# A Higher Dimensional Formulation for Robust and Interactive Distance Queries

Joon-Kyung Seong David E Johnson Elaine Cohen

School of Computing, University of Utah

## Abstract

We present an efficient and robust algorithm for computing the minimum distance between a point and freeform curve or surface by lifting the problem into a higher dimension. This higher dimensional formulation solves for all query points in the domain simultaneously, therefore providing opportunities to speed computation by applying coherency techniques. In this framework, minimum distance between a point and planar curve is solved using a single polynomial equation in three variables (two variables for a position of the point and one for the curve). This formulation yields two-manifold surfaces as a zero-set in a 3D parameter space. Given a particular query point, the solution space's remaining degrees-of-freedom are fixed and we can numerically compute the minimum distance in a very efficient way. We further recast the problem of analyzing the topological structure of the solution space to that of solving two polynomial equations in three variables. This topological information provides an elegant way to efficiently find a global minimum distance solution for spatially coherent queries. Additionally, we extend this approach to a 3D case. We formulate the problem for the surface case using two polynomial equations in five variables. The effectiveness of our approach is demonstrated with several experimental results.

**CR Categories:** I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Splines

**Keywords:** minimum distance, dimensionality lifting, spline models, problem reduction scheme

## 1 Introduction

Minimum distance queries on computer models are one of the most important geometric operations in simulation [Baraff 1990], haptics [II et al. 1997], robotics [Quinlan 1994], registration [Pottmann et al. 2003], and distance volume computation [Breen et al. 1998]. Often, efficiency and robustness are important for these applications, yet prior formulations have not been able to combine the efficiency of numerical solutions with robustness and global convergence. In this article, we develop an algorithm for reliable and efficient minimum distance queries from a point to a spline model by casting the problem into a higher-dimensional space parameterized not only by the curve or surface, but also by potential positions of the query point.

The minimum distance between a point and a spline model can be computed either symbolically or numerically. Symbolic approaches are robust and can find a global minimum solution but are relatively slow. Because the underlying equations change when the query point changes, symbolic approaches have not been able to exploit spatially coherent queries to accelerate their solution. On the other hand, numerical methods depend on spatial coherency to yield rapid solutions to distance queries, but often can be unreliable since they typically search only for local solutions and miss global solutions that may "jump" from one branch of the model to another. In this paper, a symbolic-numeric hybrid approach is presented to exploit both advantages.

Our higher-dimensional approach symbolically represents the distance between a spline model and all query points in a bounded domain. Given a planar curve $C(t)$ and a parameterization of all possible points in the plane $P(x,y)$, minimum distance queries between them are solved using a single polynomial equation. Using this single equation in three variables, the two-dimensional solution space is constructed as a zero-set in the *xyt*-parameter space. A parameter point $(x,y,t)$ located on the solution space is mapped to a distance extrema for $C(t)$ in the real space. Thus, a search for the distance extrema can be reduced to finding a solution point on the two-dimensional zero-sets in the parameter space.

After performing one minimum distance query using a symbolic approach, repeated queries can be updated using fast numerical methods. Since we have a two-dimensional solution space implicitly defined by a single polynomial equation, spatially coherent queries can be solved by numerically tracing the solution manifolds. In the parameter space, this marching technique computes solution points for the next query point based on the previous ones, which results in a very fast computation of minimum distances.

Local searches using the numerical tracing method may not guarantee that they find all the solutions, one of which may be the global minimum. A new component of the solution space may appear or an existing one disappear when the point encounters a global changes in the higher-dimensional topological structure. We recast the problem of analyzing global topology of the solution space to that of solving two polynomial equations in the parameter space – this part of the algorithm is further accelerated through preprocessing of the higher-dimensional space. This topology information supports the local numerical method to efficiently find a global solution for spatially coherent queries.

The approach developed for curves in the plane can be extended to freeform surfaces in a 3D space. For the surface case, a higher-dimensional solution space is implicitly defined by two polynomial equations in five variables. Experimental results show the robustness of the approach and the speed advantage for coherent queries. We are able to achieve simultaneous tracking of multiple extrema for global minimum distance thousands of times per second for curves and hundreds of times per second for surfaces.

The rest of this paper is organized as follows. In Section 2, some related works are discussed. Section 3 presents an algorithm for computing minimum distance between a point and planar curve, an approach that is based on the dimensionality lifting scheme in the

parameter space. In Section 4, the minimum distance algorithm is extended to freeform surfaces in a 3D space. Some examples are presented in Section 5, and finally, in Section 6, this paper is concluded.

## 2 Related Works

Distance query solution methods tend to vary depending on the type geometric primitive being used. For discrete geometry, such as triangular models, the predominant approaches accelerate queries using bounding volume hierarchies. In this case, research has focused on more efficient bounding primitives. For continuous geometry, such as spline surfaces, the distance equations can be directly formulated and solution approaches have used numerical or symbolic techniques to find distance minima.

### 2.1 Polygonal Models

An early hierarchical approach to distance queries used bounding spheres to prune away portions of the model further away than an upper bound on distance [Quinlan 1994]. In [Larsen et al. 2000], swept spheres volumes proved more efficient at pruning than spheres for near contact cases. More recently, [Ehmann and Lin 2001] has used temporal coherence on hierarchies of convex surface patches to accelerate distance queries.

These techniques all used Euclidean space bounds to find a global minimum. Hierarchical bounds on surface normals were used in [Johnson and Cohen 2001] to efficiently find local minima in distance for polygonal models as well as spline models [Johnson and Cohen 2005].

### 2.2 Spline Models

The basic approach of formulating equations that reflect the extrema of a distance function have been well-known and available in textbooks for some time [Mortensen 1985]. Research issues have been in improving robustness and speed of convergence. Baraff used distance measures on convex smooth models as an efficient collision test [Baraff 1990]. Snyder used an interval Newton's method to robustly update model penetration [Snyder 1993b]. A combination of numerical methods solved for the closest point on a space curve in [Wang et al. 2002].

Subdivision of constraint spaces have been used as more robust, yet slower, alternatives to numerical methods. Interval methods searched the four dimensional parameter space representing the distance between two parametric surfaces in [Snyder 1993a]. Elber [Elber 1992] represented the distance equation and its derivatives as NURBS surfaces and used subdivision along with numerical methods to search for solutions.

### 2.3 Distance Transform

The distance transform is a mapping between point position and a minimum distance to a model. Often, this transform is represented as a distance volume, which is a discrete sampling of point positions with their minimum distances to a model stored in a volumetric array. Distance volumes have the advantage of being extremely fast, since finding a distance is just looking up the appropriate value in a table [Museth et al. 2005]. They have the disadvantage of requiring large storage and of being a fixed, usually crude, resolution. Our approach is analogous to a continuous version of a distance volume, since we can quickly find a minimum distance for any point position in a bounded domain. Distance field computations are also explored in many papers [Frisken et al. 2000; Sigg et al. 2003; Sud et al. 2004]. We present a continuous representation of points and parameters that satisfy a local distance extremum solution, while distance fields are discrete representations of the global minimum distance.

### 2.4 Problem Reduction to Parameter Space

A variety of geometric problems involving freeform curves or surfaces can be reduced to the single question of finding the zero-set of a system of non-linear polynomial equations in the parameter space of the original curves or surfaces [Elber et al. 2001; Seong et al. 2004; Seong et al. 2005]. Techniques for solving a set of polynomial equations are developed and applied to various geometric problems as a primitive tool [Sherbrooke and Patrikalakis 1993; Elber and Kim 2005; Dokken 1985; Grandine et al. 2000]. The minimum distance algorithm employed in this paper also operates on the same premise as taken in [Kim and Elber 2000; Patrikalakis and Maekawa 2002].

## 3 Distance from a Point to Planar Curve

In this section, we present a dimensionality lifting approach for computing minimum distances between a point and planar curve. Considering all the possible position of the point, we lift the problem into a 3D parameter space and construct a solution space for the minimum distance problem. In the higher dimension, we find that the solution is simplified and it becomes easy to analyze the topological structure of the solution space. This topology information makes it possible to search the global solution in a very efficient way.

### 3.1 Computing a High Dimensional Solution Space

Given a planar curve $C(t)$ and a point in the plane, $\mathbf{P}$, the squared distance

$$D^2(t) = \langle C(t) - \mathbf{P}, C(t) - \mathbf{P} \rangle$$

is represented as a B-spline scalar function. The minimum of $D^2(t)$ can be found by computing all its extrema and choosing the smallest. Assuming that $C(t)$ is $C^1$-continuous, extrema of $D^2(t)$ occur where its derivative is zero, which in turn we seek the zeros of the scaled extremal equation $E(t)$,

$$E(t) = \left\langle C(t) - \mathbf{P}, C'(t) \right\rangle.$$

Considering all the possible position in the plane, the point $\mathbf{P}$ can be parameterized by $P(x,y)$. Then, a foot-point of $P(x,y)$ onto the curve $C(t)$ can be defined by lifting a distance extremal equation $E(t)$ into a 3D space,

**Definition 1** *A foot-point of $P(x,y)$ onto the planar curve $C(t)$ satisfies the following polynomial equation:*

$$\mathscr{F}(x,y,t) \quad = \quad \left\langle C(t) - P(x,y), C'(t) \right\rangle = 0. \qquad (1)$$

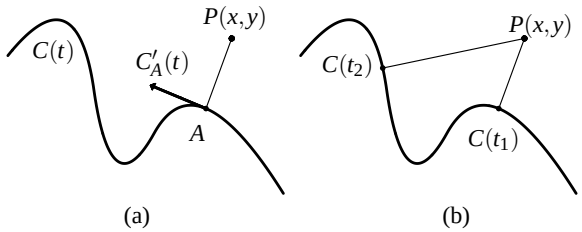



Figure 1: (a) A curve point $A$ is a foot-point of $P(x,y)$ since $C'_A(t)$ is orthogonal to the directional vector $A-P(x,y)$. Two foot-points of $P(x,y)$, $C(t_1)$ and $C(t_2)$, are shown in (b).

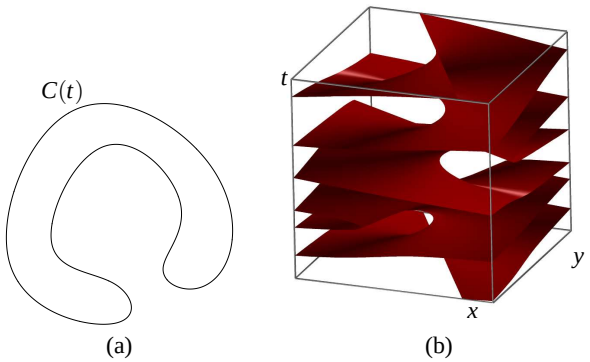



Figure 2: Given a planar curve $C(t)$ in (a), a zero-set surface of Equation (1) is represented using a red-colored surface in the *xyt*-parameter space (b).

In Figure 1(a), a curve point $A$ is a foot-point of $P(x,y)$ since $C'_A(t)$, the tangent vector evaluated at the point $A$, is orthogonal to the directional vector $(A-P(x,y))$. A general freeform curve may have a set of foot-points, $C(t_i), i=0,1,\cdots,n$, from the point $P(x,y)$ (Figure 1(b) shows two of them). Then, the problem for computing a minimum distance from the point $P(x,y)$ to the planar curve $C(t)$ can be posed as

$$\text{find} \;\; \min \|P(x,y)-C(t_i)\|^2,$$

where $C(t_i), i=0,1,\cdots,n$, are foot-points of $P(x,y)$ onto the curve $C(t)$.

The solution space to this minimum distance problem is constructed in the higher dimensional parameter space. Denoted by $\mathscr{M}$, a zero-set of Equation (1) is a two-manifold surface in the *xyt*-parameter space. Given a $C^1$-continuous curve $C(t)$, $\mathscr{M}$ is continuous and closed in the domain. The surface $\mathscr{M}$, in the parameter space, implicitly represent the solution space for the problem of minimum distance queries. A parameter point $(x,y,t)$ located on $\mathscr{M}$ corresponds to a foot-point $C(t)$ from $P(x,y)$ in the real world. Figure 2(a) shows a planar curve $C(t)$. Assuming that the plane is parameterized by $P(x,y)$, Figure 2(b) represents a zero-set of Equation (1) using a red-colored surface in the *xyt*-parameter space. Please note that the solution space $\mathscr{M}$ is constructed implicitly as a zero-set of the single polynomial equation in three variables, while the polynomial equation itself is explicitly represented using a NURBS function.

## 3.2 Solving for Single Queries

Given a specific position of the point $P(x_0,y_0)$, finding a minimum distance to the planar curve $C(t)$ is considered. Having a solution

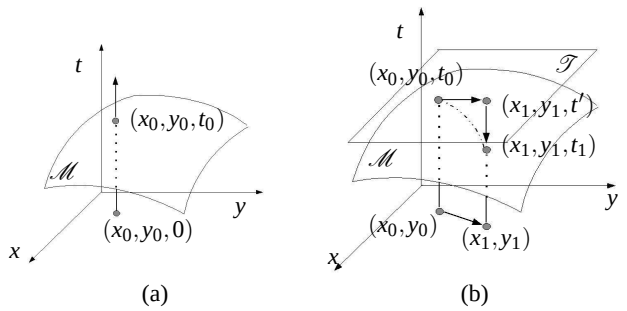



Figure 3: (a) Given a zero-set $\mathscr{M}$ of Equation (1), its intersection points with a ray staring at $(x_0,y_0,0)$ and directing positive $t$-direction correspond to foot-points of $P(x_0,y_0)$ onto the curve $C(t)$. When the point dynamically moves to the next position, an appropriate numerical marching process is shown in (b).

space $\mathscr{M}$ in the parameter space, one can imagine intersections of $\mathscr{M}$ with a ray of positive $t$-direction starting at $(x_0,y_0,0)$. A set of such intersection points, $\{(x_0,y_0,t_i)\}, i=0,1,\cdots,n$, between the ray and the solution space $\mathscr{M}$ corresponds to foot-points, $C(t_i), i=0,1,\cdots,n$, from $P(x_0,y_0)$. Figure 3(a) shows the solution space $\mathscr{M}$ and its intersection with the ray in the *xyt*-parameter space. A problem of computing a minimum distance between a point and planar curve can then be reduced to that of intersecting such a ray with the solution space $\mathscr{M}$.

The abstract idea of intersecting the ray with the zero-set $\mathscr{M}$ is computed by symbolically solving zeros of Equation (1) at a given parameter $(x_0,y_0)$ without an explicit representation of $\mathscr{M}$. Since $\mathscr{F}(x,y,t)$ is a piecewise rational function, the zero-set can be constructed by exploiting the convex hull and subdivision properties of NURBS, yielding a highly robust divide-and-conquer zero-set computation that is reasonably efficient (see [Elber and Kim 2005] for details). The subdivision process continues until a given maximum depth of subdivision or some other termination criteria is reached. At the end of the subdivision step, a leaf node of the subdivision tree contains a single solution point and a set of these discrete points are improved using multivariate Newton-Rapson method into a highly precise solutions.

## 3.3 Solution to Consecutive Queries

Without spatial coherency, solving a minimum distance query requires an evaluation and subdivision of the solution space $\mathscr{M}$ to be performed for every new position of the point. But, a point may move continuously in the plane and the distance query can be reformulated for a consecutive position of the point. In such a case, we present an efficient numerical marching method, which traces the solution space $\mathscr{M}$ in the parameter space. Let's assume that a point $P(x_0,y_0)$ in the plane moves to the next point $P(x_1,y_1)$ and that two points are close enough to each other. Then, in the *xyt*-parameter space, the corresponding solution point $(x_0,y_0,t_0)$ needs to be updated to the next point $(x_1,y_1,t_1)$, while the new point should be on the zero-set manifold $\mathscr{M}$. We compute new solution point using an iterative marching method. The point $(x_0,y_0,t_0)$ first proceeds to the point $(x_1,y_1,t')$ located on the tangent plane, $\mathscr{T}$, of the zero-set at $(x_0,y_0,t_0)$:

$$(x_1,y_1,t') = (x_0,y_0,t_0) + \Delta P - \nabla\mathscr{F}\,\langle \nabla\mathscr{F}, \Delta P\rangle,$$

where $\Delta P = (x_1,y_1,0)-(x_0,y_0,0)$. Here, we use the normal vector of the zero-set surface $\mathscr{M}$, $\nabla\mathscr{F}$, that is normalized to a unit vector. We now project the point $(x_1,y_1,t')$ back onto the manifold $\mathscr{M}$

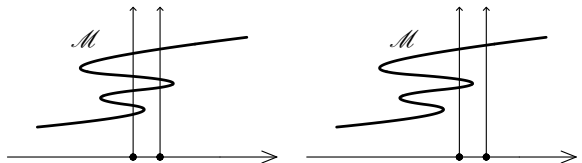


Figure 4: When a point $P$ moves dynamically, new components of the solution manifold $\mathscr{M}$ may appear (a) or an existing one may disappear (b).

using a high-dimensional Newton's method. We do this process iteratively until the point $(x_1, y_1, t')$ is placed on the solution space $\mathscr{M}$. Figure 3(b) shows the zero-set manifold $\mathscr{M}$ and the marching process over $\mathscr{M}$. Experimental results show that two or three iterations are enough for the convergence of the numerical projection operation.

### 3.4 Symbolic Analysis of the Topology

A continuous change of the position $P(x,y)$ may require a global analysis of the topological structure of the solution space. For consecutive distance queries, a numerical marching method may fail since it considers only the local properties of the solution space. Figure 4 shows a one-dimensional analogy to such a case. In Figure 4(a), when the point $P$ moves to $P_1$, a new component of the zero-set $\mathscr{M}$ appears. Similarly, a proceeding toward $P_2$ causes an event such that an existing component of the zero-set disappears (see Figure 4(b)). These critical events need to be properly handled for a global solution to a distance query. Considering the case that a new component appears (Figure 4(a)), a new part of the planar curve may contribute to a minimum distance. Local tracing algorithms have difficulties in searching for a global solution since the global solution jumps from one part of the curve to another. On the other hand, the projection of the new solution point onto the zero-set manifold may not converge if it encounters the second case of the critical events (Figure 4(b)), as the existing component disappears.

In this section, the problem for computing critical positions where such global changes occur is reduced to that of solving two polynomial equations in three variables. As presented in Section 3.1, the zero-set surface of Equation (1) determines the solution space for querying minimum distances from all the possible positions of the point to planar curve. Since the point moves continuously in the *xy*-plane, critical events occur at points where the *t*-component of the normal vector of $\mathscr{M}$ vanishes. We, therefore, reformulate the problem for computing topological changes to that of solving the following two polynomial equations:

$$\begin{aligned} \mathscr{F}(x,y,t) &= 0, \\ \mathscr{G}(x,y,t) &= \frac{\partial \mathscr{F}}{\partial t}(x,y,t) = 0. \end{aligned} \tag{2}$$

Having two equations in three variables, the simultaneous zeros of Equations (1) and (2) are one-dimensional curves in the *xyt*-parameter space. By projecting them onto the *xy*-plane, a set of critical curves that causes the topological change can be obtained. The plane is then decomposed into several small sub-regions having boundaries at the critical curves. Inside the connected sub-region, the topology (i.e. the number of foot-points) does not change. Given the set of critical curves, one can easily detect such events that bring a global change to the topology of the solution space. We check whether a point crosses one of the critical curves by simply subdividing each of the two B-spline functions $\mathscr{F}(x,y,t)$ and $\mathscr{G}(x,y,t)$ at the appropriate position and evaluating their bounding boxes in the parameter space. When the point crosses the critical curves, the corresponding event needs to be considered: new foot-points should be inserted into the set of candidate solutions in the case of Figure 4(a) or existing one is deleted encountering the case of Figure 4(b).

A geometric interpretation of the critical curves shows an interesting relationship between those curves and the *curve evolute* of the given planar curve. Equation (2) expands to

$$\frac{\partial \mathscr{F}}{\partial t}(x,y,t) = \langle C'(t), C'(t) \rangle + \langle C(t) - P(x,y), C''(t) \rangle = 0.$$

The locus of the points $P(x,y)$ at which the above equation is satisfied can be written as

$$P(x,y) = C(t) + \frac{1}{\kappa(t)} N(t),$$

where $\kappa(t)$ is the curvature and $N(t)$ is the normal vector of the curve $C(t)$ [Bruce and Giblin 1992; Johnson 2005]. Therefore, a critical point $P(x,y)$, computed by solving Equations (1) and (2) simultaneously, lies at the center of the osculating circle at $C(t)$, the locus of which form a possibly discontinuous curve called the *curve evolute*.

Figure 5(a) shows a planar curve and its curve evolute. In Figure 5(a), the curve evolute is represented in gray-colored lines, which is computed symbolically using the formula, $C(t) + \frac{1}{\kappa} N(t)$. The evolute is possibly discontinuous at inflection points of the original curve, resulting in multiple curve sections (see Figure 5(a)). The zero-set of Equation (1) is shown as a red-colored surface and the critical curves are shown in green-colored lines in Figure 5(b). As one can see from Figure 5(c), the projection of critical curves onto the *xy*-plane matches with the curve evolute of Figure 5(a).

We now consider the combination of the global topology information with the previous numerical marching method for consecutive distance queries. Assuming $\mathscr{F}(x,y,t)$ in Equation (1) as a single variate function with $t$ variable, its Jacobian becomes a matrix containing a single component $\frac{\partial \mathscr{F}}{\partial t}(x,y,t)$. For the case of finding roots of Equation (1), an iteration of Newton's method becomes degenerate when the determinant of its Jacobian vanishes:

$$\det(\mathbf{J}) = \det([\mathscr{F}_t]) = \frac{\partial \mathscr{F}}{\partial t}(x,y,t) = 0.$$

This shows that the meaning of Equation (2) can be reviewed from the Jacobian of Equation (1), which determines the degeneracy condition of the Newton's method. Thus, the critical curves provide an elegant way not only to efficiently find the global solution but also to support the numerical method even in its degenerate cases.

To get an efficient system for the minimum distance queries, a hierarchical subdivision of the plane is implemented. We first precompute critical points by symbolically solving Equations (1) and (2) with a rough tolerance. Four or five level of subdivision is used along each parameteric axes for the hierarchical subdivision in the examples of Section 5. Complex geometry of the original curve may require high level of subdivision. Each sub-region of the subdivided plane contains a list of critical points which are located inside the region. More precisely speaking, they maintain a set of bounding boxes, at which the critical point resides, of the size of the given tolerance used from the symbolic computation. Figure 5(d) shows a planar curve and the critical points with their bounding boxes. Since it is relatively expensive to test whether the point crosses one

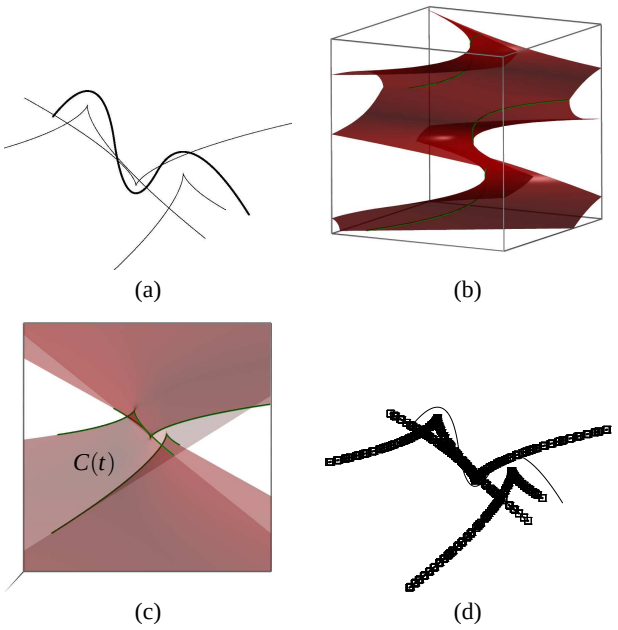



Figure 5: (a) A planar curve is shown in bold lines togethered with its curve evolute. (b) A zero-set of Equation (1) is represented in red-colored surface and critical curves are shown in green and bold lines. (c) The projection of the critical curves onto the *xy*-plane matches with the curve evolute of $C(t)$. A set of bounding boxes are shown in (d) with their center at the corresponding critical point.

of the critical curves, the system checks the crossing only when the point moves into one of the bounding boxes of the critical points. Hierarchical subdivision of the plane prunes away the movement of the point using a simple bounding box test. Experimental results show that this grid-based approach makes the performance of querying system about ten times faster.

# 4 Minimum Distance Queries Between a Point and Freeform Surface

The algorithm for computing minimum distances between a point and planar curve can be extended to the 3D case: distance queries from a point to freeform surface. For the surface case, a high-dimensional solution space is constructed by two polynomial equations in five variables. We therefore solve and update the distance query by tracing the solution space numerically in five-dimensional parameter space. Furthermore, we reduce the problem for analyzing the global topology of the solution space to that of solving three polynomial equations in five variables. Based on the topology information, we get a reliable tracing algorithm to find global minima.

## 4.1 Computing a High Dimensional Solution Space

Similarly to the curve case, the squared distance from a point in a 3D space, $\mathbf{P}$, to a freeform surface $S(u,v)$

$$D^2(u,v) = \langle S(u,v) - \mathbf{P}, S(u,v) - \mathbf{P} \rangle$$

is represented by B-spline scalar function. The minimum of $D^2(u,v)$ can be found by computing all its extrema and choosing the smallest. Assuming that $S(u,v)$ is $C^1$-continuous, extrema of $D^2(u,v)$ occur where its partial derivatives with respect to $u$ and $v$ become zero simultaneously. Using scaled extremal equations $E_1(u,v)$ and $E_2(u,v)$

$$E_1(u,v) = \langle S(u,v) - \mathbf{P}, S_u(u,v) \rangle$$
$$E_2(u,v) = \langle S(u,v) - \mathbf{P}, S_v(u,v) \rangle,$$

where $S_{u/v}$ is $u/v$-partial derivative of $S(u,v)$, we find their simultaneous zeros.

Given a parameterization of a point in a 3-D space, $P(x,y,z)$, we define a foot-point of $P(x,y,z)$ onto the surface $S(u,v)$ by lifting extremal distance equations $E_1(u,v)$ and $E_2(u,v)$ into a five variate functions,

**Definition 2** *A foot-point of the point $P(x,y,z)$ onto freeform surface $S(u,v)$ satisfies the following two polynomial equations:*

$$\mathscr{H}(x,y,z,u,v) = \left\langle S(u,v) - P(x,y,z), \frac{\partial S}{\partial u}(u,v) \right\rangle = 0, \quad (3)$$

$$\mathscr{I}(x,y,z,u,v) = \left\langle S(u,v) - P(x,y,z), \frac{\partial S}{\partial v}(u,v) \right\rangle = 0. \quad (4)$$

A general freeform surface may have a set of foot-points, $S(u_i,v_i), i = 0,1,\cdots,n$, from the point $P(x,y,z)$. Then, the problem for computing a minimum distance from $P(x,y,z)$ to $S(u,v)$ can be posed as

$$\text{find} \quad \min \|P(x,y,z) - S(u_i,v_i)\|^2,$$

where $S(u_i,v_i), i = 0,1,\cdots,n$, are foot-points of $P(x,y,z)$ onto the surface $S(u,v)$.

The solution space to the minimum distance problem for the surface case is now constructed using Equations (3) and (4). Having two equations in five variables, one gets three-dimensional manifolds, $\mathscr{Z}$, as their simultaneous zero-set in five-dimensional parameter space. A parameter point $(x,y,z,u,v)$ located on $\mathscr{Z}$ corresponds to a foot-point $S(u,v)$ from $P(x,y,z)$ in the real world. Thus, two polynomial equations implicitly define the solution space $\mathscr{Z}$ in five-dimensional parameter space.

## 4.2 Computing Minimum Distances

As the solution space to the Equations (3) and (4) has three degrees-of-freedom, specifying a point $(x,y,z)$ yields discrete set of zero-dimensional solution points. For a specific point $P(x_0,y_0,z_0)$, one can imagine a hyperplane in the *xyzuv*-parameter space. Then, intersection points between the hyperplane and the three-dimensional solution space are mapped to a set of foot-points and one can choose one of them having the smallest distance to $P(x_0,y_0,z_0)$. Similarly to the curve case, the intersection points are computed using a multivariate constraint solver [Elber and Kim 2005].

Solving for a single query requires an evaluation and subdivision of B-spline functions for every new query. However, a spatially coherent movement of a space point $P(x,y,z)$ yields an efficient numerical algorithm by utilizing a higher dimensional solution space, $\mathscr{Z}$. Assuming that a set of solution points, $\{(x_0,y_0,z_0,u_i,v_i)\}, i = 0,1,2,\cdots,n$, is given at current time step, a minimum distance query for the next position is solved numerically by extending the tracing algorithm presented in Section 3.3 to that of five dimensional space. When the point $P(x_0,y_0,z_0)$ moves to $P(x_1,y_1,z_1)$,

we numerically march each solution points $\{(x_0, y_0, z_0, u_i, v_i)\}$ in the parameter space to the next point located on the solution space $\mathscr{Z}$. For the simplicity of the explanation, we consider a single solution point $(x_0, y_0, z_0, u_0, v_0)$ in the parameter space. Then, the point $(x_0, y_0, z_0, u_0, v_0)$ proceeds to the point $(x_1, y_1, z_1, u', v')$ located on the hyperplane which is tangent to $\mathscr{Z}$ at $(x_0, y_0, z_0, u_0, v_0)$,

$$(x_1, y_1, z_1, u', v') = (x_0, y_0, z_0, u_0, v_0) + \Delta P - \nabla\mathscr{H} \langle \nabla\mathscr{H}, \Delta P \rangle - \nabla\mathscr{I} \langle \nabla\mathscr{I}, \Delta P \rangle .$$

Here, $\Delta P = (x_1, y_1, z_1, 0, 0) - (x_0, y_0, z_0, 0, 0)$ and we use two gradient vectors, $\nabla\mathscr{H}$ and $\nabla\mathscr{I}$, that are orthonormal to each other. Finally, the point $(x_1, y_1, z_1, u', v')$ is projected onto the manifold using a high dimensional Newton's method. Note, however, that we project the point onto the manifold while keeping the *xyz* components of the point unchanged. We do this process iteratively until the point $(x_1, y_1, z_1, u', v')$ is placed on the solution space $\mathscr{Z}$.

### 4.3 Analysis of the Topology

As a logical extension from the curve case, we provide a global analysis on the topology of the solution manifold for the surface case and show how this topology information helps the numeric marching method especially in degenerate cases where the numerical method fails. A geometric interpretation of the critical points also shows its relationship with the *focal set* of the surface.

Similar to the curve case, the global structure of the solution space changes when the space point $P(x, y, z)$ crosses critical points. Since the solution space $\mathscr{Z}$ is determined by two five-variate Equations (3) and (4), a condition for the critical point becomes

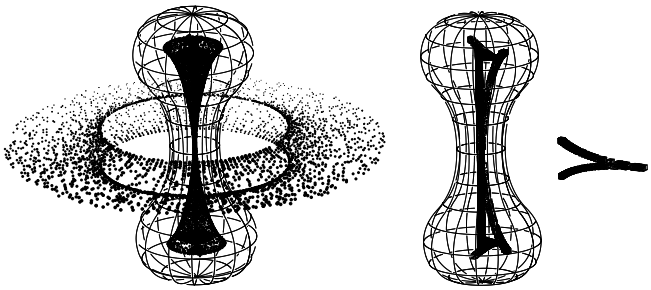
$$\mathscr{K}(x, y, z, u, v) = \det \begin{bmatrix} \frac{\partial\mathscr{H}}{\partial u} & \frac{\partial\mathscr{H}}{\partial v} \\ \frac{\partial\mathscr{I}}{\partial u} & \frac{\partial\mathscr{I}}{\partial v} \end{bmatrix} = 0. \qquad (5)$$

Having three Equations (3), (4) and (5) in five variables, one gets two-manifold surfaces as their simultaneous zero-set. By projecting them onto the *xyz*-space, a set of critical surfaces that causes the topological change can be obtained. Figure 6(a) shows a freeform surface and the set of critical points.

A hierarchical subdivision of the *xyz*-space to efficiently maintain a set of critical points is constructed like the curve case. First, critical points are precomputed by symbolically solving Equations (3), (4) and (5) with a rough tolerance. Since the symbolic solver computes critical points by recursively subdividing the parameter space, we can get a hierarchical subdivision of the *xyz*-space without any other costs, which contains a critical point at the end of the subdivision stage. Figure 6(b) shows a set of critical points with its bounding box of the size of the given tolerance. In Figure 6(b), critical points are evaluated only along one of the iso-curves for the simplicity of the representation.

A geometric meaning of critical points shows that intrinsic properties of a surface, such as the principal curvatures of a surface, characterize the topological structure of the solution space, $\mathscr{Z}$. Since critical points satisfy Equations (3) and (4), we can restrict the points to be on the normal line of a surface. Then, the directional vector $(S(u, v) - P(x, y, z)$ can be replaced by $\ell N(u, v)$, where $N(u, v)$ is a normal vector of $S(u, v)$ and $\ell$ is some real variable. Plugging in this to Equation (5) results in

$$\det \begin{bmatrix} \langle \ell N(u,v), S_{uu} \rangle + S_u^2 & \langle \ell N(u,v), S_{uv} \rangle + \langle S_u, S_v \rangle \\ \langle \ell N(u,v), S_{uv} \rangle + \langle S_u, S_v \rangle & \langle \ell N(u,v), S_{vv} \rangle + S_v^2 \end{bmatrix} = 0. \qquad (6)$$



(a) (b)

Figure 6: (a) A freeform surface and a set of critical points. (b) shows a set of critical points with its bounding box of the size of given tolerance. Critical points are evaluated only along one of the iso-curves for the simplicity of the representation.

Zeros of Equation (6) can be interpreted in terms of the principal curvatures at $S(u, v)$, $\kappa_1$ and $\kappa_2$, after a simple substitution of Equation (6) using the first and second fundamental forms. These derivations provide a rewriting of Equation (6) as

$$\ell^2 \kappa_1 \kappa_2 + \ell(\kappa_1 + \kappa_2) + 1 = 0,$$

and the zeros are just

$$\ell = -\frac{1}{\kappa_1} \qquad \text{and} \quad \ell = -\frac{1}{\kappa_2}.$$

Therefore, the critical point $P(x, y, z)$ is a distance along the normal equal to one of the principal radii of curvature of the closest point on the surface $S(u, v)$. Compare this geometric interpretation of the critical point to that of the curve case.

## 5 Experimental Results

Several examples of computing minimum distances between a point and planar curve or freeform surface are now presented. First, some examples for minimum distance queries to planar curve are shown. Figure 7 presents an example of a curve with a trajectory of the moving point. In Figure 7(a), the planar curve is shown in bold lines with a trajectory curve in gray. For each sampled position of the moving point, the corresponding curve point which gives the minimum distance to the point is connected using a line segment. Figure 7(b) shows a higher dimensional solution space using a red-colored surface in the *xyt*-parameter space. Minimum distance points in real Euclidean space have their corresponding points in the higher dimensional parameter space. In Figure 7(b), a yellow-colored sphere represents a parameter point corresponding to each of the minimum distance query shown in Figure 7(a). One more example is presented in Figure 8. The planar curves presented in these experimental examples are represented by cubic NURBS having about 10-20 control points. In these experimental examples, a tolerance of 0.05 (planar curves' dimensions span about a unit length) is used for the preparation of the critical points. And about 90% of distance queries are pruned away during the test of crossing the critical curves.

Figure 9 presents two more complex examples of minimum distance queries for planar curves. Here, planar curves are shown in bold lines and a trajectory curve for the moving point in gray. 200 positions of the trajectory curve are sampled for the minimum distance queries. With the aid of the higher dimensional solution

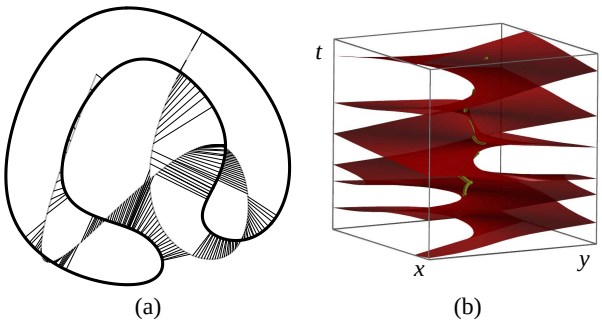



Figure 7: (a) A planar curve is shown in bold lines togethered with a trajectory curve for the moving point. For a sampled point of the trajectory curve, a line segment connects its corresponding curve point which gives a minimum distance to the curve. (b) A higher dimensional solution space is represented in red-colored surface and solution point in the parameter space, which is corresponding to the minimum distance, is shown in yellow sphere.

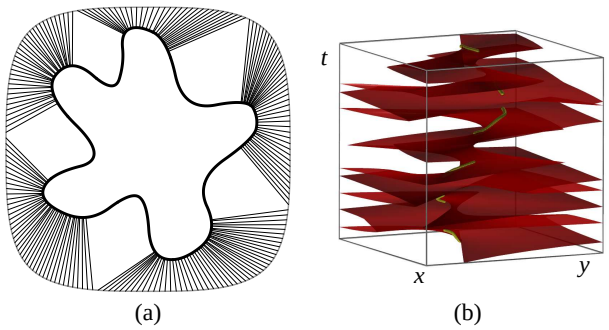



Figure 8: (a) A planar curve is shown in bold lines togethered with a trajectory curve for the moving point. For a sampled point of the trajectory curve, a line segment connects its corresponding curve point which gives a minimum distance to the curve. (b) A higher dimensional solution space is represented in red-colored surface and solution point in the parameter space, which is corresponding to the minimum distance, is shown in yellow sphere.

space, global solutions to the minimum distance queries are computed and shown in Figure 9 using line segments. Having the hierarchical subdivision of the plane and early pruning technique, the computation time for these results for the curve case are about the same, 10000 distance queries took about 1.2 to 1.4 seconds. Please note that we take all the solution points of local minima being tracked to yield a global solution. Thus, a single query contains multiple distance extrema computations. Testing was done on a Pentium IV 2GHz desktop machine.

Continuing to examples of computing minimum distance for freeform surfaces, Figure 10 shows two examples. In Figure 10(a), the same surface as in Figure 6 is used for computing a minimum distance. As one can see from Figure 6(a), the minimum distance point jumps to a different part of the surface when the moving point crosses the critical points. With the aid of global analysis on the topology of the solution space, our approach properly traces the global solution. The freeform surfaces presented in these experimental examples are represented by bicubic NURBS having about 50-70 control points. A symbolic computation of the critical points used a tolerance of 0.1 (freeform surfaces' dimensions span about a unit length) and the symbolic computation time for these examples are within a minute including a construction of the hierarchical bounding boxes, which were used in early pruning of distance queries. On a 2GHz Pentium IV machine, about 300 to 500 queries were computed in a second.

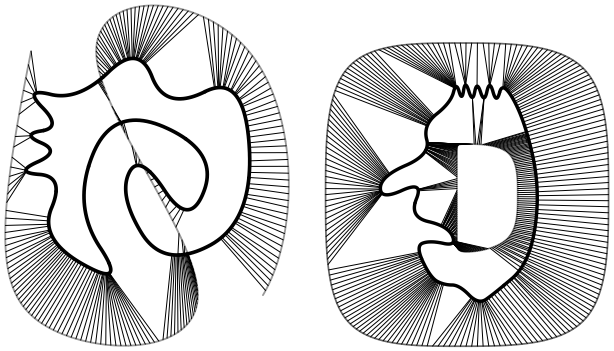


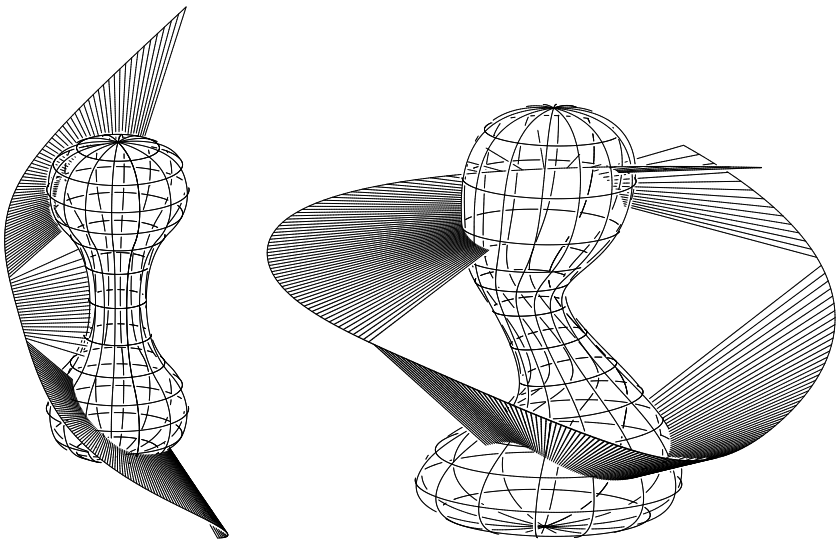

Figure 9: A planar curve is shown in bold lines togethered with a trajectory curve for the moving point. For a sampled point of the trajectory curve, a line segment connects its corresponding curve point which gives a minimum distance to the curve.

Figure 11 shows two more complex examples. Here, freeform surfaces are shown in bold lines and the trajectory curve for the moving point in gray. 200 positions of the trajectory curve are sampled for the minimum distance queries. In Figure 11(b), our minimum distance search is applied to a collection of surfaces that form a teapot. It is quite straightforward to extend our approach to such a scene that has multiple objects. Obviously, distance computations between a point and boundary curves of each surface patch may be taken into consideration to handle multiple objects, while that doesn't need to be invoked in the example of Figure 11(b).

## 6 Conclusion

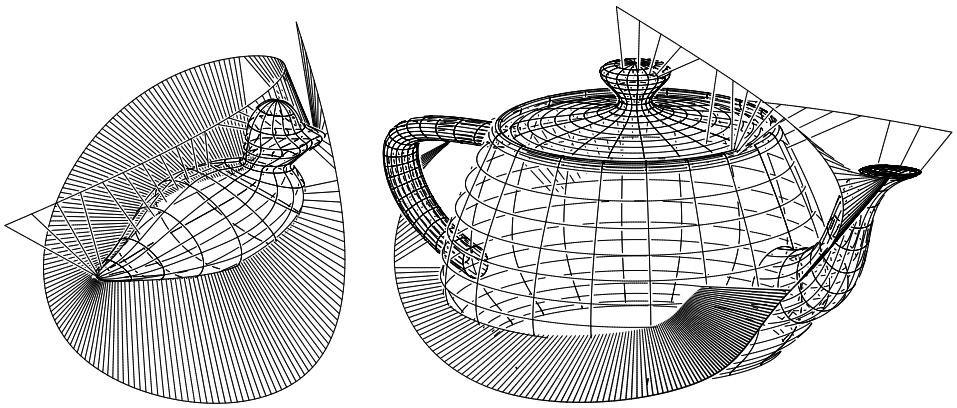
A robust and efficient algorithm for computing minimum distances between a point and planar curve or freeform surface has been presented. The presented approach is based on the dimensionality lifting of the problem into a higher-dimensional parameter space. This higher dimensional formulation solves for all query points in the domain simultaneously, which provides opportunities to speed computation by applying coherency techniques. For the curve case, a higher dimensional solution space is defined by a single equation in three variables. This formulation yields two-manifold surfaces in the parameter space, from which a minimum distance query is solved numerically in a very efficient way. Global convergence is assured by detecting changes in the higher-dimensional topological structure. We reduced this problem of analyzing the topological structure of the solution space to that of solving two polynomial equations in three variables. The topology information supports local searches in the numerical tracing algorithm to guarantee that it finds all the solutions and finally gives a global minima. This symbolic computation part of the algorithm is accelerated through preprocessing of the higher-dimensional space. The approach developed for curves in the plane has been extended to freeform surfaces in a 3D space. For the surface case, a higher-dimensional solution space is implicitly defined by two polynomial equations in five variables.

A variety of hierarchical space subdivision techniques can be applied to the current system for computing minimum distances. An efficient integration of the hierarchical technique is expected to



(a) (b)

Figure 10: A freeform surface is shown in bold lines togethered with a trajectory curve for the moving point. For a sampled point of the trajectory curve, a line segment connects its corresponding surface point which gives a minimum distance to the surface.



(a) (b)

Figure 11: A freeform surface is shown in bold lines togethered with a trajectory curve for the moving point. For a sampled point of the trajectory curve, a line segment connects its corresponding surface point which gives a minimum distance to the surface.

yield a distance querying algorithm of higher performance. The presented approach may easily be applicable to a trimmed model. Solution points for the local distance extrema only need to be tested whether they are located inside the valid parameter domain or not in the case of trimmed models. We are also working to extend the presented approach to the computation of minimum distances for the curve-curve or surface-surface case. To this end, we need to deal with even higher-dimensional solution spaces.

## Acknowledgments

All the algorithms and figures presented in this paper were implemented and generated using the IRIT solid modeling system [Elber 2000] developed at the Technion, Israel. This work was supported in part by ARO (DAAD19-01-1-0013) and NSF (IIS0218809, CCR0310705). All opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the sponsoring agencies.

## References

BARAFF, D. 1990. Curved surfaces and coherence for non-penatrating rigid body simulation. *Computer Graphics 24*, 4, 19–28.

BREEN, D., MAUCH, S., AND WHITAKER, R. 1998. 3d scan conversion of csg models into distance volumes. In *Proceedings of the 1998 Symposium on Volume Visualization*, ACM SIGGRAPH, 7–14.

BRUCE, J., AND GIBLIN, P. 1992. *Curves and Singularities*. Cambridge University Press.

DOKKEN, T. 1985. Finding intersections of b-spline represented geometries using recursive subdivision techniques. *Computer Aided Geometric Design 2*, 1, 189–195.

EHMANN, S. A., AND LIN, M. C. 2001. Accurate and fast proximity queries between polyhedra using convex surface decomposition. *Eurographics (EG) 2001 Proceedings 20*, 500–510.

ELBER, G., AND KIM, M. S. 2005. Geometric constraint solver using multivariate rational spline functions. In *Proc. of International Conference on Shape Modeling and Applications*, MIT, USA, 216–225.

ELBER, G., KIM, M. S., AND HEO, H. 2001. The convex hull of rational plane curves. *Graphical Models 63*, 151–162.

ELBER, G. 1992. *Free Form Surface Analysis using a Hybrid of Symbolic and Numeric Computation*. PhD thesis, Department of Computer Science, University of Utah.

ELBER, G., 2000. Irit 9.0 user's manual. http://www.cs.technion.ac.il/∼irit, October.

FRISKEN, S., PERRY, R., ROCKWOOD, A., AND JONES, T. 2000. Adaptively sampled distance fields: A general representation of shape for computer graphics. In *Proc. of SIGGRAPH 2000*, 249–254.

GRANDINE, T., CRACIUN, B., HEITMANN, N., INGALLS, B., GIA, Q., OU, M., AND TSAI, Y. 2000. The bivariate contouring problem. *IMA Bebruary 2000 Preprint Series*.

II, T. T., JOHNSON, D., AND COHEN, E. 1997. Direct haptic rendering of sculptured models. In *Proc. of 1997 Symposium on Interactive 3D Graphics*, 167–176.

JOHNSON, D., AND COHEN, E. 2001. Spatialized normal cone hierarchies. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics (I3D)*, 129–134.

JOHNSON, D., AND COHEN, E. 2005. Distance extrema for spline models using tangent cones. In *Graphics Interface*.

JOHNSON, D. 2005. Minimum distance queries for haptic rendering. *PhD Thesis*.

KIM, M. S., AND ELBER, G. 2000. Problem reduction to parameter space. In *The Mathematics of Surface IX (Proc. of the Ninth IMA Conference)*, London, 82–98.

LARSEN, E., GOTTSCHALK, S., LIN, M., AND MANOCHA, D. 2000. Fast distance queries with rectangular swept sphere volumes. In *IEEE International Conference on Robotics and Automation (ICRA)*, 24–48.

MORTENSEN, M. 1985. *Geometric Modeling*. John Wiley & Sons, New York.

MUSETH, K., BREEN, D., WHITAKER, R., MAUCH, S., AND JOHNSON, D. 2005. Algorithms for interactive editing of level set models. *Computer Graphics Forum 24*, 4, 1–22.

PATRIKALAKIS, N., AND MAEKAWA, T. 2002. *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer Verlag.

POTTMANN, H., LEOPOLDSEDER, S., AND ZHAO, H. 2003. The $d^2$-tree: A hierarchical representation of the squared distance function. In *Technical Report No. 101*, Institut fur Geometrie, TU Wien.

QUINLAN, S. 1994. Efficient distance computation between noncnovex objects. In *IEEE Int. Conference on Robotics and Automation*, 3324–3329.

SEONG, J., ELBER, G., JOHNSTONE, J., AND KIM, M. 2004. The convex hull of freeform surfaces. *Computing 72*, 1, 171–183.

SEONG, J., KIM, K., KIM, M., ELBER, G., AND MARTIN, R. 2005. Intersecting a freeform surfaces with a general swept surface. *Computer-Aided Design 37*, 5, 473–483.

SHERBROOKE, E., AND PATRIKALAKIS, N. 1993. Computation of the solutions of nonlinear polynomial systems. *ComputerAided Geometric Design 10*, 5, 379–405.

SIGG, C., PEIKERT, R., AND GROSS, M. 2003. Signed distance transform using graphics hardware. In *Proc of IEEE VIS*.

SNYDER, J. 1993. Interval analysis for computer graphics. *Computer Graphics 26*, 2, 121–130.

SNYDER, J. 1993. Interval methods for multi-point colilisions between time-dependent curved surfaces. *Computer Graphics 27*, 2, 321–334.

SUD, A., OTADUY, M., AND MANOCHA, D. 2004. Difi: Fast 3d distance field computation using graphics hardware. *Computer Graphics Forum 23*, 3, 557–566.

WANG, H., KEARNEY, J., AND ATKINSON, K. 2002. Robust and efficient computation of the closest point on a spline curve. In *Proceedings of the 5th International Conference on Curves and Surfaces*, San Malo, France, 397–406.