GPU-Accelerated Ray Casting of Node-Based Implicits

1 Introduction

Implicit surfaces are a popular surface representation for the modeling and animation of organic shapes and natural phenomena. We present a novel GPU ray casting engine for the direct and efficient visualization of node or skeleton-based distance surfaces.

2 Algorithm

The surface is defined as an iso-value contour of the field function S_F. The implicit surface is a superposition of local field functions defined at a finite number of nodes,

$$S_F = \sum K(\bar{x}, \bar{x_i}),$$

where $K(x, x_i)$ is a strictly monotonic decreasing function. Today's most popular techniques for visualizing these surfaces are marching cubes and surfel-based techniques, both of which rely on generating an intermediate representation. This is both memory and compute intensive. An alternative approach for visualizing node-based implicit surfaces is ray casting or ray tracing [Blinn 1982]. The large computational cost of ray-based visualization has traditionally precluded these techniques from real-time rendering of skeleton-based implicit surfaces.

Commodity graphics hardware can provide significant computing power; however, employing the hardware efficiently can be difficult due to architectural constraints. As demonstrated previously, ray casting of discrete implicit surfaces can be implemented efficiently on current GPUs. In contrast to existing techniques, we use an algebraic representation of the implicit surface, yielding pixel-accurate surfaces in real-time with a small memory footprint.

Our technique performs the ray-surface intersection tests in the fragment shader. The test is performed on a set of candidate rays determined by rasterizing the convex hull of the surface. This set is usually significantly smaller than the set of all eye rays. In general, the convex hull of the density field in space cannot be determined a priori. Only for certain applications, such as Lagrangian simulations with a proximity constraint on the nodes, is this possible.

Therefore, we propose a more general and rigorous approach to limit the domain of the surface field. Instead of S_F, an alternative field function $\hat{S}_F$ is used:

$$\hat{S}_F = \frac{\sum K(\bar{x}, \bar{x_i})^2}{\sum K(\bar{x}, \bar{x_i})}.$$

With this surface field function, the superposition of the convex hulls lies within the convex hull of the superposition of the nodes. We employ Newton iteration for the intersection test with initial guesses based on the intersection between the ray and the central node whose convex hull spawned the ray [Blinn 1982]. The sum over all neighboring nodes must be computed during each Newton iteration step. However due to hardware limitations on the current generation of GPUs, the number of terms used for the sum has to be limited. Omitting many terms may lead to non-smooth surfaces. We therefore approximate the sum by an ambient term stored in a discrete representation of the field, the *ambient field*. We note that the sum over the closest neighbors

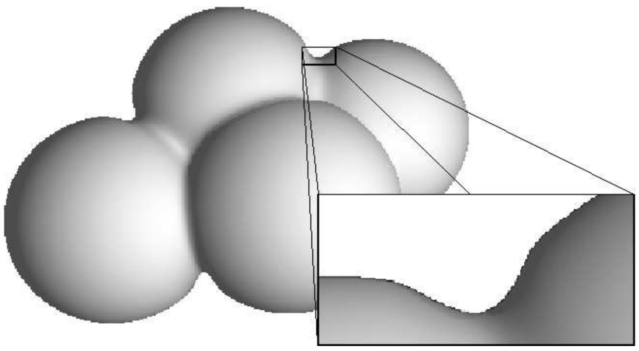

Figure 1: Surfaces blended with $\hat{S}_F$ at 35 FPS.

and the value from the ambient field are combined with a weighted sum. The volume texture resolution can be used to trade performance for quality on slower hardware. The release of DirectX 10 compliant hardware later this year will enable the computation of the entire surface field sum without requiring an ambient field.

3 Implementation

Our system is implemented in C++ with OpenGL 2.0 and GLSL on both Linux and Windows platforms. For simplicity, axis-aligned bounding boxes are employed as convex hulls. We also implemented a CPU-based Lagrangian fluid simulator with collision detection to illustrate the interactivity of our method [Müller et al.]. This demonstrates that our representation allows a node-based physical simulator to directly generate rendered results from simulation nodes.

4 Results and Future Work

We present an accurate, fast and memory efficient rendering technique for node-based implicit surfaces based on hardware-accelerated ray casting. All of the results were generated on a Pentium D 3.2 GHz with 1 Gbyte of RAM and an Nvidia 7800 GT with 256 MB of RAM. We are able to interactively simulate a Lagrangian fluid with 100 blended nodes over arbitrarily long periods of time at a sustained frame rate greater than 5 FPS.

Next-generation GPUs will eliminate the costly generation and read-write penalties incurred by the ambient field volume texturing. We also propose performing both the physical simulation and rendering calculations on the GPU as well as real-time modeling with blobbies as future work.

References

Blinn, J. F. 1982. A generalization of algebraic surface drawing. *ACM Trans. Graph. 1*, 3, 235–256.

Müller, M., Charypar, D., and Gross, M. Particle-based fluid simulation for interactive applications. *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer animation.*