Distance Extrema for Spline Models Using Tangent Cones

David E Johnson Elaine Cohen

School of Computing
University of Utah

Abstract

We present a robust search for distance extrema from a point to a curve or a surface. The robustness comes from using geometric operations rather than numerical methods to find all local extrema. Tangent cones are used to search for regions where distance extrema conditions are satisfied and patch refinement hierarchically improves the search. Instead of preprocessing and storing a large hierarchy, elements are computed as needed and retained only if useful. However, for spatially coherent queries, this provides a significant speedup.

Key words: Tangent cones, minimum distance, spline models.

1 Introduction

A distance query between a point and a model is a basic operation used in simulation [1], haptics [2], robotics [3], model simplification [4], and distance volume computation [5]. In general, distance queries either have been for a global minimum or for a local solution in the neighborhood of an initial guess. Instead, in this paper, we describe a robust search for all distance extrema between a point and a spline model. This type of search provides more information about the distance field than the global minimum does (Figure 1), and yet is more robust than typical local searches.

Our approach is based on hierarchical pruning of the spline model. This type of approach is commonly used to find a global minimum by bounding portions of the geometry with a conservative spatial primitive, such as a sphere or oriented bounding box. Instead of bounding just geometry, our approach also bounds tangent spreads on portions of the geometry and searches to satisfy the zeros of a distance extrema equation.

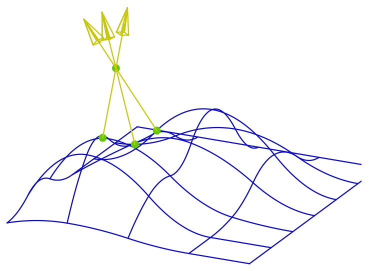


Figure 1: Finding all local extrema better characterizes the distance space than a single global minimum.

The resulting algorithm robustly finds all distance extrema to curves and surfaces. In addition, we develop acceleration techniques to take advantage of coherence in successive distance queries to surfaces. The accelerated queries compute multiple extrema to surfaces interactively.

This work is motivated by two projects in particular. In one, a force-feedback interface is used to feel spline models [2]. Latency requirements mandate that fast local distance methods are initialized by a global search prior to contact; however, a global minimum result may not detect multiple small local features that are potential contact locations. The second project is converting trimmed spline models to distance volumes [6]. In that case, a global minimum may lie in the trimmed off domain, so all local minima must be found, checked against the trimmed domain, and then the closest valid minima used.

This robust extrema search should be useful in other applications as well. However, in this paper, we describe the underlying algorithmic approach and not the applications.

2 Background

Distance techniques have often been specialized for either polygonal models or sculptured models such as splines. Because our approach uses hierarchical pruning, which is commonly associated with polygonal models, we include background on that model representation as well as on sculptured models.

2.1 Distance Queries for Polygonal Models

Polygonal models are typically composed of collections of triangles, and most distance algorithms for polygonal models deal with triangle primitives. The model may contain topological connectivity information. Models without connectivity are known as a triangle cloud, and ones with are properly described as a triangle mesh.

Lin [7] and Gilbert, Johnson, and Keerthi [8] developed fast minimum distance methods for convex polygonal models. Since local gradient search produces a global minimum for convex objects, their algorithms can converge quickly.

Quinlan [3] developed a spherical bounding hierarchy for general triangle clouds. The bounding hierarchy was used to determine an upper bound on minimum distance between the two models, and then to prune away portions of each model with lower bounds on distance larger than the upper bound. The PQP package [9] uses swept sphere volumes as a bounding hierarchy for triangle clouds. These volumes can control their aspect ratio to more tightly bound contained geometry than sphere bounds, providing faster distance queries.

More recently, the distance methods for convex model distance queries have been applied to convex decompositions of triangular models [10]. Essentially, this method reduces the number of leaf nodes by replacing triangles with convex sets.

For these general polygonal models, the predominant techniques create bounding volume hierarchies, and the advancements have come mostly from improving the tightness of the bounding volumes. This approach differs markedly from techniques used for sculptured models.

2.2 Parametric Models

Parametric models are composed of smooth surface patches, and typical models have fewer primitives than polygonal models. Thus, the emphasis in research has not been on efficient means of pruning large numbers of primitives. Instead, methods have explored various techniques for quickly and reliably solving systems of equations derived from setting up minimum distance conditions.

The system of equations for distance extrema have been variously defined as sets of cross-products [11] or augmented with explicit normal collinearity conditions [12]. These extrema conditions have been solved by employing symbolic computation [11], interval methods [12], and Newton-Rapheson iteration [13]. Having the advantage of high speed and rapid convergence, the last has been a practical choice for many implementers.

We examine the Newton-Rapheson approach in more detail for distance between a point and a curve and between a point and a surface in the following sections.

Distance from a Point to a Curve

The distance between a point in space $\mathbf{P}$ and a planar curve $\gamma(t)$ can be expressed as a parametric function,

$$D(t) = \left\| \left(\gamma(t) - \mathbf{P} \right) \right\|, \qquad (1)$$

and minimizing *D(t)* finds the minimum distance. The distance squared,

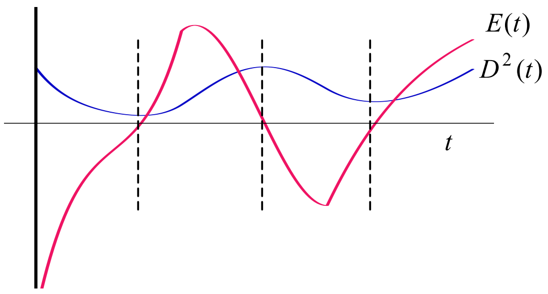


Figure 2: The derivative E(t) (in red) of the distance squared function (in blue). Zero crossings correspond to extrema in distance (the dashed lines).

$$D^2(t) = \left(\gamma(t) - \mathbf{P} \right) \cdot \left(\gamma(t) - \mathbf{P} \right), \qquad (2)$$

shares common extrema parameters with *D(t)* and avoids the square root.

The minimum of Eq. 2 can be found by computing all its extrema and choosing the smallest. Extrema occur where the derivative is zero, described by *E(t),*

$$\frac{dD^2(t)}{dt} = 2\left(\gamma(t) - \mathbf{P} \right) \cdot \frac{d\gamma}{dt} = 0 \, . \qquad (3)$$

$$E(t) = \left(\gamma(t) - \mathbf{P} \right) \cdot \frac{d\gamma}{dt} \, . \qquad (4)$$

Figure 2 shows this relationship between zeroes of *E(t)* and distance extrema. In the general case, extrema also occur at endpoints of the curve or at non-differentiable points. A curve with tangent discontinuities can be split into multiple curves, each of which is considered independently. The endpoints of the curve can also be checked as a special case.

E(t) can be interpreted as showing that distance extrema occur at orthogonal projections of $\mathbf{P}$ onto the curve, which is where the projection vector is at right angles to the tangent at the projected point. This orthogonality condition is what will be used as a search condition for the distance extrema search developed later in this paper.

Distance from a Point to a Surface

The minimum distance between a point in space $\mathbf{P}$ and a bivariate parametric surface $\mathbf{S}(u,v)$ is the minimum of the distance function

$$D(u,v) = \left\| \mathbf{S}(u,v) - \mathbf{P} \right\| . \qquad (5)$$

Following the approach for computing the minimum distance between a point and a curve, we instead look at the distance squared to a surface

$$D^2(u,v) = \left(\mathbf{S}(u,v) - \mathbf{P} \right) \cdot \left(\mathbf{S}(u,v) - \mathbf{P} \right). \qquad (6)$$

In Figure 3, local distance minima between the green query point and the surface are visible in the visualization of the distance squared function as two bumps, each corresponding to a local closest point on the original surface to **P**.

The partials of the distance squared generate a system of equations that is satisfied at an extremum.

$$\begin{aligned}(\mathbf{S}(u,v)-\mathbf{P})\cdot\mathbf{S_u} &= 0\\ (\mathbf{S}(u,v)-\mathbf{P})\cdot\mathbf{S_v} &= 0\end{aligned} \qquad (7)$$

This system is analogous to the extrema equation for minimum distance to a curve (Eq. 4), as it shows that the closest point on the surface is also an orthogonal projection of the query point onto the surface.

By mapping the partial derivatives as height fields $(u, v, \frac{\partial D^2(u,v)}{\partial u})$ and $(u, v, \frac{\partial D^2(u,v)}{\partial v})$, the zero crossings generically form curves in the *uv* plane (Figure 4), and the intersections between the zero sets are solutions to Equation 8.

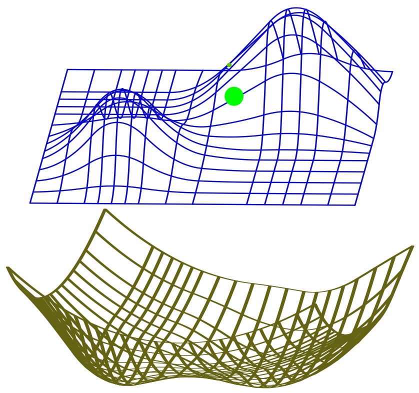


Figure 3: The squared distance between a point and surface (top) is visualized as a function mapping parameter value vs. squared distance (bottom).

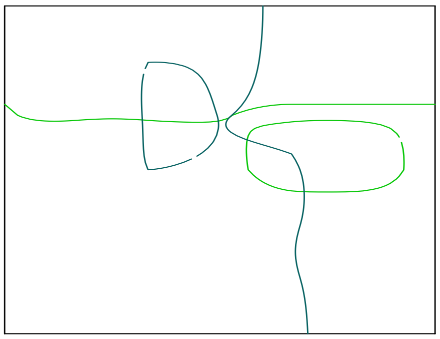


Figure 4: The zero crossings of each partial of the distance squared function generically form curves in the uv *plane. The intersections between the two curves are local extrema of the distance squared function.*

2.3 Spatialized Normal Cone Hierarchies

Prior work for triangular models has developed hierarchical geometric search for zeros of the extrema equations[14], and that work has been applied to haptic prototyping of model accessibility [15]. In that work, pre-computed hierarchies of normal cones were used to search for collinearity conditions between normal ranges and the query point. This work differs in that our approach does not use pre-computed hierarchies, the means of computing the bounding cone differs for spline surfaces, the way of determining satisfaction of the extrema equations differs for spline models, and acceleration techniques are needed for the spline models. The spline model approach, as applied to B-spline curves and tensor product surfaces, is described in the following section.

3 Tangent Cones for Distance Queries

Approaches that prune hierarchies based purely on spatial bounds ignore the role tangents and orthogonality play in defining distance extrema. In this section, we develop techniques that satisfy the equations for distance extrema using robust geometric operations. Our approach uses tangent cones to conservatively bound the range of tangents on a portion of a model.

4 Tangent Cones for Curve Distance Queries

We start by describing the application of tangent cone hierarchies to curves. The outline of the tangent cone approach is

1. Bound the range of tangents and the spatial extent of each interval of the curve.
2. See if the orthogonality condition might be satisfied for the interval for a given query point.
3. Subdivide intervals that meet the orthogonality condition.
4. Repeat until the remaining curve intervals contain small tangent ranges, and then compute exact local extrema using local methods.

4.1 Bounding the Range of Tangents with Cones

The derivative of a spline curve is a vector-valued spline curve. Thus, the properties of spline curves, such as the convex hull property, also apply to the derivative curve. This can be used to bound the range of normals for that curve.

The curve subdivision process turns pieces of spline curves into Bézier curves, which are a subset of splines. For a Bézier curve $B(t)$ of degree *d,* with control points P_i, and basis functions $\beta_{i,d}(t)$,

$$B(t) = \sum_i P_i \beta_{i,d}(t), \qquad (8)$$

and its derivative is

$$B'(t) = \sum_i d(P_{i+1} - P_i)\beta_{i,d-1}(t). \qquad (9)$$

The derivative curve is also known as the *hodograph*, and is readily computed as scaled differences of adjacent control points of the original curve. Since the convex hull property holds for the derivative curve, the range of tangent directions is bounded by the convex hull of the derivative curve control points.

We compute a 2D cone to encapsulate the range of tangent directions by averaging the normalized tangent control points and then finding the maximum angle between the average tangent and the tangent curve control points.

Since the orthogonality condition depends not only on the curve tangent, but also on the vector between the query point and the solution point, the tangent cone alone is insufficient to determine whether the orthogonality condition is satisfied. The range of vectors between the query point and possible solution points must also be taken into consideration. Those vectors are bounded by first bounding the possible locations of the solution point on the curve with a circle. The range of vectors between the query point and bounding circle then forms a cone, called the *solution line cone* (Figure 5).

4.2 Checking the Orthogonality Condition

Given the tangent cone and the solution line cone, the orthogonality check determines if any vector in the tangent cone can be orthogonal to a vector in the solution line cone. For cone axes, T_A and S_A, with spreads T_α and S_α, this is tested by seeing if

$$\begin{aligned} \cos^{-1}(T_A \cdot T_B) - T_\alpha - S_\alpha \le 90° \\ \cos^{-1}(T_A \cdot T_B) + T_\alpha + S_\alpha \ge 90° \end{aligned} \qquad (10)$$

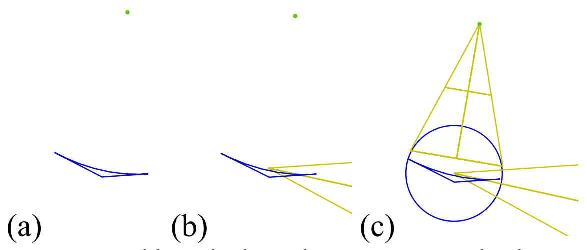


Figure 5: Building the bounds on tangent and solution vectors. (a) The query point and curve interval, with its control polygon. (b) The tangent cone computed from differences of control points. (c) The solution line cone encompasses a bounding circle around the interval.

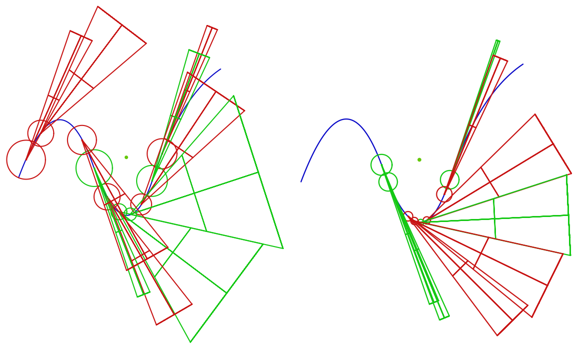


Figure 6: Two consecutive levels of subdivision are shown. The triangle fans are the tangent cones computed from the derivative curve and the circles provide the basis for the solution line cones. Intervals in green are subdivided and ones in red are removed from further computation.

Intervals passing the orthogonality test are subdivided. These subdivided curves have tighter bounds on their spatial extent, which produces tighter solution line cones and tighter bounds on their tangent cone. These tighter bounds are better able to prune intervals during the next iteration (Figure 6).

4.3 Computing an Exact Solution

The subdivision terminates when the tangent cone spread angle falls below a user specified epsilon. Exact local minima are computed from these intervals using local numerical methods. An interval accepted for exact tests performs nodal mapping between the query point and interval, followed by Newton's method.

Not all intervals will contain a valid local minimum because of the conservative nature of the bounding cone and bounding circle tests. Solutions that leave the interval during Newton's method can be removed, or converged solutions can be checked for redundancies. The latter approach is preferred since it potentially provides multiple initial guesses to Newton's method for each local minimum, improving robustness when computing the exact solution.

5 Tangent Cones for Surface Distance Queries

The tangent cone approach for point-curve distance extrema extends naturally to point-surface distance queries. In the curve case, a cone bounded the range of tangents for a curve interval. In the surface case, the same approach applies, except to surface patches instead of curve intervals.

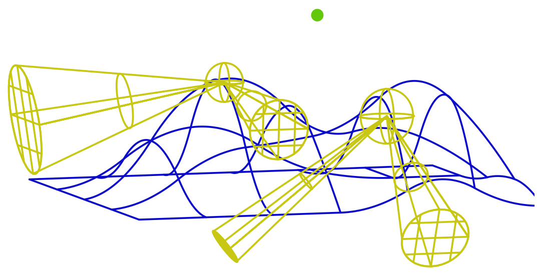


Figure 7: Two surface patches with their sample tangent cones. The spheres bound the spatial extent of the patch and help to form the solution line cone from the query point.

Surfaces have two tangent directions, one for each parameter. The orthogonality test computes bounding cones for each of the tangent directions (Figure 7) and a solution line cone from the query point to a bounding sphere around the patch. Patches are pruned by checking each tangent cone for perpendicularity with the solution line cone. A failure to find potential orthogonality between a vector in the solution line cone and vectors in either of the tangent cones means that surface patch cannot contain a local solution.

Tangent cones are computed by finding differences of control points over all the rows or columns of the control mesh, depending on the tangent direction. Similarly, the patch bounding sphere is computed from the average and extent of the patch control points. For Bézier surfaces, the number of rows or column is equal to the parametric order, so for low order surfaces the computation is not too costly.

Surface patches that may contain a solution are split into four smaller patches (Figure 8), while patches that fail the normal cone test are removed. Exact closest points are found on patches with both tangent cone spread angles smaller than an epsilon. A combination of nodal mapping followed by Newton's method computes these exact local closest points (Figure 1). Because leaf nodes may not contain a local solution, solutions that leave the leaf node domain need to be discarded, or checked against existing solutions to see if they are redundant, similar to the curve case.

Tests indicate that the subdivision hierarchy manages to stay fairly sparse during typical distance queries, which shows the subdivision and pruning test is effective.

Figure 9 shows the number of patches checked at each iteration of the algorithm for two different query points, one yielding five solutions and the other three. The chart does show that for the tested queries, little pruning occurs during the first few levels of subdivision, and that further on, the algorithm creates and destroys a lot of patches that do not yield a solution.

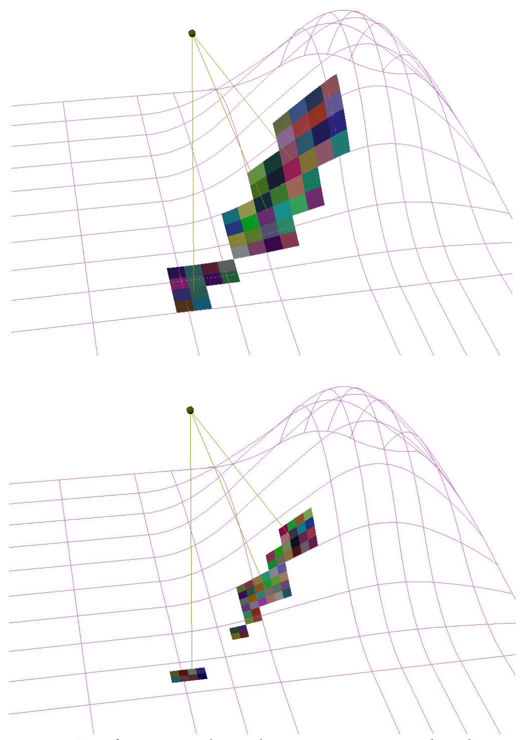


Figure 8: Surface patches that are accepted split into four smaller patches. These new patches have tighter bounds on tangent cones and spatial location.

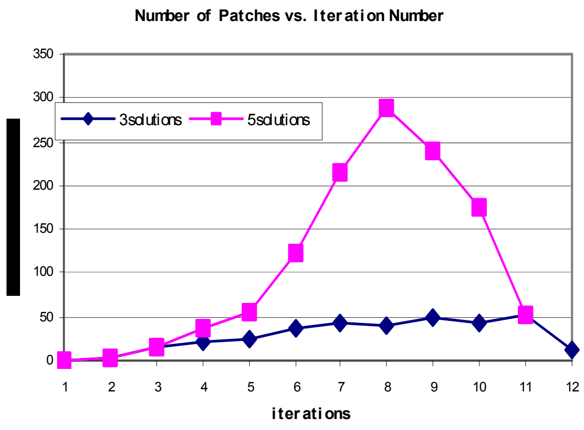


Figure 9: The chart shows the number of active surface patches during each iteration. The query that returned five local minima had to create and search many more patches than the query that returned three.

6 Coherency with the Dynamic Subdivision Tree

Since the subdivision process always splits patches in the middle of its domain, a tested child patch is identical from one query to the next, even when the query point moves. Surface patches and their associated cones could then be stored for use in following queries. However, retaining all surface patches can be costly in terms of computer memory.

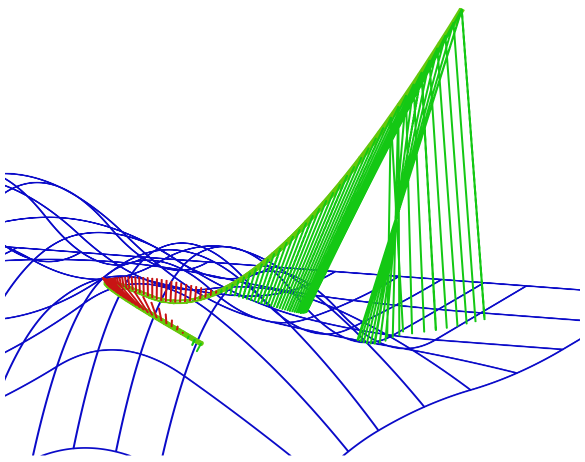


Figure 10: The query point moves along the test path, in green. Lines between the query points and local solutions are drawn to show multiple extrema. This type of test is sped using a dynamic subdivision tree

Instead, we adopt the approach of retaining portions of the subdivision tree over multiple queries, but removing portions that are no longer currently used. Patches are stored as computed in a tree data structure, a *dynamic subdivision tree*, with each node having four children. Patches (and their children) are deleted only when a query tests a patch's tangent cones and determines there is no local solution in that patch. This has the effect of retaining patches during sets of temporally and spatially coherent queries. The patch may need to be recomputed later, but this approach balances memory usage with computational efficiency.

The dynamic subdivision tree approach dramatically improves computational speed when query points are closely spaced. The dynamic subdivision tree approach was tested by creating a sample path above a surface, moving along the path in small increments, and performing a distance query at each step (Figure 10). Without coherence, 1000 distance queries took 24 seconds; with coherence, the same test took 4 seconds, 6 times faster. Testing was done on a Pentium III 1GHz laptop.

7 Robustness

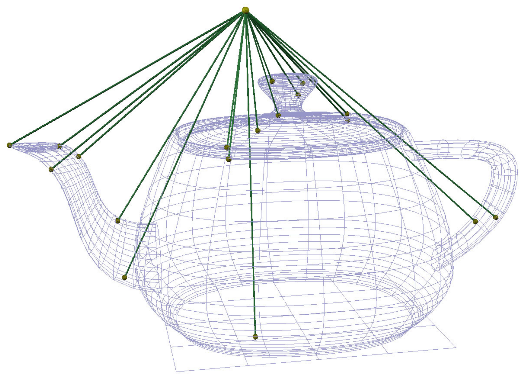
The subdivision step that provides the basis for improving the search is a very robust operation, as are the bounding operations that prune the hierarchy. However, at a user specified angle tolerance, multi-dimensional Newton's method is used to improve the final solution. This improvement step is initialized by nodal mapping between the point and the patch. If that initial guess is poor, the numerical method may not converge. Therefore, the tolerance provides a balance between robustness and speed that must be decided by the needs of the user and the type of model. In our tests, we used a tolerance of 0.001 radians, which generally provides a very good initial guess.

The numerical methods may still not converge in concave regions which form singularities in the matrix inversion step of the numerical method. If convergence is not detected, then the nodal mapping estimate can be used as a solution, although we did not use this fallback in any of our tests.

8 Discussion

A more complex example demonstrating our extrema search is shown in Figure 11, where we find local extrema to a collection of surfaces that form a trimmed teapot. Our technique was successful in extracting extrema from the high curvature lips around the teapot lid and spout as well as less challenging areas on the lid and top.

We are working to adapt this approach to computing distance volumes on such trimmed models. An advantage of this approach over other robust geometric subdivision approaches is that we work directly on the original model. Because the extrema equation changes for each new query point, the symbolic approaches are not able to retain subdivision information over coherent queries, an important speedup in our approach.



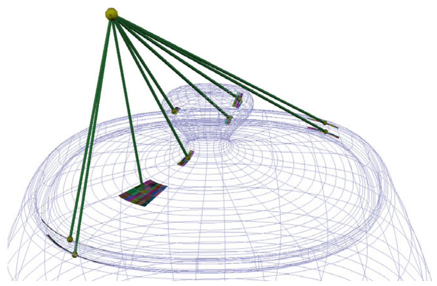


Figure 11:Distance extrema between a point and collection of surfaces are shown at top. Below, we show remaining surfaces after several steps of subdivision and pruning.

9 Conclusion

We have developed a robust search for local extrema in distance between a point and a curve or a surface. By bounding tangent ranges rather than spatial location, the algorithm searches for portions of the surface that satisfy an extrema equation rather than an upper bound on global distance. Because the refinement hierarchy retains useful information from one query to the next, retaining portions of the hierarchy provides several times speedups.

Acknowledgements

This work was supported in part by ARO (DAAD19-01-1-0013) and NSF (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

[1] Baraff, D., "Curved Surfaces and Coherence for Non-penetrating Rigid Body Simulation," Computer Graphics, Vol. 24, No. 4, pp.19-28, 1990.

[2] Thompson II, T.V., Johnson, D.E., and Cohen, E.C. "Direct Haptic Rendering Of Sculptured Models", in Proc. 1997 Symposium on Interactive 3D Graphics, (Providence, RI), April 1997, pp. 167-176.

[3] Quinlan, S. "Efficient Distance Computation between Non-Convex Objects," IEEE Int. Conference on Robotics and Automation, pp. 3324-3329, 1994.

[4] Guezlec, A. "'Meshsweeper': dynamic point-to-polygonal mesh distance and applications," in IEEE Transactions on Visualization and Computer Graphics, vol. 7-1. pp. 47-61. 2001.

[5] Breen, D., Mauch, S. and Whitaker, R."3D Scan Conversion of CSG Models Into Distance Volumes," *Proceedings of the 1998 Symposium on Volume Visualization*, ACM SIGGRAPH, October 1998, pp. 7-14.

[6] Museth, K. Breen, D., Whitaker, R., Mauch, S. and Johnson, D. "Algorithms for Interactive Editing of Level Set Models", submitted to *Journal of Eurographics: Computer Graphics Forum.*

[7] M.C. Lin, *Efficient Collision Detection For Animation and Robotics*, Ph.D. thesis, University of California, Berkeley.

[8] E. Gilbert, D. Johnson, S. Keerthi, "A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space," IEEE Journal of Robotics and Automation, pp. 193-203, April 1988.

[9] E. Larsen, S. Gottschalk, M. Lin, and D. Manocha., "Fast Distance Queries using Swept-Sphere Volumes," *Proc. IEEE International Conference on Robotics and Automation*, 2000.

[10] S. Ehmann and M. Lin. "Accurate and Fast Proximkity Queries between Polyhedra using Surface Decomposition," Computer Graphics Forum (Proc. Eurographics), 2001.

[11] M.C. Lin and D. Manocha, "Fast Interference Detection Between Geometric Models," The Visual Computer, pp. 542-561, 1995.

[12] J. Snyder, "Interval Methods For Multi-Point Collisions Between Time-Dependent Curved Surfaces", Computer Graphics, 27(2). pp. 321-334, Aug. 1993.

[13] D. Nelson, D.E. Johnson, and E. Cohen, "Haptic Rendering of Surface-to-Surface Sculpted Model Interaction," in *Proc. 8th Annual Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, (Nashville, TN), ASME, November 1999.

[14] D.E. Johnson and E. Cohen, "Spatialized Normal Cone Hierarchies," in 2001 ACM Symposium on Interactive 3D Graphics, ACM SIGGRAPH, March 2001.

[15] D.E. Johnson, and P. Willemsen, "Six Degree-of-Freedom Haptic Rendering of Complex Polygonal Models," in Proc. *2003 Haptics Symposium* , 2003.