DiFi: Fast 3D Distance Field Computation Using GPUs

Department of Computer Science University of North Carolina at Chapel Hill November 2004

The Challenge

Given a set of objects, a distance field in 3D is defined at each point by the smallest distance from the point to the objects. Applications of distance fields include shape representation, model simplification, remeshing, morphing, CSG operations, sculpting, path planning and navigation and collision and proximity computations. These applications use a signed or unsigned distance field along a discrete grid. Computation of a distance field along a uniform grid can be accelerated by using graphics rasterization hardware. Previous algorithms compute 2D slices of the 3D distance field by rendering the three dimensional distance function of each primitive. However, rendering the distance meshes of all the primitives for each slice is expensive in terms of transformation and rasterization cost. As a result, current algorithms for 3D distance field computation may be slow and not work well for complex deformable models or dynamic environments. We have developed a fast algorithm (DiFi) to efficiently compute a distance field of complex objects along a 3D grid, by using a combination of novel culling and clamping algorithms. DiFi is applicable to all geometric models and does not make any assumptions about connectivity or a manifold representation. We have demonstrated its application to compute the simplified medial axis of deformable polyhedral models and perform proximity computations in a dynamic environment.

Hugo Model (17K polys)

Cassini Spacecraft (93k polys)

3D Distance Fields: Distance to surface is color coded, increasing from red to green to blue

Highlights

- **Generality: Applicable to all polygonal and image models**
- **Efficiency: Achieves two orders of magnitude speedup over CPU and 4-20 times speedup over previous GPU approaches**
- **Geometric Properties: Exploits Voronoi region properties and spatial coherence to reduce computations**
- **GPU Optimization: Performs geometric tests efficiently using GPU features**
- **Dynamic Models: Involves no preprocessing and is applicable to dynamic models**

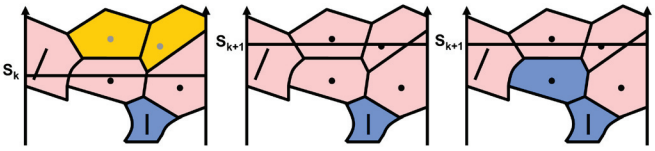
GPU-based Algorithm

Rasterization graphics hardware is particularly well suited for parallel computation of distance functions on a uniform grid. Our algorithm further exploits features of modern GPUs to perform computations efficiently. The key steps of the algorithm are:

Culling: We compute conservative bounds on the Voronoi regions to cull away primitives that do not contribute to the distance field of a given slice. This culling is performed using occlusion queries on GPUs. Furthermore, we present a conservative sampling strategy to account for sampling errors in the occlusion queries.

Culling:

- Use Voronoi region bounds to compute a conservative set of intersecting sites which contribute to distance field of a slice
- Update intersecting set incrementally for next slice



Slice k: Given an intersecting set

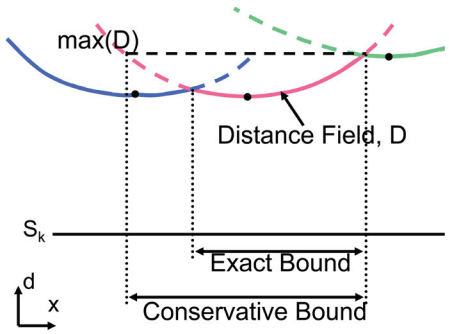
Slice k+1: Add appropriate approaching sites to get new intersecting set, Compute distance field

Slice k+1: Remove appropriate receding sites to get final intersecting set

Clamping: Conservative bounds of the Voronoi regions are used to limit the portion of a slice on which each distance function is computed. The bounds can be tightened further using connectivity information of manifold surfaces.

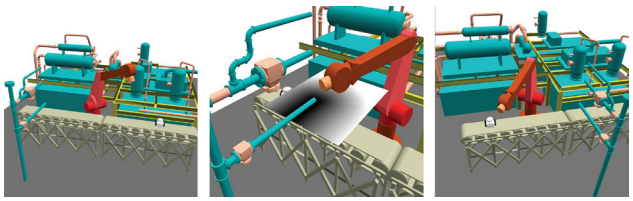
Clamping:

- Use Voronoi region bounds to limit portion of the slice on which each distance function is computed
- Compute conservative bounds using max value of distance field
- Max value of distance field incrementally computed for next slice



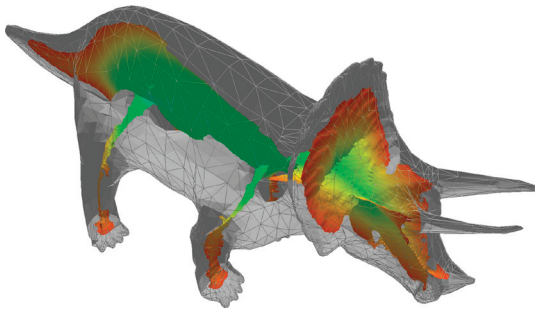
Applications and Results

Recent advances in graphics hardware, like programmability and 32-bit fioating point precision, have enabled many applications to be performed efficiently on the GPU by exploiting the SIMD nature of graphics hardware. We have applied DiFi to compute a simplified medial axis transform of polygonal models on the GPU. This avoids the costly readback of the entire distance field to the CPU. DiFi is also used for proximity computations within a constraint-based path planner. The distance field is recomputed as objects in the environment move. This path planner has been used for virtual prototyping applications. Demos of both applications are shown in the video online.



Planning in an assembly environment: The robot arm tracks a moving part across a conveyor belt, while avoiding contact with other obstacles in real time

We have used DiFi to compute the 3D distance field of several manifold and non-manifold benchmark models. DiFi obtains more than two orders of magnitude improvement over a CPU algorithm and 4-20 times speedup over a previous GPU based algorithm. For medial axis computations, the GPU implementation achieves an order of magnitude speedup over the CPU implementation.



Simplified medial axis of Triceratops model (5.6k polys). Surface is color coded by distance to boundary

Project Leader

Dinesh Manocha, professor

Graduate Research Assistants

Avneesh Sud
Miguel A. Otaduy

Research Sponsors

U.S. Army Research Office
Office of Naval Research
Defense Advanced Research Projects Agency

Selected Publications

Hoff, K., T. Culver, J. Keyser, M. Lin, and D. Manocha. Fast computation of generalized voronoi diagrams using graphics hardware. *Proceedings of ACM SIGGRAPH*, 277-286, 1999.

Sud, A., M.A. Otaduy, and D. Manocha. DiFi: Fast 3D distance field computation using graphics hardware. *Proceedings of Eurographics*, 2004.

Key Words

Distance fields, Voronoi regions, graphics hardware, proximity computations

For More Information

http://gamma.cs.unc.edu/DiFi

http://gamma.cs.unc.edu/DiFi