Six Degree-of-Freedom Haptic Rendering of Complex Polygonal Models

David E. Johnson Peter Willemsen
School of Computing, University of Utah

Abstract

This paper describes a haptic rendering algorithm for arbitrary polygonal models using a six degree-of-freedom haptic interface. The algorithm supports activities such as virtual prototyping of complex polygonal models and adding haptic interaction to virtual environments. The underlying collision system computes local minimum distances between the model controlled by the haptic device and the rest of the scene. The haptic rendering computes forces and torques on the moving model based on these local minimum distances.

1 Introduction

Haptic interfaces add a sense of touch to interactions with a virtual scene. Sensations of touch are generated by computing the forces of interaction between objects in the virtual environment and reflecting those forces back to the user. *Haptic rendering* is the term for computing these interaction forces between models. Typically, these forces must be updated much faster than for visual simulation, on the order of one kilohertz[1]. Haptic forces are usually proportional to the distance of penetration between the models. Typically, the computational bottleneck has been from not being able to compute these distances quickly enough.

Our approach finds the *local minimum distances* (LMDs) between polygonal models using spatialized normal cone hierarchies[7]. This data structure hierarchically encapsulates the position and spread of surface normals over a model. The LMD algorithm uses the surface normal information to find portions of each model that point towards each other, which represents a local minimum distance.

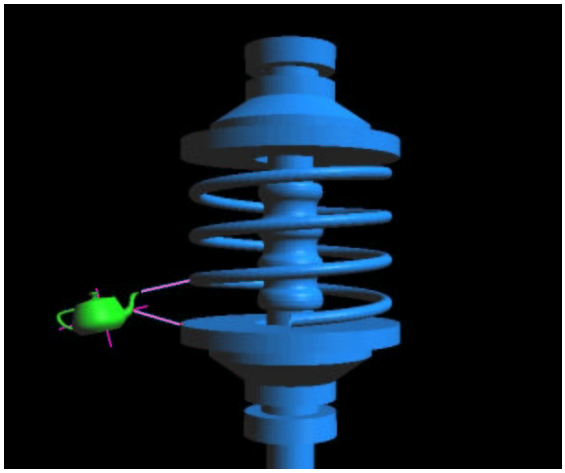


Figure 1: *Our approach finds local minimum distances between models and applies repulsive forces to prevent contact.*

Rather than finding forces that move models apart once they have collided, our haptic rendering algorithm prevents collisions by applying preventative forces as models approach each other. This technique is appropriate for representing interactions between models since allowing models to penetrate each other violates real-world constraints. Our test system uses a six degree-of-freedom (DOF) haptic interface as the means of moving a model in the scene and for reflecting the forces of model-model collision back to the user.

Some distinguishing characteristics of our approach are as follows:

- polygonal models of arbitrary shape can be used in the virtual environment.
- elements of the scene can be moved or added and deleted without requiring substantial preprocessing.
- environments with large number of triangles can be used, increasing the accuracy of simulated model interactions.

The haptic rendering algorithm we present is appropriate for activities such as virtual prototyping and adding haptic interaction to virtual environments.

Virtual prototyping systems attempt to replace the evaluative aspects of physical prototype models with virtual models in a computer. These types of systems

have utility in mechanical design and architecture, where physical models can be costly to produce and limit the ability of a designer to quickly test modifications.

A virtual prototyping environment should support activities such as assembly and placement of models. These activities are difficult to perform using purely computational means. The size of the virtual models overwhelm current algorithmic techniques. In addition, the complexity of the models also makes techniques that rely solely on visual interpretation difficult to use effectively[13].

Haptic interfaces allow the sense of touch to guide the placement of the models in the scene. This type of interaction is very natural and prior haptic virtual prototyping systems have demonstrated the usefulness of including the sense of touch in complex placement tasks. Our algorithm allows virtual prototyping of standard polygonal models with little preprocessing and permits modifying the scene to quickly test changes to the design.

2 Background

Advancements in haptic rendering and geometric computations have been tightly linked. The following sections will cover relevant work in distance computations and then review existing work in haptic rendering and virtual prototyping.

2.1 Distance computations

Almost all the literature on minimum distance, especially for polygonal models, treats the problem primarily as an Euclidean one. Approaches typically partition the model with hierarchical spatial bounding volumes, using primitives such as spheres[17], convex polytopes[16], oriented swept sphere volumes[11], and convex surface patches[4]. Nodes of the hierarchy may be pruned by obtaining a lower bound on the distance to the contained geometry and comparing that distance to an upper bound on the minimum distance obtained through a depth-first descent in the hierarchy or a sample point on the surface. The primary research thrust in this area has focused on more tightly bounding the geometry at a node. This approach returns the global minimum distance between models.

A different approach is found in work on sculptured surfaces, such as B-splines[7][15]. Often a local solution to the minimum distance is desired, rather than a global solution. The distance between two parametric surfaces, *F(u,v)* and *G(s,t)*, may be described by

$$D^2(u,v,s,t) = \langle F(u,v) - G(s,t) \rangle^2 . \quad \text{(Eq. 1)}$$

Extrema of the distance can be found by differentiating and finding the roots of the resulting set of equations. We remove the parameters for clarity and denote the partial derivative of the surface with a subscript.

$$\begin{aligned} (F-G)\cdot F_u &= 0 \\ (F-G)\cdot F_v &= 0 \\ (F-G)\cdot G_s &= 0 \\ (F-G)\cdot G_t &= 0. \end{aligned} \quad \text{(Eq. 2)}$$

The roots of these equations yield a set of local extrema in distance between the two models. Choosing the minimum solution provides the global minimum distance solution. Note that this solution depends only on the normals at the solution being parallel and parallel to the vector between the solution points. Recalling that two models at the moment of collision have parallel tangent planes at the contact point, and thus fulfill the criteria above, can help to understand the significance of this derivation.

This approach is very different from the upper and lower bound pruning used with polygonal models and is the basic approach that the spatialized normal cone adapts to polygonal models.

2.2 Haptic rendering

Haptic rendering algorithms were first developed to support three DOF haptic interfaces. These algorithms and devices support a moving point exploring a computer model. Researchers have developed algorithms for haptic point contact with polygonal[1] [5][18][20], implicit[21], and NURBS0[24] models. In [3], contact between a ray and polygonal model produced more realistic haptic rendering than with just a point contact.

More recently, efforts have focused on developing techniques to haptically render the interactions between two models. The resulting forces include torques as well as translation forces, and a six DOF haptic device is needed to accurately reflect the results back to a user.

In the polygonal model domain, the first efforts at 6DOF haptic rendering were for small convex shapes[2]. More recent work has looked at collections of convex bodies[6], as well as incremental methods for computing the penetration depth[9].

Model-model haptic rendering for general NURBS models is described in [15]. This system used a three pass approach: initial distance monitoring using polygonal approximations, local closest point initialization using Newton's method on an extremal distance formulation, and stable maintenance of the

penetration depth distance with a relation between parametric space and Euclidean movement.

Researchers at Boeing have taken a different approach by creating a voxel-based scene and allowing a point-sampled free-moving model to interact with the voxels[13]. The advantage is that the computation time can be tightly bound by the number of voxels and the number of points in the free-moving model. They also created a voxel boundary around the models in the scene to prevent interpenetration of models, which would invalidate the correctness of the virtual prototyping.

2.3 Spatialized Normal Hierarchies

Spatialized normal cone hierarchies are introduced in [8], although various hierarchical normal bounds have been developed for applications as diverse as silhouette edge extraction[22], back-face primitive culling[23][10], and level-of-detail display of large-scale models[18].

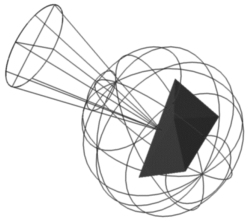


Figure 2: *The spatialized normal cone data structure bounds the Euclidian extent with a sphere and the range of surface normals with a cone.*

The spatialized normal cone hierarchy data structure is simple, consisting of a cone, represented by a cone axis vector and a cone semi-angle; and a Euclidean bounding volume, in this case a sphere, represented by a center and a radius. Each node of the data structure also contains pointers to two child nodes, and a pointer to an underlying triangle if it is a leaf node.

The triangle holds pointers to its neighboring triangles and storage for the face normal and vertex normals. These normals are computed in a preprocess if they aren't available from the file format.

The hierarchy is computed using the publicly available PQP distance code. The details are described in [11].

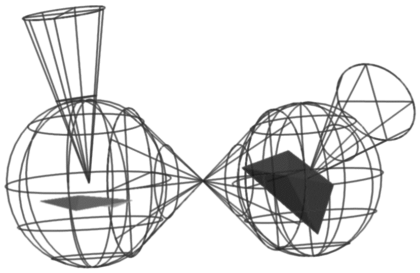


Figure 3: *At each pair of nodes, the algorithm checks to see if the normal spans overlap and are along a vector contained within the span of possible minimum distance lines.*

The algorithm computes the LMD by finding nodes in the hierarchy that satisfy the constraints of Eq. 2. That is, it finds nodes such that the normal cones point towards one another and along a vector contained within the span of vectors computed from the spherical geometric bounds of each node. The algorithm recursively descends the hierarchy to winnow away large portions of the models until exact leaf tests can be applied.

This algorithm returns the set of LMDs between the polygonal models. These minimum distances describe a fuller set of potential contacts than the single distance returned by a global minimum distance algorithm.

3 System overview

Our virtual prototyping system is based on a Sensable six DOF PHaNTOM haptic interface. The computations run on a dual processor Pentium 4 2.4 GHz Linux computer with a gigabyte of RAM and a GeForce 4 Ti 4400 graphics card. The application is multi-threaded, with the haptic force computation thread running at a high priority to ensure fast update rates. The graphic display reproduces the force computation at a lower rate and displays the polygonal models in their current positions.

4 Approach

We adopt the approach of the Boeing voxel sampling virtual prototyping system[13], which prevents models from colliding rather than moving them apart once interpenetration has occurred. This has several advantages:

- accuracy of virtual prototyping is maintained since real-world constraints are maintained,
- lower rates than the typical kilohertz haptic rate are acceptable, since we are

not attempting to create the impulsive forces of hard contact,

- minimum distances are faster to compute than penetration depths, allowing more detailed scenes to be haptically rendered.

4.1 Local Minimum Distances

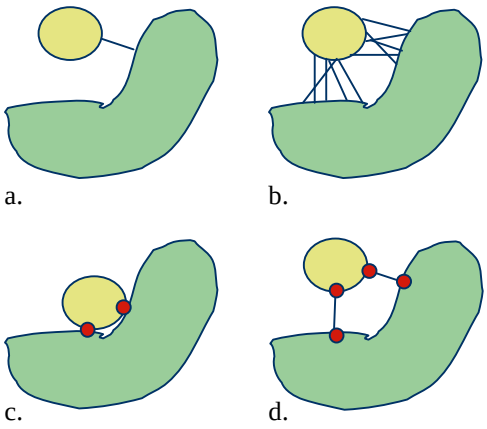


Figure 5: *(a) The global minimum distance. (b) All pairs within a distance. (c) Contact points between two colliding models. (d) Local minimum distances.*

Global minimum distances may be rapidly computed between polygonal models using a number of algorithms[11]. However, if these techniques were used in a haptic rendering system, the global minimum would only generate a single penalty force at a time. This force could rapidly change direction, creating haptic instabilities.

One could easily imagine modifying a distance computation to return all pairs that are within a certain distance, rather than just the global minimum. However, this could potentially create very large numbers of penalty forces, which would swamp the haptic computation (Figure 4).

We argue that an appropriate solution is to compute the local minimum distances between models. Imagine two models that have just collided. This collision can be represented at a single point on each surface (even for manifold contacts, a single point encapsulates that area of contact). If the models move apart, this pair of points tracks the local minimum distance and represents the potential future contact between entire sections of these two models. Additional pairs of contact points for those sections are redundant predictors of future contacts for those regions, thus the local minimum distance pairs are adequate. This formulation keeps a manageable number of contacts for the haptic computation, yet is complete enough to safely predict all potential contacts.

4.2 Modifying the LMD computation

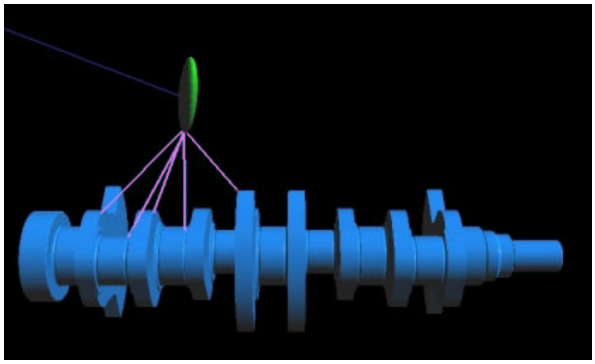


Figure 4: *Finding all LMDs will create forces between portions of the models that are not interacting.*

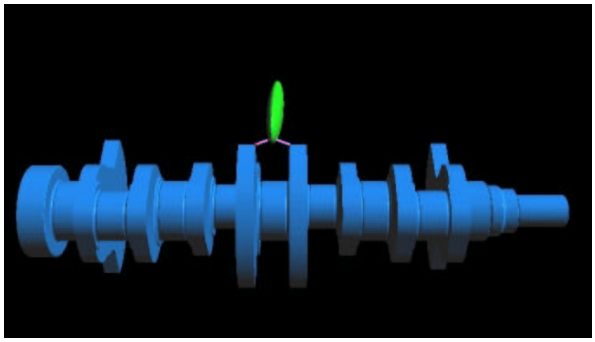


Figure 6: *Using a small cutoff distance removes unnecessary LMDs from consideration and controls the onset of forces.*

We use a modified LMD computation based on the spatialized normal cone hierarchies to quickly determine all the potential areas of contact. The main modification is to introduce a cutoff distance that prunes pairs of nodes that are further apart than this distance. This is appropriate for haptic rendering, where we are only interested in computing penalty forces for models in close proximity (Figure 4 and Figure 6).

4.3 Forces and Torques

At each time step in the haptic rendering loop, our algorithm computes the LMDs that are closer than the cutoff distance between the model that is controlled by the haptic interface and the rest of the models in the scene. Each LMD is considered a virtual spring with a rest length equal to the cutoff distance. Each spring is attached to the models by the pairs of points that form

the LMD. The force applied to the moving model is then

$$\vec{f}_i = (cutoffDist - \|LMD_i\|)LMD_i$$

$$\vec{F} = \sum_i -k\vec{f}_i$$

And the torque is

$$\vec{\tau} = \sum_i \vec{p}_i \times \vec{f}_i$$

where $\vec{p}_i$ is the distance from the center of mass to the tracking point on the model for the local minimum distance.

The center of mass and the first-order moments are approximated by the geometric extent of a PQP generated, oriented swept sphere bounding box surrounding and approximating the shape of the model. More precise values could be easily used when available.

The repulsive forces between models begin at zero at the cutoff distance, so LMDs that are created and destroyed as sections of the two models approach the cutoff distance only modify the total force and torque a small amount. Furthermore, since we are not attempting to render the forces of hard contact, only guiding the placement of models, the springs can be fairly soft, smoothing the haptic rendering.

4.4 Preprocessing

The LMD computations require precomputing a spatialized normal cone hierarchy for each polygonal model in the virtual prototyping environment. However, models in the scene can be added and deleted, or moved around interactively, without needing further precomputation. The preprocessing step takes a few seconds for models of a few thousand triangles.

5 Results

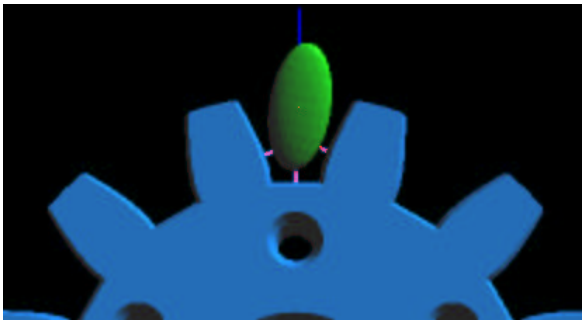


Figure 7: *We are able to feel translation forces and torques generated by interacting arbitrary models.*

We have tested our algorithms on a variety of models. The six DOF force feedback allows the model controlled by the haptic device to slide around the objects in the stationary scene, providing good intuition for the user (Figure 7). We were able to explore concave portions of the stationary model, with repulsive forces keeping us from all the potential contact areas (Figure 8).

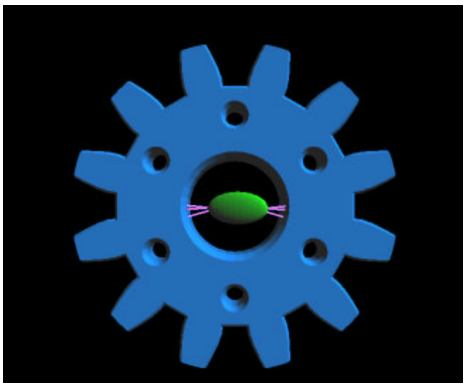


Figure 8: *Our technique handles concave regions of models.*

It is difficult to give a chart with timings for the rendering of these models, since the computation time varies with the cutoff distance, the number of LMDs found, and the relative configuration of the models. Instead, in Appendix A, we show a variety of sample interactions with a chart of polygon counts, timings, and number of LMDs. Typically, haptic rates in the hundreds of Hz are achieved between models with hundreds and thousands of polygons.

6 Conclusion

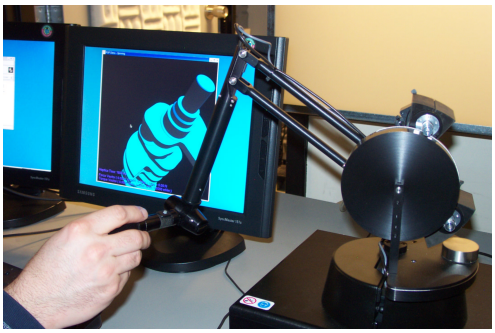


Figure 9: *The 6DOF Phantom used in a virtual prototyping session.*

This paper demonstrates an algorithm for six DOF haptic rendering of arbitrary polygonal models. The underlying collision code computes local minimum distances between models and derives repulsive forces and torques to maintain collision-free status. This technique is appropriate as a foundation for a virtual prototyping application for complex models.

7 Acknowledgements

The authors would like to acknowledge support in part from the following grants: NSF DMI9978603, NSF EIA 9871440, NSF CDA-96-23614, and ARO DAAD 19-01-1-0013.

8 References

[1] Adachi, Y. and Kumano, T. and Ogino, K., "Intermediate Representation For Stiff Virtual Objects," in *Proc. Virtual Reality Annual Intl. Symp*., Research Triangle Park, N.C., pp. 203-210, March 11-15, 1995.

[2] Baraff, D., "Fast contact force computations for non-penetrating rigid bodies", in *Computer Graphics Proceedings, Annual Conference Series*, vol. 28, 1994, pp. 23-34.

[3] Basdogan, C., Ho, C.-H., and Srinivasan, M.A, "A ray-based haptic rendering technique for displaying shape and texture of 3D objects in virtual environments", in *Proc. ASME Dynamic Systems and Control Division*, DSC-Vol. 61, pp. 77-84.

[4] Ehmann, S. and Lin, M., "Accurate and Fast Proximity Queries between Polyhedra Using Surface Decomposition", in *Computer Graphics Forum (Proc. of Eurographics)*, 2001.

[5] Gregory, A., Lin, M., Gottschalk, S., and Taylor, R., "H-COLLIDE: A Framework for Fast and Accurate Collision Detection for Haptic Interaction", in *Proc. IEEE Virtual Reality '99*

[6] Gregory, A., Mascarenhas, A., Ehmann, S., Lin, M., and Manocha., "Six Degree-of-Freedom Haptic Display of Polygonal Models", in Proc. IEEE Visualization, 2000.

[7] Johnson, David E. and Cohen, Elaine, "A framework for efficient minimum distance computations," *Proc. IEEE Intl. Conf. Robotics & Automation*, Leuven, Belgium, May 16-21, 1998, pp. 3678-3684.

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

[9] Kim, Y., Otaduy, M., Lin, M. and Manocha, D. "Six Degree-of Freedom Haptic Display Using Localized Contact Computations", in the *Tenth Symposium on Haptic Interfaces For Virtual Environment and Teleoperator Systems*, March 24-25, 2002.

[10] Kumar, S., Manocha, D., Garrett, W. and Lin, M., "Hierarchical Backface Computation," in *Proc. of 7th Eurographics Workshop on Rendering*, pp. 231-240. 1996.

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

[12] Lin, Ming and Manocha, Dinesh. "Fast Interference Detection Between Geometric Models," *The Visual Computer*, pp. 542-561, 1995.

[13] McNeely, W., Puterbaugh, K., and J. Troy. "Six degree-of-freedom haptic rendering using voxel sampling". *Proc. of ACM SIGGRAPH*, pages 401–408, 1999.

[14] Mirtich, Brian, *Impulse-based Dynamic Simulation of Rigid Body Systems*, Ph.D. Thesis, University of California, Berkeley, December, 1996.

[15] Nelson, D., Johnson, D., and Cohen, E., "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.

[16] Ponamgi, M., Manocha, D., and Lin, M., "Incremental algorithms for collision detection between solid models", *Proceedings of ACM/SIGGRAPH Symposium on Solid Modeling*, pp. 293-304, 1995.

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

[18] Rusinkiewicz, S. and Levoy, M., "QSplat: A Multiresolution Point Rendering System for Large Meshes", in SIGGRAPH 2000. 2000.

[19] Ruspini, D., Kolarov, K., and Khatib, O., "The Haptic Display of Complex Graphical Environments," *in Computer Graphics Proceedings, SIGGRAPH 1997*. Aug. 3-8. pp. 345-352.

[20] Salisbury, K., Brock, D., Massie, T., Swarup, N., and Zilles, C., "Haptic rendering: programming touch interaction with virtual objects", in *Symp. on Interactive 3D Graphics*, 1995. pp. 123-130.

[21] Salisbury, K., and Tarr, C, "Haptic rendering of surfaces defined by implicit functions", in *Proc. ASME Dynamic Systems and Control Division*, DSC-Vol. 61, 1997. pp. 61-67.

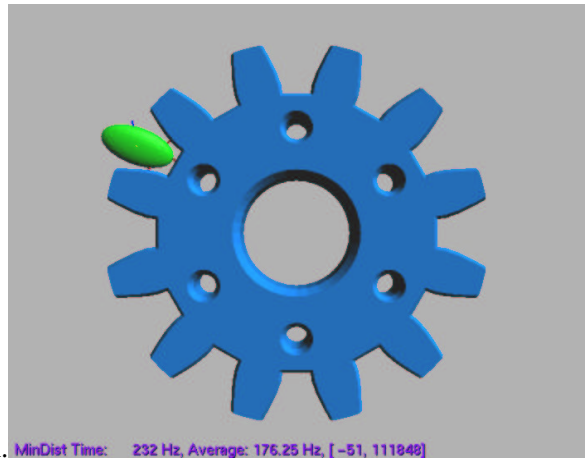
[22] Sander, P., Gu, X., Gortler, S., Hoppe, H., Snyder, J., "Silhouette Clipping." in *Computer Graphics Proceedings, SIGGRAPH 2000*. New Orleans. pp. 327-334.

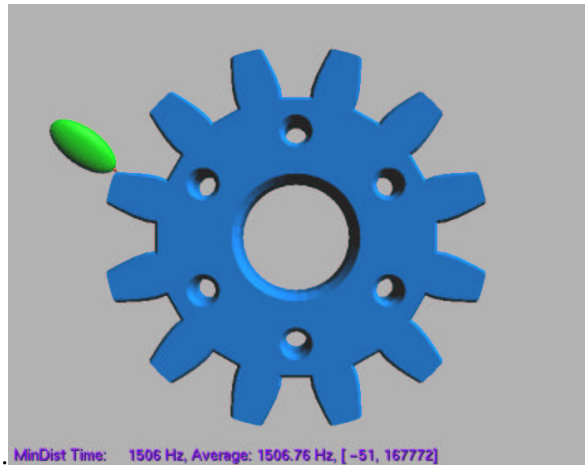
[23] Shirman, L. and Abi-Ezzi, S., "The Cone of Normals Technique for Fast Processing of Curved Patches," *EUROGRAPHICS'93*, Vol. 12.3., pp. 261-272. 1993.

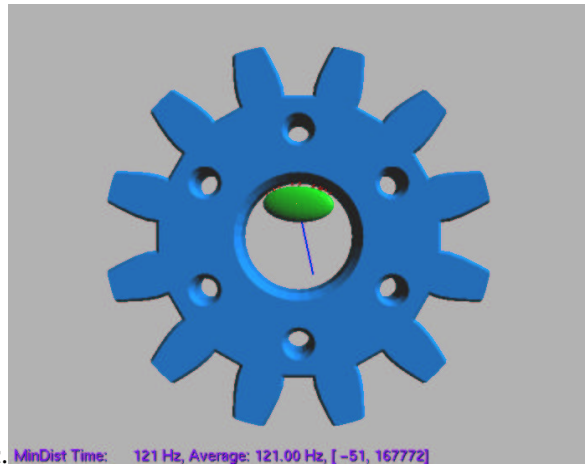
[24] Stewart, Paul, et al, "CAD Data Representations For Haptic Virtual Prototyping", Proceedings of DETC'97, *1997 ASME Design Engineering Technical Conferences*, Sept. 14-17, 1997, Sacramento, California.

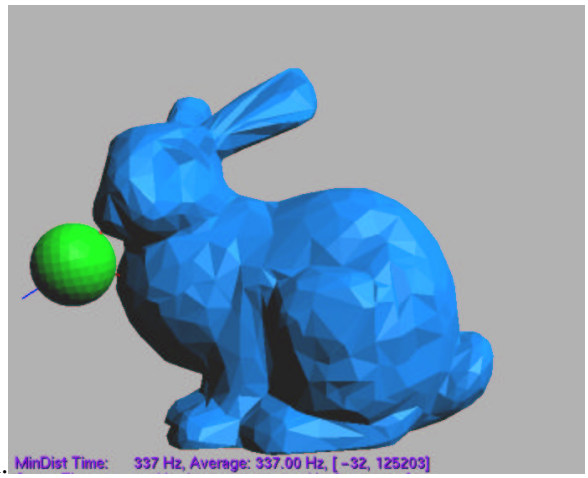
[25] Thompson II, T.V., Johnson, D.E., and Cohen, E.C. "Direct Haptic Rendering Of Sculptured Models", in *Proc. 1997 Symposium on Interactive 3D Graphics*, (Providence, RI), April 1997, pp. 167-176.

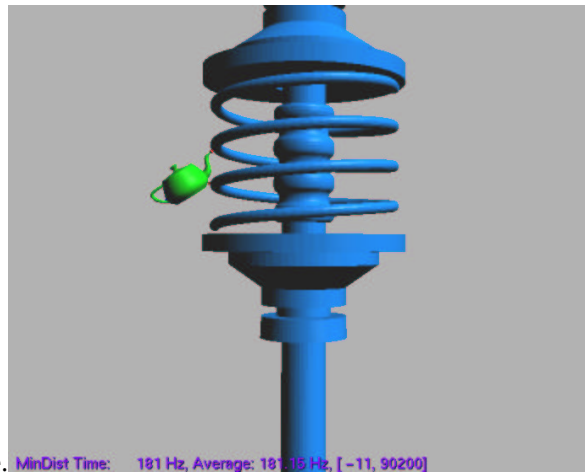
9 Appendix A

a. 

b. 

c. 

d. 

e. 

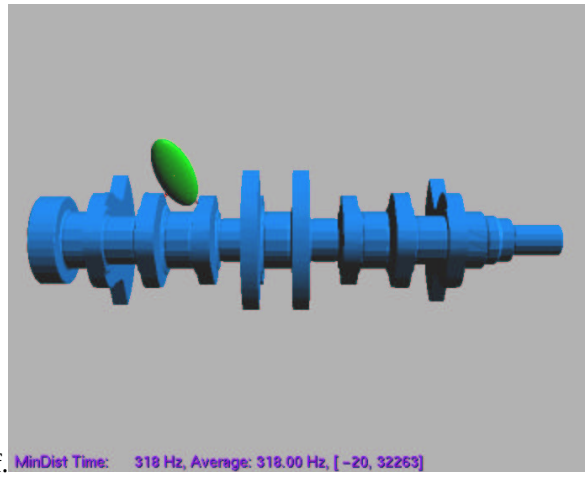
f. 

Image	Model (# tris)	Scene (# tris)	# of LMDs	Rate (Hz)
a	Disc (512)	Gear (6302)	3	176
b	Disc (512)	Gear (6302)	1	1506
c	Disc (512)	Gear (6302)	10	121
d	Sphere (128)	Bunny (2204)	2	337
e	Teapot (5648)	Spring (23578)	2	181
f	Disc (512)	Crankshaft (12802)	2	318