Proceedings of the 2000 IEEE
International Conference on Robotics & Automation
San Francisco, CA • April 2000

Computing Signed Distances between Free-Form Objects

Federico Thomas†, Colin Turnbull‡, Lluis Ros†, and Stephen Cameron‡

‡Oxford University Computing Laboratory
Parks Road, Oxford OX1 3QD, UK
{ct,cameron}@comlab.ox.ac.uk

†Institut de Robòtica i Informàtica Industrial (CSIC – UPC)
Gran Capità 2-4, 2 planta, 08034 Barcelona. SPAIN
{fthomas,llros}@iri.upc.es

Abstract

Given two sculptured objects, described by a collection of rational Bézier patches, we propose an algorithm that provides the distance between them if the objects are not intersecting, or a measure of penetration otherwise. The algorithm extends the upper-lower bound subdivision approach to the computation of the nearest points by considering the relative orientations between the subdivided patches.

All required geometric constructions can be described as rational Bézier patches so that their control points can be precomputed from those of the original patches. Additional operations have been designed to exhibit linear complexity with the total number of involved control points.

1 Introduction

Free-form objects, also called sculptured objects, are widely used in scientific and engineering applications. Propeller and turbine blades, or ships, automobiles, and aircraft bodies are good examples of this kind of objects.

Free-form object surface representations can be classified into two categories: parametric surfaces and implicit surfaces. For the first category Bézier patches, B-spline patches, and NURBS are widely used for the representation, design, and data exchange of geometric information in most industrial applications. We assume that our objects are represented by Bézier patches. This assumption is not too restrictive because surfaces expressed as NURBS or B-splines can be decomposed into rational Bézier patches [11, p. 162].

Different applications involving free-form objects need fast answers to separation or penetration distance queries. Most of the work on these computations comes from the robotics, computer graphics, and computational geometry communities. In robotics, distance information is useful, for example, for computing interaction forces and penalty functions in robot motion planning. In computer graphics, minimum distance computations play roles in physical simulation and model prototyping. Unfortunately, most attained results are limited to separation distances and/or to convex objects [12].

A solution to the problem of computing minimum distances between sculptured objects based on parametric surfaces is to resort to conversion to polyhedral forms. However, when accuracy is a must, such an approach may not be adequate, as it is possible that the actual surfaces intersect with each other, but their polyhedral approximations do not and vice versa.

The exact computation of the minimum distance for parametric models can be translated into a minimization or root finding method for a system of equations that describe the conditions for the minimum distance [23]. Alternative approaches keep the geometric flavor of methods for polyhedral models, thus avoiding many of the numerical issues that complicate the numerical methods [19, 9, 10].

Contrary to the computation of distances between object models, past research in the computation of penetration distances is quite sparse and limited to convex objects and, to our knowledge, no results have been reported for free-form objects.

A widespread approach to compute the minimum distance between two objects with complex geometries consists of (a) recursively decomposing them up to a given resolution level; (b) generating pairs of candi-

date regions possibly containing the nearest points; and (c) removing pairs that have lower bounds in their distance greater than the upper bound on the minimum distance between both objects. This basic approach neglects the fact that the closest points of the two surfaces satisfy the necessary condition that their normals are aligned and opposite in direction. This paper shows how three important benefits are obtained by introducing this constraint into the basic subdivision approach, namely: (a) the convergence to the nearest points is speeded up; (b) a measure of penetration distance, in the case that both objects intersect, can be obtained; and (c) degenerate cases are treated more efficiently.

This paper is organized as follows. Section 2 reviews some properties of Bézier patches needed throughout this paper. Section 3 shows how the computation of points with collinear normals is related to the concept of convolution between surfaces. Local methods, needed to refine approximate nearest points, are briefly discussed in section 4. The upper-lower subdivision strategy is reviewed in section 5. Section 6 describes how the collinearity condition is introduced in the subdivision strategy and, finally, section 7 contains the conclusions and points that deserve further attention.

2 Background

Unlike implicit surfaces, parametric surfaces are not generally closed manifolds so that they do not represent a complete solid model, but rather a piece of its boundary. Thus, our solid models are assumed to be compact subsets of $\Re^3$ whose boundary is described by a collection of untrimmed rational Bézier patches, $\mathbf{F}^i(s,t)$, defined in the domain $(s,t) \in [0,1] \times [0,1]$, for $i = 1 \ldots N$. Models are also assumed to be smooth, i.e. normals are continuous between adjacent patches. Although the general case can be tackled by considering separately the boundary curve between patches, this simplification makes the description given below much simpler. Moreover, each patch equation has been chosen such that the normal vector points to the exterior of the object.

Each Bézier patch is a piecewise rational surface in $\Re^3$ of the homogeneous form:

$$\mathbf{F}(s,t) = (X(s,t), Y(s,t), Z(s,t), W(s,t)).$$

Geometrically, a patch of degree $m \times n$ is represented in terms of $(m+1) \times (n+1)$ control points and a linear combination of the standard Bernstein basis functions:

$$\mathbf{F}(s,t) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{V}_{ij} B_{im}(s) B_{jn}(t),$$

where $\mathbf{V}_{ij} = (x_{ij}, y_{ij}, z_{ij}, w_{ij})$ are the control point homogeneous coordinates. The entire patch is contained in the convex hull of the control points either in the homogeneous [22] or Euclidean [6] space.

A Bézier patch of degree $m \times n$ defined in the domain $[0,1] \times [0,1]$ can also be defined over any domain $[a,b] \times [c,d] \subset [0,1] \times [0,1]$. The part of the patch that corresponds to $[a,b] \times [c,d]$ can be described as a Bézier patch of the same degree whose control points can be directly obtained from those of the original Bézier patch. This process is referred to as subdivision [6].

Given a Bézier patch, $\mathbf{F}(s,t)$, of degree $m \times n$, its pseudo-normal patch is defined as:

$$\mathbf{N}_{s,t} = \mathbf{F}_s(s,t) \times \mathbf{F}_t(s,t)$$

where $\mathbf{F}_s$ and $\mathbf{F}_t$ are the partial derivative vectors. The projection of the pseudo-normal patch onto the unit sphere is the Gauss map of $\mathbf{F}(s,t)$. The pseudo-normal patch is also a Bézier patch and its parametric degree is $(2m-1) \times (2n-1)$ for a polynomial patch [21] (i.e., $w_{ij} = 1$, $\forall i,j$) and $(3m-2) \times (3n-2)$ for a rational patch [18]. The control points of the pseudo-normal patch can be obtained directly from those of the original patch.

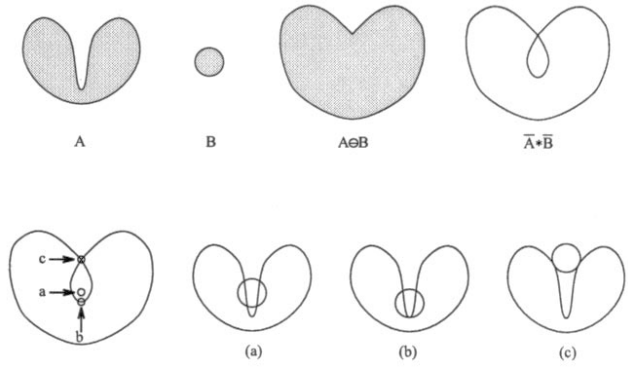

Fig. 1 *Objects A and B, their Minkowski difference, $A \ominus B$, and their surface convolution, $\overline{A} * \overline{B}$. The nearest configuration on the convolution to configunation (a) does not give the minimum translation that separates A and B.*

3 Minkowski differences and convolutions

Let A and B denote two objects, compact subsets of $\Re^3$, and $\bar{A}$ and $\bar{B}$ their boundaries, respectively. The

Minkowski difference between A and B is defined as:

$$A \ominus B = \{\mathbf{P} \in \Re^3 \mid \mathbf{P} = \mathbf{P}_a - \mathbf{P}_b, \mathbf{P}_a \in A, \mathbf{P}_b \in B\},$$

and the convolution of their boundaries as:

$$\overline{A} * \overline{B} = \{\mathbf{P} \in \Re^3 \mid \mathbf{P} = \mathbf{P}_a - \mathbf{P}_b, \mathbf{P}_a \in \overline{A}, \mathbf{P}_b \in \overline{B}, \mathbf{N}(\mathbf{P}_a) \times \mathbf{N}(\mathbf{P}_b) = 0, \mathbf{N}(\mathbf{P}_a) \cdot \mathbf{N}(\mathbf{P}_b) < 0\},$$

where $\mathbf{N}(\mathbf{P})$ denotes the normal at point $\mathbf{P}$ on the boundary of either A or B.

If we consider object B to be free to move with fixed orientation, the Minkowski difference is a set containing all the translations that bring B to intersect with A, and the convolution all the translations that bring B to touch A so that the normals at the contact point are exactly opposite.

The closest point from the Minkowski difference boundary, $\overline{A \ominus B}$, to the origin gives us the minimum distance between A and B. If both objects intersect, the origin is inside the Minkowski difference and the obtained distance can be interpreted as a penetration distance [3].

When A and B are convex, the boundary of their Minkowski difference coincide with the convolution of their boundaries. In general, it was proven in [1] that

$$\overline{A \ominus B} \subseteq \bar{A} * \bar{B} \subset A \ominus B.$$

When A and B are not intersecting, the nearest point on the boundary of the Minkowski difference to the origin is also on the convolution. Unfortunately, when both objects intersect the nearest point of the convolution to the origin is not necessarily on the boundary of the Minkowski difference. As an example, consider the objects A and B whose Minkowski difference and boundary convolution appear in *fig. 1.* While the nearest point of the convolution to the origin (represented by point a) is point b, the nearest point to the boundary of the Minkowski difference is point c; the point that provides us the actual penetration distance.

As a conclusion, when we search for the nearest points between two surfaces such that their normals are collinear, we get the exact separation distance when they do not intersect but, in general, a lower bound of the penetration distance when they intersect, unless both surfaces are convex or locally convex.

4 Local methods

For a point $\mathbf{P}$, the closest point on a surface $\mathbf{F}$ is among those satisfying the equation

$$\mathbf{H} = (\mathbf{F} - \mathbf{P}) \times (\mathbf{F}_u \times \mathbf{F}_v) = \mathbf{0}.$$

If $\mathbf{F}$ is a rational Bézier surface, $\mathbf{H}$ can be itself expressed as a rational Bézier patch, so that the problem becomes that of a point inversion problem [11, p.229]. Unfortunately, inverse point mapping is subject to numerical instability and it is in general quite costly.

When we deal with two surfaces, $\mathbf{F}$ and $\mathbf{G}$, instead of a point and a surface, the following two perpendicularity conditions must be simultaneously satisfied:

$$(\mathbf{F} - \mathbf{G}) \times (\mathbf{F}_u \times \mathbf{F}_v) = (\mathbf{G} - \mathbf{F}) \times (\mathbf{G}_u \times \mathbf{G}_v) = 0$$

A solution can be obtained by repeatedly finding the closest point on alternating surfaces, a technique equivalent to the alternating orthogonal projection described in [15]. Unless $\mathbf{F}$ and $\mathbf{G}$ are convex, there is no guarantee of convergence and, in the best of the cases, just a local minimum is obtained. Approximations can be obtained by sampling, a process which can not provide full assurance that the nearest points have been found.

If we have a good approximation of the nearest points between two surfaces, we can always rely on a local numerical technique to refine them. The same techniques used in surface/surface intersection to relax approximate points onto intersections can be adapted to obtain the local nearest points [2, 15]. They employ some variation of the Newton-Raphson iteration or numerical optimization. Also, optimization techniques used for the computation of the nearest points between convex polyhedra can be adapted.

In our implementation we have used the Newton method described in [2, p. 266] because of its quadratic convergence.

5 The upper-lower bound subdivision strategy

For complex concave geometries a useful approach to compute the minimum distance between objects has been to surround them with hierarchies of bounding volumes. These volumes can be pruned by establishing and refining an upper bound on the minimum distance between both objects then removing bounding volumes that have lower bounds on their distance greater than the current upper bound [9]. As the algorithm descends the bounding volume hierarchies, the lower bound tighten to the underlying geometry. The most common types of bounding volumes are box and sphere but more sophisticated ones, such as spherical shells [14], are possible.

In our case, an alternative to pre-compute hierarchies of bounding volumes is the subdivision of the patches for which we want to obtain the nearest points. Then, bounds can be obtained using the convex hull

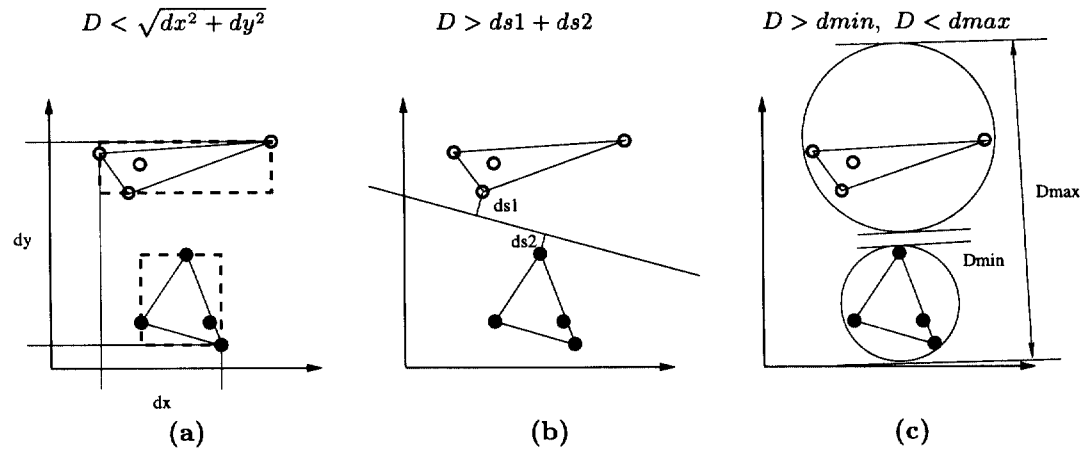

Fig. 2 *Computing bounds in linear time:* **(a)** *an upper bound using bounding boxes;* **(b)** *a lower bound using a randomized separating plane; and* **(c)** *an upper and lower bound using enclosing spheres. Bounds can always be combined.*

property of Bézier patches. Since this operation at each subdivision is quite expensive, bounding volumes instead of the convex hulls themselves should be used. As a consequence of this approximation, more subdivisions are needed to obtain good distance bounds.

Three-dimensional bounding volume is a ubiquitous need in a wide variety of applications and has received a considerable attention. Unfortunately, the designed linear time algorithms for optimal bounding volumes require mathematical sophistication, their implementation is not easy, and furthermore the constants of proportionality in their complexities are generally large. This is why in practice only near-optimal bounding regions are computed at run time. In our implementation we use the near-optimal spheres described in [24], but obviously other alternatives, and even combinations of them, are possible (*fig. 2*).

The first step of the subdivision approach consists of obtaining the distance bounds between all pairs of patches. To avoid the quadratic complexity of this initial step we use a variation of the sweep-and-prune technique described in [4] to find the initial set of candidates in nearly linear time.

The recursive subdivision of two Bézier patches first requires selecting the necessary level of subdivision at which a point on the subdivided patches can be refined by local methods to give the exact solution. Fortunately, the number of subdivisions needed to give an excellent degree of accuracy is, in practice, very small.

Subdivision is robust but computationally expensive compared to solutions based on a hierarchy of bounding volumes. Nevertheless, it allows us to easily introduce the collinearity condition as explained in the next section.

6 Searching for collinear normals

The closest points of two surfaces satisfy the necessary condition that their normals are aligned and opposite in direction. From the surface intersection literature, it is well known that for two surfaces to intersect in a loop there must be a collinear normal between them [17]. Thus, much of the literature in this area has been very helpful in coming up with a technique that combines this necessary condition with the basic upper/lower bound subdivision algorithm.

As a consequence of the collinearity condition, if patches $\mathbf{F}^1(s,t)$ and $\mathbf{F}^2(s,t)$ contain the nearest points, then the projections onto the unit sphere of $\mathbf{N}^1$ and $-\mathbf{N}^2$ overlap. This condition can be tested by computing the control points of each of the pseudo-normal patches and checking, using linear programming, whether they can be separated by a plane passing through the origin. If this test fails, the corresponding Gaussian maps might overlap, and the corresponding patches can be further subdivided. Since computing the control points of the orientation patch is linear with the total number of original control points and determining the existence of a separating plane is also linear, the overall complexity of this operation is linear. Although this is clear and simple, a different approach has been finally adopted for the reason given below.

Although a tight bound on the orientations of the normals to $\mathbf{F}$ can be obtained by considering the controls points of $\mathbf{N}_{s,t}$, a cheaper solution consists in individually bounding the orientations of $\mathbf{F}_s$ and $\mathbf{F}_t$ and then computing the region that contains any cross product between a vector in both regions.

Based on the fact that $\mathbf{F}_s$ is a rational Bézier patch of degree $(2m-2) \times 2n$, a near-optimal algorithm to obtain a cone that bounds the orientations of $\mathbf{F}_s$ in linear time with the number of control points of $\mathbf{F}$ is described in [14] following the approach given in [17]. The algorithm updates a cone in a single iteration over the $(2m-1) \times (2n+1)$ control points of $\mathbf{F}_s$. Each control point is checked to see if it lies within the cone. If not, a new cone is formed which is the smallest cone containing the old cone and the new control point (*fig. 3a*). Any cone containing a single control point can be used as initial cone.

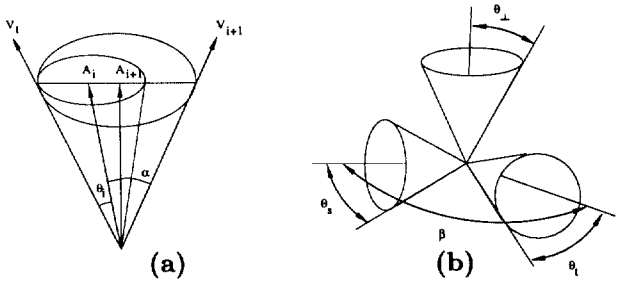

Fig. 3 *Bounding cone creation* (a), *and normal cone with s and t cones* (b) *(adapted from* [17]*).*

The cone bounding the resulting control points, or s cone, will bound the direction of all the iso-parameter curves on the surface. If this cone is translated so that its vertex lies on any point on the surface, the cone will bound the curve which passes through that point. Given the s and t cone, the cone which contains any cross product between a vector in the s cone and the vector in the t cone is computed. This leads to a conservative normal cone whose axis is mutually perpendicular to the s and t cone axes (*fig. 3b*). From spherical trigonometry, we can compute its half angle $\theta_\perp$ as:

$$\theta_\perp = \sin^{-1}\left(\frac{\sqrt{\sin^2\theta_s + 2\sin\theta_s \sin\theta_t \cos\beta + \sin^2\theta_t}}{\sin\beta}\right)$$

provided that $\beta < \theta_s + \theta_t$.

As recognized in [17], the normal bounding cone just discussed has the virtue that it can be computed very quickly, but it bounds the normal vectors rather loosely. A trade-off between the number of subdivisions and cost of the operations carried out in each subdision arise, but preliminary experiments show that this solution provides a good balance.

Now, depending on the relative orientation between patches, we can assign signed bounds. Consider the patches $\mathbf{F}^1$ and $\mathbf{F}^2$. Let $\mathcal{S}(\mathbf{F}^1)$ and $\mathcal{S}(\mathbf{F}^2)$ denote the smallest spheres bounding their control points (and thus the patches themselves). Also let $\mathcal{C}(\mathbf{N}^1)$ and $\mathcal{C}(\mathbf{N}^2)$ denote the cones bounding their associated pseudo-normal patches and $\mathcal{C}(\mathbf{N}^{12})$ the cone bounding all possible vectors emanating from a point on $\mathbf{F}^1$ and pointing to a point on $\mathbf{F}^2$ (which can be approximated from $\mathcal{S}(\mathbf{F}^1)$ and $\mathcal{S}(\mathbf{F}^2)$). Then, the distance D between $\mathbf{F}^1$ and $\mathbf{F}^2$ will be bounded as follows:

- If spheres do not intersect, i.e. $\mathcal{S}(\mathbf{F}^1) \cap \mathcal{S}(\mathbf{F}^2) = \emptyset$, then

  (1) If $\mathcal{C}(\mathbf{N}^1) \cap \mathcal{C}(-\mathbf{N}^2) \cap \mathcal{C}(\mathbf{N}^{12}) \neq \emptyset$ then the sphere-sphere distance bounds the solution:

  $$D \in [d - r_a - r_b, d + r_a + r_b]$$

  (2) If $\mathcal{C}(\mathbf{N}^1) \cap \mathcal{C}(\mathbf{N}^2) \cap \mathcal{C}(\mathbf{N}^{12}) \neq \emptyset$ then we obtain the negative bounds:

  $$D \in [-d - r_a - r_b, -d + r_a + r_b]$$

  (3) If $\mathcal{C}(\mathbf{N}^1) \cap \mathcal{C}(-\mathbf{N}^2) \cap \mathcal{C}(\mathbf{N}^2) \cap \mathcal{C}(\mathbf{N}^{12}) \neq \emptyset$ then we obtain the largest interval:

  $$D \in [-d - r_a - r_b, d + r_a + r_b]$$

- If spheres intersect, i.e. $\mathcal{S}(\mathbf{F}^1) \cap \mathcal{S}(\mathbf{F}^2) \neq \emptyset$, then

  (4) If $\mathcal{C}(\mathbf{N}^1) \cap \mathcal{C}(-\mathbf{N}^2) \neq \emptyset$, then as case (3):

  $$D \in [-d - r_a - r_b, d + r_a + r_b]$$

where r_a and r_b are the radii of $\mathcal{S}(\mathbf{F}^1)$ and $\mathcal{S}(\mathbf{F}^2)$, respectively, and d the distance between their centres. If none of the four above situations arise, the pair of patches formed by $\mathbf{F}^1$ and $\mathbf{F}^2$ can be removed from the list of candidates.

Now, the subdivision strategy proceeds as explained above, the only difference being that bounds are signed and the sought points, which are bound to satisfy the collinearity condition, must realize the minimum absolute distance. Subdivision ends when the computed spheres or orientation cones are small enough. Thus, the introduction of orientation cones also avoids the unnecessary subdivision carried out by the basic upper-lower subdivision strategy in many degenerate cases.

7 Conclusions

The typical degree of the patches involved in most model descriptions are bicubic (16 control points). For example, a torus is described using eight rational Bézier patches. The general expressions for the control points of subdivided patches and control points of the derivative patches – which can be found in [6] and [18], respectively – have been particularized to bicubic patches.

The whole algorithm is being implemented using ACIS [5]. Preliminary results confirm that the introduction of the collinearity condition speeds up the convergence of the subdivision approach and compares favorably to the approaches based on hierarchical representations, at least in degenerate configurations.

Finally, the way the orientations cones intersect provides valuable information on where to subdivide the patches. Adaptive subdivisions based on this or other informations is a point that deserves further attention.

Acknowledgments

F. Thomas received financial support from the Catalan Goverment during his stay in the Oxford University Computing Laboratory. The work at Oxford is supported by EPSRC grant number GR/L32408.

References

[1] C. Bajaj and M. Kim, "Generation of Configuration Space Obstacles: Moving Algebraic Surfaces," *The International Journal of Robotics Research*, Vol. 9, No. 1, pp. 92-112, 1990.

[2] R.E. Barnhill and S.N. Kersey, "A Marching Method for Parametric Surface/surface Intersection," *Computer Aided Geometric Design*, Vol. 7, pp. 257-280, 1990.

[3] S.A. Cameron and R.K. Culley, "Determining the Minimum Translational Distance between two Convex Polyhedra," *IEEE Int. Conf. on Robotics and Automation*, Vol. 1, pp. 591-596, 1986.

[4] J. Cohen, M. Lin, D. Manocha, and M. Ponamgi, "I-COLLIDE: An Interactive and Exact Collision Detection System for Large-scale Environments", *ACM Interactive Graphics Conference*, pp. 189-196, 1995.

[5] J. Corney, "3D Modeling with the ACIS Kernel and Toolkit", John Wiley, 1997.

[6] G. Farin, *Curves and Surfaces for Computer Aided Geometric Design. A Practical Guide*, Academic Press, San Diego, CA, 1993.

[7] E.G. Gilbert and C.J. Ong, "New Distances for the Separation and Penetration of Objects," *IEEE Int. Conf. on Robotics and Automation*, Vol. 1, pp. 579-586, 1994.

[8] M. Hughes, C. DiMattia, M.C. Lin, and D. Manocha, "Efficient and Accurate Interference Detection for Polynomial Deformation," *IEEE Conf. on Computer Animation*, pp. 155-166, 1996.

[9] D.E. Johnson, and E. Cohen, "A Framework for Efficient Minimum Distance Computations," *IEEE Int. Conf. on Robotics and Automation*, pp. 3678-3684, 1998.

[10] D.E. Johnson, and E. Cohen, "Bound Coherence for Minimum Distance Computations," *IEEE Int. Conf. on Robotics and Automation*, pp. 1843-1848, 1999.

[11] L. Piegl, and W. Tiller, *The NURBS Book*, Springer-Verlag, Berlin, 1997.

[12] P. Jiménez, C. Torras, and F. Thomas, "Collision Detection Algorithms for Motion Planning", in *Robot Motion. Planning and Control*, J-P. Laumond (editor), Lecture Notes in Control and Information Sciences, Vol. 229, pp. 305-344, Springer Verlag, 1998.

[13] M. Kohler and M. Spreng, "Fast Computation of the C-Space of Convex 2D Algebraic Objects," *The International Journal of Robotics Research*, Vol. 14, No. 6, pp. 590-608, 1995.

[14] S. Krishnan, A. Pattekar, M.C. Lin, and D. Manocha, "Spherical Shell: A Higher Order Bounding Volumen for Fast Proximity Queries," in *Proc. of the Third Int. Workshop on Algebraic Foundations of Robotics*, 1998.

[15] A. Limaiem and F. Trochu, "Geometric Algorithms for the Intersection of Curves and Surfaces," *Computers and Graphics*, Vol. 19, No. 3, pp. 391-403, 1995.

[16] M.C. Lin and D. Manocha, "Interference Detection between Curved Objects for Computer Animation," In *Models and Techniques in Computer Animation*, pp. 43-57. Springer-Verlag, 1993.

[17] T.W. Sederberg, and R.J. Meyers, "Loop Detection in Surface Patch Intersection," *Computer Aided Geometric Design*, Vol. 5, pp. 161-171, 1988.

[18] T. Saito, G.J. Wang, and T.W. Sederberg, "Hodographs and Normals of Rational Curves and Surfaces," *Computer Aided Geometric Design*, Vol. 12, pp. 417-430, 1995.

[19] C. Turnbull and S. Cameron, "Computing Distances between NURBS-defined Objects," *IEEE Int. Conf. on Robotics and Automation*, pp. 3685-3691, 1998.

[20] G.-J. Wang, T.W. Sederberg, and T. Saito, "Partial Derivatives of Rational Bézier Surfaces," *Computer Aided Geometric Design*, Vol. 14, pp. 377-381, 1997.

[21] Y. Yamaguchi, "Bézier Normal Vector Surface and Its Applications," *IEEE Int. Conf. on Shape Modeling and Applications*, pp. 26-35, 1997.

[22] A. Yamada, and Y. Yamaguchi, "Homogeneous Bounding Boxes as Tools for Intersection Algorithms of Rational Bézier Curves and Surfaces," *Visual Computer*, Vol. 12, pp. 202-214, 1996.

[23] J. Zhou, E.C. Sherbrooke, and N.M. Patrikalakis, "Computation of Stationary Points of Distance Functions," *Engineering with Computers*, Vol. 9, No. 4, pp. 231-246, 1993.

[24] X. Wu, "A Linear-Time Simple Bounding Volume Algorithm," in *Graphics Gems III*, pp. 301-306, Academic Press, 1992.