Early Experiences and Challenges in Building and Using A Scalable Display Wall System

Kai Li, Han Chen, Yuqun Chen, Douglas W. Clark,
Perry Cook, Stefanos Damianakis, Georg Essl,
Adam Finkelstein, Thomas Funkhouser,
Timothy Housel, Allison Klein, Zhiyan Liu,
Emil Praun, Rudrajit Samanta, Ben Shedd,
Jaswinder Pal Singh, George Tzanetakis,
Jiannan Zheng

Department of Computer Science,
Princeton University, Princeton, NJ 08544
http://www.cs.princeton.edu/omnimedia/

Introduction

The Princeton Scalable Display Wall project explores how to build and use a large-format display with multiple commodity components. Our goal is to construct a collaborative space that fully utilizes a large-format display, immersive sound, and natural user interfaces.

Unlike most display wall systems today, which are built with high-end graphics machines and high-end projectors, our prototype system is built with low-cost commodity components: a cluster of PCs, PC graphics accelerators, consumer video and sound equipment, and portable presentation projectors. The advantages of this approach are low cost and technology tracking, as high-volume commodity components typically have better price/performance ratios and improve at faster rates than special-purpose hardware. The challenge is to use commodity components to construct a high-quality collaborative environment that delivers display, rendering, input, and sound performance competitive with, or better than, that delivered by the custom-designed, high-end graphics machine approach.

A schematic representation of our current display wall system is shown in Figure 1. It comprises an 8′×18′ rear projection screen with a 4×2 array of Proxima LCD polysilicon projectors, each driven by a 450 Mhz Pentium II PC with an off-the-shelf graphics accelerator. The resolution of the resulting image is 4,096×1,536. The display system is integrated with several components, including: a *sound server*, a PC that uses two 8-channel sound cards to drive 16 speakers placed around the area in front of the wall; an *input cluster*, which uses two 300 Mhz Pentium II PCs to capture video images from an array of video cameras, to gather input from a gyroscope mouse, and to receive audio input from a microphone; a *storage server*, which uses two PCs each with 5 inexpensive EIDE disks; a *local compute cluster* of 4 PCs, which provides high-bandwidth access to compute cycles; a *remote compute cluster* containing 32 PCs; and, a *console PC*, which controls execution of the system.

All the PCs are connected by a 100 Base-T Ethernet network. In addition, the PCs of the display cluster, local compute cluster, and storage server are connected by a Myrinet system area network. We are using the *Virtual Memory-Mapped Communication* (VMMC) mechanism developed in the *Scalable High-performance Really Inexpensive MultiProcessor* (SHRIMP) project.[3] VMMC implements a protected, reliable, user-level communication protocol. Its end-to-end end-to-end latency at the user level is about 13 microseconds and its peak user-level bandwidth is about 100 Mbytes/sec on the Myrinet.[74]

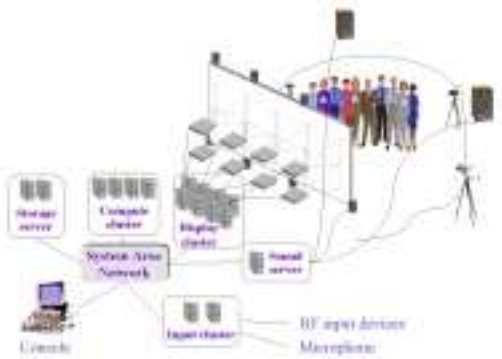

Figure 1: A Scalable, Inexpensive Display Wall System

The foci of our research are usability and scalability. In order to address usability: we must investigate new user interfaces, new content design methodologies, and learn from human perception studies in teaching design courses. In order to achieve scalability, we must carefully address three key system design issues:

- *Coordination among multiple components:* Commodity components are usually designed for individual use rather than as building blocks for a larger, seamless system. To achieve seamless imaging and sound, one must develop methods to coordinate multiple components effectively.
- *Communication performance and requirements:* Immersive and collaborative applications require that multiple components communicate effectively. A scalable system should provide a low-latency, high-bandwidth mechanism to deliver high-performance communication among multiple commodity components. At the same time, software systems and applications must be carefully designed to achieve high quality and performance while minimizing communication requirements.
- *Resource allocation:* Effective resource allocation and partitioning of work among components is critical at both the system and application levels.

In the following sections, we report our early experiences in building and using a display wall system, and we describe our approach to research challenges in several specific research areas, including seamless tiling, parallel rendering, parallel data visualization, parallel MPEG decod-

ing, layered multi-resolution video input, multi-channel immersive sound, user interfaces, application tools, and content-creation.

Seamless Tiling

Image Blending: Although a lot of progress has been made recently in the development of new display technologies such as *Organic Light Emitting Diodes* (OLEDs), the current economical approach to making a large-format, high-resolution display is to use an array of projectors. In this case, an important issue is the coordination of multiple commodity projectors to achieve seamless edge blending and precise alignment.

Seamless edge blending can remove the visible discontinuities between adjacent projectors. Edge blending techniques overlap the edges of projected, tiled images and blend the overlapped pixels to smooth the luminance and chromaticity transition from one image to another. The current state-of-the-art technique is to use specially designed hardware to modulate the video signals that correspond to the overlapped region.[11,18] This electrical edge-blending approach works only with CRT projectors but does not work well with commodity LCD or DLP projectors. This is because these new projectors leak light when projected pixels are black, making them appear dark gray. Overlapped dark gray regions are then lighter gray – brighter than non-overlapped regions. In order to avoid seams we reduce the light projected in the overlapped regions.

Our approach is based on the technique of *aperture modulation*, that is, placing an opaque object in front of a lens (between the projector lens and the screen) to reduce the luminance of the image without distorting the image itself. Thus, by carefully placing an opaque rectangular frame, we can make its shadow penumbra coincide with the inter-projector overlap regions.[8]

Computational Alignment: To make a multi-projector display appear seamless, projected images must have precise alignment with each other in all directions. Aligning projectors manually is a time-consuming task. The traditional alignment method is to use a sophisticated adjustable platform to fine-tune projector position and orientation. This approach requires expensive mechanical devices and extensive human time. In addition, it does not work for commodity projectors whose lenses tend to produce image distortions.

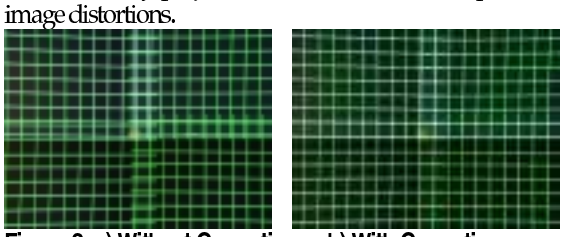

Figure 2: a) Without Correction b) With Correction

To overcome both misalignment and image distortion problems, we use image-processing techniques to "correct" the source image before it is displayed by misaligned projectors. In other words, we pre-warp the image in such a way that the projected images are aligned. We call this approach *computational alignment*. It requires only the coarsest physical alignment of the projectors. Our alignment algorithm currently calculates for each projector a 3×3 projection matrix, with which an image warping process resamples the images to counter the effects of physical misalignment. Figure 2a shows a picture without correction. Figure 2b shows the picture after each projector re-samples the image according to its correct perspective matrix. As a work in progress, we adapt our alignment algorithm to correct some distortions caused by imperfect lenses, e.g. radial distortions.

We obtain precise alignment (or misalignment) information with an off-the-shelf camera that has much lower resolution than our display wall. We zoom the camera to focus on a relatively small region of the display wall, and pan the camera across the wall to get a broader coverage. The camera measures point correspondences and line matches between neighboring projectors. We then use simulated annealing to minimize alignment error globally, and solve for the projection matrices. Our approach differs from the solutions of Rasker, *et al.*,[14] which uses carefully calibrated, fixed-zoomed camera(s) to obtain projector distortion measurements. The cameras in their approach have to see the entire screen or a significant portion of it; and, therefore, cannot easily obtain sub-pixel alignment information.

Parallel rendering

We are investigating parallel rendering algorithms[6] for real-time display of very large, high-resolution images partitioned over multiple projectors. Here we face all three general types of research challenges: coordination of PCs and graphics accelerators to create consistent, real-time images, communication among multiple PCs and their graphics accelerators, and resource allocation to achieve good utilization.

The focus of our efforts is on developing "sort-first" and "sort-last" parallel rendering methods that minimize communication requirements and balance the rendering load across a cluster of PCs.[12] Our general approach is to partition each frame into a number of "virtual tiles." Each rendering machine is then assigned a set of virtual tiles so that the load is as evenly balanced as possible. Since the virtual tiles usually do not correspond to the physical tiles on the wall, rendered pixels must often be read back from the rendering PC's frame buffer and transferred over the network to the projecting PC's frame buffer. We use the VMMC mechanism to achieve low latency and high bandwidth communication for the pixel redistribution phase, as well as to provide fast synchronization of the frame buffer swapping.

The research issues are to develop algorithms that compute the shapes and arrangement of virtual tiles dynamically, sort graphics primitives among virtual tiles in real-time, deliver graphics primitives to multiple PCs in parallel, and redistribute pixels across a network efficiently. To explore this space we have designed and implemented several "sort-first" virtual tiling algorithms. The best of these algorithms uses a KD-tree partition of the screen space followed by an optimization step to ensure the best possible balance of the load.[15] Figure 3 and Figure 4 show the cases with a static screen-space partition without load balancing and a KD-tree partition after load balancing, respectively. The colors indicate which machines render the different parts of the scene. The imbalance in the first case can be observed by looking at the "load bars" on the bottom right of the figure. The load is much better balanced in the KD-

tree case, and as a result the final frame-time is up to four times lower with eight PCs.

Figure 3: Parallel Rendering without Loading Balancing

Figure 4: Parallel Rendering with Load Balancing

Parallel Data Visualization

Increases in computing power have enabled researchers in areas ranging from astrophysics to zoology to amass vast data sets resulting from both observation and simulation. Since the data itself is quite rich, the display wall presents an ideal medium for scientific visualization at high resolution. The magnitude of the data sets motivates the use of parallel computation, a fast network, and separation of computation and rendering across different machines.

The initial focus of our research is to develop parallel algorithms that permit the users to interactively view isosurfaces in volumetric data on the display wall. Our system uses the PCs in the display cluster to perform rendering, the PCs in compute cluster to perform isosurface extraction, and storage servers to hold datasets. We coordinate these three sets of PCs in a pipelined fashion on a per frame basis. Data are sent from the storage servers to the isosurface extraction PC cluster. Triangles for an isosurface are generated in parallel using a *marching cubes* algorithm[10] accelerated with an interval method[5] based on Chazelle's filtering search. They are then sent to the appropriate rendering PCs.

We have experimented with lossless compression methods to reduce communication requirements. Even with compression, we find that low-latency, high-bandwidth communication between the isosurface extraction PCs and rendering PCs is critical.

Figure 5: Parallel Visualization of "Visible Woman"

Figure 5 shows the result of using our parallel visualization system to visualize part of the Visible Woman data set.[13] We are currently focusing on better isosurface extraction algorithms, large-scale storage server development, and load-balancing methods to improve the utilization of computing resources.

Parallel MPEG-2 Decoding

MPEG-2 is the current standard format for delivering high-quality video streams for entertainment, collaboration and digital art. Our goal is to develop fast pure-software MPEG-2 decoding methods on a PC cluster to bring HDTV or even higher resolution MPEG-2 videos to a scalable display wall. To achieve the 60 fps real time frame rate including the overhead in scaling and loading pixels into the frame buffer, a decoder should be capable of decompressing one frame in less than about 14 ms. We approach the problem in two steps: developing a fast decoder on a single PC and designing a fast parallel, scalable decoder for a PC cluster. The key research challenges here are coordination among PCs to split an MPEG-2 stream and fast communication among PCs to decode high-resolution streams in real time.

To improve the MPEG-2 decoding performance on a single PC, we exploited both instruction level parallelism and memory/cache behavior. We develop our decoder based on the open source *MPEG Software Simulation Group* reference design, which decodes 720p HDTV (1280×720) at about 13 fps on a 733 MHz Pentium III PC. We extensively use Intel MMX/SSE instructions to accelerate arithmetic operations and carefully design the data structures and their layouts to improve the data cache utilization. Our preliminary result is a decoder capable of decompressing 720p HDTV stream at over 56 fps on a 733 MHz Pentium III PC. The speed-up is over a factor of four.

To further improve the performance, we use parallel decoding on a PC cluster. Previous work on parallel MPEG decoding is done almost exclusively on shared memory multiprocessors.[2] They parallelize MPEG-2 video decoder at either the *picture* or *slice* level. However, the amount of data movement among the PCs is too high if these methods are used for a PC cluster. We develop a novel *macroblock* level parallelization. We use a single PC to split an MPEG-2 stream into multiple sub-streams at macroblock level and send them to the PCs in the display cluster to be decoded, scaled and displayed.

With the previous picture-level or slice-level parallelization, the per-link bandwidth requirement of the decoding PC depends on the whole video size. With our macroblock-level parallelization, it depends only on the size of the portion that the local node is decoding. This makes our approach highly scalable. Our preliminary result shows that with 4 PCs (in a 2×2 setup) decoding 720p HDTV streams in parallel, the aggregate communication bandwidth requirement among all nodes is only about 100 Mbits/sec. As a comparison, this number can be as high as 1.7 Gbits/sec when a picture or slice level parallelization is used.

Multi-layered Video Input

Video resolution has always been limited by the TV standards. In order to take advantage of the high resolution of a scalable display wall, we are working on methods to create video streams at a scalable resolution that matches the display resolution, using a small number of commodity video cameras. The main research challenge is the coordination among video cameras.

The traditional approach is to use juxtaposed cameras with edge overlapping and stitch multiple images together.[17] It has several disadvantages. First, juxtaposed cam-

eras make zooming awkward – the cameras must be synchronized and the angles between them must be adjusted mechanically at the same time. Second, since each camera has its own focal point, scenes with a lot of depths can look unnatural with multiple focal points. Third, it requires many video cameras. For example, it requires 28 640×480 video cameras for a 4,096×1,536 resolution display wall. The aggregate communication requirement of the video streams is also too high for the network. We would like to overcome all of these problems.

Figure 6: Multi-Layered Video Registration Program

Our approach is called layered multi-resolution video. We use a number of cameras to cover different fields of view. Each camera can be panned, tilted, and zoomed individually. We are developing a fast registration algorithm to find the correspondence of the different layers and merge them into one. This method not only solves the three problems above, but also fits nicely into MPEG-4 video compression framework. Our current registration algorithm runs at 30 registration-passes per second for 2 images. Figure 6 demonstrates the registration process. Our goal is to develop a registration algorithm that runs at real time.

Multi-channel Immersive Sound

Sound guides the eyes, enhances the sense of reality, and provides extra channels of communication. Since the visual display is spread over a large surface, large amounts of the displayed data might be out of the visual field of any user. Sound can be used to draw directional auditory attention to an event, causing users of a large display system to turn their heads toward the sound and thus bringing important visual information into their field of view. To investigate the integration of immersive sound with a large-scale display wall, we use a large number of speakers positioned around the space in front of the display wall to provide immersive sound synthesis and processing in real time. The key challenge is the coordination of multiple sound devices to create immersive sound.

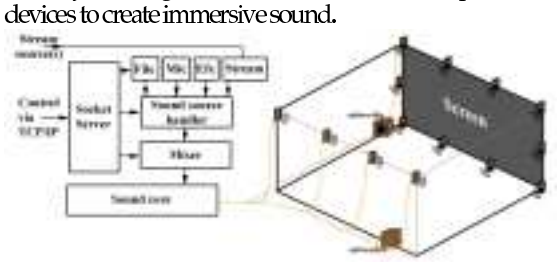

Figure 7: Multi-Channel Sound System

The display wall sound system is implemented on commodity PCs, using inexpensive multi-channel sound cards. These cards are designed for digital home-recording use, and can be synchronized through SPDIF/AESEBU cables and special calls to the software drivers. We have written a sound server that takes commands from any computer via a TCP/IP connection. The server can playback sound files through any combination of the 16 speakers in the present configuration (See Figure 7). Other possible sound sources include onboard synthesis of sound effects, microphone signals, sound streams from any machine on the network or web, and effects (reverb, echo, etc.) processing of any sound source.

User Interfaces

A large collaborative space presents interesting challenges for user interfaces, both display and control. Because of the scale of the display wall, it is important to track human positions, recognize gestures, and construct imagery and sound appropriate for the user's position. Many methods developed in the past require users to carry cumbersome tracking or sensing devices. Our focus has been on developing natural methods for users to interact with the system. We use multiple cameras in the viewing field to track human and input device. We also develop image processing algorithms to understand gestures in a collaborative environment. The main research challenge is the coordination of among commodity input devices and with the computers in the display wall PC cluster.

Figure 8: a) Magic Wand Input b) Voice Recognition

We write a multi-input mouse server program that runs on a master cursor control computer. Any other computer can take control of the display wall mouse by running a mouse client program and connecting to the server. This has allowed us to quickly construct and test a number of new pointing devices, including a swivel chair (the Quake Chair), voice input mouse control, and pressure sensitive floor panels. Figure 8a and Figure 8b shows the use of a camera-tracked wand as a pointer device and a wireless microphone as a speech recognition device, respectively. Research challenges include allowing multiple cursors at once, as well as further refinement and integration of camera tracking.

Methods to Design Application Tools

It is important and non-trivial to bring many applications to a scalable display wall and run them at the *intrinsic* resolution supported by the display surface. Most video-wall products use special-purpose hardware to scale relatively lower-resolution content, such as NTSC, VGA, SVGA, HDTV formats to fit large display surfaces. Only a few expensive solutions use high-end graphics machines to render directly in the intrinsic resolution of a multi-projector display system. Coordination and communication are the two main challenges in developing tools to port off-the-shelf applications to a scalable display wall using its native display resolution.

We have investigated four methods to design tools for applications: custom-designed, distributed application, distributed 2D primitive, and distributed 3D primitive. The

following subsections illustrate each method by an example.

Figure 9: Looking at Image with Still Image Viewer

Custom-Designed Method: Our first tool on the display wall is a *Still Image Viewer*, which allows a naive user to display still images and perform cross fading between images on the wall. The image viewer contains two parts: a controller program and an image viewer program. An image viewer program runs on every PC in the display cluster. The controller program runs on a different PC and sends commands over the network, such as loading an image from the disk, displaying a cached image, or cross fading between two cached images. The image viewer loads JPEG images from a shared file system, decodes only its portion of the image, and synchronizes with other viewers on other PCs prior to swapping the frame buffer. The controller program also implements a scripting interface so that the users can write scripts to control image and video playback that are synchronized with our multi-channel sound playback. Many students have made multimedia presentations on our display wall using the image viewer and the multi-channel sound system. Figure 9 shows an image of the International Space Station on the display wall.

Figure 10: A Distributed Building Walkthrough Program

Distributed Application Method: We distribute application-level input commands to bring a *Building Walkthrough* system designed for a uniprocessor system to the display wall. We run an instance of the building walkthrough program on every PC in the display cluster. In order to coordinate among these programs, we run another instance on the console PC. A user drives the walkthrough using the console PC. The console translates the user inputs and sends the camera information and screen space information to each PC that drives a tile of the display wall. PCs in the display cluster execute copies of a uniprocessor walkthrough application, each of which renders a different part of the screen from its own copy of the scene database. They synchronize frame updates with network messages under the control of console. This method provides interactive frame rates (e.g. 20 fps) without noticeable synchronization problems. Figure 10 shows the walkthrough program being run with a 3D model created by Lucent Technologies.

Figure 11: Windows 2000 Virtual Display Driver

Distributed 2D Primitive Method: We have developed a *Virtual Display Driver* (VDD) to bring existing Windows applications to the display wall, using a distributed 2D primitive method. VDD is a Windows display driver that "fakes" a high-resolution graphics adapter to the Windows 2000 operating system. It leverages the feature in Windows 2000 that supports multiple monitors on a single PC. VDD intercepts all *Device Driver Interface* (DDI) calls and execute them remotely as remote procedure calls on the PCs in the display cluster. The users can drag application windows from the regular CRT display into our virtual display, the contents of which are subsequently drawn on the display wall. All drawing done by the application on VDD is performed in the intrinsic resolution of the virtual display, which is the same as the display wall. Therefore, users can see a lot more details in any Windows applications than with existing commercial video-walls. Figure 11 shows Microsoft PowerPoint and Internet Explorer running on our Display Wall through VDD. At the close range where people are standing in front of the display wall, both applications show adequate details and no fuzziness with line drawings and text.

Figure 12: GlQuake Running on the Display Wall

Distributed 3D Primitive Method: We developed a user-level, *Distributed OpenGL* tool using a 3D primitive distribution method. Unlike the distributed 2D primitive method where our tool works at the device driver level, the distributed OpenGL library lives at the user level. We take advantage of the fact that on all Windows platforms, an application's calls to the OpenGL API are made through a *Dynamically Linked Library* (DLL), opengl32.dll. Our approach is to implement our own opengl32.dll to intercept all the OpenGL calls, and forward them to the PCs in the display cluster. These PCs receive the RPC calls and directly execute them, with the exception that the view frustums are properly modified so that each projector renders only its own tile portion of the screen space. This distributed OpenGL mechanism allows many off-the-shelf Windows OpenGL applications to run on the display wall without any modifications. We have brought up many such applications including games, CAD and visualization tools.

Figure 12 shows the game GlQuake being run on the display wall using our distributed OpenGL mechanism. Currently, we are investigating methods for integrating our parallel rendering algorithms into this OpenGL library.

Content Creation and Design Implications

We started studying content creation and design methods at the same time as other research topics. We taught two design courses using the display wall. The main point of these courses is to provide opportunity and experience utilizing desktop-size screens to create effective wall-size images. Figure 13 to 15 show students' creations on the display wall.

Figure 13: Multiple Small Windows © IAN BUCK

Compared to the traditional, expensive display walls, the inexpensive aspect of the scalable display wall makes a big difference in content creation. Suddenly, we are presented with a new design space available to all users, in particular non-technical users. This rapid democratization of billboard-size display space is quite provocative. Students in the design class are asked to imagine future applications and implications when many such walls are widely in use, and to investigate the best uses for these large displays.

Figure 14: Sketches on a Digital Canvas © JON HARRIS

One implication of a wall-size image is that it completely fills our visual field, which creates a one-to-one experience with the onscreen imagery. There is no border or frame for scale reference as on small monitors. This single shift creates a whole new design paradigm.[16] Areas of interest and focus must be added into the image composition. A second implication is that a group can interact with information on just a portion of the screen while others focus on a different area. Different viewers can be at different distances from the high-resolution screen and move around in the room space while viewing. Third, objects can be seen life-size or intensely magnified. For example, an image of a dense computer chip reads like a road map. Fourth, there is not necessarily a need to rapidly change the images, as they can be so densely filled with data that it takes a while to absorb it all. Often, a single high-resolution screen can be displayed for 10 to 20 minutes and remain continuously interesting. Fifth, the light from the screen can become the room light for the working group. All of these elements, especially the frameless nature of the image, require new thinking and new ways of approaching design.[1,19]

Figure 15: A Fractal Image © WILMOT KIDD

This new design paradigm motivates future work in composition tools for large-format displays. Self-expression has a new form. TCL scripting adds the dimension of time to wall presentations, providing capabilities for timed displays and dissolves from image to image. By synchronizing music and sounds to changing images, the wall has become a storytelling space for presentations of 5 to 10 minutes, as complex and engrossing as any short film or video. The wall room, with its billboard size images, has been used three times as a performance art and theater space. Virtually everyone who visits the display wall expresses some kind of emotional response about being in the huge visual and aural space.[9]

Summary and Conclusions

The Princeton Scalable Display Wall prototype system has been operational since March 1998. It has been used as an infrastructure to conduct our research as well as to teach two design courses.

The approach of using a multiplicity of commodity parts to construct a scalable display wall system works well, but it requires us to address design tradeoffs to deal with coordination, communication and resource allocation issues. We have successfully addressed these tradeoffs and developed solutions in several research areas as outlined in this paper. In seamless rendering, we have developed a combination of optical edge-blending and software image manipulation for projector alignment. In parallel rendering, we have developed a "sort-first" screen partitioning method that achieves good load balance and parallel speedup. In parallel data visualization, we have developed a parallel isosurface extraction algorithm for a PC cluster architecture. In parallel MPEG-2 decoding, we have developed a fast splitter and a fast decoder that achieve real-time decoding entirely in software with minimal communication overhead. In layered multi-resolution video, we interactively combine multiple video streams with a fast registration algorithm. And in application tools design, we developed four methods to let existing applications use the native resolution of the display system while minimizing communication requirements.

Study of user interface issues and human perceptions is very important in building a collaborative environment with a scalable display wall system. We have developed and experimented with several user interfaces beyond the traditional keyboard and mouse, including a gyroscope mouse, a "magic wand" implemented by multi-camera

tracking, and a speech recognition user interface. Our experience shows that natural, unencumbered user interfaces based on passive sensors are useful in such an environment and that it is very desirable to allow multiple users to control a shared display wall simultaneously.

Finally, in teaching design courses using our display wall system, we have found that the resolution and scale of the display require new ways of approaching design. For instance, vast amounts of information can be presented in a single image, rather than as a sequence of images as would be required in a desktop display. Typographic layouts where the font sizes can range from 2 to 600 points bring new capabilities to the use and meaning of text. Sound, especially spatial sound integrated with imagery, is critical for storytelling. A design aesthetic is emerging for large scale, high-resolution images that are dependent on the center of the images rather than on the frame of the wall. Perhaps, from the high magnifications seen in wall size imagery, we will discover new insights and experiences that had not previously been available. ■

Acknowledgements

The Princeton Display Wall Project is supported in part by Department of Energy under grant ANI-9906704 and grant DE-FC02-99ER25387, by Intel Research Council and Intel Technology 2000 equipment grant, and by National Science Foundation under grant CDA-9624099 and grant EIA-9975011. The research programs of Adam Finkelstein and Thomas Funkhouser are also supported, respectively, by an NSF CAREER award and an Alfred P. Sloan Fellowship. We are also grateful to Arial Foundation, Interval Research, Microsoft Corporation and Viewsonic for their generous equipment and software donations.

We would like to thank John DiLoreto for building special large-format screens, and several Intel colleagues Konrad Lai, Dick Hofsheier, Steve Hunt, Paul Pierce, and Wen-Hann Wang for sharing their ideas, projector-mount design, and contents. We also would like to thank all students who took the design classes and who conducted independent studies using the display wall for their content creation.

References

1. R. Arnheim. *The Power of the Center*. University of California, Berkeley, CA, 1988.
2. A. Bilas, J. Fritts, and J. P. Singh. "Real-Time Parallel MPEG-2 Decoding in Software." In *Proceedings of International Parallel Processing Symposium*, 1997.
3. M. Blumrich, K. Li, R. Alpert, C. Dubnicki, E. Felten, and J. Sandberg. "Virtual Memory Mapped Network Interface for the Shrimp Multicomputer." In *ACM/IEEE Proceedings of the 21st Annual International Symposium on Computer Architecture*, pp 142-153, April 1994.
4. Y. Chen, C. Dubnicki, S. Damianakis, A. Bilas, and K. Li. "UTLB: A Mechanism for Translations on Network Interface." In *Proceedings of ACM Architectural Support for Programming Languages and Operating Systems (ASPLOS-VIII)*, pp 193-204, October 1998.
5. P. Cignoni, P. Marino, C. Montani, E. Puppo, and R. Scopigno. "Speeding Up Isosurface Extraction using Interval Trees." *IEEE Transactions on Visualization and Computer Graphics*, Vol 3(2), pp 158-170, June 1997.
6. T. W. Crockett. "An Introduction to Parallel Rendering." *Parallel Computing*, Vol 23, pp 819-843, 1997.
7. C. Dubnicki, A. Bilas, K. Li and J. Philbin. "Design and Implementation of Virtual Memory-Mapped Communication on Myrinet." In *Proceedings of the IEEE 11th International Parallel Processing Symposium*, April 1997.
8. K. Li and Y. Chen, "Optical Blending for Multi-Projector Display Wall System." In *Proceedings of the 12th Lasers and Electro-Optics Society 1999 Annual Meeting*, November 1999.
9. M. Lombard and T. Ditton. "At the Heart of It All: The Concept of Presence." http://www.ascusc.org/jcmc/vol3/issue2/lombard.html
10. W. Lorensen and H. Cline. "Marching cubes: a high resolution 3D surface construction algorithm." *ACM Computer Graphics (SIGGRAPH '87 Conference Proceedings)*, Vol 21(4), pp 163-170, 1987.
11. T. Mayer. "New Options and Considerations for Creating Enhanced Viewing Experiences." *Computer Graphics*, Vol 31(2), pp 32-34, May 1997.
12. S. Molnar, M. Cox, D. Ellsworth, and H. Fuchs, "A Sorting Classification of Parallel Rendering." *IEEE Computer Graphics and Applications*, Vol 14(4), pp 23-32, July 1994.
13. Visible Human Project at the National Library of Medicine. http://www.nlm.nih.gov/research/visible/
14. R. Raskar, M. S. Brown, R. Yang, W.-C. Chen, G. Welch and H. Towles. "Multi-Projector Displays Using Camera-Based Registration." In *Proceedings of IEEE Visualization 1999*. October 1999.
15. R. Samanta, J. Zheng, T. Funkhouser, K. Li, and J. P. Singh. "Load Balancing for Multi-Projector Rendering Systems." *SIGGRAPH/Eurographics Workshop on Graphics Hardware*, Los Angeles, CA, August 1999.
16. B. Shedd. "Exploding the Frame: Seeking a New Cinematic Language", *SMPTE 135th Conference*, 1994
17. R. Szeliski, and H.-Y. Shum. "Creating Full View Panoramic Image Mosaics and Environment Maps." In *Proceedings of ACM Siggraph 1995*, 1995.
18. http://www.trimension.com/
19. E. R. Tufte. *Visual Explanations*. Graphics Press, Cheshire, CT, 1997