Adaptive Visualization of Dynamic Unstructured Meshes

Steven P. Callahan*
Scientific Computing and Imaging Institute, University of Utah

Abstract

Visualization of time-varying volume data is essential for understanding the results of scientific simulations. Interactive exploration of these datasets is a challenging problem since the data is often unstructured and may contain many time and domain field instances. We propose an adaptive framework for interactively volume rendering large dynamic unstructured meshes on a commodity PC by efficiently leveraging programmable graphics hardware. Our goal is to provide a user with an tool for interactively managing and exploring datasets too large to render with existing methods.

Keywords: Unstructured meshes, direct volume rendering, level-of-detail, time-varying data

Index Terms: I.3.3 [Computer Graphics]: Picture/Image Generation; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

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 research 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 mesh also changes over time. Rendering of dynamic data has the potential to be applied in many areas of science, engineering, and even medicine.

Since it is now common for machines to contain multiple processors along with a programmable graphics card, a major focus of our work is on balancing the computation between the available resources. Our proposed work is based on a fundamentally new approach where the volumetric data is processed as a set of triangles on the CPU and streamed through the GPU, removing the need for extensive preprocessing or mesh connectivity information. This method also gives us the flexibility to adaptively render the geometry for improved interactivity.

2 Research Challenges

Our goal is to create a framework for volume rendering large dynamic meshes without significant penalty over similar static approaches. To remain interactive, it is imperative that the speed and quality of the visualization is easily controlled and that redundant or unnecessary computation is avoided.

An important challenge for creating an interactive framework is providing an adaptive rendering system for time-varying data. Though level-of-detail techniques exist for the static case, applying them to dynamic data is not trivial. Thus, a portion of our research addresses the issue of creating high quality approximations for interactive rendering. Another major challenge for handling time-varying data is developing techniques to update the renderable data at each time step. Compression is essential to avoid costly data transfers between the CPU and GPU. Because changing all the data with every rendering pass may not be feasible for interactivity, we focus on developing techniques for incrementally updating the dynamic data and avoid transferring data that is either off screen or unchanged between time-steps. By leveraging temporal coherence, our adaptive framework avoids unnecessary computation and data transfer.

*e-mail: stevec@sci.utah.edu

3 Background

A large step in creating a framework for adaptive visualization was presented in previous thesis work [2]. The main contributions of this thesis is 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 when rasterized. In image-space, the fragment stream is incrementally sorted using a fixed-depth A-buffer [6] implemented in hardware, called the k-buffer.

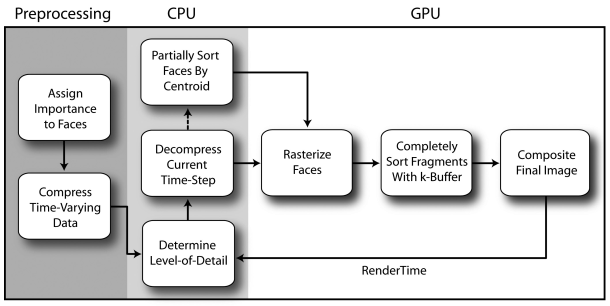
A key advantage of the HAVS algorithm is that it operates on a collection of triangles instead of tetrahedra, thus it requires little preprocessing and no neighbor information. Because the algorithm can use a different set of data for each frame, it can *naturally handle dynamic geometry*. This was recently 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 to form a mesh with fewer primitives. This new level-of-detail approach is very simple and efficient because it requires no hierarchical mesh representation.

4 Proposed Research

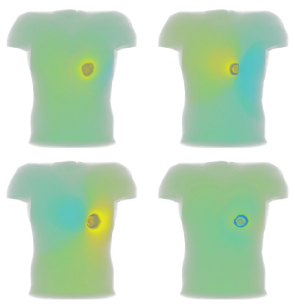
This work build on the volume rendering system proposed in [2] to handle time-varying data. An overview of the adaptive framework for dynamic data and an example of its use are shown in Figure 1. The proposed research is separated into three major components: providing tools for interactive exploration 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*.

4.1 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


Preprocessing
CPU
GPU
Assign Importance to Faces
Compress Time-Varying Data
Partially Sort Faces By Centroid
Decompress Current Time-Step
Determine Level-of-Detail
Rasterize Faces
Completely Sort Fragments With k-Buffer
Composite Final Image
RenderTime

(a)



(b)

Figure 1: (a) A framework for adaptive visualization of dynamic unstructured grids. Compression and level-of-detail techniques are essential to minimize the overhead of time-varying data. (b) An example of the torso dataset (50K tetrahedra) with a time-varying scalar field (360 time-steps) rendered interactively with our framework.

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 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. An important feature of this algorithm is that it only renders data inside the current view, thus reducing unnecessary computation and facilitating detailed exploration. Our algorithm is 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.

Remaining Tasks. 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.

4.2 Dynamic Scalar Fields

Datasets with static geometry and dynamic 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 (shown in Figure 1) for moderately-sized datasets with little additional rendering overhead.

Remaining Tasks. 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.

4.3 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.

Remaining Tasks. 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. In particular, we would like to take advantage of the temporal coherence of these datasets by incrementing existing geometry at each rendering step, thus avoiding the update of the entire geometry and topology at each time step. We believe that the framework shown in Figure 1 can be applied to this type of data to allow the user to control the speed and quality of the visualization and allow interactive exploration of the time-steps.

Acknowledgments

The proposed research is being performed under direction and guidance of Cláudio T. Silva. Other contributors to the work outlined in this proposal include Louis Bavoil, Fábio F. Bernardon, João L. D. Comba, and Valerio Pascucci. Steven P. Callahan is funded by the National Science Foundation (grants CCF-0401498, EIA-0323604, OISE-0405402, IIS-0513692, and CCF-0528201), the Department of Energy, an IBM Faculty Award, Sandia National Laboratories, Lawrence Livermore National Laboratory, the Army Research Office, and a University of Utah Seed Grant.

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. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of Visualization / Information Visualization 2006)*, 12(5), 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 sorting 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.