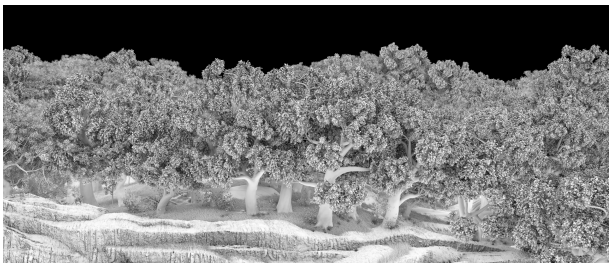
Coherent Out-of-Core Point-Based Global Illumination

Janne Kontkanen * Eric Tabellion† Ryan S. Overbeck‡

DreamWorks Animation

(a) "City" scene from the movie *"Kung Fu Panda 2"*.
128 million points, 7 GB, octree build time: 4:18s.
PBGI shading for 3.2 million visible micropolygons: 6:23s.
PBGI used for diffuse and glossy global illumination.



(b) "Forest" scene from the movie *"Shrek Forever After"*.
927 million points, 51 GB, octree build time: 37:02s.
PBGI shading for 18.8 million visible micropolygons: 24:19s.
PBGI used for ambient occlusion.

Figure 1: *Two production scenes from DreamWorks Animation movies that were rendered using the out-of-core method described in this talk. Timings are given in mm:ss. Both scenes were built and shaded within a modest memory cap of 2 GB for (a) and 3.1 GB for (b). In our talk, we show scenes of up to 1.7 billion points (88 GB of data) built and shaded within a 2 GB memory cap.*

1 Introduction

We introduce a new set of techniques for coherent out-of-core point-based global illumination and ambient occlusion.

As described in [Christensen 2008], point-based global illumination (PBGI) operates on a dense point representation of the scene. Points are diffusely shaded and are roughly distributed according to a micropolygon tessellation. An octree is constructed over the points, where each octree node describes the light reflected from the corresponding cluster of points to any direction. The reflected radiance is stored as spherical harmonics coefficients to compactly capture directional variation. After the octree has been constructed, each micropolygon is shaded by first selecting a set of octree nodes that define a suitable level of detail, and then accumulating the radiance stored in these nodes using a cube rasterizer.

PBGI is used in film production to render tremendously complex scenes, therefore in-core storage of point and octree data structures quickly becomes a problem. Furthermore, a simple out-of-core extension of a classical top-down octree building algorithm would be extremely slow due to the large amount of I/O required. Our work addresses these problems and extends previous PBGI algorithms with efficient out-of-core techniques. Using properties of a space-filling Z-curve, our octree build requires minimal I/O and additionally stores data on disk compactly and in coherent chunks. Later during shading, data is accessed using on-demand paging and caching. Our parallel system runs entirely within a small fixed memory cap while handling extremely complex scenes, as shown in Figure 1.

2 Out-of-Core Octree Build

Our primary contribution is an out-of-core octree build algorithm that is able to preprocess the point data in just two passes: an external 1D-sort and a sequential octree construction pass. Our method requires minimal amount of I/O and stores both the points and the octree nodes coherently on disk. We first sort the points along a Z-curve, based on their Morton code. Utilizing the well-known property that the Z-curve exactly follows the depth first traversal [Morton 1966], we build the PBGI octree in a single sequential streaming pass over the sorted points. This algorithm builds upon an N-body simulation method developed by Salmon and Warren [1997]. We also introduce a technique to organize the octree nodes into coherent chunks at no additional cost, which can improve the final out-of-core shading performance more than $4\times$.

3 Out-of-Core Shading

During the final shading, we shade each visible micropolygon by choosing the suitable level of detail from the out-of-core octree. We perform a top-down recursive depth-first traversal of the octree, until a refinement criteria is satisfied. During this traversal, we load pages of octree nodes and points on-demand to minimize memory use. We also utilize a two-level LRU in-core cache to avoid redundant I/O and to scale to multiple processors. The portions of the octree and point data not required for shading are never loaded into memory.

4 Conclusion

Our out-of-core PBGI algorithm allows us to handle unprecedented scene complexity while operating within a modest memory cap. Our novel out-of-core octree construction algorithm only requires a single streaming pass over the points after they have been sorted in Morton order. We organize the octree into coherent chunks, speeding up shading by up to $4\times$. Our fully parallel final shading algorithm only loads the parts of the octree that are required. All these pieces come together to provide a coherent, parallel out-of-core system for rendering high quality global illumination for extremely complex scenes. We demonstrate our system on real production scenes of up to 1.7 billion points. Fore more details: **www.tabellion.org/et/paper11/index.html**

References

CHRISTENSEN, P. H. 2008. Point-based approximate color bleeding. Tech. Rep. 08-01, Pixar.

MORTON, G. M. 1966. A computer oriented geodetic data base; and a new technique in file sequencing. Tech. rep., IBM Ltd.

SALMON, J., AND WARREN, M. S. 1997. Parallel, Out-of-core Methods for N-body Simulation. In *SIAM Parallel Processing for Scientific Computing*.