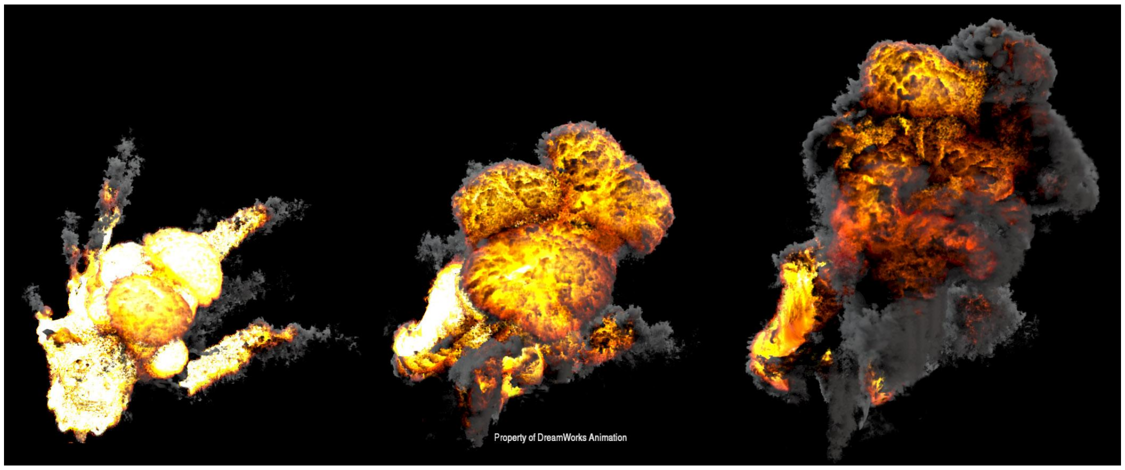
Amorphous: An OpenGL Sparse Volume Renderer

Mark J. Matthews

DreamWorks Animation, SKG



We present *Amorphous*, a GPU volume renderer implemented in OpenGL and optimized for sparse volumes. Amorphous correctly computes the volume rendering integral along each ray, giving physically accurate results. Our collection of techniques elegantly solves common issues with volume slice renderers while offering interactive performance. High visual quality at interactive rates translates into significant production gains, tightening the loop between volumetric modeling, simulation and visual feedback, and reducing the need for offline renders.

Visual Correctness

Image-order rendering algorithms, such as ray-marching [Maury et al. 2010], iterate over viewing rays and access volume data as needed. This is elementary for dense data, but sparse access requires multiple indirection and code-branches, which does not pipeline well on GPUs. Instead we use object-order slice renderering [Cullip and Neumann 1993] that exploits the inherent speed of GPU texture units, while easily coping with our sparse data.

Many slice renderers incorrectly compute fragment thickness. This leads to popping artifacts if axis-aligned slicing is used. Camera-aligned slices eliminate this popping, but do not necessarily compute the proper fragment thickness.We eliminate the need for camera-aligned slices and instead exploit the covariant transformation of the slice normal. Doing so allows us to quickly compute the precise per-fragment thickness, yet slice in convenient axis-aligned planes. As long as values for extinction, albedo and incandescence are specified, the ray integral can be computed correctly.

Floating point frame buffers are important for obtaining good results. Without them, banding artifacts become prevalent unless adaptive slicing is used. Floating point frame buffers are also necessary for explosions, since incendiary explosions generally have high internal intensity, much greater than 1, which is then attenuated by the outer volume to a value below 1. Using these techniques we achieve results that have accuracy similar to ray marching on the CPU.

We discuss special considerations for frustum buffers, which are particularly useful for rendering volumes only visible from a camera. We present a novel transformation that allows a proper computation of covariant slice normals.

For shadowing, we use multiple dense opacity maps to provide a complete single-scattering solution. Computing a true multiple-scattering solution is expensive, and simpler techniques can be used to approximate the same look.

Sparse Volumes

The ability to render sparse volumes is extremely desirable given the $O(N^3)$ memory requirement of dense volumes. Amorphous natively supports the sparse volume format OpenVDB [Museth 2012], a library comprising a compact hierarchical data structure and a suite of tools for volumetric modeling and simulation. However, sparse volumes present a number of problems for dense GPU rendering techniques.

Amorphous relies on blocked data which is tightly packed into GPU memory, granting significant memory gains. Blocks are seamlessly stitched together for linear interpolation by storing an extra upwind layer of voxels and picking appropriate texture values. Special consideration must be given to determining the slicing direction and the block drawing order to reduce artifacts for perspective viewing.

Production Examples

We show examples based on cloud modeling using frustum buffers developed during *Puss in Boots*. Interactive visualization using our custom renderer was critical for production because offline renders were too slow for interactive modeling work, and volume resolution was generally to high for conversion to an equivalent dense volume for display. We also show examples from several current productions for visualization of explosions that makes use of our incandescence model. Although other shaders were used for final renders in these cases, the ability to interactively visualize simulations makes Amorphous a valuable tool for modeling and simulation development.

CULLIP, T., AND NEUMANN, U. 1993. Accelerating volume reconstruction with 3D texture hardware. Tech. Rep. TR93-027, University of North Carolina, Chapel Hill.

MAURY, O., PIPONI, D., ANDORRA, F., AND HAMMAČK, C. 2010. Bending fire with Plume, a CUDA-based 3D fluid solver an d volume renderer. In *ACM SIGGRAPH 2010 Talks*, ACM, New York, NY, USA, SIGGRAPH '10.

MUSETH, K. 2012. Vdb: High-resolution sparse volumes with dynamic topology. *In submission*.

Copyright is held by the author / owner(s).
SIGGRAPH 2012, Los Angeles, California, August 5 – 9, 2012.
ISBN 978-1-4503-1435-0/12/0008