Coupling Massively Parallel Volume Rendering with Autostereoscopic 3D Display Environments

Tom Peterka[1], Robert Ross[1], Hongfeng Yu[2], Kwan-Liu Ma[2], Robert L. Kooima[3]
[1]Argonne National Laboratory
[2]University of California at Davis
[3]University of Illinois at Chicago
Address correspondence to tpeterka@mcs.anl.gov

Abstract

Scientific datasets are growing at incredible rates, resulting in two challenges for visualization researchers and domain scientists: maintaining performance and extracting meaningful content. These tasks have always been goals of scientific visualization, but their importance – and difficulty – grows with the scale of the data. At Argonne National Laboratory, we have been attacking these two challenges independently. We can mitigate the effects of extreme data scale by performing software visualization directly on the same supercomputer as the simulation, in our case the IBM Blue Gene/P (BG/P). We can improve information perception with autostereoscopic (autostereo, 3D with no glasses) display technology by building immersive, interactive workspaces where scientists can engage their data. By coupling these two areas of research, massively parallel rendering with novel 3D display systems, we expect to spur new development in each from the requirements imposed by the other.

In this research, we connect the two facets of our previous work into a complete solution by generating a stereo pair of direct volume renderings for each time-step of a time-varying dataset and streaming the pair of images to several autostereo displays. At the display side, we interleave left and right image pairs into a parallax barrier pattern and render the resulting autostereoscopic image. In this first attempt at coupling massive volume rendering running in real time on BG/P with novel 3D displays, we solve functionality problems of streaming, pipelining, and show how stereo can be approximated with a parallel projection ray casting algorithm. We collect initial performance measures that indicate a promising start but that also expose room for improvement. The future challenges in this process are to generate large visualizations at interactive rates and to provide meaningful interaction paradigms that extract more meaning from large data than would have been possible otherwise.

Classification: I3.1 [Hardware Architecture]: Parallel processing, I3.2 [Graphics Systems]: Distributed / network graphics, I3.7 [Three-Dimensional Graphics and Realism]: Raytracing, Virtual Reality, I3.8 [Applications]

Keywords: Large scale simulation and visualization; Real-time and/or interactive visualization; innovative volume visualization; massively parallel volume visualization; autostereoscopic 3D displays

1. Introduction

The purpose of scientific visualization is to convert electronic information into human information. Visualization bridges these two information formats, mapping data into forms amenable for human understanding, and its success depends on the best management of both scientific data and human perception. As data grow in size and complexity, functions of both the digital visualization systems and the human cognitive systems are strained. Hence, as groups such as the Institute for Ultra-Scale Visualization [1] prepare for datasets in the scale of petabytes, it is imperative to attack both problems concurrently: digital performance and human perception.

Peterka et al. are actively studying the data management problem by utilizing a new supercomputer, the IBM Blue Gene/P (BG/P), not only for computing simulations but also for visualizing the resulting data [19, 20]. This paper is the first research that links massively parallel visualization across thousands of supercomputer cores to virtual reality (VR) display interfaces that enhance visual understanding. The BG/P supercomputer generates high-quality, illuminated stereo pairs of direct volume rendered images and streams these images to an autostereo VR system. The display client interleaves left- and right-eye images, and in conjunction with a parallax barrier [23], displays the resulting 3D image in first-person perspective to the scientist.

Stereoscopic 3D offers two perceptual advantages over monoscopic 2D. First, binocular disparity is a powerful depth cue not only in computer graphics applications [24], but in our daily lives as well [22]. 3D depth disambiguates data. While 2D depth cues such as occlusion and motion parallax provide hints about the *relative* depth of objects, it is only 3D stereopsis that gives the *absolute* positions of objects in depth. Second, stereo effectively extends the available resolution of the display device by multiplying the viewable area by depth. With stereo, more data can be packed into the same physical display space without cluttering the scene because data are spaced in the depth direction.

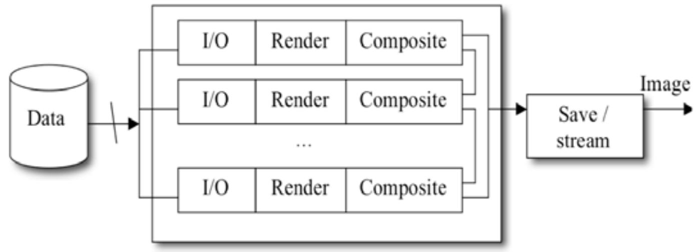

Figure 1: Massively parallel volume rendering consists of I/O, rendering, and compositing running in parallel on thousands of cores.

Autostereoscopy is the removal of stereo viewing glasses from the stereo viewing system. This represents the current state-of-the-art in VR display devices because it frees the user from the encumbrances normally associated with stereoscopic viewing. This is an important development for three reasons. First, it raises the level of engagement within the virtual world. Second, it more closely resembles a natural, human-like interface to the data. Lastly, the removal of encumbrances makes it possible to multiplex the visualization task into the scientist's normal, everyday, unpredictable workflow.

2. Background

This research draws on two previous areas that have been disconnected until now: parallel visualization and 3D display systems.

2.1 Massively Parallel Volume Rendering

In [20], Peterka et al. implemented and tested a massively parallel volume rendering algorithm on the BG/P. Figure 1 shows that the algorithm consists of three serial stages: file I/O, rendering, and compositing, which are performed in parallel on many cores. We demonstrated scalability up to 650 million data elements (864^3) per time-step and 4K BG/P cores, resulting in an end-to-end frame time of 3 seconds, including file I/O. File I/O was the largest contributing factor to the total frame time at this scale. In a follow-up work [19], Peterka et al. demonstrate three improvements to the parallel volume rendering algorithm on BG/P. The first is improved load balancing that leads to better scalability and efficiency. The second is the addition of a lighting model and its use in performance tests, along with larger datasets and more cores. The largest example to date is 16K BG/P cores used to render a dataset consisting of 11 billion elements per time-step. The third improvement, parallel pipelining, is to distribute the total number of cores among a number of consecutive time-steps in order to successfully hide the I/O time. With these features, the same 864^3 dataset as in [20] can now be rendered with high quality lighting at .8 frames per second. 8K total cores arranged into 16 parallel pipelines of 512 cores each were used to achieve this result.

2.2 Stereoscopic and Autostereoscopic Display Systems

Tiled LCD and projector walls, while high resolution and wide field-of-view, are 2D devices that do not exploit our most effective depth discriminator: binocular vision. Stereoscopic displays began with head mounted displays (HMDs), evolved to the CAVE in 1992 [5, 6], and subsequently down-sized to single-wall versions of the CAVE such as the GeoWall [9]. These technologies are still used today, and all require some form of active or passive eyewear to be worn by the user. CAVEs have also been coupled with supercomputer volume rendering by Ohno and Kageyama [15] on datasets as large as 13 GB.

Autostereo displays represent the natural next step in the evolution of VR systems: stereo without the glasses. Quality matching that of the CAVE has

Figure 2: Varrier autostereo display systems exist in a variety of form factors and sites.

been demonstrated with the *Varrier* system [23], and this system has been deployed in a variety of form factors and at a number of physical sites shown in Figure 2 [16]. Varrier (see the left side of Figure 3) is an example of a static parallax barrier: two viewpoints spatially multiplexed such that only the image stripes for a given viewpoint are visible by that eye. Only two viewpoints are generated and follow the user in space through the use of a head tracking position sensing system. The parallax barrier is *static* because it is a physical device mounted in front of the LCD display. Alternatively, the parallax barrier can be *dynamic*. In the right side of Figure 3, the printed film is replaced with a second LCD display, such that the monitor consists of a stack of two LCD layers illuminated by a common backlight [17, 18]. We have implemented both static and dynamic display types at Argonne National Laboratory.

3. Method and Implementation

This section describes implementation details of both hardware and software at the server (BG/P) and client (Varrier). We begin, however, with an overview of the dataset characteristics.

3.1 Scientific Data

Our test dataset comes from astrophysics data representing various physical quantities during the early stages of supernova core collapse. Quantities such as pressure, density, velocity and momentum are simulated by a computational code run by John Blondin of North Carolina State University and Anthony Mezzacappa of Oak Ridge National Laboratory [3]. Scalar data are computed on a structured grid. In the results for this paper, entropy is stored as a 32-bit scalar value over a structured grid size of dimensions 864^3, or 2.5 GB per time-step. The data are time-varying, so each time step must be first read from storage before rendering. Nearly ¾ billion elements per time-step are rendered on 4K BG/P cores, including two high-quality images per time-step with lighting.

3.2 Volume Rendering Algorithm

The classic software ray casting algorithm of Levoy [12, 13] is executed in parallel by dividing the data statically among 4K cores. Partial images are then composited using a sort-last scheme. Ultimately the resulting image is either saved to a file or streamed to a display device and the process repeats for the next time-step. Details of the parallel implementation and its performance appear in Peterka et al. [19, 20], but a few salient features of the algorithm are summarized below:

- The programming model is MPI distributed memory message passing.
- Collective I/O using MPI-2 and PVFS (Parallel Virtual File System) [4] permits all cores to simultaneously read only their subset of the dataset. No single core reads the entire dataset.
- Data are partitioned statically and the load is approximately balanced via round-robin allocation of multiple data blocks to each core.
- Composition along a ray occurs in front-to-back order with the “over” operator [21] and includes early ray termination. No other optimizations such as level of detail (LOD), octree hierarchy, or empty space skipping are employed.
- The lighting model includes ambient, diffuse, and specular reflection.
- The direct-send compositing algorithm [8, 14] is used to combine subimages into a final result.
- Multiple parallel pipelines execute several time-steps concurrently.

3.3 Local Display Algorithm

A client process displays autostereo images at the local display system. This process is written in a high-level VR scripting environment called Electro [10]. Electro provides support for rapid deployment of virtual worlds that are structured in a scene-graph hierarchy. Among its many features, Electro supports configuration for numerous VR display devices such as CAVE, GeoWall, and Varrier. A common programming interface drives mono, stereo, and autostereo displays, from single-tile desktops to multitile wall displays. Scene content can originate locally as scene-graph entities or rendered images may stream from remote sources. A flipbook animation of prerendered frames may also be stored locally for later playback. Electro supports various tracking systems for input of users’ positional and rotational information, and navigation devices such as keyboards, mice, or 3D wands.

Electro natively includes the Varrier method to spatially combine left- and right-eye images into an

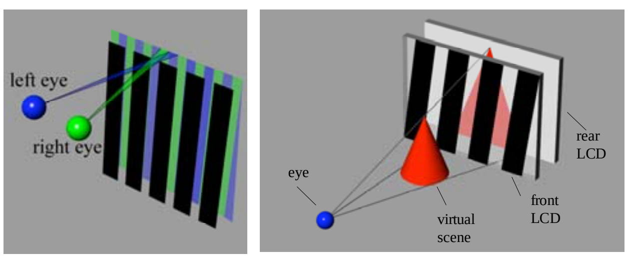

Figure 3: Varrier method generates two views corresponding to the user’s eye positions. The left image depicts a static barrier, a physical device placed in front of an LCD display. In the right image, a second LCD layer renders a dynamic barrier.

interleaved autostereo image that is calibrated for the physical parameters of the parallax barrier. The autostereo combiner implementation is highly optimized through the use of GPU vertex and fragment shaders, and Kooima et al. have shown that it adds no significant overhead beyond the computation of two images in quad-buffered stereo [11].

3.4 Display Devices

We have built two Varrier static barrier displays at Argonne National Laboratory for this research: the single-panel desktop 30-inch Personal Varrier and the six-panel 60-inch Varrier. We have also built a dynamic barrier display per Peterka et al. [18] and will link it to the same BG/P visualizations. Due to its unique hardware requirements, the dynamic barrier programming interface has not been incorporated into Electro yet, but no technical details impede the streaming of volume rendered images to this display system as well. All three autostereo display systems at Argonne National Laboratory appear in Figure 4, and a single Linux desktop machine drives each. The machine for the 60-inch display has 4 NVIDIA 8600 GTS graphics cards. Two cards are located in PCI-Express 16X slots while two are in PCI-Express 8X slots. The entire system effectively runs at the slower bus rate, PCI-E 8X.

First-person perspective, two-view autostereo requires positional and optional rotational information about the viewer's head location and orientation. This information is provided to Electro by a tracking system. The smaller desktop displays are fitted with a low-cost camera tracker designed for the home video-game industry [2]. A tiny, self-contained infrared camera captures images of a triad of retroreflective markers worn on a visor. A Windows laptop reads position and orientation data and transmits it to Electro over a UDP port. The larger 60-inch display is fitted with a custom tracking solution developed by Girado et al. [7] It requires no gear to be worn and captures and decodes only position (not orientation) information at 340 Hz through a set of infrared cameras and neural network algorithms. A dedicated Windows machine with an Intel quad-core processor executes the neural networks and sends 3D position data to Electro over a UDP socket.

4. Results

Several details are described in this section, including combining the BG/P parallel pipelines into pairs of stereo pipelines, combining image pairs into data streams, and approximating stereo perspective from a pair of parallel projections.

Figure 4: We deployed three autostereo systems to date at Argonne: a 30-inch personal Varrier display (left side of left image), a 17-inch active barrier display (center of left image), and a 60-inch 6-panel Varrier display (right image)

Preliminary end-to-end frame rate data are also presented.

4.1 Communication and Pipelining

BG/P requires at least two hops to transmit data over a socket to a display device. The first hop is from the BG/P core to one of the front-end login nodes, followed by a hop from the login node to an outside IP address. A "forwarder daemon" runs on the login node and forwards the data through it to the display system. At the display side, Electro organizes incoming packets via a 12-byte header that precedes each image section. The header specifies where each section fits within the total image; a section can be as small as a single image row or as large as a complete image, depending on network limitations. Before streaming, the left- and right-eye images of an image pair are combined into one image that is the original width but twice the height of a single view. That is, left and right images are stacked one on top of the other. Each of the individual views is sent in a separate packet with its own header, but the contents of the header reflect the fact that the two packets constitute one larger image. Figure 5 shows the structure of one entire stereo image transmission.

Peterka et al. describe a multiple pipeline architecture in [19] that is used to hide file I/O latency by processing several time-steps concurrently. They have demonstrated that up to 16 parallel pipelines have produced speedups for this same 864^3 dataset. We adopt the same parallel pipeline architecture but modify it so that half of the pipelines are processing each eye's view. Images exit BG/P's cores via multiple, parallel sockets, one socket per pipeline. Eventually these transmissions must be serialized because Electro receives and displays frames sequentially. This task is performed by the forwarder daemon. It takes turns reading each

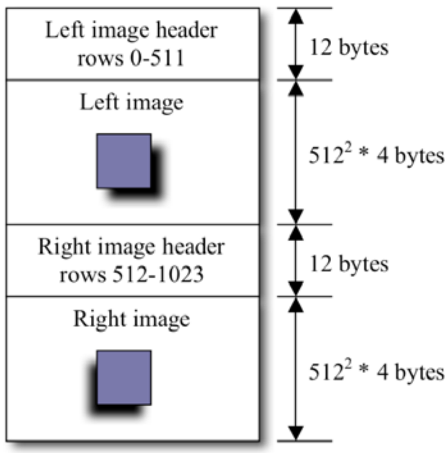

Figure 5: A sample packet structure is shown for a pair of left and right 512 x 512 images that constitute a single 512 x 1024 stereo image.

of the incoming sockets from BG/P and forwards the header and payload sequentially on a single outbound socket to the display device. It also meters the outgoing packets in time so they are sent at a consistent rate, despite arriving in bursts.

4.2 Projection Models

Stereo and autostereo projections are almost always performed using perspective projection. In ray casting, each ray would originate at the eye position and extend through each pixel in the image plane. This corresponds to the way that we see in real life, including foreshortening of distance. Parallel projection is an alternative approximation where all viewing rays are parallel; at infinite distance the two models are equivalent. Our volume rendering application currently uses parallel projection for historical reasons in its ray casting algorithm. We do not expect changing the algorithm to perspective projection to violate any optimizations requiring the use of parallel projection.

Meanwhile, for this research we incorporate a temporary workaround that produces believable results to the casual observer, while we work to change the ray casting algorithm permanently. The temporary method produces two parallel projections from the volume rendering algorithm, combines them by the Varrier autostereo method, and displays the result in correct perspective projection at the autostereo display. Figure 6 illustrates this workflow.

To display the final, combined autostereo image, the two views are spatially interleaved and textured onto a polygon in the virtual world. This polygon is intentionally set back some distance behind the screen, as if "inside" the display. The trained observer will realize that the resulting image is correctly located within the 3D space inside of the display monitor, but that the polygon is indeed flat and that the image on it is created by parallel rather than perspective projection. Many viewers, however, persuaded by the 3D position of the image and assisted in their depth perception by cues such as accurate lighting, choose to perceive 3D without noticing the lack of perspective projection. We reiterate that in our case this technique is temporary, but suggest that the method can offer satisfactory results in absence of true perspective projection.

4.3 Performance Results

Figure 7 shows the timing results of the end-to-end process of volume rendering, streaming, and viewing stereo pairs of images from BG/P to Varrier. The frame time is the total interframe latency between the arrivals of new time-steps at the autostereo client display. This is the time for a complete stereo image, consisting of two separate renderings of the two eye positions. Two image sizes were tested, 512^2 and 1024^2. Lighting is enabled, and Figure 7 also shows a sample of the left-eye image from one time-step. The best stereo frame time achieved so far for the 512^2 image is 3.9 s, and for the 1024^2 image is 7.3 s. For this dataset, approximately 70% of the frame time is spent performing file I/O when rendering a single image [20]. With stereo rendering, this fraction is one-half of the original, or 35%, because two views are rendered from a single access to storage. A maximum of 4K cores were used for this test, arranged into parallel pipelines of 512, 1K, and 2K cores per pipe. A minimum of two pipes were used,

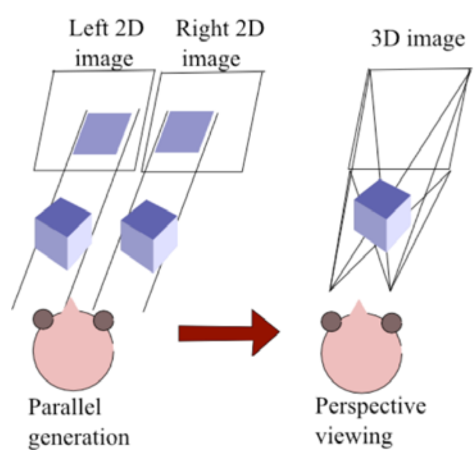

Figure 6: A stereo perspective projection is approximated from two parallel projections that are subsequently rendered onto an image plane by perspective projection.

one for each eye's view. In all cases, more pipelines were faster than fewer pipelines because of the increased potential for overlapping I/O operations [19].

Although new time-steps are generated relatively slowly by BG/P, refreshing the same textured polygon, for example, when moving to a new head position, occurs at full frame rate at the client (60 Hz). The decoupling of generation rate and viewing rate permits the scientist to move freely, and the stereo imagery tracks the new position in real time. However, thus far there is no user interaction in the workflow. Once the head position is used to actually steer the visualization, the generation frame rate will need to be much faster.

There is room for improvement in the frame time. We have shown in previous tests [19] that we can cut the time for a single image by a factor of 2 by applying more processor cores. Moreover, when combining two views into one stereo image, it is unnecessary to generate the original views in their full resolution. One half of the horizontal resolution will be lost during the autostereo combination step; in fact, Electro does a similar optimization through a quality factor parameter [11]. We can generate the original images using half of the horizontal resolution and save another factor of 2 in the frame time. One last optimization is to increase the spacing of the samples taken along the rays of the ray casting computation. We normally match this sample spacing to the data spacing in order to maximize quality. Like resolution however, switching to stereo mode permits the reduction of the amount of work done per eye while maintaining constant overall quality. We will experiment with all of these optimizations in the future, but we are confident that a factor of 4-6 speedup is possible from the current performance.

With these optimizations, we expect 2 frames per second (2 Hz) for the 512^2 image size and 1 Hz for 1024^2. This is still too slow for real-time interactions with the data. This may be improved further with deeper algorithmic optimizations such as level-of-detail data organization and empty space skipping, although the trade-off is preprocessing time. Other possibilities for improvement involve local image-based approximations at the display client. We are continuing to pursue these directions and consider this initial research a success in defining the required performance goals and highlighting the next steps to take.

We also implemented a method of volume rendering stereo images offline, copying theses images to the local display client, and displaying a flipbook animation in autostereo at full frame rate. The image files are stored in raw RGBA mode, 4 bytes per pixel, and left-right images are stacked on top of each other similar to the way that they are streamed, minus the headers. The flipbook feature is convenient for demonstrations and permits scientists to review previously rendered results.

5. Conclusions and Future Work

We consider this research to be a successful first step toward the merging of very large scale, massively parallel visualization, with engaging work environments for scientists. Only by addressing both the scale and the perceptual issues can we successfully migrate to the petascale era of scientific computation and simulation. We have

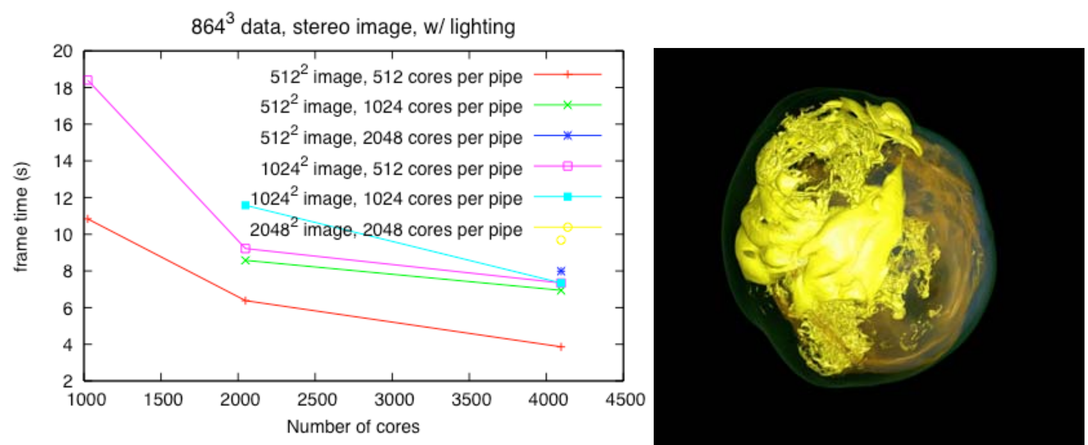

Figure 7: Frame time for receiving stereo images of two sizes, 512^2 and 1024^2, is plotted. Pipeline sizes of 512, 1024, and 2048 cores per pipe are tested. The right image shows the result of one eye's image, rendered with high-quality lighting.

demonstrated working functionality at modest scale and have explored what next steps can improve performance. We discussed implementation details at both the server and client sides, including pipeline organization, communication, projection models, and performance results.

This is, however, only a start, and this research identifies several future needs before real-time autostereo visualization of supercomputer volume rendering can be used in practice. First, minimum frame rates of 15 Hz are necessary. This implies that an order of magnitude improvement is needed, and local interpolation between views is one possibility that we are considering. Once this rate has been achieved, interactive steering of the visualization via head movements and navigation devices can successfully be realized. Perspective projection of viewing rays during the ray casting algorithm is also necessary, although we do not expect this to affect performance.

Acknowledgments

We thank John Blondin and Anthony Mezzacappa for making their dataset available for this research. This work was supported in part by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357. Work is also supported in part by NSF through grants CNS-0551727 and CCF-0325934, and DOE with agreement No. DE-FC02-06ER25777. Varrier is a trademark of the Board of Trustees of the University of Illinois.

References

[1] SciDAC Institute for Ultra-Scale Visualization. http://ultravis.ucdavis.edu/ 2007.

[2] TrackIR. http://www.naturalpoint.com/trackir/02-products/product-TrackIR-4-PRO.html 2008.

[3] BLONDIN, J. M., MEZZACAPPA, A. DEMARINO, C.: Stability of Standing Accretion Shocks, with an Eye Toward Core Collapse Supernovae. *The Astrophysics Journal*, 584, 2, (2003), 971.

[4] CARNS, P., LIGON, W. B. I., ROSS, R. THAKUR, R.: PVFS: A Parallel File System for Linux Clusters. *Proceedings of 4th Annual Linux Showcase & Conference*, Atlanta, GA, (2000), 28.

[5] CRUZ-NEIRA, C., SANDIN, D. DEFANTI, T.: Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE. *Proceedings of ACM SIGGRAPH 1993*, (1993), 135-142.

[6] CRUZ-NEIRA, C., SANDIN, D., DEFANTI, T., KENYON, R. HART, J.: The CAVE: Audio Visual Experience Automatic Virtual Environment. *Communications of the ACM*, 35, 6, (1992), 64-72.

[7] GIRADO, J., PETERKA, T., KOOIMA, R., GIRADO, J., GE, J., SANDIN, D., JOHNSON, A., LEIGH, J. DEFANTI, T.: Real Time Neural Network-based Face Tracker for VR Displays. *Proceedings of IEEE Virtual Reality 2007*, Charlotte NC, (March 2007).

[8] HSU, W. M.: Segmented Ray Casting for Data Parallel Volume Rendering. *Proceedings of 1993 Parallel Rendering Symposium*, San Jose, CA, (1993), 7-14.

[9] JOHNSON, A., LEIGH, J., MORIN, P. VAN KEKEN, P.: GeoWall: Stereoscopic Visualization for Geoscience Research and Education. *IEEE Computer Graphics and Applications*, 26, 6, (2006), 10-14.

[10] Electro. http://www.evl.uic.edu/rlk/electro/ 2008.

[11] KOOIMA, R., PETERKA, T., GIRADO, J., GE, J., SANDIN, D. DEFANTI, T.: A GPU Sub-pixel Algorithm for Autostereoscopic Virtual Reality. *Proceedings of IEEE Virtual Reality 2007*, Charlotte, NC, (March 2007).

[12] LEVOY, M.: Display of Surfaces from Volume Data. *IEEE Computer Graphics and Applications*, 8, 3, (May 1988), 29-37.

[13] LEVOY, M.: Efficient Ray Tracing of Volume Data. *ACM Transactions on Graphics*, 9, 3, (July 1990), 245-261.

[14] NEUMANN, U.: Communication Costs for Parallel Volume-Rendering Algorithms. *IEEE Computer Graphics and Applications*, 14, 4, (July 1994), 49-58.

[15] OHNO, N. KAGEYAMA, A.: Scientific Visualization of Geophysical Simulation Data by the CAVE VR System with Volume Rendering. *Physics of the Earth and Planetary Interiors*, 163, 1-4, (August 2007), 305-311.

[16] PETERKA, T., KOOIMA, R., GIRADO, J., GE, J., SANDIN, D. DEFANTI, T.: Evolution of the Varrier Autostereoscopic VR Display. *Proceedings of IS&T / SPIE Electronic Imaging 2007*, San Jose, CA, (January 2007).

[17] PETERKA, T., KOOIMA, R., GIRADO, J., GE, J., SANDIN, D., JOHNSON, A., LEIGH, J., SCHULZE, J. DEFANTI, T.: Dynallax: Solid State Dynamic Barrier Autostereoscopic VR Display. *Proceedings of IEEE Virtual Reality 2007*, Charlotte, NC, (March 2007).

[18] PETERKA, T., KOOIMA, R., SANDIN, D., JOHNSON, A., LEIGH, J. DEFANTI, T.: Advances in the Dynallax Solid-State Dynamic Parallax Barrier Autostereoscopic Visualization Display System. *IEEE Transactions on Visualization and Computer Graphics*, 14, 3, (May / June 2008), 487-499.

[19] PETERKA, T., ROSS, R., YU, H. MA, K.-L.: Assessing and Improving Large-Scale Parallel Volume Rendering on the IBM Blue Gene/P. *Proceedings of Supercomputing 2008*, Austin TX, (November 2008).
[20] PETERKA, T., YU, H., ROSS, R. MA, K.-L.: Parallel Volume Rendering on the IBM Blue Gene/P. *Proceedings of Eurographics Parallel Graphics and Visualization Symposium 2008,* Crete, Greece, (April 2008).
[21] PORTER, T. DUFF, T.: Compositing Digital Images. *Proceedings of 11th Annual Conference on Computer Graphics and Interactive Techniques*, (1984), 253-259.
[22] SACKS, O.: A Neurologists Notebook, "Stereo Sue" *New Yorker*, pp. 64, 2006.
[23] SANDIN, D., MARGOLIS, T., GE, J., GIRADO, J., PETERKA, T. DEFANTI, T.: The Varrier Autostereoscopic Virtual Reality Display. *ACM Transactions on Graphics, Proceedings of ACM*, 24, 3, (2005), 894-903.
[24] WARE, C. MITCHELL, P.: Reevaluating Stereo and Motion Cues for Visualizing Graphs in Three Dimensions. 51-58.