Interactive Visualization of Exceptionally Complex Industrial CAD Datasets

Andreas Dietrich
Saarland University
dietrich@cs.uni-sb.de

Ingo Wald
Max-Planck-Institute for Computer Science
wald@mpi-sb.mpg.de

Philipp Slusallek
Saarland University
slusallek@cs.uni-sb.de

Abstract

Graphics hardware has dramatically evolved over the last years. Even consumer cards are able to render many million triangles per second. While this seems sufficient for most small and mid-sized scenes, many industrial CAD datasets are way to complex to be directly displayed interactively on current graphics accelerators, due to the limitations of the underlying rasterization algorithms.

This project aims at interactively visualizing extremely large datasets with a software ray tracing approach. We demonstrate this using a "Boeing 777" model containing *350 million* individual triangles. The model can be rendered at several frames per second including pixel-accurate shadows and highlights at full detail without any simplifications, even on a single commodity PC.

1 Introduction

The "Boeing 777" has been Boeing's first aircraft to be completely designed using a CAD framework for every single part. Although our evaluation model is missing some components, it already consists of more than 350 million triangles (see Figure 1). With minimal geometric information, i.e. even without any normal or shading data, the raw model exceeds 12 GByte in size, requiring many minutes simply to be loaded from disk.

Current high-end graphics hardware featuring a rasterization performance of a few hundred million triangles per second could theoretically produce an image in a few seconds. In practice, however, this can take up to minutes, especially if the data does not fit into main memory.

In contrast to rasterization technology, ray tracing algorithms exhibit logarithmic time complexity with respect to the number of triangles, due to the employment of acceleration structures, i.e. a spatial index. Because of the *output sensitive* nature of ray tracing only those parts of a scene are accessed that are actually visible. Therefore, only a fraction of the model data has to be kept in core, while the remaining components can be loaded *on demand*.

With recent advances in interactive ray tracing [Wald et al. 2003; Wald 2004] massively complex models can now be handled at rates of several frames per second. Our system incorporates an approach similar to [Wald et al. 2001] and also builds on the OpenRT interactive ray tracing architecture, but has been significantly enhanced and optimized in order to handle the large amount of data in the "Boeing 777" airplane model.

2 System Architecture

For our experiments we used a single dual-processor AMD Opteron PC with 6 GByte RAM. The 64-bit architecture of the Opteron makes it possible to map the complete disk data of the scene into the ray tracer's virtual address space including all acceleration structures. Special care must be taken to prevent the operating system from stalling the ray tracing process while demand-paging new data. We implemented a memory management system that allows

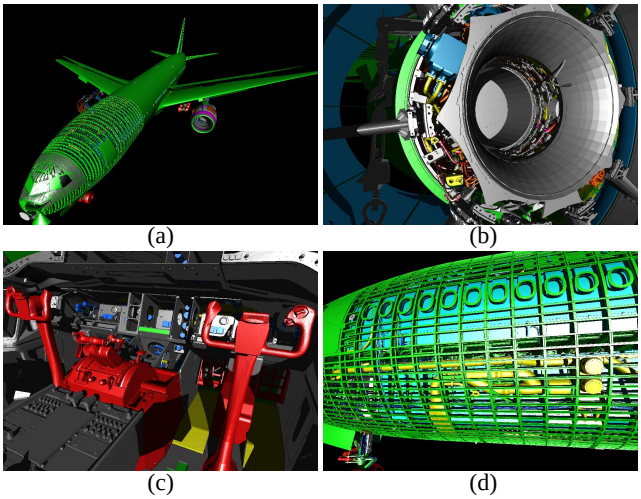

Figure 1: *Various sections of the "Boeing 777" model. These views can be rendered with up to 5 fps at video resolution (*640×480*) including pixel-accurate shadows and highlights. (a) Overview of the complete model. Through the open hull parts of the interior structure are visible. (b) Look into one of the engines. Note the highly interwoven components. (c) Some parts of the cockpit. (d) Closeup view on the open fuselage. The shadows considerably improve depth perception.*

to detect potential page faults when trying to access unavailable memory pages. Additionally, the application retains full control over what pages are cached in RAM.

Missing pages are loaded asynchronously parallel to rendering. Until this data becomes accessible several strategies are applied to bridge loading time and to provide instant visual feedback while retaining full interactivity:

As our acceleration structures (kd-trees) closely encompass the scene's geometry, they are used for approximating the missing triangle data. In combination with pre-computed additional information , such as normals and colors (so-called "Proxy" structures), this enables progressive visualization (much like progressive JPEG) until the fully detailed information is present in main memory.

References

WALD, I., SLUSALLEK, P., AND BENTHIN, C. 2001. Interactive Distributed Ray Tracing of Highly Complex Models. *Rendering Techniques 2001*, 274–285. (Proceedings of the 12th Eurographics Workshop on Rendering).

WALD, I., BENTHIN, C., DIETRICH, A., AND SLUSALLEK, P. 2003. Interactive Ray Tracing on Commodity PC Clusters – State of the Art and Practical Applications. *Lecture notes on Computer Science*. (Proceedings of EuroPar).

WALD, I. 2004. *Realtime Ray Tracing and Interactive Global Illumination*. PhD thesis, Computer Graphics Group, Saarland University. (to appear, preprint available on request).