collection of polyhedral parts in contact, is there some infinitesimal motion that rigidly removes some subset of the parts from the rest? A negative answer means that the parts cannot be assembled by a sequence of rigid motions that fit them together. A positive answer, with the possible directions of motion, does not provide a plan on how to disassemble the parts, since the subsets need to be separated completely —and not only infinitesimally. However, this constitutes a very useful information for an assembly sequence planner [20].

The paper is structured as follows. Section 2 summarizes the contributions of this work and its connections with previous research in the field. Section 3 introduces several basic concepts needed throughout the paper. Afterwards, section 4 gives a general description of the algorithm, which is followed by explanations on the role of non-effective contacts (section 5), and details on the main geometric constructions: the computation of tangent hyperplanes (section 6) and the construction of the resulting plane arrangement (section 7). The characterization of free and occupied cones follows in section 8. Finally, section 9 gives the conclusions and points that deserve further attention.

2 Related work

Many different approaches to the motion planning and related problems have been presented over the years. A common topic of many of them is *Configuration Space*, the space of position parameters of the moving body, hereafter denoted by CS.

When using a configuration space description of the motion planning problem, one of the most widely
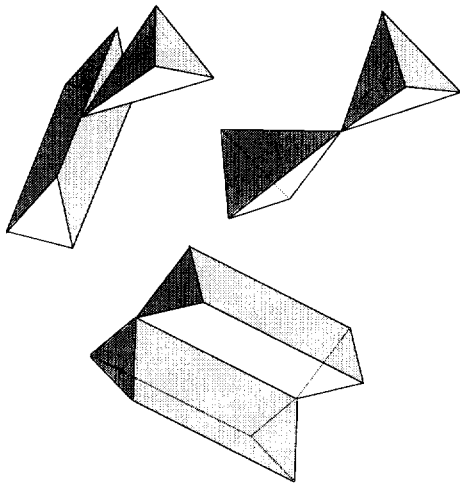
Figure 1: Contacts inducing a non-convex set of separating motions (adapted from [20]).

used techniques in path planning is *cell decomposition*, where the set of free configurations is decomposed into simple non-overlapping regions called *cells*. In this context, we can distinguish between topological and geometrical local characterizations of CS. Roughly speaking, while topological characterizations try to efficiently identify those cells neighbouring a given one [19], geometric characterizations provide a linearized description of the wall cells [5]. Our technique can be classified into this latter group, and relates mainly with the following research.

In [10], Hirukawa et al. presented a technique for automatically deriving the possible velocities and applicable forces between two polyhedra in contact. Their basic idea is to find separating planes between two neighborhoods of the contact points and use them to represent the possible motions of the moving object in terms of a system of homogeneous linear inequalities, which they efficiently solve. Although the overall algorithm is complete in the same sense as ours –it finds *all* feasible infinitesimal motions separating the two polyhedra– it lacks simplicity. Indeed, contrarily to us, they do not represent contacts by a composition of the two basic ones (edge-edge and vertex-face) but, instead, they introduce six types of contacts to cover all situations. Moreover, if at a contact one of the parts is non-convex, they must decompose it into convex ones, a step that is not required in our case.

In [21], Wilson and Matsui studied the assembly partitioning problem in its generality and found a

polynomial-time algorithm to solve it. In the same direction, Guibas et al. [8] provided a more efficient refinement using a similar approach. Their key observation was to realise that whenever there is a motion that infinitesimaly removes a subassembly from the rest of the parts, then there is also a motion inside the so-called *maximally covered cells*. Hence, to decide whether there is a solution they just focus the search in these cells. Due to its generality, their method can also be applied to our case of an assembly with just two parts. However, they only consider types of contact inducing a convex set of feasible motions, and are not able to cope with the three situations depicted in fig. 1. For these, the set of feasible solutions is non-convex, but this poses no special difficulty in our scheme, which also implicitly considers them. For example, in the situation of fig. 2, the algorithm in [8] would only provide part of the feasible separating motions.

Finally, it is important to mention that once we know all infinitesimal motions between objects in contact, one can derive the set of applicable forces on them. This is a useful information when using force control to assemble the parts. With this aim, in [12], Mason described the feasible velocities of two objects in contact with a linear system of equations, and showed that the orthogonal complement to its solution space is the set of applicable forces. Also in the same line, but using Screw Theory, in [13] Ohwovoriole and Roth showed this dual relationship between forces and velocities, and the way they can be computed from each other.
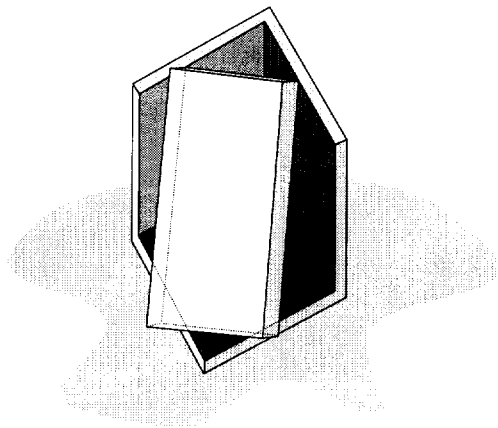

Figure 2: The algorithm in [8] would not provide all separating motions (adapted from [10]).
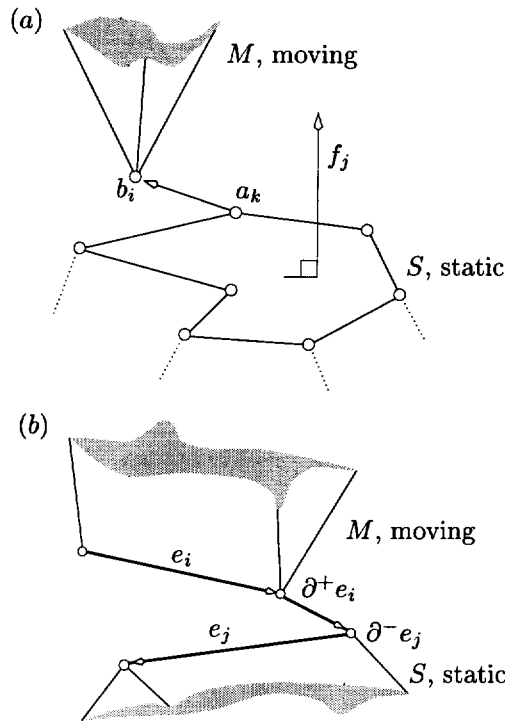
Figure 3: Elements intervening in the C-functions A_{b_i,f_j} (a), and C_{e_i,e_j} (b).

3 Basic contacts, C-functions, C-surfaces and predicates

There are three basic contacts between a moving polyhedron, say M, and a static polyhedral environment, say S, in 3D euclidean space [2, 3]; namely:

Type-A A vertex of M is in contact with a face of S.

Type-B A face of M is in contact with a vertex of S.

Type-C An edge of M is intersecting an edge of S.

All other contacts can be expressed as a combination of these three basic contacts.

What follows is a brief summary of needed results. The reader can find a detailed explanation in [18].

According to fig. 3, we define the following function for any possible type-A contact, called *type-A C-function*,

$$A_{b_i,f_j} = < f_j, b_i - a_k >,$$

where $< \cdot, \cdot >$ denotes the standard dot product, and a_k is any vertex on face f_j.

If vertex b_i meets the plane supporting face f_j, then $A_{b_i,f_j} = 0$. All configurations that satisfy this equation define a smooth manifold in the corresponding $\mathcal{CS}$ called *type-A C-surface*.

We also use the predicate $\mathbf{A}_{b_i,f_j}$, associated with the function A_{b_i,f_j}, which is true when $A_{b_i,f_j} > 0$, and false otherwise.

Likewise, a *type-B C-function*, B_{f_i,b_j}, a *type-B C-surface* and the corresponding *type-B predicate*, $\mathbf{B}_{f_i,b_j}$ are defined. $\mathbf{B}_{f_i,b_j}$ is true when $B_{f_i,b_j} > 0$, that is, when vertex b_j of S is over face f_i of M, and false otherwise.

According to fig. 3, we define a *type-C function* as:

$$C_{e_i,e_j} = < e_i \times e_j, \partial^- e_j - \partial^+ e_i >,$$

where $\partial^+ e_i$ and $\partial^- e_i$ are the vertices defining edge e_i such that $< \partial^+ e_i - \partial^- e_i, e_i >$ is positive. Thus, if the line supporting edge e_i meets the line supporting edge e_j, then $C_{e_i,e_j} = 0$. As above, the configurations that satisfy this equation define a smooth manifold in $\mathcal{CS}$ called *type-C C-surface*. We also define the predicate $\mathbf{C}_{e_i,e_j}$, associated with the function C_{e_i,e_j}, which is true when $C_{e_i,e_j} > 0$, and false otherwise.

Surface intersection of two polyhedra occurs if and only if an edge of one object intersects a face of the other. Using the results in [18], it can be proved that edge e of the moving polyhedron is piercing face f of the static one if, and only if, the boolean expression

$$\left[\bigoplus_{e_j \in \partial f} \begin{matrix} (\mathbf{A}_{\partial^+ e, f} \oplus \mathbf{A}_{\partial^- e, f}) \cap \\ (\mathbf{B}_{f_0, \partial^+ e_j} \oplus \mathbf{B}_{f_0, \partial^- e_j}) \cap \\ (\mathbf{B}_{f_0, \partial^- e_j} \oplus \mathbf{C}_{e, e_j}) \end{matrix} \right] \tag{1}$$

is true, where $\oplus$ denotes the XOR boolean operator, and f_0 is any of the two faces containing e.

Another similar boolean expression allows us to detect the case in which a face of the moving object is pierced by an edge of the static one.

The cases in which a vertex is effectively touching a face (not just the supporting plane but the face itself) or two edges are in contact (not just their supporting lines) are called type-A and type-B *effective contacts*, respectively, and *non-effective* otherwise. While the detection of a type-A effective contact boils down to a point-in-polygon problem [9], the detection of a type-B effective contact amounts to a simple line segments intersection problem [16]. In any case, these situations can be detected, as above, using boolean expressions combining basic predicates through XOR operators. As a consequence, only sign changes are relevant to
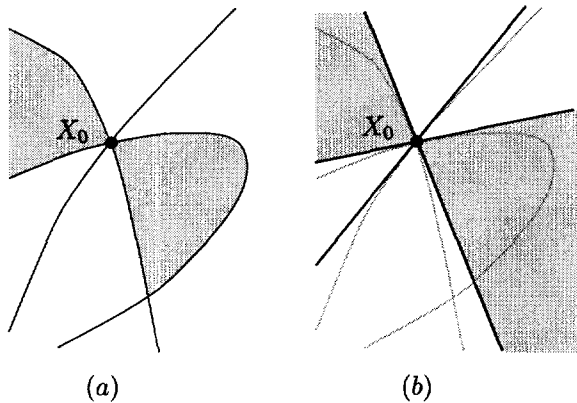
(a)   (b)

Figure 4: The C-surfaces intersecting at X_0 (a) are approximated by their tangent hyperplane at this point (occupied cells and cones are shaded). The resulting arrangement of hyperplanes (b) provides a one-to-one assignment between resulting cones in $\Re^6$ and cells adjacent to X_0 in $\mathcal{CS}$.

detect intersections and contacts, and hence edge directions and face normals are not required to follow a specific criterion.

4   Outlook of the algorithm

We assume that our polyhedra are in contact. In other words, our configuration in the $\mathcal{CS}$ lies on a given set of C-surfaces —those for which the corresponding C-functions vanish.

The proposed algorithm acts in four steps.

First, the above boolean expressions are evaluated for all faces of one polyhedron and all edges of the other, and vice versa. All involved C-functions should be evaluated and those that vanish, recorded. It is important to realize that all possible C-functions must be evaluated, which entails a $O(ms)$ number of C-functions, m and s being the number of vertices in the moving and static polyhedron, respectively [18].

Second, the tangent hyperplanes of the C-surfaces associated with the vanishing C-functions at the current configuration, X_0, are computed. This leads to an arrangement of hyperplanes in $\Re^6$ intersecting at X_0 (fig. 4). In other words, we get a set of cones that fully cover $\Re^6$ and whose apex is X_0.

At this point it should be clear that the arrangement of hyperplanes in $\Re^6$ is equivalent to the arrangement of C-surfaces in $\mathcal{SO}(3) \times \Re^3$ within an infinitesimal neighborhood of X_0.

Third, the assignment between cells and cones is effectively computed. All cells adjacent to X_0 can be characterized by a combination of signs for the C-functions vanishing at X_0. Nevertheless, not all sign combinations correspond to one of these adjacent cells and this can be easily detected by analyzing the arrangement.

Fourth, each cone is classified into free or occupied according to this circumstance for the corresponding cell. To this end, the boolean expressions (1) are reevaluated substituting the result of the vanishing C-functions by 1 or -1 according to the sign sequence for the cone. This allows us to classify the set of cones into free or occupied, the free ones describing the infinitesimal motions that separate the polyhedra. This step only requires evaluating boolean expressions.

5   The irrelevance of non-effective contacts

There is a factor that introduces an unnecessary complexity in our arrangement which is better seen with an example. Let us consider the two cubes C_S and C_M in fig. 5. The corresponding configuration of C_M in $\mathcal{CS}$ lies on 24 different C-surfaces: 8 correspond to vertex-face contacts and the other 16 to edge-edge contacts. However, as seen in the figure, only six contacts are effective.

If we consider all contacts, effective and non-effective, the number of cones in the arrangement may be huge. Fortunately, tangent planes associated with

• *edge-edge effective*
◦ *edge-edge non-effective*
■ *vertex-face effective*
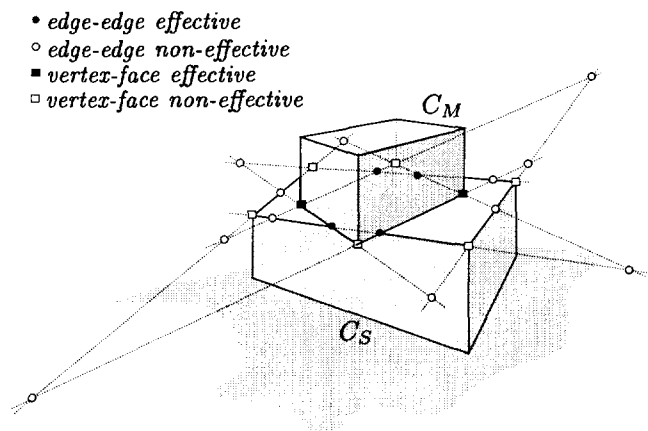□ *vertex-face non-effective*

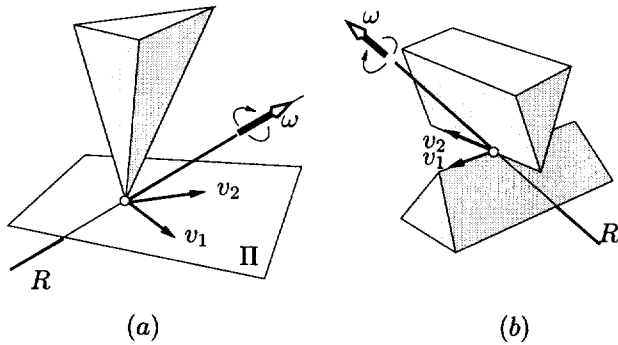Figure 5: A configuration of two cubes lying on 24 C-surfaces.

Figure 6: Infinitesimal motions that keep the basic contacts.

non-effective contacts need not be considered in the arrangement, because the adjacent cells they separate are either both free, or both occupied. To prove this, let us consider two adjacent cones separated by a tangent plane associated with a non-effective contact at X_0. Let us assume that one of the cones is occupied and the other free. In other words, an infinitesimal motion in CS from a point in the neighborhood of X_0 moving from the corresponding free cell to the occupied one should provoke an intersection between both polyhedra. Nevertheless, this is impossible since, by construction, the trajectory only passes through a non-effective contact configuration.

Since the result of the intersection test to classify a cone as occupied or free is independent from the signs associated with non-effective contacts, planes corresponding to them are not strictly required to be considered in the arrangement.

Now, the following sections expand on each of the steps of our algorithm.

6 Computing tangent hyperplanes

The normal to a C-surface at X_0 is the gradient of the corresponding C-function evaluated at X_0, i.e. $\nabla f(X_0)$. Then, the equation of the tangent hyperplane is

$$\nabla f(X_0) \cdot (X - X_0) = 0.$$

The use of normals to C-surfaces is in general of great importance for motion planning, as already recognized in the early eighties in [3, page 173].

The expressions for the required gradients can be obtained mainly in two ways, either algebraically, as

in [3], or geometrically, as in [13].

The former consists in first deriving the explicit expressions of the C-surfaces for arbitrary vertices, edges and faces in terms of the position parameters of the moving object, and then obtaining the required partial derivatives using a symbolic manipulator. Note that, at least using Euler angles, the general expression for a type-C C-surface is almost two pages long [3, page 302].

The geometric approach consists in obtaining the infinitesimal motions that separate or keep in contact both polyhedra directly from the geometry of the basic contacts. For example, in fig. 6a any rotation ω around the axis R followed by a translation on the plane Π generated by v_1 and v_2 keeps the contact between both polyhedra. The same holds for the edge-edge contact of fig. 6b. These motions can be easily recorded in terms of sets of screws, as described in [13] which, in turn, when expressed in the chosen motion parameters, provide the required tangent hyperplanes. Other alternatives that avoid the explicit derivation of position equations can be found in [7, Appendix B] and [17].

Tangent hyperplanes have also been obtained from discrete approximations of configuration spaces [5] which can be a useful alternative in some applications.

In our case, we have generated the general algebraic expressions for normals to C-surfaces, and their tangent hyperplanes, using Maple V.

7 The hyperplane arrangement

The n tangent hyperplanes to the C-surfaces on which the current configuration lies define an arrangement that partitions $\Re^6$ into several polyhedral cones, all with coincident apexes in X_0 (fig. 4).

As defined, every hyperplane receives the same orientation as its corresponding C-surface, namely the one induced by the gradient $\nabla f(X_0)$. Thus, every cone can be characterized by a sequence of signs, each sign telling whether the cone is on one side or the other of the corresponding hyperplane. However, note that with n hyperplanes not all possible 2^n sign sequences correspond to existing cones. In order to enumerate the cones generated by the hyperplane arrangement, we employ the Reverse Search Algorithm due to Avis and Fukuda [1, page 29].

To explain the basic idea of this algorithm, let C be the set of cones to be listed and suppose we have some objective function to be maximized over the elements of C. The algorithm proceeds by applying a

local search on C that moves from any cone to a neighboring one in which the objective function is greater, until a local optimum is found. Now, let us imagine the simple case in which there is only one optimum, say the cone c^*, and consider the directed graph D in which each node represents a cone and there is an arc linking two nodes if the associated cones are consecutively found in the local search. It is easy to see that D is a tree spanning all cones with c^* as the only sink, and therefore we can enumerate all its nodes by means of, for example, a depth-first search. Then, the major operation is crossing each edge against its orientation that correspond to reversing the local search. Since D is a tree, it is evident that we do not have to store any information about the visited nodes.

The time complexity of this algorithm is proportional to the number of cones times a polynomial in the number of hyperplanes, and its space complexity is polynomial in the number of hyperplanes.

The following considerations must be taken into account when implementing this enumeration.

- Every cone has a polar symmetric one with respect to X_0, so that polar cones are characterized by opposite signs. Now, if we intersect our arrangement with an external hyperplane $\mathcal{H}$ not through X_0, the induced arrangement on $\mathcal{H}$ allows to fully characterize all valid sign sequences. In other words, we can reduce the dimensionality of our arrangement from 6 in $\mathcal{CS}$ to 5 in $\mathcal{H}$.

- According to theorem 6.4.1 in [14], the number of existing cells in a 5-dimensional arrangement is $O(n^5)$. This result assumes that the arrangement is *simple*, i.e. all involved hyperplanes are in general position. If not, the number of cones is lower. Actually, this is what happens to our arrangement: it is far from simple because many hyperplanes intersect at the same subspaces. The ultimate reason for this to happen is that many vertices in our polyhedra are coplanar —faces are not in general triangular.

- In practice, rounding errors in the computation of normals make the arrangement to behave as simple in $\mathcal{H}$. An example shows the situation: if the normals to six hyperplanes of our arrangement in $\mathcal{CS}$ are coplanar, these six hyperplanes intersect in a line. Nevertheless, rounding errors in the computation of their normals give rise to a cone and its polar symmetric that, actually, do not exist.

8 Free and occupied cones

Consider a configuration in $\mathcal{CS}$, $X_f = X_0 + \Delta X$, such that $|\Delta X| \to 0$, i.e. ΔX defines an infinitesimal motion. Assume that no C-function vanishes at X_f. Then, the signs of all C-functions evaluated at X_f are the same as for X_0, the only difference being the signs of the C-functions vanishing at X_0, and note that these signs are the ones that characterize the cone in which X_f lies. Now, deciding if X_f is free from intersections –and hence the corresponding cone–, is straightforward and perhaps is one of the key points of our algorithm: it can be done solely based on the evaluation of boolean expressions.

Then, deciding which cones of the arrangement encompass infinitesimal motions that separate or interpenetrate the polyhedra can be carried out by evaluating the predicates (1) at X_0 substituting the values of the vanishing C-surfaces by -1 or 1 according to the signs for the cone. This should be repeated for each cone to obtain the desired classification. Remind that signs for non-effective contacts are irrelevant, leading to the same result regardless of their value.

Neighboring free or occupied cones can be finally grouped to obtain a more compact representation. This can be carried out using, for example, the algebraic technique described in [6].

9 Conclusions

We have presented an algorithm that computes the separating motions between two general polyhedra in contact. It relies on the study of the partition induced by the arrangement of hyperplanes tangent to the C-surfaces associated with effective contacts, at the given configuration of the moving polyhedron.

The problem of finding separating motions has been reduced to that of computing all occupied convex cones of maximum dimension, induced by the hyperplane arrangement, by evaluating boolean expressions.

Our local characterization of the configuration space only relies on the classical basic contacts between polyhedra, i.e. the vertex-face and edge-edge contacts. Other contacts, such as vertex-edge or vertex-vertex contacts, are not explicitly required. Also, since the used boolean expressions are valid for general polyhedra, no concavities have to be decomposed into convex parts. Thus, our algorithm is a simpler alternative to the one presented in [10].

The presented algorithm is being implemented as a utility within the PLASM package [15]. While the re-

quired geometric routines (the computation of the tangent hyperplanes and the resulting arrangement) is being implemented in C, PLASM provides the required solid modeling capabilities. Although the implementation details are outside the scope of this paper, it is worth to mention that the presented algorithm is simple to implement, its only complexity being the great deal of indexing tables required.

The natural extension of the presented work is the characterization of not only those motions that separate two polyhedra, but also those that preserve their contact. To this end, all cells in the hyperplane arrangement must be computed, not only those of maximum dimension; i.e. those herein referred as cones.

Acknowledgements

References

[1] D. Avis, K. Fukuda, "Reverse Search for Enumeration," *Discrete Applied Mathematics,* Vol. 6, pp. 21-46, 1996.

[2] J. F. Canny, *The Complexity of Robot Motion Planning,* MIT Press, 1987.

[3] B. R. Donald, *Local and Global Techniques for Motion Planning,* Master Thesis, Massachusetts Institute of Technology, Cambridge MA, USA, 1984.

[4] B. R. Donald, "A Search Algorithm for Motion Planning with Six Degrees of Freedom," *Artificial Intelligence,* Vol. 31, pp. 295-353, 1987.

[5] A. B. Doyle, D. I. Jones, "A Tangent Based Method for Robot Path Planning," *IEEE International Conference on Robotics and Automation,* pp. 1561-1566, San Diego CA, USA, 1994.

[6] H. Edelsbrunner, "Algebraic Decomposition of Nonconvex Polyhedra," 36th Annual Symposium on Foundations of Computer Science, pp. 248–257, 1995.

[7] K.-S. Fu, R. C. Gonzalez, C. S. Lee, "Robotics: Control, Sensing, Vision, and Intelligence," McGraw-Hill, 1987.

[8] L. J. Guibas, D. Halperin, H. Hirukawa, J.-C. Latombe, R. H. Wilson, "A Simple and Efficient Procedure for Polyhedral Assembly Partitioning Under Infinitesimal Motions," *IEEE International Conference on Robotics and Automation,* pp. 2553-2560, Nagoya, Japan, 1995.

[9] E. Haines, "Point in Polygon Strategies," in *Graphics Gems I,* edited by A. Glassner, Academic Press, Boston, 1990.

[10] H. Hirukawa, T. Matsui, K. Takase, "Automatic Determination of Possible Velocity and Applicable Force of Frictionless Objects in Contact from a Geometric Model," *IEEE Transactions on Robotics and Automation,* Vol. RA-10, no.3, pp. 309-322, 1994.

[11] J.-C. Latombe, *Robot Motion Planning,* Kluwer Academic Publishers, 1991.

[12] M. T. Mason, "Compliance and Force Control for Computer Controlled Manipulators," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. SMC-11, no. 6, pp. 418-432, 1981.

[13] M. S. Ohwovoriole, B. Roth, "An Extension of Screw Theory," *Journal of Mechanical Design,* Vol. 103, pp. 725-735, 1981.

[14] J. O'Rourke, *Computational Geometry in C,* Cambridge University Press, 1993.

[15] A. Paoluzzi, V. Pascucci, M. Vicentino, "Geometric Programming: A Programming Approach to Geometric Design," *ACM Transactions on Graphics,* Vol. 14, no. 4, pp. 266–306, July 1995.

[16] M. Prasad, "Intersection of Line Segments," in *Graphics Gems II,* edited by J. Arvo, pp. 7-9, Academic Press, Boston, 1991.

[17] E. Staffetti, L. Ros, F. Thomas, "Finding Infinitesimal Motions of Objects in Assemblies Using the Grassmann-Cayley Algebra," submitted to the Tenth World Congress on the Theory of Machines and Mechanisms, June 1999.

[18] F. Thomas, C. Torras, "Interference Detection Between Non-Convex Polyhedra Revisited with a Practical Aim," *1994 IEEE International Conference on Robotics and Automation,* Vol. I, pp. 587-594, San Diego CA, USA, 1994.

[19] F. Thomas, "An Approach to the Movers' Problem that Combines Oriented Matroid Theory and Algebraic Geometry," *IEEE International Conference on Robotics and Automation,* Vol. III, pp. 2285-2293, Nagoya, Japan, 1995.

[20] R. H. Wilson, *On Geometric Assembly Planning,* Ph.D. Thesis, Stanford University, Stanford CA, USA, 1992.

[21] R. H. Wilson, T. Matsui, "Partitioning an Assembly for Infinitesimal Motions in Translation and Rotation," *IEEE International Conference on Intelligent Robots and Systems,* pp. 1311-1318, 1992.