Large Model Visualization: Techniques and Applications

Dirk Bartz
WSI/GRIS
University of Tübingen
Sand 14,
D-72076 Tübingen, Germany
bartz@gris.uni-tuebingen.de

ABSTRACT

The visual representation of complex data has always been a major motivation for computer graphics. However, there have always been computer graphics scenes which were too complex to be rendered within a reasonable time limit, or even too complex to be rendered at all with the available resources. Nevertheless, computer graphics has become one of the primary tools used for the interpretation of data from engineering or science. By the end of the eighties, the application of computer graphics methods for the visual interpretation of scientific data became a research field on its own and was called *Scientific Visualization*. By now, it is almost unthinkable to understand data, which is generated by simulations or is measured, without any graphics or visualization technology. The technology used ranges from simple plot drawings, via direct volume rendering techniques to provide semi-transparent view through a dataset, to illuminated streamlines, and line integral convolutions to visualize the flow of particles through multi-dimensional and multi-variate data.

Keywords
Large Models Visualization, Mesh Reduction, Mesh Compression, Parallel Computing, Occlusion Culling, Image-based Rendering, Point-based Rendering, Volume Rendering, Virtual Environments

1 INTRODUCTION

In recent years, large model visualization or Large Scale Data Visualization (LSDV) became one of the most important research fields in scientific computing. The reason of the emergence of LSDV lies in the fast increasing size of datasets from various sources. In the United States, research efforts are mostly driven by the Accelerated Strategic Computing Initiative (ASCI) of the US Department of Energy (DOE), focusing on nuclear weapon research, and the Large Scientific and Software Data Set Visualization program (LSSDSV) of the US National Science Foundation (NSF), motivated by simulation of natural phenomena (i.e., global and regional weather, ocean dynamics, high energy and astro-physics, etc.). Besides these initiatives, the increasing dataset size of medical scanners (i.e., multi-slice Computer Tomography, rotational biplanar X-ray) and design review tasks in product data management systems (PDM) drive the need for techniques for large datasets:

- The generated data volumes of simulations from scientific computing can easily grow into the range of tera-bytes.
- The size of scientific measured data frequently exceeds tera-bytes of storage space, not only in academic experiments, but also in commercially driven scientific tasks like in flow experiments in the aircraft and automotive industry, or the oil-and-gas exploration.
- Design review tasks in computer-aided engineering (CAE) have to deal with tessellated, polygonal models of up to 100 million polygons.
- Medical scanners routinely generate data volumes with a resolution of 512^3 voxels – some

Journal of WSCG, Vol.11, No.1., ISSN 1213-6972
WSCG 2003, February 3-7, 2003, Plzen, Czech Republic.

scanners like multi-slice CT even generate more than 1000 image slices in one scan; modern high field MRI scanners can go up to a slice resolution of 2048^2 voxels.

Techniques for the handling of large datasets include database management, architectural aspects of large computing systems, parallel computing, and last but not least, rendering techniques for the visualization of large datasets. Here, we focus on techniques for the advanced visualization of large datasets.

2 LARGE MODEL VISUALIZATION

Two issues are usually the major subject of large data handling; memory efficiency and rendering performance. However, many standard visualization techniques require substantial auxiliary data like spatial data-structures or distance fields which are usually computed in a pre-process. Storing this data can exceed the memory capacities of the visualization host computer, prompting the use of different visualization algorithms. Some visualization applications, i.e. design review tasks or intra-operative navigation-based visualization, require a certain rendering performance to provide *interactive* or even *real-time frame-rates*, where an interactive frame-rate usually specifies more than five frames-per-second (fps), and a real-time frame-rate more than 20 fps. Currently available top-of-the-line computer graphics accelerators achieve a sustained performance of several million triangles per second, which is only satisfactory for the interactive rendering of medium sized models. Unfortunately, the data volume generated by applications in architecture, medicine, mechanical engineering, or scientific computing grows faster than the rapidly increasing graphics performance of modern graphics subsystems. This growing divide requires approaches which reduce the complexity by an order of magnitude.

Several methods have been proposed in the recent years to address this divide [SBK+02]. Many of these methods have a hierarchical scene representation in common which usually provides different levels-of-detail. However, this hierarchy has to be constructed, an operation which is potentially very expensive, and imposes additional space requirements on the actual application. Consequently, some of these approaches are extended to be able to work *out-of-core*, which means that they no longer rely on the size of the main memory, but only on the size of the hard disk [SBK+02].

Probably the best known class of methods are mesh-reduction approaches, which reduce the rendering complexity of the given geometry data depending on the required rendering performance or quality. A recent survey on mesh-reduction approaches can be found in [Gar99]. In contrast, subdivision progressively refines a coarse polygonal base mesh until a specifiable error threshold is satisfied [ZSD+00]. If only a limited transfer bandwidth is available, geometry compression methods can be applied to reduce the storage size of a model [Tau99; TDG+00]. Parallel processing of a given problem reduces the per-pipeline rendering complexity by increasing the number of processing pipelines with the number of CPUs. However, potential bottlenecks, required data replication, or synchronization overheads prevent many applications from achieving an optimal speed-up. While parallel rendering concentrated on large SIMD supercomputers in the past, it experienced a renaissance on modern symmetric multi-processing (SMP) systems, large non-uniform memory access (NUMA) computers, or on clusters of single PC-class or RISC-based workstations [BSS00]. In particular the ASCI and LSSDSV programs drive the development of methods for large NUMA- and cluster-based systems.

All the approaches so far address the lack of rendering performance by reducing the polygonal complexity of objects, or by distributing the rendering load to several processing entities. However, the overall rendering complexity of individual pixels remains the same. In contrast, visibility and occlusion culling approaches reduce that pixel complexity by removing geometry which is not visible from a specific view-point. In depth-complex scenes – where many polygons are rasterized at the same pixels of the framebuffer, due to the same location in image-space – visibility and occlusion culling enables a reduction of the polygonal complexity of up 90% [BMH99; BS99]. If interactive rendering needs to be guaranteed, a budget-oriented rendering system can be applied, which may skip rendering of parts of the models if the budget is not sufficient for the entire model [HP98; BSS+01; BMH99; KS99].

A technique especially suitable for architectural walkthroughs is image-based rendering [DBC+00], where distant parts of the geometric model are approximated by an image (i.e., a texture), since their visual appearance is not changing much [AMC+00]. This method also reduces the per-pixel-complexity of a rendered frame. A somewhat related approach is point rendering which computes the required object-space geometry based on a sampling of the image-space [PZBG00; RL00; WFP+01]. Therefore, the complexity of the rendering is determined by the image-space complexity, not by the geometric complexity of the model. A similar approach was already proposed by Cline et al. in 1988 [CLL+88], which used attributed points instead of triangles to render volume datasets from medical scanners. In contrast to the recent methods, the point rendering complexity was determined in object-space.

Volume rendering approaches the problem from a different side [CMVK02]. The model is no longer represented as a set of polygons, but as a discrete, volumetric set of samples. Depending on the requested modeling details, the space requirements can be substantially smaller than with a polygonal representation [BM99]. For an evaluation of the FabFour of volume rendering, please check [MHB+00]. Methods of virtual environments provide different interaction methods for the user of large data. In contrast to traditional rotating, translating, and zooming, the user can interact with the models in a more intuitive way. See [SC02] for a recent introduction into virtual environments as interaction methodology.

3. REFERENCES

[AMC+00] D. Aliaga, D. Manocha, J. Cohen, S. Kumar, T. Funkhouse, M. Lin, A. Wilson, and D. Luebke. Interactive Walkthroughs of Large Geometric Datasets. In *ACM SIGGRAPH Course 18*, 2000.

[BM99] D. Bartz and M. Meißner. Voxels versus Polygons: A Comparative Approach for Volume Graphics. In *Proc. of Volume Graphics*, pages 33–48, 1999.

[BMH99] D. Bartz, M. Meißner, and T. Hüttner. OpenGL-assisted Occlusion Culling of Large Polygonal Models. *Computers & Graphics*, 23(5):667–679, 1999.

[BS99] D. Bartz and M. Skalej. VIVENDI - A Virtual Ventricle Endoscopy System for Virtual Medicine. In *Proc. of Symposium on Visualization*, pages 155–166,324, 1999.

[BSS00] D. Bartz, C. Silva, and B. Schneider. Rendering and Visualization in Parallel Environments. In *ACM SIGGRAPH Course 13*, 2000.

[BSS+01] D. Bartz, D. Staneker, W. Straßer, B. Cripe, T. Gaskins, K. Orton, M. Carter, A. Johannsen, and J. Trom. Jupiter: A Toolkit for Interactive Large Model Visualization. In *Proc. of Symposium on Parallel and Large Data Visualization and Graphics*, pages 129–134, 2001.

[CLL+88] H. Cline, W. Lorensen, S. Ludke, C. Crawford, and B. Teeter. Two Algorithms for the Three-Dimensional Construction of Tomograms. *Medical Physics*, 15(3)(3):320–327, 1988.

[CMVK02] B. Chen, K. Mueller, A. Varshney, and A. Kaufman. State of the Art in Data Representation for Visualization. In *IEEE Visualization Tutorial T9*, 2002.

[DBC+00] P. Debevec, C. Bregler, M. Cohen, R. Szeliski, L. McMillan, and F. Sillion. Image-Based Modeling, Rendering, and Lighting. In *ACM SIGGRAPH Course 35*, 2000.

[Gar99] M. Garland. Multiresolution Modeling: Survey and Future Opportunities. In *Eurographics STAR report 2*, 1999.

[HP98] Hewlett-Packard. Jupiter 1.0 Specification. Technical report, Hewlett Packard Company, Palo Alto, 1998.

[KS99] J. Klosowski and C. Silva. Rendering on a Budget. In *Proc. of IEEE Visualization*, pages 155–122, 1999.

[MHB+00] M. Meißner, J. Huang, D. Bartz, K. Mueller, and R. Crawfis. A Practical Evaluation of Four Popular Volume Rendering Algorithms. In *Proc. of Symposium on Volume Visualization and Graphics*, pages 81–90, 2000.

[PZBG00] H. Pfister, M. Zwicker, J. Baar, and M. Gross. Surfels: Surface Elements as Rendering Primitives. In *Proc. of ACM SIGGRAPH*, pages 335–342, 2000.

[RL00] S. Rusinkiewicz and M. Levoy. Qsplats: A Multiresolution Point Rendering System for Large Meshes. In *Proc. of ACM SIGGRAPH*, pages 343–352, 2000.

[SBK+02] C. Silva, D. Bartz, J. Klosowski, W. Schroeder, and P. Lindstroem. High-Performance Visualization of Large and Complex Scientific Datasets. In *SuperComputing Tutorial M9*, 2002.

[SC02] R. Stone and K. Clark. Human-Centered Processes in Virtual Environments: Methodologies and Real-Life Case Studies. In *ACM SIGGRAPH Course 6*, 2002.

[Tau99] G. Taubin. 3D Geometry Compression and Progressive Transmission. In *Eurographics STAR report 3*, 1999.

[TDG+00] G. Taubin, M. Deering, C. Gotsman, S. Gumhold, and J. Rossignac. 3D Geometry Compression. In *ACM SIGGRAPH Course 38*, 2000.

[WFP+01] M. Wand, M. Fischer, I. Peter, F. Meyer auf der Heide, and W. Straßer. The Radomized Z-Buffer Algorithm: Interactive Rendering of Highly Complex Scenes. In *Proc. of ACM SIGGRAPH*, 2001.

[ZSD+00] D. Zorin, P. Schröder, T. DeRose, L. Kobbelt, A. Levin, and W. Sweldens. Subdivision for Modeling and Animation. In *ACM SIGGRAPH Course 23*, 2000.