An Efficient Algorithm (FAPRIC) for Finding the Principal Contacts Possibly Established due to Uncertainties *

Jing Xiao and Lixin Zhang
Computer Science Department
University of North Carolina - Charlotte
Charlotte, NC 28223, USA

Abstract

An implemented algorithm is presented for finding all principal contacts possibly established between two surface features (i.e., faces, edges, or vertices) of two polyhedral objects due to location (i.e., position and orientation) uncertainties. The algorithm FAPRIC (for Finding All PRIncipal Contacts) requires as inputs: the estimated locations of the two objects, the geometric models of the objects, and the bounds on position and orientation uncertainties. The algorithm is based on checking intersections between the grown regions of the objects by location uncertainties [12] with the S-tope model [5]. It is very efficient and apt for real-time processing. The information obtained can serve as a guide for more accurate extraction and further reasoning of the contact data, possibly with additional sensors. As the surface of an arbitrary object can be approximated by planar patches, the algorithm can also be applied to general objects. The work is motivated by the need for automatically recognizing contacts in spite of uncertainties in robotic tasks.

1 Introduction

Automatic recognition of contacts is crucial for part-mating or assembly tasks where both task descriptions and operations are contact-based, particularly because of the presence of uncertainties. First, due to uncertainties, unintended contact between the part held (by the manipulator) and other parts in the environment can occur. Thus it is necessary to be able to recognize such a contact and to distinguish unintended contacts from intended ones, such as the contacts which define the goal state of an assembly. Secondly, the recognition task itself is also much complicated by the presence of uncertainties, especially for nonconvex objects, even if the environment can be well controlled with known objects.

Although there exist many attempts to determining contact states or constraints, only a few addressed the effect of uncertainties in contact recognition, and they were based almost universally on the approach of hypotheses-and-tests [1, 11, 4, 8]. For these methods to work, a key problem is how to obtain effectively the initial contact hypotheses in the first place. Xiao[10] specifically defined the problem as how to obtain automatically the possible set of topological contacts from the sensed locations of the objects in contact, taking into account sensing uncertainties. Xiao later [12] developed a general approach, which would first "grow" the two objects by location uncertainties feature by feature and then determine the set of all possible contacts between two features of the two objects respectively based on the intersection between the grown regions of the two features.

The algorithm FAPRIC provided in this paper is for implementing Xiao's approach. Although in [12], the analytical descriptions of the grown regions of any features are given, since these regions are non-polyhedral and mostly non-convex, there is no efficient algorithm to compute the intersection between two such regions. Thus, we use the S-tope model introduced by Hamlin, Kelley and Tornero to approximate the grown regions and transform the problem to that of checking intersections between S-topes, for which there is an efficient algorithm [5].

2 Background

In this section, we review the contact model, uncertainty model, and the approach of contact recognition in terms of those models as described in [12].

*This research is supported by a National Science Foundation grant under number IRI-9210423.

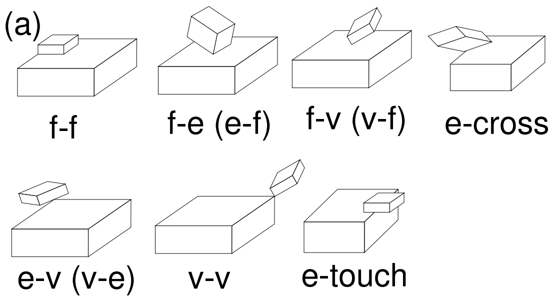

Figure 1: Principal contacts

2.1 Contact Model

Contacts between two polyhedra can be described in terms of the surface features — faces (f), edges (e), and vertices (v) of the polyhedra. A *principal contact* (PC) [10] captures the single contact between a pair of surface features which are not the boundaries of other contacting surface features (if there is more than one pair in contact)[1]. For example, Fig. 1a shows a PC of face-face (f-f) and not of edge-face or face-edge (f-e or e-f) or other types. Theoretically, there are 10 types of possible PCs as shown in Fig. 1. A contact between two polyhedra can be described by the set of PCs formed, called a *contact formation* (CF) [1]. Note that the concept of CF is mostly aimed at describing contacts between *nonconvex* objects. For two convex objects in contact, the CF contains only a single PC.

2.2 Uncertainty Model

We define *position uncertainty* ϵ_p as the magnitude of the maximum possible difference between an estimated (or sensed) position and the actual one. We define *orientation uncertainty* ϵ_o as the magnitude of the maximum possible angle between an estimated (or sensed) vector and the actual one. Note that for a solid object P, any point r in or on P defines a vector $\mathbf{r}$ with respect to the origin of the object frame. Thus, given a sensed orientation of P, a possible actual orientation is such that for any $r \in P$, the angle between the vector $\mathbf{r}$ and its estimated value is bounded by ϵ_o.

2.3 Approach for Contact Recognition

With the above models, the problem of contact recognition can be formulated as: *given two polyhedral objects P_1 and P_2 in collision, their geometric models, and their estimated locations, find the contact formation between them taking into account location uncertainties.*

[1]This definition is slightly different from what defined in [1].

Since contacts only occur between the surface features of two objects, our approach [12] is to find first the union of the regions which a feature of an object occupies at each possible location of the object — subject to the given estimated location of the object and uncertainty bounds ϵ_p and ϵ_o, and call this union the *grown region* of the feature. Then by analyzing the intersection between the two grown regions of two features from P_1 and P_2 respectively, the possible PCs between the two surface features can be determined. Specifically, if the two grown regions do not intersect, then the two features are *surely* not in contact. If the two grown regions do intersect, depending on how they intersect, the possible PCs between the two features can be determined. The set of all possible PCs between objects P_1 and P_2, denoted as S_{pc}, can be found this way.

Once S_{pc} is determined, further analyses can be conducted [14] to determine those subsets of S_{pc} which form geometrically valid CFs, and force sensing can be used to verify if any of such CFs is most likely the actual CF between the two objects [4, 1]. As a complementary or alternative measure, vision sensing can also be used to verify directly if a PC in S_{pc} actually exists or not [11]; the confirmed existing PCs should form the actual CF.

3 Assumption and Problem Analysis

Based on the above approach, we now focus on how to determine efficiently the set S_{pc}. For simplicity, we assume that one of the two objects in contact is fixed at a known location, denoted as object F, and the other object is held (by a manipulator) with position uncertainty ϵ_p and orientation uncertainty ϵ_o, denoted as object H. Then, as shown in [13], the possible PCs between two features u_i^F and u_j^H of F and H respectively can be obtained from analyzing the intersection between $\mathsf{U}^F_{p,i}$ and $\mathsf{U}^H_{o,j}$, where $\mathsf{U}^F_{p,i}$ denotes the grown region of u_i^F by the position uncertainty ϵ_p, and $\mathsf{U}^H_{o,j}$ denotes the grown region of u_j^H by the orientation uncertainty ϵ_o.

Although [12, 13] described analytically the grown region of each type of features by either ϵ_p or ϵ_o, since all these regions are nonpolyhedral and the regions grown by ϵ_o are also nonconvex, how to compute the intersection between two such regions (i.e., $\mathsf{U}^F_{p,i}$ and $s\mathsf{U}^H_{o,j}$) efficiently poses a great challenge. Nevertheless, to be computationally efficient is very important as we aim at real-time recognition of contacts.

There is little work in computational geometry addressing intersection checking between nonconvex objects. One approach [2] is to decompose a concave object into convex parts [2]. However, it cannot be applied to the grown regions of features which we address here, since the nonpolyhedral concave regions cannot be decomposed into convex parts. Another approach is to deal with the concave objects directly [7, 9]. However, the method in [7] can only compute the distance between non-polygonal object in 2D space from a specific direction, and it seems difficult to generalize this method to concave objects in 3D space. The method in [9] is for concave polyhedra and not for objects with curved surfaces. In general, there seems no efficient algorithm to deal with intersection checking of concave and nonpolyhedral objects directly.

Therefore, our practical solution is to approximate the precisely grown regions by regions of simpler shapes and then to check the intersections between the simpler regions circumscribing the precisely grown regions with an efficient algorithm. Clearly, if two precisely grown regions intersect, the regions circumscribing them also intersect, but *not* vice versa. Thus the approximation will not miss possible contacts but may introduce more contacts than the actually possible ones. Therefore, an approximating region is preferred to be as close to the precisely grown region as possible. In particular, major concavities of the precisely grown region should be preserved by the approximation. Hence, we still need to deal with concave regions but with simpler shapes.

The spherically-extended polytopes (s-topes) introduced by Hamlin, Kelley, and Tornero [5, 6] provide a good building block for representing the grown regions in our case. Of particular convenience is the efficient algorithm which they developed for intersection checking between s-topes based on a similar algorithm for polytopes introduced by Gilbert et al.[3].

4 Use of the S-tope Model

Instead of representing an object by a polytope model as the convex hull of a set of points, an *s-tope* is the convex hull of a set of spheres [6, 5]. Let $s(\mathbf{c}, r)$ denote a sphere of radius r centered at position $\mathbf{c}$. Then the s-tope of a set of spheres $S = \{s(\mathbf{c}_0, r_0), s(\mathbf{c}_1, r_1), \cdots, s(\mathbf{c}_n, r_n)\}$ is expressed as:

$$CoS = \{s(\mathbf{c}, r) | \mathbf{c} = \sum_{i=0}^{n} \lambda_i \mathbf{c}_i, r = \sum_{i=0}^{n} \lambda_i r_i\},$$

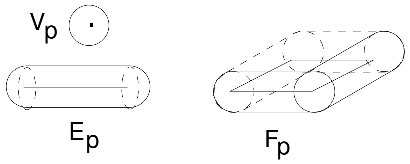

Figure 2: The grown regions V_p, E_p and F_p

where $\sum_{i=0}^{n} \lambda_i = 1, \lambda_i \geq 0$. The spheres in the set S are called the *spherical-vertices* of the s-tope.

4.1 Representing the grown regions V_p, E_p and F_p of a convex polyhedron

With the s-tope model, the grown regions of a vertex, an edge, or a face of a convex polyhedron P by its position uncertainty ϵ_p [12] can be precisely described. The grown region V_p of a vertex v by ϵ_p (Fig. 2a) is simply the s-tope of a single sphere $s(\mathbf{v}, \epsilon_p)$, where $\mathbf{v}$ is the position vector of v. The grown region E_p of an edge e by ϵ_p (Fig. 2b) is the s-tope of two spheres $s(\mathbf{v}_0^e, \epsilon_p)$ and $s(\mathbf{v}_1^e, \epsilon_p)$, where $\mathbf{v}_0^e$ and $\mathbf{v}_1^e$ are the position vectors of the two end-vertices v_0^e and v_1^e of the edge e respectively. The grown region F_p of a face f by ϵ_p (Fig. 2c) is the s-tope of the set of spheres $S = \{s(\mathbf{v}_i^f, \epsilon_p) | i = 1, ..., n_{v/f}\}$, where $\mathbf{v}_i^f$ is the position vector of the ith vertex of face f and $n_{v/f}$ is the number of vertices of the face f.

4.2 Approximating the grown regions V_o, E_o and F_o of a convex polyhedron

As described in [12], the grown regions of a vertex, an edge, or a face of even a convex polyhedron P by its orientation uncertainty ϵ_o are non-polyhedral and nonconvex. However, we can approximate these regions by circumscribing s-topes or unions of s-topes.

Generally, the grown region Q_o of a point q of P by ϵ_o is a spherical segment of one base as shown in Fig. 3, where O denotes the origin of the object frame of P and $\mathbf{q}$ is the distance vector from O to q. The region Q_o is on the sphere centered at O with radius $q = ||\mathbf{q}||$ and bounded by the circle $l(q)$ which is centered at position $\mathbf{q} \cos \epsilon_o$ relative to the object frame of P and with radius $q \sin \epsilon_o$. It is obvious that Q_o is circumscribed by the sphere $s(\mathbf{q} \cos \epsilon_o, q \sin \epsilon_o)$, which is the dotted sphere in Fig. 3. Thus, we use the s-tope of this single sphere to approximate Q_o, and we denote the s-tope approximation as $\tilde{\mathsf{Q}}_o$. For a vertex v of P, we denote the s-tope approximation of its grown

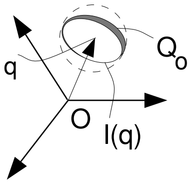

Figure 3: The grown region Q_o of q

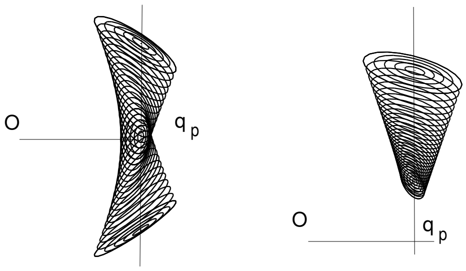

Figure 4: Examples of E_o

region V_o by ϵ_o as $\tilde{\mathsf{V}}_o$.

The grown region E_o of an edge e by ϵ_o is formed by the union of the grown regions of each point on the edge e, and similarly, the grown region F_o of a face f by ϵ_o is formed by the union of the grown regions of each point on the face f. The shape of E_o (or F_o) can be different depending on the relative location of e (or f) with respect to the origin of the object frame O[12]. Particularly, let q_p denotes the projection of O on the line where e lies (or the plane where f lies). If q_p is within e (or f), it will result in a major concavity in the grown region E_o of e (or F_o of f) in addition to the otherwise slightly concave shape (see Fig. 4 for examples of E_o). This phenomenon can be explained intuitively: since the size of the grown region Q_o for each point q on e (or on f) is proportional to the distance from O to q, it reaches minimum where that distance reaches minimum; since such minimum distance occurs at the point q_p, if q_p is an internal point on e (or f), its grown region, which is smaller than the grown regions of the rest of the points on e (or f), results in the major concavity in E_o (or F_o).

Let $\{v_i\}$ denote the set of vertices of e where $i = 1, 2$ (or the set of vertices of f where $i = 1, ..., n_{v/f}$ and $n_{v/f}$ is the number of vertices of f). Let $s_i = s(\mathbf{v}_i \cos\epsilon_o, v_i \sin\epsilon_o)$ denotes the sphere that forms the s-tope $\tilde{\mathsf{V}}_o$ of the i-th vertex. It can be easily proven that the s-tope of the spheres $\{s_i\}$ circumscribes E_o

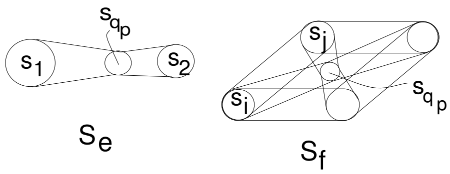

Figure 5: The union S_e and S_f of s-topes

(or F_o).

Moreover, for cases where q_p is within e (or f) so that there is a major concavity in E_o (or F_o), two (or more) s-topes can be used to approximate E_o (or F_o) while preserving that concavity. Specifically, let $s_{q_p} = s(\mathbf{q}_p \cos\epsilon_o, q_p \sin\epsilon_o)$ denote the sphere which forms the s-tope $\tilde{\mathsf{Q}}_o$ of the point q_p. It can be proven that an E_o is circumscribed by the union S_e of the following two s-topes: the s-tope of spheres s_1 and s_{q_p} and the s-tope of spheres s_2 and s_{q_p} (see Fig. 5a). Similarly, it can be proven that an F_o is circumscribed by the union S_f of such s-topes: the s-tope of spheres s_i, s_j, and s_{q_p}, where s_i and s_j ($i \neq j, i, j \in \{1, 2, ..., n_{v/f}\}$) correspond to spheres of two *ajacent* vertices of the face f (see Fig. 5b). Note that the sphere s_{q_p} is smaller than any of the spheres in $\{s_i\}$; this very property preserves the major concavity of E_o (or F_o). Therefore, we use the s-tope of the spheres $\{s_i\}$ to approximate E_o (or F_o) of the cases where q_p is outside e (or f), the union of s-topes S_e to approximate E_o of the cases where q_p is inside e, and the union of s-topes S_f to approximate F_o of the cases where q_p is inside f. We use $\tilde{\mathsf{E}}_o$ and $\tilde{\mathsf{F}}_o$ to denote such approximations of E_o and F_o respectively.

4.3 Modeling the grown regions of a concave polyhedron

What makes a concave polyhedron different from a convex one is that the concave polyhedron has concave faces (i.e. concave polygons) while a convex polyhedron has only convex faces. Thus, the s-tope models of the grown regions of vertices and edges and (convex) faces for a convex polyhedron (described above) also apply to those of a concave polyhedron; only the concave faces of the concave polyhedron should be modeled differently. Our method is to divide a concave polygon into several convex ones and model the grown region of a concave polygon by the union of the grown regions (more precisely, the s-tope models of the grown regions) of those convex patches. In this way, we can well preserve the concavity of the concave polygon.

5 The FAPRIC Algorithm

Now we introduce the algorithm FAPRIC of finding all possible principal contacts between the objects F and H based on intersection checking of their (approximate) grown regions based on s-tope models. The intersection between two grown regions can be efficiently detected by the algorithm described in [5], which we implemented with certain modifications and additions.

The FAPRIC program takes as *input* an **object database** providing geometric and topological data of polyhedra F and H, the location of F and the estimated (or sensed) location of H with respect to a world coordinate system, and the uncertainty bounds ϵ_p and ϵ_o. It produces the set S_{pc} of all possible PCs as *output*. The ideas and methods for identifying possible PCs (as described earlier) are implemented as a rule-base. The rule-base contains conditions which make a PC possible. For example, the following condition is used to check if a PC possibly exists between an edge of F and a face of H:

> If the PC e_i^F-f_j^H possibly exists, then $\mathsf{E}_{p,i} \cap \tilde{\mathsf{F}}_{o,j} \neq \emptyset$ and $\angle(e_i^F, f_j^H) \leq \epsilon_o$,

where $\mathsf{E}_{p,i}$ is the grown region of e_i^F by ϵ_p, $\tilde{\mathsf{F}}_{o,j}$ is the (s-tope approximated) grown region of f_j^H by ϵ_o, and $\angle(e_i^F, f_j^H)$ is the estimated angle between the two features. Conditions for other PCs are in similar forms. Clearly, such conditions are only *necessary* conditions mainly due to the s-tope approximations[2] of the grown regions of features by ϵ_o, which circumscribe the precisely grown regions. Thus, the set S_{pc} obtained by checking if these conditions are satisfied may include PCs *not* possibly present. However, possibly present PCs will not be absent from the set.

It can be shown that the worst-case time complexity of the FAPRIC algorithm is $O(N_v^F \cdot N_v^H \cdot N_{f/v}^F \cdot N_{f/v}^H)$, where N_v^F and N_v^H denote the numbers of vertices and $N_{f/v}^F$ and $N_{f/v}^H$ denote the average numbers of faces which form a vertex of the objects F and H respectively. If the objects H and F are trihedral, i.e., $N_{f/v}^F = N_{f/v}^H = 3$, then the worst-case time complexity of FAPRIC is $O(N_v^F \cdot N_v^H)$.

It should be noted that the FAPRIC program actually takes into account the connectivity among surface features of an object to eliminate redundant checking, which greatly speeds up the process of obtaining S_{pc}. For example, if both vertices of an edge e^F are already checked to be possibly in contact with a face f^H, then FAPRIC considers that e^F-f^H is a possible PC without further checking. Thus, the average time complexity of FAPRIC should be much improved upon the worst-case time complexity. The test results presented in the next section demonstrate that the FAPRIC is very fast and is apt for real-time application.

[2] See [12] or [13] for more discussion.

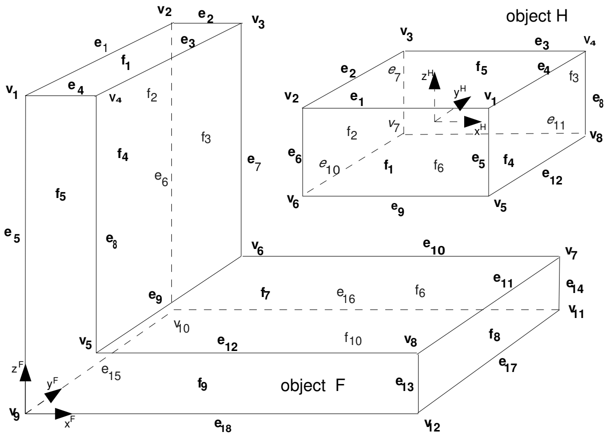

Figure 6: The objects F and H for testing

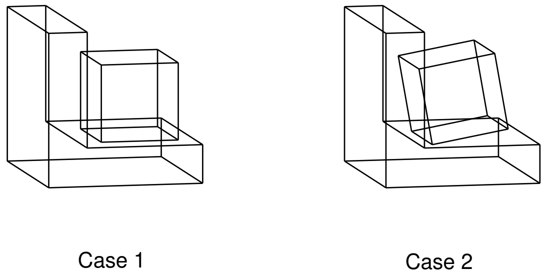

Figure 7: Two configurations for testing

6 Results and Performance

We tested the FAPRIC system with the objects displayed in Fig. 6, where F is concave and H is convex, and their features are labeled. The objects' frames are also shown. Let ${}^F\mathbf{p}_H$ denote the estimated relative position of H with respect to the object frame of F. Let $M_{\phi,\theta,\psi} = M_{z,\phi} M_{y,\theta} M_{x,\psi}$ denote the estimated rotation matrix of H with respect to the frame of F, where ϕ, θ, and ψ are estimated Euler angles. By varying the relative configuration of H with respect to F and the values of ϵ_p and ϵ_o, we created numerous test cases for FAPRIC. In all cases, the FAPRIC gave reasonable results.

Fig. 7 displays two estimated (i.e., sensed) configurations of H with respect to F, which the FAPRIC used as input. Fig. 8 lists the values of ${}^F\mathbf{p}_H$ and $\Theta = [\phi, \theta, \psi]^T$ of the two configurations in Fig. 7 and

	$\mathbf{p}$	$\mathbf{\Theta}$	ϵ_p	ϵ_o	Recognized PCs
1	$\begin{bmatrix} 5 \\ 3 \\ 4.1 \end{bmatrix}$	$\begin{bmatrix} 0^o \\ -3^o \\ 0^o \end{bmatrix}$	0.1	3^o	f_7^F-v_5^H, f_7^F-v_6^H, f_7^F-v_7^H f_7^F-v_8^H, f_7^F-e_9^H, f_7^F-e_{10}^H f_7^F-e_{11}^H, f_7^F-e_{12}^H f_7^F-f_6^H
			0.1	0^o	f_7^F-v_6^H, f_7^F-v_7^H, f_7^F-e_{10}^H
			0	3^o	f_7^F-v_6^H, f_7^F-v_7^H, f_7^F-e_9^H f_7^F-e_{10}^H, f_7^F-e_{11}^H, f_7^F-f_6^H
			0	0^o	$\emptyset$
2	$\begin{bmatrix} 4.3 \\ 3 \\ 4.3 \end{bmatrix}$	$\begin{bmatrix} 0^o \\ -10^o \\ 0^o \end{bmatrix}$	0.1	3^o	f_4^F-v_2^H, f_4^F-v_3^H, f_7^F-v_6^H f_7^F-v_7^H, f_4^F-e_2^H, f_7^F-e_{10}^H

Figure 8: Inputs and results

the respective output sets of possible PCs by FAPRIC under different uncertainty values. Observe that for Case 1, if there were no uncertainty, based on the estimated location of H with respect to F, the two objects would not even touch. However, the larger the uncertainties (ϵ_p and ϵ_o) in the estimated location, the more possible PCs there will be between the two objects as the result. For Case 2, varying the value of ϵ_o from $1°$ to $3°$ happens to generate the same result.

We also tested the speed of the FAPRIC system. $1,000$ different configurations of the object H with respect to the object F and their location uncertainties were randomly generated by uniformly distributing the values of x, y, z (of ${}^F\mathbf{p}_H$), $\phi, \theta, \psi, \epsilon_p, \epsilon_o$ in the ranges of $[-4, 12], [-4, 12], [-4, 12]$, $[-\pi, \pi], [-\pi, \pi], [-\pi, \pi]$, $[0, 0.2], [0, 2^o]$ respectively. The average CPU time that the FAPRIC used to generate the output set of possible PCs per case was 0.674 seconds on a SUN SPARC IPX (during which, an average of $1,341$ calls to the s-tope procedure for intersection checking were made).

7 Conclusions

We presented an implemented algorithm FAPRIC for recognizing all possible PCs between two arbitrary polyhedra in the presence of location uncertainties. The algorithm is efficient and apt for real-time use. The output can be used as input for further reasoning [14] and further recognition, such as by vision [11] or force [1, 4] sensing. More improvement on the speed of FAPRIC can be achieved by localizing the features of one object that may touch the other object based on the objects' estimated locations and by parallel processing. It may also be desirable to investigate better models for approximating the grown regions of features and/or faster algorithms for intersection checking.

References

[1] R. Desai, "On Fine Motion in Mechanical Assembly in Presence of Uncertainty," PhD thesis, the Univ. of Michigan, 1989.

[2] E. Gilbert, D. Johnson, "Distance Functions and Their Application to Robot Path Planning in the Presence of Obstacles," *IEEE J. R&A*, RA-1, pp. 21-30, Mar. 1985.

[3] E. Gilbert, D. Johnson, S. Keerthi, "A Fast Procedure for Computing the Distance between Complex Objects in Three-dimensional Space," *IEEE J. R&A*, 4(2), pp. 193-203, April 1988.

[4] A. Farahat, B. Graves, J. Trinkle, "Identifying Contact Formations in the Presence of Uncertainty," submitted to *IEEE Trans. R&A*, 1994.

[5] G. Hamlin, R. Kelley, J. Tornero, "Efficient Distance Calculation Using the Spherically-Extended Polytope (S-tope) Model," IEEE ICRA, Nice, France, pp. 2502-2507, May 1992.

[6] J. Tornero, G. Hamlin, R. Kelley, "Spherical-Object Representation and Fast Distance Computation for Robotic Applications," IEEE ICRA, Sacramento, pp. 1602-1607, April 1991.

[7] F. Thomas, C. Torras, "Interference Detection between Non-Convex Polyhedra Revisited with a Practical Aim," IEEE ICRA, San Diego, pp. 587-594, May 1994.

[8] M. Spreng, "Dealing with Unexpected Situations During the Execution of Robot Motions," IEEE ICRA, Sacramento, pp. 64-69, April 1991.

[9] K. Sridharan, H. Stephanou "Algorithms for Rapid Computation of Some Distance Functions between Objects for Path Planning," IEEE ICRA, San Diego, pp. 967-972, May 1994.

[10] J. Xiao, "Automatic Determination of Topological Contacts in the Presence of Sensing Uncertainties," IEEE ICRA, Atlanta, pp. 65-70, May 1993.

[11] J. Xiao, M. Su, "Automatic Verification of Topological Contacts by Vision," IEEE ISIC, Chicago, pp. 314-319, August 1993.

[12] J. Xiao, "Towards Obtaining All Possible Contacts - Growing a Polyhedron by Its Location Uncertainty," IROS'94, Munich, Germany, pp. 1788-1795, September 1994.

[13] J. Xiao, L. Zhang, "Towards Obtaining All Possible Contacts - Growing a Polyhedron by Its Location Uncertainty," submitted to the *IEEE Trans. R&A*, 1994.

[14] J. Xiao, L. Zhang, "A General Strategy to Determine Geometrically Valid Contact Formations from Possible Contact Primitives," IEEE ICRA'95, Nagoya, Japan.