Hierarchical Digital Differential Analyzer for Efficient Ray-Marching in OpenVDB

Ken Museth
DreamWorks Animation

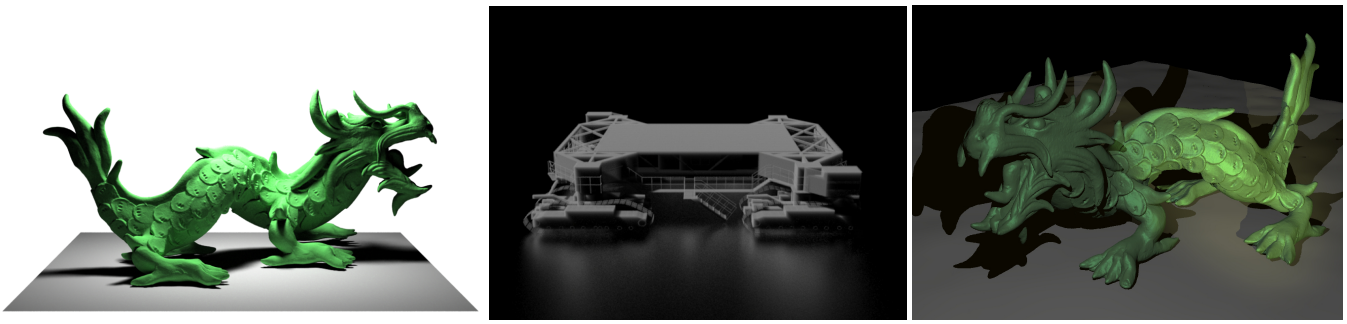

Figure 1: Ray-tracing utilizing our accelerated hierarchical digital differential analyzer. **Left:** Surface rendering of a VDB using Pixar's RenderMan. **Middle:** Volume rendering of a VDB using Solid Angle's Arnold. **Right:** Surface rendering of a VDB using SideFX's Mantra.

Introduction

VDB[Museth 2013] is a compact data structure and toolset developed at DreamWorks Animation for high-resolution volumetric effects typically encountered in movie production. Since its open source release in 2012, as OpenVDB[1], it has been adopted by major third-party renders and VFX tools, including Houdini by SideFX, RenderMan by Pixar, Arnold by Solid Angle, and RealFlow by Next Limit. Thus, it should come as no surprise that it is highly desirable to have efficient algorithms for ray-marching of sparse VDB volumes. The core problem is therefore, how to best utilize the hierarchical data structure of VDB for applications that require efficient ray-marching. As we will demonstrate, the solution is a novel hierarchical digital differential analyzer.

Hierarchical Digital Differential Analyzer

A Digital Differential Analyzer, or DDA, is in fact an old idea in computer graphics that, either in software or hardware, facilitates efficient rasterization of lines. As such DDAs are conceptually related to the celebrated Bresenham's line algorithm. In the context of ray-marching DDAs have been successfully adopted to efficiently determine the voxels or tiles that are intersected by a given ray. The main advantage of a DDA is of course its ability to quickly traverse, or leapfrog, empty space and produce intersection points or intervals parameterized along the ray.

What sets our work on VDB apart from this large body of existing applications of DDAs, is the fact that by virtue of the hierarchical structure of VDB we have developed a highly specialized (and optimized) Hierarchical DDA that takes full advantage of the underlying B+tree structure of VDB. The result is a novel algorithm that performs multi-resolution leapfrogging at the various levels of the VDB structure. Since VDB is also characterized by having several properties of its tree configuration fixed at compile-time, we have utilized techniques like Template Meta Programming to effectively inline and collapse the hierarchical algorithm, so as to reduce the overhead of having to navigate the tree vertically. The implementation is already available in OpenVDB v2, and we present the details of the underlying algorithm as well as its applications.

Applications

We have found the Hierarchical Digital Differential Analyzer to be useful for a multitude of graphics related problems. Two of the most obvious applications are rendering of narrow-band level set surfaces, and volumes that represents density, e.g. clouds.

To facilitate ray-tracing of level set surfaces we have developed a simple but fast ray-level-set intersection algorithm that internally relies on a HDDA. Another distinguishing feature of this algorithm is the fact that it minimizes interpolation of voxels, which can be costly with high-order kernels.

Volume rendering is arguably an even more important candidate for acceleration by the HDDA. Volumes associated with a linear transformation are relatively straight forward to accelerate with the HDDA, because rays are linear in both world and index space. However, grids with non-linear transforms, e.g. a grid warped to a camera frustum, pose a challenge since shadow (vs camera) rays are not guaranteed to march along straight lines in index space. Our solution is to construct an auxiliary VDB with a linear transform that can be ray-marched using a HDDA. Since this auxiliary VDB basically acts as a mask for the active values in the frustum volume, it can be generated at a lower voxel resolution, and the voxel values are encoded with just a single bit, making them extremely compact. The computational effort to construct this auxiliary grid is relatively low since it can be performed as a concurrent resampling of the density grid with a frustum transform onto a binary grid with a linear transform. Finally, we can perform fast bit-wise topological dilation of the binary grid to better account for the size of the interpolation kernel employed during sampling along the ray.

References

MUSETH, K. 2013. VDB: High-resolution sparse volumes with dynamic topology. *ACM Trans. Graph. 32*, 3 (July), 27:1–27:22.

[1] http://www.openvdb.org