A Multi-viewer Tiled Autostereoscopic Virtual Reality Display

Robert Kooima[1], Andrew Prudhomme[2], Jurgen Schulze[2], Daniel Sandin[2,3], and Thomas DeFanti[2]

[1]Center for Computation & Technology, Louisiana State University (LSU)
[2]California Institute for Telecommunications and Information Technology, University of California San Diego (UCSD)
[3]Electronic Visualization Laboratory, University of Illinois at Chicago (UIC)

Abstract

Recognizing the value of autostereoscopy for 3D displays in public contexts, we pursue the goal of large-scale, high-resolution, immersive virtual reality using lenticular displays. Our contributions include the scalable tiling of lenticular displays to large fields of view and the use of GPU image interleaving and application optimization for real-time performance. In this context, we examine several ways to improve group-viewing by combining user tracking with multi-view displays.

CR Categories: I.3.1 [Computer Graphics]: Hardware Architecture—Three-Dimensional Displays; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual Reality

Keywords: autostereo, lenticular, immersive, multi-user, VR

1 Introduction

Active stereo viewing technologies hold a valued place in the research laboratory, and passive stereo viewing technologies continue to see increasing adoption in movie theaters and other stable display venues. Yet, there remain many contexts in which the inconvenience of stereo glasses preclude the use of stereoscopic display. Examples include museums, where one or more exhibits in a traditional gallery may benefit from 3D visualization, but where the expected viewer engagement with each is not sufficient to warrant the expense of handling, maintaining, and securing a large collection of stereo glasses.

Autostereoscopic display is a viable alternative in such contexts. Several multi-viewer, multi-channel autostereoscopic displays are currently on the market, but their small size limits their areas of application. Before such displays will find broader use they must be scaled up to provide wider comfortable viewing areas for larger groups of people.

Our research pursues this goal. Using large arrays of small, off-the-shelf lenticular displays, we have installed a number of scalable autostereoscopic display systems capable of serving sizable groups. We have devised and implemented an efficient, GPU-accellerated mechanism for performing real-time rendering to lenticular displays, detailed in Section 2. We have established processes for superimposing the autostereo functionality of individual displays and synchronizing rendering to them, resulting in large arrays that behave as a single display, described in Section 3. And we have experimented with active measures for the mitigation of the fundamental discontinuities in lenticular displays using user tracking, discussed in Section 4.

VRST 2010, Hong Kong, November 22 – 24, 2010.
© 2010 ACM 978-1-4503-0441-2/10/0010 $10.00

2 Lenticular Rendering

Our installations use the Alioscopy 24″ and 42″ 3DHD displays, though our approaches have been demonstrated on a variety of similar lenticular and parallax barrier displays. The Alioscopy displays are capable of presenting eight independent image channels, projecting them onto a plane of focus at a distance of $2.95\,m$. Other displays have different sizes, resolutions, lenticular pitches, focal lengths, and channel counts, but the principles remain the same.

We take real-time interactivity to be a defining characteristic of virtual reality, and rendering efficiency is critically important to the maintenance of interactive frame rates. Lenticular display systems pose a challenge to rendering efficiency due to the need to render the scene from multiple different view points, plus the added cost of interleaving the individual sub-pixels of these renderings in accordance with the layout of the lenticular array. Despite this complexity, we achieve real-time display using the programmable graphics processor (GPU).

The approach is a generalization of a GPU-based algorithm for autostereo rendering [Kooima et al. 2007] that targeted the Varrier parallax barrier display [Sandin et al. 2005]. As a single-user system, the Varrier presents exactly two image channels. The lenticular implementation generalizes this to an arbitrary number of views, limited only by the number of texture sampler units supplied by the graphics hardware.

2.1 Scene Rendering

Prior to display on-screen, the 3D scene is rendered once per channel into off-screen render buffers. The most straightforward approach is to iterate over all channels, bind a color buffer for each, set the appropriate perspective projection for the channel, and render normally. A single depth buffer suffices for all color buffers. This approach places a minimum of graphics state requirements upon the application, and simplifies the porting of existing 3D applications for lenticular display. An optimized alternative uses geometry shaders with multiple render targets to render all channels simultaneously, though at the cost of some complexity and imposition upon the application.

The most significant rendering optimization recognizes that only a fraction of rendered sub-pixels will ultimately appear on-screen. In the case of the Alioscopy, only one out of every eight sub-pixels is used. It follows that the off-screen render buffer may be shrunk to one eighth its width without a perceptible loss of quality. In this way, the total fragment cost of the lenticular rendering is equal to the fragment cost of normal display, which is very significant for fill-limited applications.

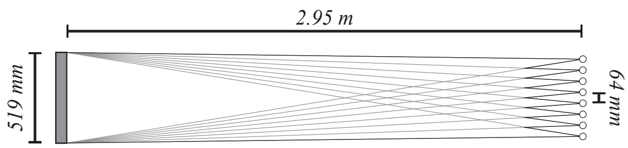

Figure 1: *The view positions (right) and off-axis perspective frusta used to render the eight channels for the 24″ Alioscopy (left).*

2.2 Interleaving

The sub-pixel interleaving of rendered scene images is determined by two sets of parameters. The first is the set of all view positions, a static array of points equally distributed on the plane of focus, separated from one another by the interocular distance. These positions give the centers of the projected channels, as depicted in Figure 1. User tracking causes these positions to be dynamically recomputed per frame, as we will discuss in Section 4.

The second set of interleaving parameters gives a definition of the physical nature of the lenticular array, described here along with example values for the 24″ Alioscopy. *Pitch* gives the width of each lenticule ($0.682\,mm$). *Angle* gives the rotation of the lenticular array relative to the pixel grid ($18.435°$). *Duty cycle* gives the fraction of a lenticule dedicated to each channel ($1/8$). *Optical thickness* gives the effective focal length of the lenticular array ($\approx 4.2\,mm$). *Shift* gives a left-to-right calibration value that tunes the focus direction.

A completed channel interleaving is equivalent to the *sum* of all channels, with each channel *modulated* by a mask that is light where a channel is visible and dark where not. This mask is 2D, but in the coordinate system rotated into alignment with the lenticular array, it is vertically constant and horizontally repeating. From this perspective, the mask is 1D and representable as a rectangular wave, with the lenticular pitch giving the wavelength and the duty cycle giving the ratio of light to dark.

The process of interleaving follows from this interpretation. Each LCD sub-pixel has a position in the modulating waveform, called its *phase*, determined by the position of the sub-pixel on the screen, the view position of the associated channel, and the lenticular parameters. We compute this phase per sub-pixel and evaluate the waveform as a simple step function, with the duty cycle giving the edge value.

To begin the process, a screen-filling rectangle is rendered, and a vertex program executes for each of its four vertices. Vertex positions give the measured locations of the screen corners, and an appropriate perspective projection brings these vertices to the corners of the frame buffer. Each vertex is also transformed from user space into lenticular phase space, and the composition of this transformation follows.

Let v be the vector from the center of the display to the view position, in the display-centric coordinate system. Also, let p be lenticular pitch, s be shift, t be optical thickness, and a be angle. The first step of the lenticular transform is the projection of positions on the plane of the pixel grid onto the plane of the lenticular, a distance given by the optical thickness. This introduces a shortened shift value $s' = s \cdot (v_z - t)/v_z$ and a reduced pitch $p' = p \cdot (v_z - t)/v_z$, and a parallax offset along the horizontal and vertical display axes, $d_x = t \cdot v_x/v_z$ and $d_y = t \cdot v_y/v_z$.

The lenticular transform translates the 3D position of each sub-pixel by the projected shift and offset, rotates them about the center of the screen through the lenticular angle, and scales them by the projected pitch, thus normalizing the wavelength to one. $M = Scale(p', p', 1) \cdot Rotate(0, 0, a) \cdot Translate(d_x - s', d_y, 0)$.

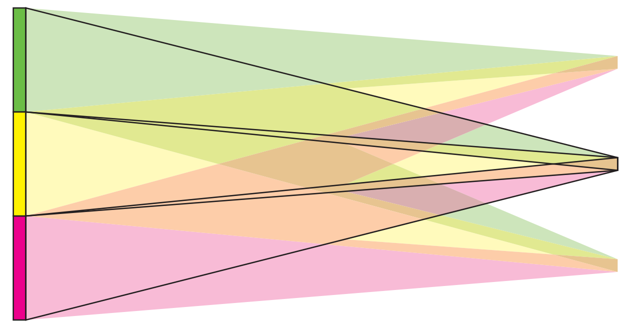

Figure 3: *The tiling of lenticular displays achieved by shifting the center channels of all displays into alignment at the plane of focus.*

The x value of the resulting vector gives the normalized 1D phase for each sub-pixel. Corner sub-pixel phases are output from the vertex shader as varying values, one vector per channel. The linearly-interpolating rasterizer produces the correct phase for each sub-pixel of the screen. The fragment shader receives the interpolated phases and computes the step function of the fraction of each, giving the RGB masks. Each channel is sampled from its off-screen render buffer and modulated, with the sum of all outputs written to the frame buffer. In total, the interleaving process incurs the overhead of rendering a single screen-sized rectangle, plus one coherent texture access per channel per pixel. The cost of this has been negligible since the introduction of programmable graphics hardware.

3 Lenticular Clustering

Immersion is an important property of VR usually achieved by filling the user's field of view with stereoscopic imagery. This is the motivating principle behind large VR environments for multiple users, such as the CAVE [Cruz-Neira et al. 1993]. But the CAVE benefits from the scalability of rear-projection, and lenticular displays of similar size are far from being feasible. To achieve multi-user immersion with lenticular displays we use an array of them. We apply software techniques to ensure that their stereoscopic display functionalities coincide, and cluster technologies to ensure that their contents synchronize. This effect is highly scalable, as demonstrated by the 6×3 installation shown in Figure 2.

3.1 Autostereo Superimposition

Normally, a lenticular display projects its channels forward, and thus two adjacent lenticular displays project their channels parallel to one another. For multiple lenticular displays to appear to the user as a single continuous display, the projected channels must be brought into alignment at the plane of focus. The shift parameter of the lenticular interleaver described in Section 2.2 enables this.

Figure 3 shows the scenario. One display is chosen to be the primary and its center channel defines the standard. It is illuminated with a constant color per channel, which projects a pattern of colored bars onto a white card, Figure 4. A second display is illuminated and its shift parameter is adjusted until the color patterns are visibly aligned. This process is repeated for all displays, bringing each into alignment with the primary. While the repeat discontinuity is still necessarily present, the alignment places the discontinuity at the same view position across the entire display. As a consequence of this, all channel repetitions align, enabling group viewing.

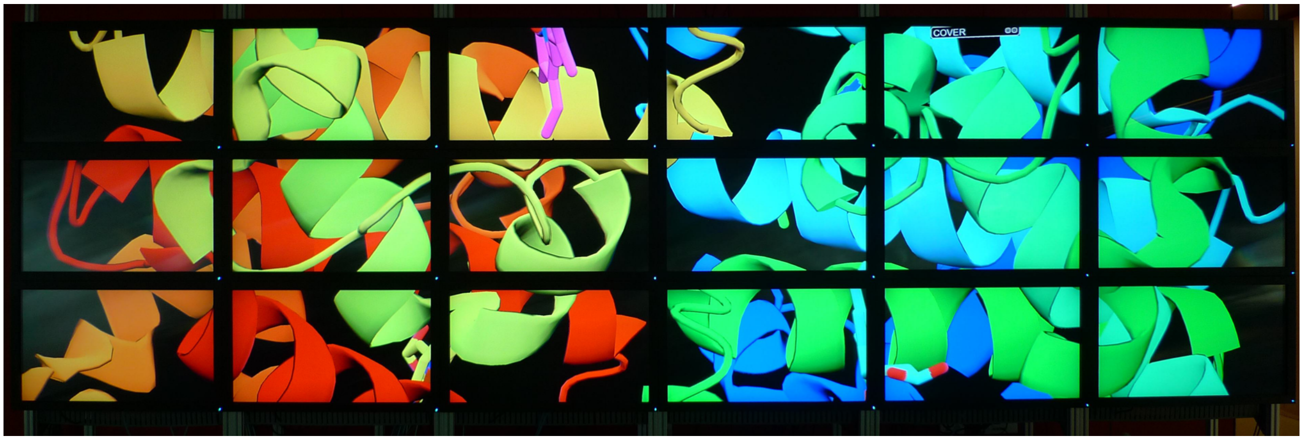

Figure 2: *The Rapidly Expandable Virtual Environment (REVE) , a 6×3 array of Alioscopy $42''$ 3DHD displays at the King Abdullah University of Science and Technology (KAUST) using COVISE. Photo by Tom DeFanti.*

Figure 4: *The repeating channel array, with solid colors projected onto a user and a white card at the plane of focus.*

3.2 Cluster Synchronization

For an array of small displays to appear as a single contiguous large display, the scene rendered to each display must be in synchronization. Fortunately, in our circumstance, the degree of synchronization necessary to provide the illusion of continuity is not high.

Perfectly adjacent displays must have extremely precise synchronization to appear continuous, and any update delay is apparent as a discontinuity in the frame. Hardware genlock synchronization is required. However, if there is even a small physical discontinuity between images, then the discontinuity is masked. With the combined bezels of adjacent Alioscopy displays over $40\,mm$ wide, a discontinuity is perceivable only when viewing critically. If the cluster synchronization mechanism and rendering hardware finishes a frame within the $60\,Hz$ refresh period then no significant discontinuity is apparent. Our implementation is compatible with hardware such as the NVIDIA Quadro G-Sync, and we have demonstrated on small scales that this does eliminate the discontinuity completely, but the cost of this solution is prohibitive at scale of our larger installations.

Software synchronization is straightforward, and a number of existing clustering software approaches satisfy the relatively lax timing requirement. There are three fundamental synchronization tasks to be performed: the application as a whole must be started across all rendering nodes, user interactions and other application states must be distributed across the cluster, and display updates and buffer swaps must be coordinated. The most flexible solution is to simply run an independent copy of the rendering application on each node, each configured with a subset of the display. Start-up is commonly performed using the secure shell with public-key authentication. State synchronization and screen update coordination use TCP socket communication. Examples of applications using this approach include the distributed visualization systems COVISE [Rantzau et al. 1996] and CGLX [Doerr and Kuester 2010]. Another effective solution is the Message Passing Interface (MPI). As an API for distributed software development, MPI handles application start-up and provides communication primitives for application state and display coordination.

4 User Tracking

One of the defining properties of a virtual reality system is the viewer-centered perspective, where-in the position of the viewer is tracked and the rendering updated accordingly. If the system supports group viewing, other *passive* viewers see the perspective of the tracked *active* viewer. The visual experience of a passive viewer in immersive systems such as CAVEs is not perspectively correct, but is compelling when positioned near the active viewer, facing the same direction. Thus, the CAVE is an effective environment for small groups of people. In contrast, the Varrier autostereo VR system produces a very disturbing experience for passive viewers, and is thus only appropriate for a single user.

Our research with multi-view displays is motivated by the goal of an autostereo VR system appropriate for small groups. Toward this end, we implemented three different methods of combining tracking with a multi-view lenticular display, with the goal of reducing the visibility and occurrence of autostereo disturbances. We informally evaluated the experience for the tracked and passive viewers.

Case 0. No Tracking: The default case of Figure 5-0 has no tracking. All viewers are passive. There is limited look-around of approximately eight degrees provided by the eight channels of the display. The viewer sees a correct stereo image across most of the viewing area, interrupted by the repeat discontinuity.

Case 1. Perspective Only: In Figure 5-1, the perspective is updated based on head-tracking, but no change is made to the direction in which channels are projected. The problem with this case is exaggerated perspective, as it effectively adds the perspective change due to tracking to the perspective change of the eight channels. The tracked viewer has complete look-around, but still sees the channel repeat discontinuity. During this transition, his perspective changes by twice that in the non-tracked case. The passive viewer sees a good stereo image with a perspective updated following the tracked

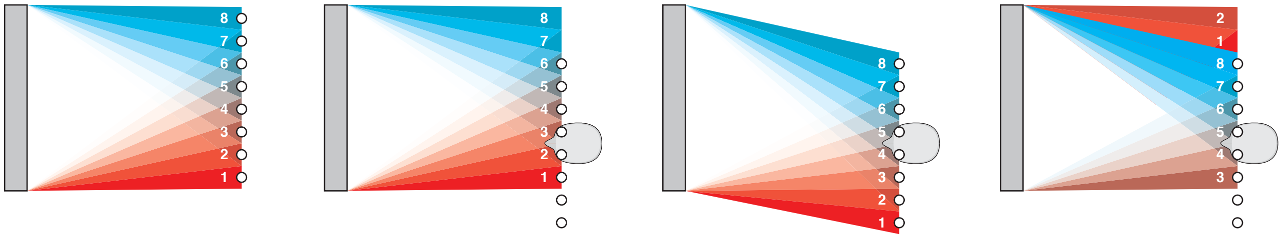

Figure 5: *The four experimental tracking cases. Numbered areas show channel projections. Circles show perspective view positions. The head shows the position of the actively-tracked user. 0) Untracked. 1) Perspective tracking. 2) Channel tracking. 3) Channel reassignment.*

viewer. As the passive viewer moves he may still pass through the repeat discontinuity as in the non-tracked case.

Case 2. Channel Adjustment: In Figure 5-2 we move the channels with the tracked positions, extrapolating eight view positions and supplying them to the interleaver. Unfortunately, in lenticular autostereo systems such as the Alioscopy, the channels cannot be moved smoothly, and only in discrete steps. As the tracked viewer moves into a transition between channels he begins to see the adjacent view before the channel perspective is updated to follow. When the update occurs, the perspective snaps back, jarring the image. The passive viewer also observes this snap-back as well as the repeat discontinuity as the tracked viewer moves.

Case 3. Channel Reassignment: In an effort to correct the flaws of Case 2, we keep the channel projection directions constant and change only how view points are mapped into them, as in Figure 5-3. Tracked viewer positions are quantized to the two nearest channel center points, and these positions are assigned to the middle channels, 4 and 5. This quantization eliminates the perspective snap-back. The remaining 6 view positions are extrapolated sequentially as usual. Of all tested tracking methods, Case 3 performs best. The tracked viewer has full look-around and never perceives the repeat discontinuity. Passive viewers enjoy good stereo, though the repeat discontinuity may pass over them due to tracked movement.

5 Future Work and Conclusion

The current research utilizes a camera-based tracking system requiring retro-reflective targets, which is not appropriate for public venues. In future installations we plan to use camera tracking without targets. We have developed such a tracker for use with the Personal Varrier autostereo display system [Girado et al. 2003] , and we plan to expand upon that work.

Depth range and orthostereo representation remain the most significant lacking issues, and we have tested a variety of adjacent-channel blending and depth-compression techniques in an effort to control the visibility of channel crosstalk and enable deeper scenes. These have been successful for some applications but less so in areas such as architectural walk-through. We plan to continue to work in this area, including the investigation non-linear stereo disparity control.

Despite a broad front of future work waiting to be pursued, we have met our initial goals of creating multi-user, large-scale, high-resolution autostereoscopic displays. These installations are big, bright, and immersive. They provide a solid foundation for continued research in autostereo display and virtual reality application development.

Acknowledgements

This publication is based on work supported in part by Award No US-2008-107/SA-C0064, made by King Abdullah University of Science and Technology (KAUST).

Additional support was provided by the Arts, Visualization, Advanced Technologies and Research (AVATAR) Initiative at the Center for Computation & Technology at Louisiana State University.

UCSD received major funding from the State of California for the StarCAVE. UIC receives major funding from the National Science Foundation (NSF), awards CNS-0420477 (LambdaVision), OCI-0943559 (SAGE) and CNS-0959053 (NG-CAVE), and also receives support from Sharp Laboratories of America and the State of Illinois. Also, UCSD and UIC were lead institutions of the NSF-funded OptIPuter project, award OCI-0225642.

References

CRUZ-NEIRA, C., SANDIN, D., AND DEFANTI, T. 1993. Surround-screen projection-based virtual reality: the design and implementation of the CAVE. In *Proceedings of the 20th Annual Conference on Computer graphics and Interactive Techniques*, ACM, 142.

DOERR, K., AND KUESTER, F. 2010. CGLX: A Scalable, High-performance Visualization Framework for Networked Display Environments. *IEEE Transactions on Visualization and Computer Graphics 99*, PrePrints.

GIRADO, J., SANDIN, D., DEFANTI, T., AND WOLF, L. 2003. Real-time camera-based face detection using a modified LAMSTAR neural network system. In *Proceedings of SPIE-IS&T Electronic Imaging*, 20–24.

KOOIMA, R., PETERKA, T., GIRADO, J., GE, J., SANDIN, D., AND DEFANTI, T. 2007. A GPU Sub-pixel Algorithm for Autostereoscopic Virtual Reality. In *IEEE Virtual Reality Conference, 2007. VR'07*, 131–137.

RANTZAU, D., LANG, U., AND RÜHLE, R. 1996. Collaborative and Interactive Visualization in a Distributed High Performance Software Environment. In *Proceedings of International Workshop on High Performance Computing for Graphics and Visualization, Swansea, Wales, '96*.

SANDIN, D., MARGOLIS, T., GE, J., GIRADO, J., PETERKA, T., AND DEFANTI, T. 2005. The Varrier™autostereoscopic virtual reality display. *ACM Transactions on Graphics (TOG) 24*, 3, 903.