INTERACTIVE VISUALIZATION IN A HIGH PERFORMANCE COMPUTING VIRTUAL ENVIRONMENT

Trina M. Roy and Thomas A. DeFanti

Electronic Visualization Laboratory
University of Illinois at Chicago
Chicago, IL 60607-7053
email: trina@evl.eecs.uic.edu

Keywords: interactive simulation, graphics, virtual reality, visualization, physics.

ABSTRACT

An intuitive means of controlling numerical simulations can be achieved by integrating virtual environments with High Performance Computing (HPC) resources. The Cosmic Worm is an HPC visualization tool which runs in the CAVE virtual reality theater. It is designed to visualize the three-dimensional output from a numerical simulation while giving a scientist interactive control of that simulation from within a virtual environment. It is being developed in collaboration with the Astrophysics and Gravitation groups at the National Center for Supercomputing Applications (NCSA). Both groups are finding this to be a valuable research tool in their studies of cosmic phenomena. For the first time they have the ability to steer their three-dimensional simulations, and visualization in the CAVE has enabled them to see aspects of the data which had not appeared in standard workstation visualizations.

INTRODUCTION

Visualization is not only a tool for understanding data but can also create an intuitive means of guiding a numerical simulation. This paper discusses the design and implementation of a distributed visualization system which allows the user interactive control of a running simulation from within a virtual reality (VR) based visualization. The *Cosmic Worm* (named after the theoretical stellar object called a wormhole) is being developed for use in conjunction with large, computationally complex simulations, specifically computational fluid dynamics (CFD) simulations for cosmology research and the solution of Einstein's equations for general relativity in three-dimensions. As a simulation is running on a supercomputer, its data is sent out over high-speed network and visualized in a virtual environment. The visualization system is not restricted to a particular supercomputer or simulation. A communications library which converts data transparently is used for data transport, which removes any dependence on specific architectures. Consequently data can be accepted from any numerical simulation producing three-dimensional volumes of data. Part of the visualization computation can also be offloaded onto a supercomputer, resulting in a flexible multi-platform HPC visualization system.

The next section will discuss previous work done with interactive visualization and distributed VR. We will then talk about the research being done with the system, after which we will go into the details of the design and implemenation of the system.

PREVIOUS WORK

Typically visualizaton of simulation data is done on a workstation using precomputed data from disk. Recently the workstations have been connected to supercomputers allowing distributed visualization of precomputed data or run-time visualization of a live simulation (Robertson 1991; Rosenblum 1989). Adding two-way communication to run-time visualization systems has enabled interactive steering of simulations and experiments (Sluis 1991; Brodlie *et al.* 1993 allowing the scientist to change simulation parameters on the fly, and see the results in real-time.

Virtual reality is another recent development in the evolution of scientific visualization. Various VR systems are being used for the visualization of precomputed data (Cruz-Neira et al. 1993; Song and Norman 1993). They are also being successfully integrated with supercomputers, resulting in distributed VR visualization systems (Bryson and Gerald-Yamasaki 1992). We are using two-way communication between a VR system and a supercomputer in order to directly control a running simulation. By integrating a virtual environment with HPC resources we have achieved run-time visualization in VR with the ability to interactively steer a live simulation.

STUDY OF COSMIC PHENOMENA

We have been collaborating with the Astrophysics and Gravitation groups at NCSA from the earliest design stages of this tool. These physicists run large supercomputer simulations producing tremendous amounts of data. Here is a brief description of each group and the simulations we have been working with.

The Early Universe

The Astrophysics group at NCSA is part of the Grand Challenge Cosmology Consortium, an NSF funded collaboration of 7 universities formed in order to research the large scale structure of the universe and the formation of galaxies. The NCSA group is trying to do this through the study of X-ray clusters.

It has long been known that galaxies tend to group together into galaxy clusters; however it wasn't until orbiting X-ray telelscopes observed these clusters in the seventies that astronomers realized that the clusters are surrounded by halos of hot gas. Simulations of the formation of these clusters, and subsequent comparisons of the results to observational data, holds the key to understanding why they form, which will offer insight into what the early universe looked like. In order to model the cluster formation, scientists start with a theoretical configuration of the early universe. CFD codes are used to evolve the configuration over time, ending at the present time (Bryan 1994). Figure 1 shows temperature surfaces from the final timestep of this evolution. Figure 1a is a full 270^3 dataset subsampled by four. Figure 1b is a full-resolution subvolume from the upper left corner of Figure 1a. By comparing the end results of these simulations to observational data they can get an idea of how close the initial configuration was to the actual early universe.

Einstein's Equations

NCSA's Gravitation group is also part of an NSF funded collaboration. The Grand Challenge Alliance is comprised of eight institutions and is dedicated to the development of numerical codes to compute the collision of two black holes in 3D. A black hole collision is one cosmic event which is considered to be one of the most promising sources for gravitational waves - a physical phenomenon predicted by Einstein's General Theory of Relativity.

The theory of Relativity, developed by Einstein at the beginning of this century, involves a complex set of 14 nonlinear, coupled, second order partial differential equations. The equations are so complex, containing many thousands of terms, that general analytic solutions of physical relevance have not been found in spite of nearly 80 years of

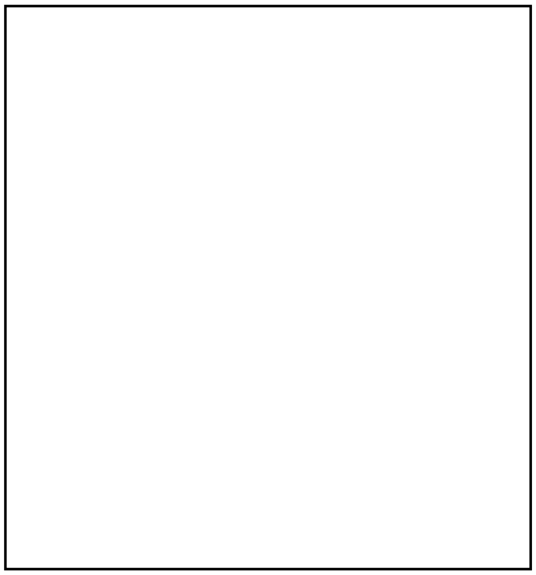

Figure 1a. Cosmological temperature isosurface. Data resolution is 270^3 and is subsampled by 4. Temperature for this surface is 3 million degrees Kelvin.

Figure 1b. Subvolume of cosmological temperature isosurface. Subvolume was selected from upper-left hand corner of Figure 1a, and is visualized at full resolution.

Figure 2. Gravitational wave (gxx) component. The data resolution is 64^3, the isosurface threshold is 0.99999.

study. The NCSA group has developed full 3D numerical codes to solve these equations for the gravitational field using advanced supercomputers (Seidel 1994). They are capable of simulating many phenomena in the Universe, such as the interactions bewteen black holes, the evolution of highly relativistic matter, or evolving pure gravitational waves.

The simulation discussed here starts with a volume representing flat space. A small perturbation is put in the gravitational field which then propagates through space over time, similar to the waves produced when dropping a pebble into a pond. Figure 2 is a visualization of one component of the gravitational field (the *gxx* component). It is hoped that studies such as these will provide an understanding of gravitational waves and their sources that will be essential to interpreting the gravitational waves that should be detected for the first time by the turn of the century.

THE COSMIC WORM

This section goes over the design and implentation of the Cosmic Worm (or "Worm" for short). An overview of the virtual environment will be given, then discussions of the system configuration, communication, interactive steering and the handling of large datasets will follow. This section concludes with an overview of other visualization tools offered by Worm.

The Virtual Environment

The CAVE is a room-sized immersive virtual environment designed as a vehicle for scientific research and interaction with HPC applications running on remote supercomputers (Cruz-Neira *et al.* 1992). It runs on a Silicon Graphics' (SGI) Onyx and can be connected via high-speed network to supercomputers such as Thinking Machines' CM-5, IBM's SP1 or the SGI Power Challenge. Stereo images are rear projected onto two walls and front projected onto the floor and are viewed by users wearing Stereographics' LCD shutter glasses. User input is accomplished with a hand held, six degree-of-freedom wand.

The Cosmic Worm is an application running within the CAVE environment, controlling a numerical simulation residing on the remote supercomputer. Worm accepts as input a stream of 3D datasets, from which it extracts surfaces of constant value ("isosurfaces") representing a specific physical parameter such as temperature or density. The surfaces are then displayed in the CAVE, resulting in an animation of the physical parameter as it changes over time. The scientist interacts directly with the simulation from within the CAVE, using the immediate visual feedback to guide progression of the simulation, make adjustments to parameters, or zoom in on interesting portions of the data using successive refinement. There are also several visualization tools available to aid in the exploration of the data. A previous version of the Cosmic Worm was presented in (Roy 1995).

System Configuration

Figure 3 shows the different parts of the system and the dataflow through the system. Table 1 gives an overview of each part including the computer system each runs on. As datasets are produced by the *simulation* they are processed by the *visualization module* and then rendered in the CAVE by the *control module*. This is done in a pipeline manner: as the simulation is calculating timestep *x+1*, the visualization module is processing timestep *x* and the user is seeing the visualization of timestep *x+1* in the CAVE.

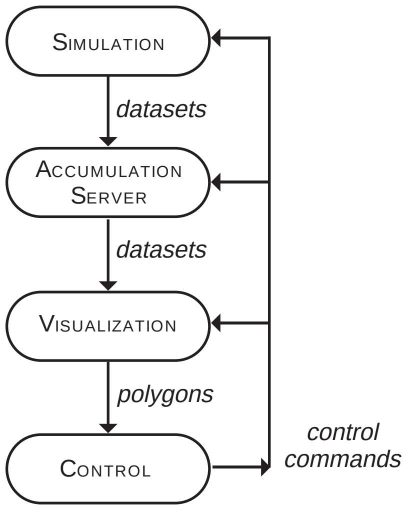

Figure 3. System Configuration.

Table 1. Components of Worm

Module	Computer	Description
Simulation	Supercomputer	Numerical simulation
Accumulation Server	Supercomputer	Optional "buffer" to hold accumulated datasets from the simulation and to handle control messages from the Control Module.
Visualization	SGI Challenge or Onyx	Parallel isosurface extraction algorithm. Takes dataset as input and produces a polygon list for rendering.
Control	SGI Onyx	Renders polygons from visualization module in the CAVE. Accepts input from user and broadcasts control messages to other modules.

The isosurfaces produced by the visualization module are generated using a parallelized surface extraction algorithm based on Lorensen and Cline's Marching Cubes algorithm (Lorensen and Cline 1987). It uses the data parallel model of parallelization, distributing voxels of data across multiple processors. It is currently written in C using IRIS Power C, SGI's multi-processing C compiler. The code is parallelized by insertion of compiler directives, which can be ignored by any non-Power C compiler, enabling the code to run in serial on any platform. When running in parallel it can run at interactive speeds, generating isosurfaces as fast as the simulation can generate datasets.

The *accumulation server* runs in conjunction with the simulation, acting as a data buffer. Currently it is only a mediator between the simulation and the visualization module, used to offload control message processing from the simulation and to minimize confusion when pausing and un-pausing the simulation. Eventually it will be the point of convergence, collecting datasets from a simulation distributed over several different supercomputers and passing them on in the proper order to the visualization module. The accumulation server allows the visualization module to be ignorant of the configuration of the simulation.

Physically, the simulation resides on a supercomputer, typically a CM-5 or SGI Challenge. The control module runs on the Onyx with the CAVE processes. The visualization module runs either on the Onyx or a Challenge. Ideally the visualization and simulation are run on separate supercomputers, either in a Challenge/CM-5 configuration or a Challenge Array.

Inter-Module Communication

Communication between modules is handled by the Data Transfer Mechanism (DTM) which was developed by NCSA. DTM is a message-based communications library which provides synchronization and transparent data conversion (NCSA 1992). Messages are sent and received through unidirectional ports which are implemented on UNIX machines using Berkeley sockets. The complexity associated with a socket implementation is hidden from the user by the DTM library, so the user can simply open the ports and read or write messages.

DTM comes with several predefined message classes. Worm uses the Scientific Dataset (SDS) class for sending raw data from the simulation to the accumulation server and the visualization module. SDS is a message class designed for sending multi-dimensional arrays of data. It contains all the necessary information (number and size of dimensions etc.) for interpreting and processing the dataset.

The isosurfaces are sent using a message class which we designed specifically for Worm. It contains the vertices and normals of one or more isosurfaces, plus other information such as threshold and stride, which is displayed along with the surface in the CAVE. Control messages consist of directives for the various modules. Their content varies with each message; see below for a more detailed description of control messages.

Steering The Simulation

Interaction with the simulation is accomplished with the use of control messages which are broadcast by the control module to all other modules (see Figure 3). The messages are sent by the user in the CAVE through the use of a wand-controlled menu system. Input control messages generally consist of directives intended for the simulation, however there are also request messages which are usually intended for the visualization module. Table 2 describes the different types of messages used by Worm and who handles each message.

Table 2. Worm Control Messages.

Message	Handled By	Description
Input Control Messages	Accumulation Server	Start, Stop, Pause, Continue and Restart the simulation
Change Parameters	Simulation	Modify simulation parameters.
Change Threshold	Visualization	Request a new surface threshold value.
Subvolume	Accumulation Server	Request a portion of the data at a higher resolution.

Input control messages allow the user to maintain control of data generation. The user initially sends a start message to begin the simulation. The simulation can then be paused and continued, or can be stopped entirely and restarted from the beginning. When a simulation is stopped the input parameters can be modified and the simulation restarted with new initial conditions. The scientist has complete control over the progress of the simulation; if it becomes apparent during a run that the output is not as expected, the simulation can simply be stopped, tweaked with and restarted from within the CAVE environment.

Request messages are used when manipulating the visualization parameters. A user can change the isosurface thresholds in order to view a different parameter value, or can request different simulation parameter values as mentioned above. He/she can also request a subvolume as part of a powerful successive refinement tool used when viewing datasets which are too large to visualize in real-time, as discussed in the next section.

Handling Large Datasets

Even with today's high-speed networks and high-performance workstations, Worm can still only visualize small-to-medium sized datasets (up to about 64^3) in real-time. In order to handle larger datasets we can subsample the data down to a manageble size. The user selects a *stride* which specifies a fraction of the resolution which will be used (i.e. a stride of 2 means that every other data point, half of the total data, will be used). The subsampling occurs in the accumulation server (or in the simulation if there is no accumulation server). The amount of data transferred to the visualization module is greatly reduced, as is the amount processed by the visualization program, resulting in less time spent in the visualization module, fewer polygons sent to the main program, and consequently a higher frame rate in the CAVE. The surface displayed in the CAVE becomes a rough approximation of the full resolution surface, but it gives the user a good idea of the general structure of the data. He/she can explore an area of interest at higher (or full) resolution with a subvoluming tool. This tool enables the user to select a portion of the data and tell the accumulation server or simulation to send only that subvolume with a smaller resampling rate (or none for full resolution). Figure 1a is a 270^3 dataset with a stride of 4. Figure 1b shows a full-resolution subvolume from the upper left corner of the volume in Figure 1a. You can see the same filament in both images, however Figure 1b shows much more detail for that small area.

Visualization Tools

Other visualization tools offered by Worm include a data slicer, exploration tools, and particle display. The data slicer allows the user to view a 2D slice of the volume as a height-field surface. The wand is used to select an axis and move the slice through the volume. The height-field surface is continually updated as the slice moves. The exploration tools are useful for analyzing the isosurfaces of data. When a simulation is paused or stopped the scientist can replay data that has been collected so far. The control here is similar to that of a VCR. One can replay from the beginning, stop, step forward or backward and so on. There is also a tool for "grabbing" a surface. This allows the user to "pick up" a surface with the wand and move it around in order to see all sides of the data. This is useful in exploring the side of the data that is away from you (which is not always easily attainable in the CAVE) and for stepping back and getting more of an outside-looking-in perspective if so desired.

The particle display is a new tool under development. It gives the option of displaying particle data in addition to the isosurfaces. If particle data is sent from the simulation, it will be visualized in the CAVE as small spheres. Stream lines will show the path of the particles if the user requests it.

Worm can also be used as a visualization tool for archived data. In this case only two modules are running: the control and visualization modules. The visualization module reads the data from files and uses the simulation control messages sent from the control module as directives for controlling loading. Functionality in the CAVE remains the same as real-time mode.

CONCLUSION AND FUTURE WORK

The Cosmic Worm is an HPC application which interactively visualizes 3D output from numerical simulations in the CAVE virtual reality environment. We have been successful in integrating HPC resources into a multi-platform, distributed virtual reality visualization system, and as technology grows we will continue to improve the computation power of this system. In the near future we plan to add and improve visualization tools (e.g. the particle tool mentioned earlier) and to extend the simulation end to incorporate multiple supercomputers, greatly expanding the domain of scientific problems being studied with this tool. As it stands scientists now have an intuitive way of interactively steering their 3D simulations. They have the opportunity to explore their data in ways which were previously out of reach.

ACKNOWLEDGMENTS

There are a lot of people without whom this work would not have been possible. Thanks to Carolina Cruz-Neira of the Electronic Visualization Laboratory (EVL) for the CAVE software, Gary Lindahl of EVL for technical support, and to Jon Goldman of EVL for his contribution to the visualization software. Thanks also goes to Jeff Terstriep and Dan Weber of NCSA for developing the DTM software and working with us during integration; and to Greg Bryan, Joan Masso and Ed Seidel of NCSA for writing the simulation codes and providing us with valuable input. A special thanks goes to Maxine Brown for her feedback and encouragement. This work is supported in part by NSF grant #ASC-9407777, #IRI-9213822 and #CDA-9303433, which includes support from ARPA and NASA.

REFERENCES

Brodlie, K., *et al.* 1993. "GRASPARC - A Problem Solving Environment Integrating Computation and Visualization." In *Proceedings of IEEE Visualization*

'93 (San Jose, CA, Oct. 25-29). IEEE Computer Society Press, Piscataway, NJ, 102-109.

Bryan, G.L. *et al.* 1994. "X-Ray Clusters From a High-Resolution Hydrodynamic PPM Simulation of the CDM Universe." *Astrophysical Journal 428* (June 20): 405-418.

Bryson, S. and Gerald-Yamasaki, M. 1992. "The Distributed Windtunnel." In *Proceedings of Superomputing '92* (Minneapolis, MN, Nov 16-20) IEEE Computer Society Press, Piscataway, NJ, 275-284.

Cruz-Neira, C. et. al. 1992. "The CAVE Audio Visual Experience Automatic Virtual Environment.". *Communications of the ACM 35,* no. 6 (June): 64-72.

Cruz-Neira, C. et. al. 1993. "Scientists in Wonderland: A Report on Visualization Applicatoins in the CAVE Virutal Reality Environment." In *Proceedings of IEEE 1993 Symposium on Research Frontiers in Virual Reality*. (San Jose, CA, Oct. 25-26). IEEE Computer Society Press, Piscataway, NJ, 59-66.

Lorensen, W.E., and Cline, H.E. 1987. "Marching Cubes: A High Resolution 3D Surface Construction Algorithm." *In Proceedings of SIGGRAPH '87* (Anaheim, CA, July 27-31). ACM, New York, NY, 163-169.

National Center for Supercomputing Applications. 1992. *NCSA Data Transfer Mechanism Programming Manual. Version 2.3.* University of Illinois at Urbana-Champaign, Champaign, IL. (Feb.).

Robertson, D.W. *et al.* 1991. "Distributed Visualization Using Workstations, Supercomputers and High Speed Network Protocols." In *Proceedings of IEEE Visualization '91* (San Diego, CA, Oct. 21-25). IEEE Computer Society Press, Piscataway, NJ, 379-382.

Rosenblum, L.J. 1989. "Scientific Visualization at Research Laboratories." *Computer 22,* no. 8 (Aug): 68-70.

Roy, T.M., Cruz-Neira, C., and DeFanti, T.A. 1995 "Cosmic Worm in the CAVE: Steering a High Performance Computing Application from a Virtual Environment." *PRESENCE: Teleoperators and Virtual Environments 4.* no. 2 (Spring): to be published.

Seidel, E. and Suen, W. 1994. "Numerical Relativity." *International Journal of Modern Physics C 5:* 181-187.

Sluis, L.V. 1991. "Designing a Distributed Scientific Visualization Tool." In *Proceedings of IEEE Visualization '91* (San Diego, CA, Oct. 21-25). IEEE Computer Society Press, Piscataway, NJ, 383-386.

Song, D. and Normal, M.L. 1993. "Cosmic Explorer: A Virtual Reality Environment for Exploring Cosmic Data." In *Proceedings of IEEE 1993 Symposium on Research Frontiers in Virual Reality* (San Jose, CA, Oct. 25-26). IEEE Computer Society Press, Piscataway, NJ., 75-79.