VOLUMETRIC VISUALIZATION METHODS FOR ATMOSPHERIC MODEL DATA IN AN IMMERSIVE VIRTUAL ENVIRONMENT

Michael P. Dye[1,2]
Philip A. McDonald[2]
Frederick C. Harris Jr.[1,2]
William R Sherman[2]

[1]University of Nevada, Reno
Reno, NV 89557

[2]Desert Research Institute
Reno, NV 89512

mdye@cse.unr.edu
phil.mcdonald@dri.edu
fredh@cse.unr.edu
bill.sherman@dri.edu

KEYWORDS
3D, Scientific Visualization, Surround Screen Virtual Environment

ABSTRACT

We present the design and implementation of an immersive interactive volumetric visualization system. This system was designed to allow atmospheric modelers to visualize their simulations in an immersive environment such as our Fakespace FLEX system. By allowing them to play back the entire simulation as well as choose what airborne particulates to turn on and off atmospheric scientists are given an incredible degree of control regarding what to study and look at from any angle with amazing detail at the interactions between particulates in a simulation and interactions of the particulates with surrounding terrain.

INTRODUCTION

Atmospheric simulation is an important means of understanding the environment around us. Through atmospheric simulation we can predict how various airborne particulates such as dirt, smog, and fire can affect our cities and overall public health. However, problems can arise when trying to interpret the results of the simulation. For example, it can be difficult to determine the density or shape of an individual particulate if the area under study is large. A similar problem arises when attempting to find how various terrain formations interact with and affect the atmosphere. However, little in the way of research exists on how to accurately model atmospheric data at interactive frame rates.

Since raw data can be hard to conceptualize, particularly when the size of the data sets can be hundreds of megabytes if not gigabytes, an alternative method of interpreting atmospheric data is needed. Virtual reality technology allows us to accurately and realistically model atmospheric data in a meaningful way for the user. Virtual reality has long been used as a way of creating realistic visual simulations to help aid in interpreting large and complex data sets. Recent advances in visualization and supporting technologies now offers the possibilities of creating realistic, real-time atmospheric visualizations for research.

By combining virtual reality technology with atmospheric simulation users are able to visually conceptualize large amounts of atmospheric data in an interactive way. This paper describes a package we call Vesuvius which is a virtual reality library for visualizing large sets of atmospheric data. Vesuvius is intended to allow users to visualize various atmospheric data sets from a variety of viewpoints and to allow for playback of atmospheric data sets that contain temporal information.

The rest of this paper is structured as follows. First we present some related work on atmospheric visualization, virtual reality, and previous research on volume rendering. Next we present an overview of the execution environments followed by our immersive volume visualization library for visualizing atmospheric data. Lastly presents our conclusion and possibilities for future work.

BACKGROUND

Atmospheric visualization would benefit from the merger of both volume visualization and virtual reality. Marrying the two, users can see realistic 3-dimensional representations of their atmospheric simulations. In addition, it allows users to immerse themselves in the simulation, enabling them to watch it from different points of view in a highly realistic environment.

Atmospheric Simulation

Atmospheric modeling and simulation is done with numerical models like the National Center for Atmospheric Research (NCAR)/Pennsylvania State University Mesoscale Model (MM5). Models such as MM5 simulate the behavior of a wide range of atmospheric parameters, including mass parameters (e.g. temperature and humidity) as well as momentum parameters (e.g. wind U, V, and W fields). When these models are initialized with archived meteorological data for prior conditions, their simulations can be used to analyze the state of the atmosphere during a past event. When they are initialized with current meteorological observation data, their simulations serve as atmospheric forecasts. These models are quite adaptable and can be utilized for atmospheric simulations over domains with resolutions ranging from tens of kilometers down to just a few kilometers. Ongoing research is further refining the physics in MM5 so that simulations over domains with horizontal

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)

resolutions of less than one kilometer can be achieved.

In addition to models like MM5 which simulate the state of the atmosphere, there are specialized models that simulate a specific component of the atmosphere. One such model is the Comprehensive Air Quality Model with Extensions (CAMx) which simulates the behavior of atmospheric chemicals and its use is common in air pollution studies. Like MM5, CAMx can simulate past, current, or future events and can be scaled to adapt to a wide variety of domain sizes and resolutions.

Atmospheric Visualization

In 1988, the Space Science and Engineering Center (SSEC) at the University of Wisconsin, Madison released the Vis5D atmospheric visualization tool and in 1994, a virtual reality port of this open-source application was created for the CAVE immersive display system. Vis5D, with it's grid limitations, displaying smog data over Los Angeles can be seen in Figure 1. Since these early efforts, a great deal of the work done in the area of atmospheric visualization has been in the field of cloud simulation and rendering (Dobashi et al., 2000; Ebert and Parent, 1990; Harris and Lastra, 2001). Because of the level of complexity in rendering clouds (realistic shading, light scattering, etc) much previous work has been done in non-interactive rendering techniques for cloud rendering. Work into interactive cloud rendering has produced methods whereby most scattering and shading are done in a preprocessing step and textured polygons known as imposters are used to bypass fill rate limitations (Harris and Lastra, 2001).

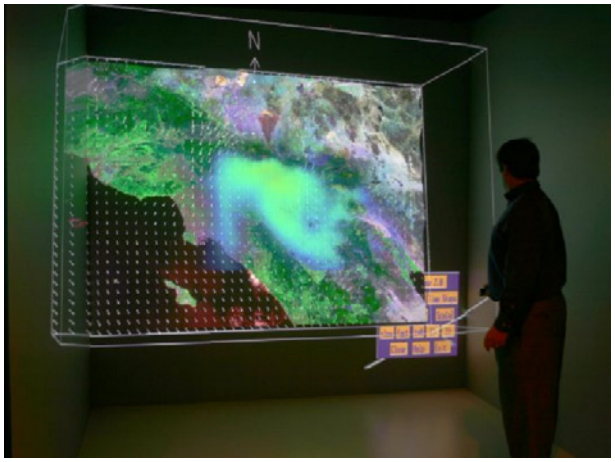

Figure 1. Volume Rendering in Vis5D

Volume Rendering

Volume visualization deals with displaying volumetric data sets, represented as sample points. There are two means of achieving this. First is indirect volume rendering (IVR) which involves converting the volumetric data into a set of polygonal iso-surfaces which are then rendered using traditional graphics hardware. The second means is called direct volume rendering (DVR) which involves rendering directly to the screen without an intermediate step such as converting to iso-surfaces.

Indirect volume rendering assumes that extractable iso-surfaces exist, which is not always the case (such as flow fields, clouds, etc.). In addition, the complexity of the iso-surfaces might be so complex that they may overwhelm the capabilities of the graphics hardware. Because of this, direct volume rendering may be more efficient. What follows is an explanation of the primary methods of direct volume rendering.

Raycasting

Raycasting is perhaps the most researched of direct volume rendering algorithms (Levoy, 1988, 1990) with several acceleration techniques for more interactive volume raycasting being proposed over the last decade (Grimm et al., 2004; Knittel, 2000; Mora et al., 2002; Wan et al., 1999).

For each pixel in the image, a ray is cast into the volume. At fixed intervals along the ray the volume data is resampled, most commonly using tri-linear interpolation. Trilinear interpolation involves taking the scalar values of the eight neighbors to a particular pixel and weighing them according to their distance to the actual location. The solutions to each interval are combined either in a front-to-back or back-to-front order to determine the final color and opacity of the pixel. Because each pixel has to be computed, raycasting is the slowest of volume rendering algorithms. However, because volumes are determined on a pixel-by-pixel basis, they can generate high-quality images without any blurring or loss of detail.

Shear-warp

Shear-warp (Lacroute and Levoy, 1994) is a fast means of software volume rendering. Unlike raycasting, no rays are cast into the volume. Instead, the volume is projected slice by slice onto the image plane using bi-linear interpolation within the slices. In shear-warp, an intermediate image plane is created and aligned with the volume. The volume itself is then sheared to turn the projection direction into a direction perpendicular to the intermediate image plane. This intermediate image is finally warped to the final image plane. Due to the warping to the final image plane only being required once per image, not once per slice and run-length encoding of the volume data, shear-warping is considered the fastest software base volume rendering algorithm. However, the speed of shear-warp does not come without a price, magnification of volume data results in blurring of details and the addition of artifacts. In addition, it requires three copies of the volume data to remain in memory, one copy for each major axis.

Texture-Based

3D texture mapping uses 3D textures to render volumes. The volume is loaded into texture memory and sampled as a series of slices. The resulting planes are then drawn as a series of textured polygons that are blended together, thus creating the final image. Because texture mapping and compositing are performed in hardware, the rendering is actually faster than shear-warping for small datasets. However, if a dataset if too large to fit completely into texture memory,

then performance is decreased considerably as data must continually be paged back and forth between the hard drive and graphical memory. To minimize the performance impact of this, the volume is encoded into an octree structure (Boada et al., 2001; Fang et al., 1996).

Virtual Reality

VR provides a medium composed of immersive interactive computer simulations that provide real-time feedback to the users (Sherman and Craig, 2003). VR is a technology that can provide sophisticated real time 3D user interface for users to interact with 3D applications. Therefore, VR technology is a good candidate as a user interface for interaction with 3D atmospheric data. There have been various research efforts regarding the use of VR technology in different contexts (Chen et al., 2001; Koller et al., 1995; Lin et al., 2000; Loftin et al., 1998).

Interaction within a virtual environment can generally be categorized into selection, manipulation, navigation, and system control. Selection and manipulation techniques will be very different from one application to another as each application's interaction requirements are different. Navigation techniques are mandatory for large scaled VR applications like terrain visualization since the user will need to be able to get from one point to another within the virtual environment. In system control interaction, the user will be able to control the state of the application at the system level where the execution mode of the application is changed.

The research of applying VR technology into the field of atmospheric research is a little explored research direction. The capability to see various types of volumetric data in a 3D large screen immersive environment has not been explored. In the research of large displays, (Huang et al., 2006) described some of the unique features provided by their use. These unique features provided by large displays could be beneficial in the viewing and interaction with large atmospheric datasets within a virtual environment.

HARDWARE AND SOFTWARE ENVIRONMENT

Our immersive visualization facility hardware includes both a four-screen CAVE-like Fakespace FLEX display (Figure 2), and a single-screen Visbox-P1 (Figure 3). The FLEX display is driven by an SGI Prism running SuSE 9.0 Enterprise edition, with four active-stereo capable graphics channels. Tracking of the participants is accomplished with an Intersense IS-900 VETracker with wireless MiniTrax Head Tracker and wireless Wand with 5 Buttons and center click joystick. Both the viewpoint and the dominant hand are tracked at interactive rates to enable participants to interact with the application

For the VisBox-P1 virtual environment system, a custom built dual Opteron graphics workstation running OpenSUSE 10.0 with an NVIDIA GeForce 6800 GT with dual outputs is used to drive two projectors. Tracking of the participants is accomplished with an Ascension Technology Flock of Birds with one sensor for positional tracking of the gamepad used for button and joystick inputs. The tracked gamepad is used by the participants to interact with the application. The viewpoint of the participant's head is tracked using proprietary optical tracking solution.

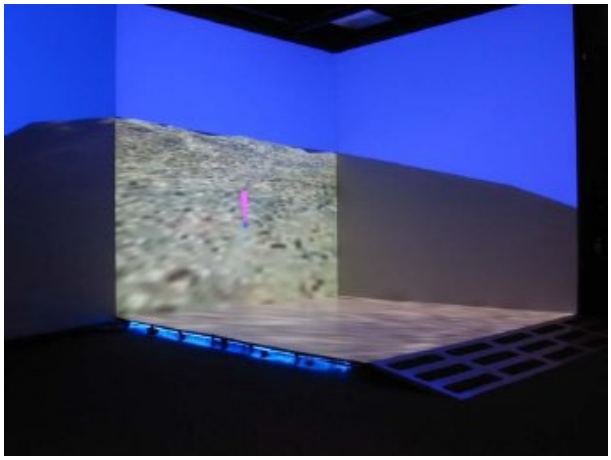

Figure 2. FakeSpace FLEX Display

Figure 3. VisBox-P1 Display

Vesuvius is designed to work well inside the open-source FreeVR (Sherman, 2007) and OpenSceneGraph (OSG) libraries. The FreeVR virtual reality integration library is a cross-display VR library with built-in interfaces for many input and output devices. It allows programmers to develop on a standard desktop machine, with inputs and display windows that simulate a projection or head- based immersive system. The application can then run on either the Visbox-P1 or FLEX displays, or the display of a collaborator on just about any type of VR system. The OpenSceneGraph library is used to help with world rendering. OSG allows 3D objects

to be hierarchically organized within the environment, and also provides a system that optimizes the rendering through the use of various culling and sorting techniques.

Although we chose the VisBox-P1 and the FLEX as the display devices, the application can easily be modified to display on different virtual environment displays with various configurations supported by the FreeVR library. In addition, the FreeVR library also supports various virtual reality devices that enable the use of other tracking devices with the application.

PROPOSED ARCHITECTURE

Although much has been done in the realm of volume visualization little has been done in atmospheric visualization, specifically with the MM5 model. Most volume visualization research has focused on visualizing static medical data such as CAT scans and MRIs. Unlike medical imagery, there is often a temporal variable associated with atmospheric data sets. This means that for a given particulate, the user might wish to see its simulated interaction over the course of minutes, hours, or even days. In addition, the size of atmospheric data tends to be considerably more than that of medical data if for no other reason than the scope of the datasets. Work in this area of dynamic volumes is far from complete. In addition, atmospheric data contains many types of divergent data, each of which may or may not be wished to be displayed by the user. The purpose of Vesuvius is to address these differences by visualizing dynamic and varied atmospheric data at interactive framerates.

Design

At the core of Vesuvius are two direct volume rendering algorithms: raycasting and shear-warping. Vesuvius was designed with the algorithms derived from a common base to create a minimum amount of code duplication and increase overall program cohesion. As an added result of this derived structure, other volume rendering algorithms such as texture-based volume rendering can be easily added in the future. In addition, this allows for a common interface through which the algorithms can receive data (such as camera position) and send data (such as the final volumetric image). An overview of this design is given in Figure 5.

Raycasting was chosen as one of the direct volume rendering algorithms to implement because of its ability to display highly detailed volumes. Unlike other algorithms such as shear-warping, increasing the resolution of a volume or its magnification will not result in any loss of detail when using the raycasting algorithm. However, due to the computationally intensive nature of the algorithm, even with optimizations such as early ray termination (Levoy, 1990), spatial encoding of the volumetric data (Grimm et al., 2004; Sobierajski and Avila, 1995), or an object-ordered approach (Mora et al., 2002) the framerate could barely be considered interactive, especially in a virtual environment. This algorithm is particularly suited to static, highly detailed volumetric data such as that of medical data which comes from CT and MRI scanners.

The second algorithm chosen was shear-warping because of how it complements raycasting. While shear-warping tends to blur detail when the resolution is increased or the volume is magnified, under normal viewing conditions it is the fastest of the software volume rendering algorithms. The speed of shear-warp allows the amount and size of information normally contained in volumetric data to run at interactive framerates. This is made more important when temporal data is factored into the volume data, where the volume will be required to dynamically move and change in real-time. With the amount of data being displayed in such a way, it becomes essential that things run at interactive framerates.

Once the image is obtained, a texture is created and placed on a billboarded polygon at the position of the actual volume data. This minimizes the amount of data that actually needs to be rendered as well as minimizes the amount of calculating that needs to be done, as the polygon texture only needs to be recalculated when the camera position changes.

Vesuvius works with an external program that will read and convert MM5 files into a custom file format. Because of the sheer size and complexity of MM5 files it is difficult to find and extract just the relevant volumetric information for a given frame in real-time. This external program extracts all relevant volume information (including bounds, opacities, etc.) for each particulate and stores this into the custom file format. In cases when there is temporal data in the MM5, the step is recorded and the temporal data is stored as deltas at each step interval. A command line option in the converter program allows the user to specify a resolution for the time interval. This means that the user can specify a higher resolution for the temporal data stored in the MM5 file and the program will interpolate the temporal data and add this new information into the file. While this greatly increases the file size, it also makes animations of atmospheric data both smoother and less computationally intensive, which results in greater framerates. The layout of this file structure is given in Figure 4.

What makes Vesuvius unique is the amount of control it gives the user when viewing the atmospheric data. It encapsulates controls for moving through the temporal data in the dataset, allowing the user to rewind, fast-forward, play, and pause atmospheric simulations in real-time. In addition, Vesuvius can render overlapping volume data such as that of multiple particulates. This is done by first keeping track of each particulate separately and computing how much each particulate adds to the overall volume data at a given point. From this, the user can choose to display, hide, or adjust the transparency for any given particulate type. This allows the user to focus on the interactions of one or several certain particulates with the atmosphere and for the user to see how these particulates are affected by the varying conditions of the simulation without the added information of every other particulate in the dataset.

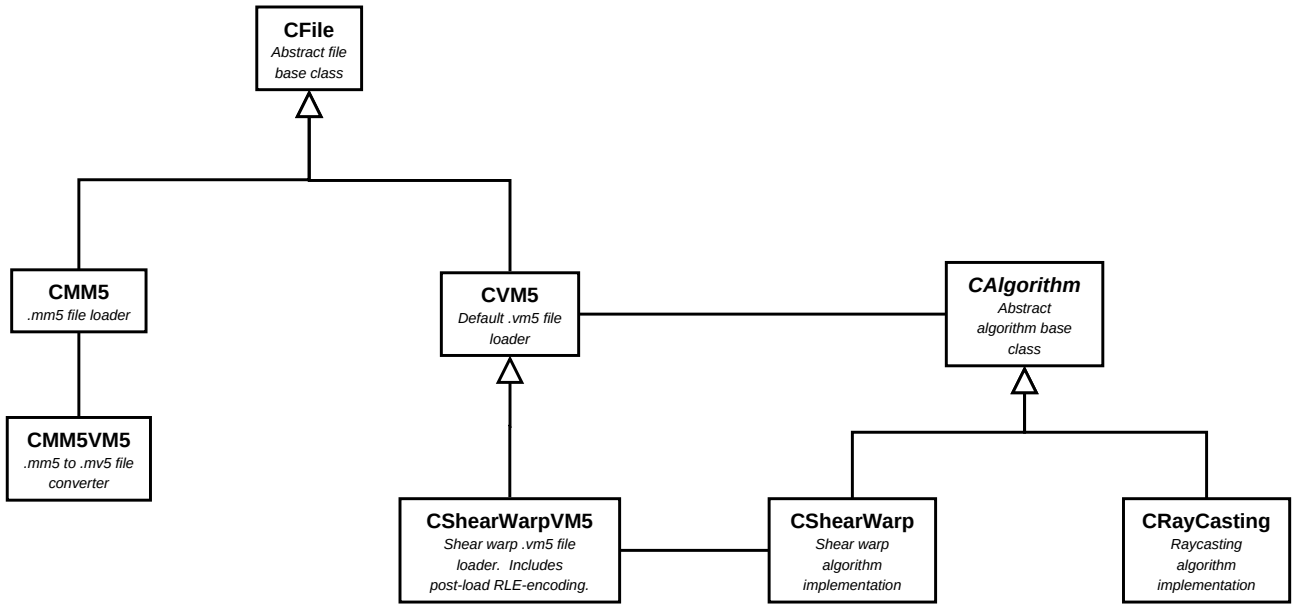

Figure 5. Software Design

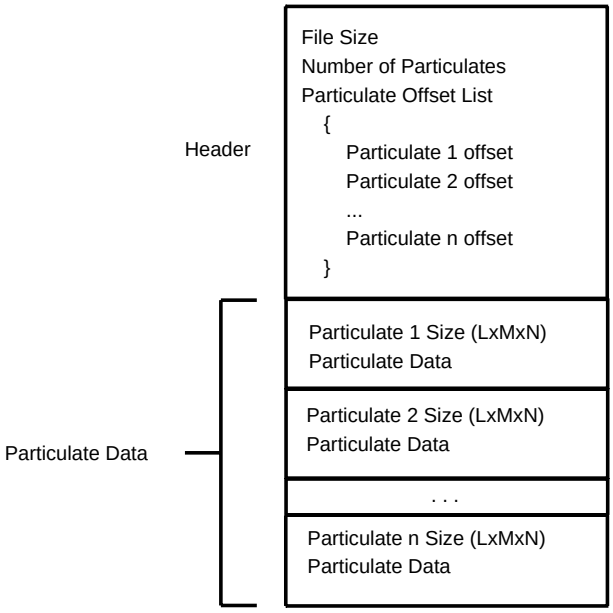

Figure 4. File Structure

CONCLUSIONS

Vesuvius provides atmospheric scientists a chance to interactively evaluate the large amounts of data that are generated from an atmospheric simulation. Instead of gigabytes of raw numbers they can utilize the three-dimensional interaction and visualization capability provided by FLEX and VisBox-P1 or possibly other virtual reality environments.

By allowing them to play back the entire simulation as well as choose particulate opacity and which particulates to enable and disable, atmospheric scientists are given an incredible degree of control regarding what they choose to study and look at. In addition, by rendering the entire simulation in a full 3D virtual reality environment they are allowed an unparalleled view from any angle with amazing detail at the interactions between particulates in a simulation and interactions of the particulates with surrounding terrain.

FUTURE WORK

Optimizing the renderer to achieve higher framerates is the main focus of our future work. By paralellizing the algorithm to fully utilize the many CPUs available to us, the computation of the volume rendering should be done in far less time. Also, by rethinking how data is shared between the different screens in the virtual environment we hope to minimize redundant data in memory. We feel that by re-evaluating how overlapping volumes affect each other we can come up with a new, less computationally expensive process. A texture-based volume rendering algorithm is also being considered for hardware acceleration of small datasets or small particulates in a dataset. In addition to rendering optimizations, optimizations can also be made to the file structure. By reorganizing the custom volume file we hope to be able to fit more into memory and have less hard drive access and seeking, which are notoriously slow.

ACKNOWLEDGEMENTS

This work was partially supported by the US Army PEO-STRI, NAVAIR Contract N61339-04-C-0072

REFERENCES

Boada, I., Navazo, I. and Scopigno, R. 2001. Multiresolution volume visualization with a texture-based octree. *The Visual Computer*, *17(3)*, 185–197.

Chen, J., Fang, Y., Loftin, R., Leiss, E., Lin, C. and Su, S. 2001. An immersive virtual environment training system on real-time motion platform. *Proc. of the Computer Aided Design and Computer Graphics*, *2*, 951–954.

Dobashi, Y., Kaneda, K., Yamashita, H., Okita, T. and Nishita, T. 2000. A simple, efficient method for realistic animation of clouds. In *SIGGRAPH '00: Proceed-*

ings of the 27th annual conference on Computer graphics and interactive techniques (pp. 19–28). New York, NY, USA: ACM Press/Addison-Wesley Publishing Co.

Ebert, D. S. and Parent, R. E. 1990. Rendering and animation of gaseous phenomena by combining fast volume and scanline a-buffer techniques. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques* (pp. 357–366). New York, NY, USA: ACM Press.

Fang, S., Srinivasan, R., Huang, S. and Raghavan, R. 1996. Deformable volume rendering by 3d texture mapping and octree encoding. *IEEE Visualization '96*, (pp. 73–80).

Grimm, S., Bruckner, S., Kanitsar, A. and Gröller, E. 2004. Memory efficient acceleration structures and techniques for cpu-based volume raycasting of large data. In *2004 IEEE Symposium on Volume Visualization and Graphics* (pp. 1–8).

Harris, M. J. and Lastra, A. 2001. Real-time cloud rendering. *Computer Graphics Forum*, *20(3)*, 76–85.

Huang, E. M., Mynatt, E. D., Russell, D. M. and Sue, A. E. 2006. Secrets to success and fatal flaws: The design of large-display groupware. *IEEE Computer Graphics and Applications*, *26(1)*, 37–45.

Knittel, G. 2000. The UltraVis system. In *IEEE Visualization '2000*.

Koller, D., Lindstrom, P., Ribarsky, W., Hodges, L. F., Faust, N. and Turner, G. 1995. Virtual gis: A real-time 3d geographic information system. *vis*, *0*, 94.

Lacroute, P. and Levoy, M. 1994. Fast volume rendering using a shear-warp factorization of the viewing transformation. *Computer Graphics*, *28(Annual Conference Series)*, 451–458.

Levoy, M. 1988. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, *8(3)*, 29–37.

Levoy, M. 1990. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, *9(3)*, 245–261.

Lin, C., Loftin, B., Kakadiaris, I., Chen, D. and Su, S. 2000. Interaction with medical volume data on a projection workbench. In *The Proceedings of the 10th International Conference on Artificial Reality and Telexistence* (pp. 148–152). Taipei, Taiwan.

Loftin, R. B., Pettitt, B. M., Su, S., Chuter, C., McCammon, J. A., Dede, C., Bannon, B. and Ash, K. 1998. Paulingworld: An immersive environment for collaborative exploration of molecular structures and interactions. In *NORDUnet'98, 17th Nordic Internet Conference*.

Meissner, M., Huang, J., Bartz, D., Mueller, K. and Crawfis, R. 2000. A practical evaluation of popular volume rendering algorithms. In *VVS '00: Proceedings of the 2000 IEEE symposium on Volume visualization* (pp. 81–90). New York, NY, USA: ACM Press.

Mora, B., Jessel, J. P. and Caubet, R. 2002. A new object-order ray-casting algorithm. In *VIS '02: Proceedings of the conference on Visualization '02* (pp. 203–210). Washington, DC, USA: IEEE Computer Society.

Sherman, W. R. 2007. Freevr.

Sherman, W. R. and Craig, A. B. 2003. *Understanding Virtual Reality*. San Francisco, CA, USA: Morgan Kaufmann Publishers.

Sobierajski, L. M. and Avila, R. S. 1995. A hardware acceleration method for volumetric ray tracing. *vis*, *0*, 27.

Wan, M., Kaufman, A. and Bryson, S. 1999. High performance presence-accelerated ray casting. In *VIS '99: Proceedings of the conference on Visualization '99* (pp. 379–386). Los Alamitos, CA, USA: IEEE Computer Society Press.

AUTHOR BIOGRAPHIES

MICHAEL P. DYE was born in Orange, California, USA. He received his BS in Computer Science from the University of Nevada in 2005, and is currently an MS student in Computer Science and should finish in August 2007. He has been employed in the Center for Advanced Visualization, Computation, and Modeling (CAVCaM) at DRI for the past two years. His research is in graphics, computer games, and immersive scientific visualization. His e-mail address is: `mdye@cse.unr.edu`.

FREDERICK C. HARRIS, JR. was born in San Jose, California, USA. He received his BS in Mathematics and MS in Educational Administration from Bob Jones University in 1986 and 1988, and his MS and PhD in Computer Science from Clemson University in 1991 and 1994. He has been at the University of Nevada, Reno since 1994, and is currently a Professor in the Computer Science and Engineering Department. He also has an affiliated faculty position with Desert Research Institute in Reno, where he is in the Center for Advanced Visualization, Computation and Modeling. His research is in the areas of Parallel and Distributed Computing, as well Graphics and Virtual Reality. His e-mail address is: `fredh@cse.unr.edu` and his Web page can be found at `http://www.cse.unr.edu/~fredh`

PHILIP A. McDONALD was born in Lebanon, Indiana. He received his BS in Forestry from the University of California, Berkeley in 1971 and his MS in Environmental Design and Meteorology from the University of Oklahoma in 1984. He has been a visualization software engineer serving the atmospheric sciences community for more than twenty years. He has been a faculty member in the Center for Advanced Visualization, Computation and Modeling at the Desert Research Institute since 2005 where he is an Assistant Research Visualization Scientist. His research areas include the application of advanced visualization methods to the earth sciences. His email address is: `phil.mcdonald@dri.edu`

WILLIAM R. SHERMAN was born in Madison, Wisconsin, USA. He received his BS in Computer Engineering and MS in Computer Science University of Illinois in 1986 and 1988 respectively. He was a Visualization Scientist for the National Center for Supercomputing Applications (NCSA) for 15 years. He came to the Desert Research Institute in 2004 as the Technical and Acting Director of the Center for Advanced Visualization, Computation,and Modeling (CAVCaM). His research areas include Virtual Reality and Scientific Data Visualization. His email address is: `bill.sherman@dri.edu`