Adaptive Visualization of Dynamic Unstructured Meshes

Student: Steven P. Callahan[1]
Advisor: Cláudio T. Silva[2]
Institution: University of Utah

1 Introduction

Research in scientific visualization has advanced to the point where there are now good and effective techniques for the analysis of static regular and unstructured volumetric data. Although not as advanced, there are also many techniques that address the problem of visualizing time-varying regular data sets [7]. This thesis deals with the visualization of dynamic unstructured meshes, an area that has been virtually untouched in the literature and in ongoing work.

The motivation for working on dynamic meshes comes from the fact that a large number of applications require this functionality. Advanced simulation codes now generate large amounts of time-varying unstructured meshes. In some of them, the actual geometry of the mesh stays the same over time, while only the computed fields change. In others, the geometry and topology of the meshes also change over time. Rendering of dynamic data has the potential to be applied in many areas of science, engineering, and even medicine.

Our proposed work is based on a fundamentally new approach where the volumetric data is streamed through the GPU, removing the need for extensive preprocessing or mesh connectivity information. It also provides the means for adaptively rendering the geometry for improved interactivity.

This work is seperated into three major components: providing tools for interactive exploration through *adaptive visualization* of large datasets; efficiently volume rendering the common case of unstructured meshes with *dynamic scalar fields*; and finally, handling the more complex case of volume rendering unstructured meshes with *dynamic geometry and topology*.

2 Background

A large step in creating a framework for adaptive visualization was presented in our previous thesis work [2]. The main contributions of this work are a hardware-assisted visibility sorting (HAVS) algorithm for unstructured volume rendering [5] and a dynamic level-of-detail approach for interactively rendering large meshes [4].

The HAVS algorithm [5] efficiently balances CPU and GPU computation, resulting in one of the fastest volume rendering algorithms for unstructured grids. In contrast to previous techniques, HAVS operates in both object-space and image-space. In object-space, the algorithm performs a partial sort of triangle primitives on the CPU in preparation for rasterization. The goal of the partial sort is to create a list of primitives that generate fragments in nearly sorted order. In image-space, the fragment stream is incrementally sorted using a fixed-depth A-buffer [6] implemented in hardware, called the k-buffer.

A key advantage of the HAVS algorithm is that it operates on a collection of triangles instead of tetrahedra, thus it requires little preprocessing and no neighbor information. Because the algorithm can use a different set of data for each frame, it can *naturally handle dynamic geometry*. This was recently exploited to perform dynamic level-of-detail rendering for large datasets [4]. The idea is to perform *sample-based simplification* of the data by rendering only a subset of it, in place of the more traditional *domain-based simplification* which collapses vertices or edges and forms a mesh with fewer primitives. This new level-of-detail approach is very simple and efficient because it requires no hierarchical mesh representation.

3 Adaptive Visualization

As data size increases, it becomes more difficult to manage the data efficiently. By keeping the data in one repository either on a server or in a database, we can reduce the storage cost of keeping it locally. As an example, consider a scientist working remotely who would like to visualize a large dataset on his laptop computer. Recently, we introduced a new progressive technique that allows real-time rendering of extremely large tetrahedral meshes [3]. The main idea is to create an effect similar to progressive image transmission over the internet. A server becomes a data repository

[1]stevec@sci.utah.edu
[2]csilva@cs.utah.edu

and a client (*i.e.*, a laptop with programmable graphics hardware) becomes a renderer that accumulates the incoming geometry and displays it in a progressively improving manner. This progressive strategy is unique because it only requires the storage of a few images on the client for the incremental refinement. For interactivity, a small portion of the mesh is stored on the client which uses a bounded amount of memory. Because the geometry is rendered in steps, the user can stop a progression and change the view without penalty, thus facilitating exploration. Our algorithm is robust, memory efficient, and provides the ability to create and manage approximate and full quality volume renderings of unstructured grids too large to render interactively at full resolution.

Proposed Research. Currently, our adaptive visualization algorithm only handles static meshes. We would like to extend the algorithm to handle dynamic data by streaming the changing scalar field, geometry, and topology.

4 Dynamic Scalar Fields

Datasets with static geometry and dyanamic scalar fields are common. There are four fundamental pieces to adaptively volume render dynamic data. First, compression of the dynamic data for efficient storage is necessary to avoid exhausting available resources. Second, handling the data transfer of the compressed data is important to maintain interactivity. Third, efficient volume rendering solutions that handle changing data are necessary. Finally, maintaining a desired level of interactivity or allowing the user to change the speed of the animation is important for the user experience. In recent work [1], we address these issues in a system that extends the HAVS algorithm with a sample-based level-of-detail strategy that targets changing scalar fields and data compression that reduces the data transfer of the additional data from the CPU to the GPU. Our solution provides an adaptive framework for moderately-sized datasets with little additional rendering overhead.

Proposed Research. We would like to explore better compression algorithms that can be more efficiently used on the GPU and allow much larger datasets by reducing the CPU to GPU transfer.

5 Dynamic Geometry and Topology

The most complex type of dynamic data is that in which geometry and/or topology change over time. Like the case of dynamic scalar fields, this poses a difficult task in both compression and visualization algorithms. Consider, for example, a mesh created from sensors placed directly on a beating heart. New vertices and primitives may be added or removed between subsequent time-steps. Currently, this type of data is handled in a completely brute-force manner because the technical challenges to be overcome have thus far eluded visualization researchers. Our adaptive framework provides the necessary flexibility to handle this type of data.

Proposed Research. By extending existing compression schemes to handle changes in space and time, we can leverage the power of our streaming HAVS volume renderer to efficiently render these large datasets. We would like to explore methods to reduce the data size, allow user control of the speed and quality of the visualizations, and facilitate interactive exploration of the time-steps.

References

[1] F. Bernardon, S. Callahan, J. Comba, and C. Silva. Interactive volume rendering of unstructured grids with time-varying scalar fields. In *Eurographics Symposium on Parallel Graphics and Visualization*, pages 51–58, 2006.

[2] S. Callahan. The K-buffer and its applications to volume rendering. Master's thesis, University of Utah, 2005.

[3] S. Callahan, L. Bavoil, V. Pascucci, and C. Silva. Progressive volume rendering of large unstructured grids. In *IEEE Visualization '06*, 2006. to appear.

[4] S. Callahan, J. Comba, P. Shirley, and C. Silva. Interactive rendering of large unstructured grids using dynamic level-of-detail. In *IEEE Visualization '05*, pages 199–206, 2005.

[5] S. Callahan, M. Ikits, J. Comba, and C. Silva. Hardware-assisted visibility ordering for unstructured volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):285–295, 2005.

[6] L. Carpenter. The A-buffer, an antialiased hidden surface method. In *Computer Graphics (Proceedings of ACM SIGGRAPH)*, volume 18, pages 103–108, July 1984.

[7] K.-L. Ma and E. Lum. Techniques for visualizing time-varying volume data. In C. D. Hansen and C. Johnson, editors, *Visualization Handbook*. Academic Press, 2004.