Dynamic Projection Environments for Immersive Visualization

Theodore C. Yapo Yu Sheng Joshua Nasman Andrew Dolce Eric Li Barbara Cutler

Department of Computer Science
Rensselaer Polytechnic Institute
{yapot, shengyu, nasmaj, dolcea, lie2, cutler} @cs.rpi.edu

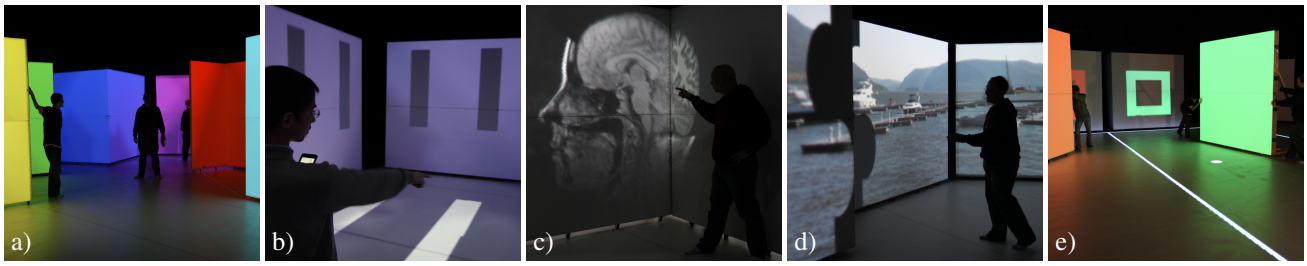


Figure 1. Our immersive and dynamic projection environment enables users to control visualization by manipulating the position and orientation of the projection surfaces. We present a variety of interactive demonstration applications leveraging our new interface.

Abstract

We present a system for dynamic projection on large, human-scale, moving projection screens and demonstrate this system for immersive visualization applications in several fields. We have designed and implemented efficient, low-cost methods for robust tracking of projection surfaces, and a method to provide high frame rate output for computationally-intensive, low frame rate applications. We present a distributed rendering environment which allows many projectors to work together to illuminate the projection surfaces. This physically immersive visualization environment promotes innovation and creativity in design and analysis applications and facilitates exploration of alternative visualization styles and modes. The system provides for multiple participants to interact in a shared environment in a natural manner. Our new human-scale user interface is intuitive and novice users require essentially no instruction to operate the visualization.

1. Introduction

We present a system for physically immersive visualization supporting a number of different applications. In each application, users are able to interact dynamically with the application in real time via a number of movable projection surfaces. These surfaces can be used for displaying application-generated data, as user interface elements, or both in the same application. We use large projection surfaces to provide an immersive, human-body scale visualization and mount these surfaces on wheels so the surfaces can easily be moved by users. Our system tracks these projection surfaces, and provides a mechanism where applications can easily project their output to one or more surfaces (Figure 1).

The resulting environment provides a platform on which we prototyped several applications. In some applications, the projection surfaces represent physical entities, such as in architectural lighting simulations. In other applications, the walls can represent user interface elements, such as being used as a "cut-plane" to visualize volumetric data. Our third prototype application uses the projection surfaces as paddles in a game of virtual ping-pong.

Our system is designed to provide an immersive feel: users walk within the environment and move projection surfaces in real-time, and the application re-calculates the visualization on the fly. Within this framework, we provide facilities for applications that may run at significantly slower than real-time frame rates. We use a technique we call *projection keyframing* to provide continuity on moving surfaces while waiting for simulations to complete. The system allows multiple users to participate interactively with each other and the visualization application. This shared experience allows users to actively collaborate on design or data analysis tasks with enhanced interaction.

1.1. Related Work

Our system leverages previous research in Spatial Augmented Reality (SAR) [4] that mixes physical elements (projection surfaces) and virtual elements (projected entities) to create an enhanced experience. For example, ex-

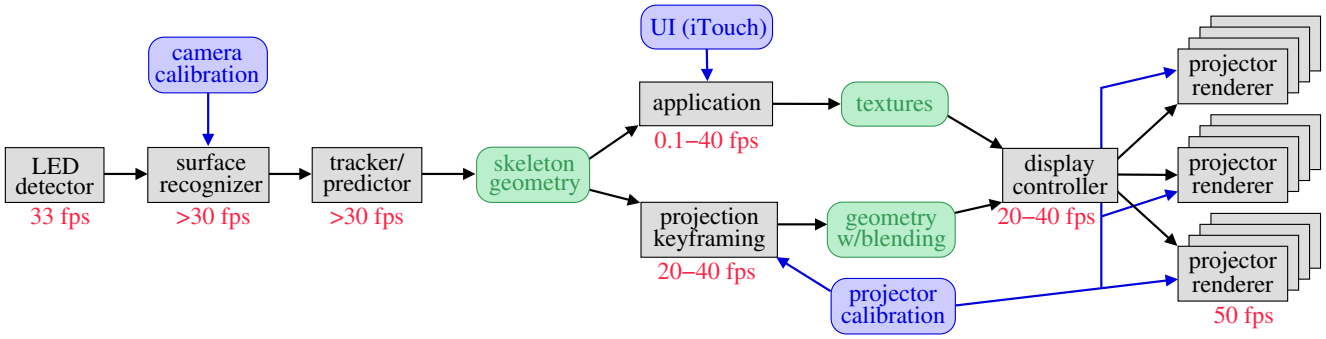


Figure 2. The architecture of our system is composed of several asynchronous computation modules. Once the skeleton geometry has been detected from the camera imagery, the system splits into two paths: the application path, which may not maintain an adequately high frame rate for real-time rendering, and a fast fixed frame rate path to ensure that the most recent application surface textures are projected on the surfaces in their current locations. A distributed rendering process sends the data across a local gigabit Ethernet network to multiple projection servers, each of which handles up to four projectors.

isting wall and desk surfaces can be used as projection surfaces to extend the computer interface [18]. Similarly, Raskar et al. [19] used these techniques to project virtual textures on physical surfaces, for instance to project light and shadows on the exterior of an architectural model. In these systems, the geometry of the physical objects is known and the surfaces are assumed to be a uniform diffuse white material. These techniques were subsequently extended to produce apparent motion on static scenes [20].

When the projection surface geometry is not known *a priori*, it must be captured in addition to determining the projection imagery. Systems for projection onto planar surfaces have used cameras and structured light patterns to reconstruct the geometry of a multi-planar display prior to projecting on the surfaces [1, 12]. Bandyopadhyay et al. [2] project on movable objects using either optical or magnetic-based trackers to capture the motion of the projection surfaces. Lee et al. [13] track movable projection surfaces using a structured light technique with light sensors embedded in the projection surfaces, using an initial pass to obtain surface positions and incrementally updating the position estimates with less perceptible projected patterns. Lee et al. [14] use a Nintendo Wii remote to track infrared LEDs placed on moving and deformable (folding) projection surfaces, simulating, for example, a foldable newspaper. Lee et al. [11] used a custom hybrid infrared and visible light projector to allow unobtrusive tracking of projection surfaces. Invisible infrared structured light patterns are received by IR sensors on projection surfaces, while visible light projects appropriate images. A different approach to imperceptible simultaneous scene capture and display was explored by Cotting et al. [5]. A short-exposure camera captured patterns produced by the pulse-width modulation of DLP projector micro-mirrors for acquisition, which were imperceptible to the eye.

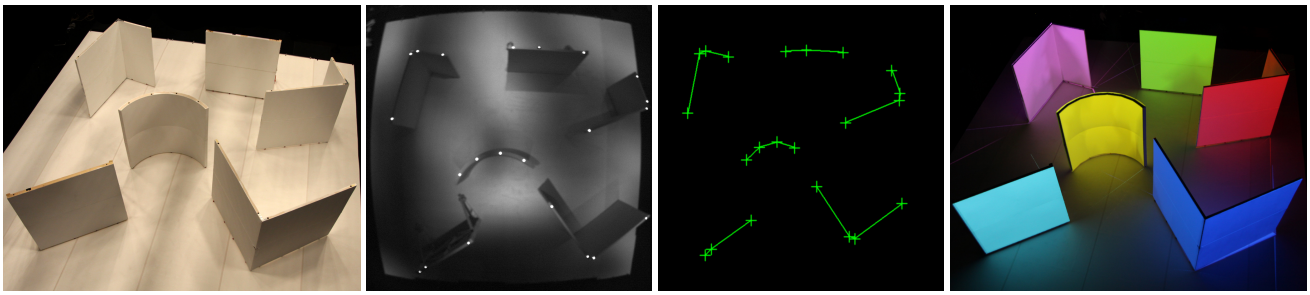
While the systems described above tracked relatively small surfaces that could be moved within a limited volume, CAVE-style environments [6, 7] produce an immersive experience with large projection surfaces in a space that the user can physically enter. Low et al. demonstrate the advantages of a life-sized projection surfaces with customized geometry [15]; however, the projection surface geometry is fixed and cannot be interactively moved by the users.

To produce accurate illumination on surfaces at various locations and orientations, the angle and distance between the projector and surface must be taken into account when adjusting the focal length and relative pixel intensity [17]. Color correction is necessary to accommodate both slight color mismatches between different projectors and projection surfaces with varying material properties [16, 8]. Finally, when projecting onto multi-planar scenes or non-planar objects, there will be secondary scattering of the projected light. If the scene geometry and surface reflectance properties can be estimated, inverse global illumination is necessary to compensate for these inter-reflections [3, 22].

1.2. System Requirements

Our initial target application for the system was interactive architectural lighting visualization. In this application, architects and/or clients gather within the simulated environment to evaluate the functionality and aesthetics of natural and artificial lighting for a proposed architectural design. Hence, interactivity and ease of use are of paramount importance. To facilitate collaborative design, we require a system which:

- Allows people to freely move and interact within the design space without disrupting the system,
- Supports projection surfaces that can be moved in real-time,
- Tracks surfaces despite varying illumination from projectors and ambient sources and the presence of users within the scene, and
- Achieves real-time tracking and display rates.



a) full room lighting b) infrared camera image c) detected surfaces d) projection-applied "paint"

Figure 3. a) A variety of diffuse white projection surfaces are positioned within the active visualization volume. b) An overhead camera captures the position of infrared LEDs marking the top edges of each surface. c) The surface configurations are matched to known distance and angle constraints in a surface library. d) The surfaces can be "painted" different colors by the projectors.

1.3. Our Contributions

We have extended previous research to include:

- Moving (dynamic) projection surfaces, and
- A human-scale interactive immersive environment.

The distributed system we have created allows for:

- *Projector keyframing* - a technique to impart slow applications with a dynamic, responsive feel
- Tracking projection surfaces of known geometry with simple IR-based LED markers, and
- A distributed rendering system which can be extended to drive an arbitrary number of projectors.

2. System Implementation Details

Our system comprises several parts, which we divided into processes distributed over a number of quad-core computers. Since the various components of the system may run at different rates, we couple them asynchronously; each process can obtain the latest output from the previous stage at any time desired (Figure 2). In the sections below we will describe all major components of the system, including: the physical environment, detection and tracking of projection surfaces, our rendering algorithms, and a method for determining the projector placement.

2.1. Physical Environment and Projection Surfaces

We constructed a set of six projection surface modules in a variety of shapes. The projection surface of each module is 2.44m tall and is mounted on 6.35cm tall casters. The surfaces of the wall modules are covered on one or both sides with either thin plywood or stretched canvas and painted with bright white diffuse paint. The venue for our project is roughly 19m x 14.5m x 9m tall. The floor of the room is covered with a 12m x 12m white vinyl dance floor, which ideally would also have a diffuse surface; however, the material available does show a non-zero specular reflection component. Most of the other surfaces in the room are black.

Overall, the collection of modules (Figure 3a) are sufficiently complex and expressive for use in a wide variety of applications (presented in §3) and demonstrate our ability to detect and project on non-convex and non-planar surfaces.

2.2. LED Detection

Our system uses a single camera to obtain images of the scene and determine the projection surface geometry (Figure 3b). We use a gigabit-Ethernet connected camera of 1280x960 monochrome pixels. Since the height of our working environment is only about half the width of the desired active area, a wide-angle fisheye lens (1.4-3.1mm varifocal CS-mount) is required to capture the entire scene.

Since people are free to move throughout the space, and indeed having them move projection surfaces in real time is a fundamental aspect of our system, we wish to ignore their presence when tracking surfaces. To achieve this, we attach three or four near-infrared (850nm) LEDs to each projection surface. The spacing of the LEDs is designed so that the position and orientation of individual projection surfaces can be robustly and uniquely inferred from the LED configuration. The camera lens is fitted with an optical long-pass filter that absorbs visible light from the projectors and ambient sources while passing the 850nm infrared light from the LEDs. The resulting images are extremely high-contrast and allow the LED positions to be detected with high accuracy. To avoid bias introduced by distortion of the fisheye lens, we first detect the small set of pixels corresponding to each LED's image, then back-project to world coordinates before estimating the LED position with the centroid of the world-space points. Backprojection of pixels to world-space coordinates is facilitated by calibrating the floor to lie in the xz coordinate plane. Since the height of the projection surfaces above the floor is fixed and known, we can find the intersection of the line of sight from the camera with a plane at the height of the projection surface tops, yielding an estimate of the 3D position of each LED.

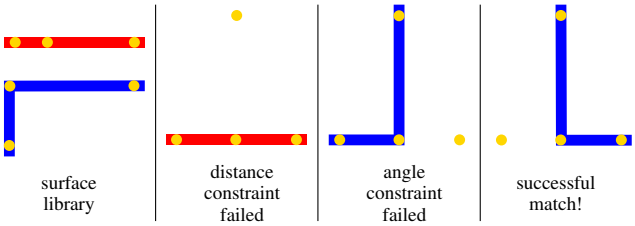


Figure 4. The surface recognizer searches for each wall in the set of detected LEDs by considering all n choose k possible configurations and eliminating configurations which violate the distance or angle constraints.

2.3. Surface Recognition & Tracking

The 3D LED marker coordinates for the current camera frame are passed to the *surface recognizer*, which analyzes potential scene configurations based on geometric constraints in the *known surface library*. Each surface in the library has a specified number of marker locations, corresponding to LEDs on the physical surfaces, as well as a unique set of distance and angle constraints. Distance constraints specify the distance between a pair of markers, while angle constraints describe the clockwise or counter-clockwise orientation of a set of three markers.

First, the algorithm attempts to locate each surface in the library within the scene by enumerating all possible permutations of the detected markers (Figure 4). Each marker, p_i, is associated with N_i measured distance constraints and a maximum error value, e_i, equal to the spatially varying expected LED position error (our fisheye lens yields better accuracy in the center of the space). For each distance constraint, d_0, we calculate d, the distance between the corresponding markers for the permutation under consideration. If $|d - d_0| > e_i + e_j$, the configuration is deemed invalid and the corresponding branch of the permutation tree is pruned. If all distance constraints are satisfied (within the error threshold), then a normalized error value between markers p_i and p_j is computed as follows:

$$E_{ij} = \left(\frac{e_i}{N_i} + \frac{e_j}{N_j}\right)\left(\frac{|d - d_0|}{e_i + e_j}\right) \quad (1)$$

This normalization ensures that the total error for an acceptable configuration will not exceed the sum of the maximum expected errors of its markers, a property that is important during the next step of recognition. Each permutation is also checked for agreement with the known angle constraints, and each valid surface configuration is assigned an error equal to the sum of the normalized error values of all its distance constraints.

Next, the recognizer combines individual surface configurations into the optimal scene configuration (Figure 3c) using a best-first tree search. At each step, the partial scene configuration with minimum total error is expanded by creating new partial configurations that add one more surfaces into the scene in all locations identified in the previous step. Configurations that re-use LEDs are eliminated. The total error of a given configuration is equal to the sum of the configuration error for each placed surface plus a penalty for each unused LED (equal to the maximum calibration error at that spatial location). Equation (1) ensures that the total cost for including a surface in the configuration is less than the penalty for the un-used LEDs.

Figure 5. Our jigsaw puzzle application demonstrates the system's capability to "paint" a texture on a surface wherever it is placed in the visualization volume and the ability to swap in a new textures in response to the relative wall position — the piece edges *snap* together when matching edges are placed in close proximity.

The *surface tracker and predictor* module compensates for system latency by predicting future surface locations. This module tracks markers by maintaining a Kalman filter for each marker output by the *surface recognizer* and uses the detected locations to update the corresponding Kalman filters and generate the posterior state estimations, including locations and velocities. The predicted surface location is the sum of posterior location estimation from the Kalman filter and a displacement computed by multiplying the Kalman-corrected velocity and a timestep tuned to match the latency of the current application.

Our current surface library with six moving projection surfaces contains a total of 22 markers (Figure 5). During typical use of the system, we occasionally receive one or two falsely detected markers as a result of noise from extraneous infrared sources. These erroneous markers could produce an incorrect scene configuration with lower total error than the correct solution. The frequency of incorrect solutions was found to be negligible in our normal test environment, but could become problematic in scenarios with significant noise. For this dataset, the *LED detector*, *surface recognizer*, and *surface tracker and predictor* modules each run significantly faster than the maximum transfer speed of our current camera (33 fps).

2.4. Dynamic Projection Surfaces

Our projection system differentiates between two classes

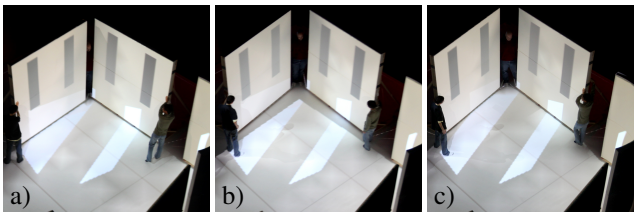


Figure 6. If the visualization application does not have a real-time frame rate, our system a) captures the surface data as textures, and b) maps these textures to the walls as they move. When the application finishes recalculation c) the new textures are used.

of applications: real-time and interactive. For real-time applications, where the computed frame rate is at least as fast as the camera's frame rate, the application directly renders to the projector displays. For interactive frame rate applications, where the computation time per frame is too long to provide smooth display updates while the surfaces are in motion, we introduce a two-pass approach called *projection keyframing*. Raskar et al. [18] use a two-pass scheme to project images corrected for a tracked viewer's perspective; our method uses a similar scheme to solve a different system problem: mismatched application updates and display rates. In projection keyframing, the application does not generate projection frames directly; instead, it generates textures for each projection surface. The *projector renderer* then renders the scene using these infrequently updated textures (Figure 6).

Our system is modular and can be used with a variety of projector configurations. We tested the system with 4, 6, and 10 projectors. We implemented a distributed rendering system for both real-time and interactive applications across multiple desktop machines. In our current 10 projector setup, we use three rendering machines that either directly run real-time applications or alternatively run the projection keyframing rendering program for lower frame rate applications. Two additional machines process camera frames, update the positions of the projection surfaces, calculate blending weights to seamlessly transition between different projectors based on surface normals and occlusions, and run the computationally-intensive, less than real-time frame rate applications, such as architectural lighting visualization. For each camera frame, the updated positions of projection surfaces are used to build a *skeleton geometry* which is sent to each of the rendering machines. The slower frame rate applications prepare new surface textures (projector keyframes) that are sent via network to the rendering machines. The rendering machines then generate images from the projectors' perspective views with the textures and skeleton geometry. To ensure that the rendering machines display projection images simultaneously, we use Message Passing Interface (MPI) barrier calls between each frame.

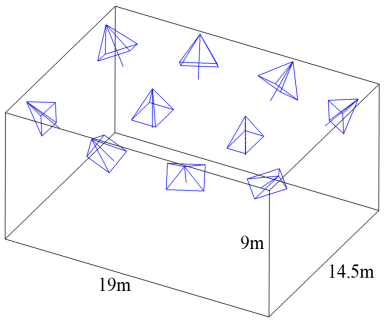


Figure 7. A snapshot from our visualization application showing the 10-projector configuration we selected for our current installation. Two central projectors primarily illuminate the floor; the remaining eight illuminate vertical surfaces.

2.5. Projector Placement

In our system, where projection surfaces can be freely moved through an active volume, the placement and orientation of projectors can greatly affect the quality of the experience produced by the resulting system. Poorly positioned projectors can result in areas which are not covered, appear dimly lit, or suffer from low resolution. Additionally, when multiple projection surfaces are used, occlusions may prevent parts of projection surfaces from being lit. An effective placement of projectors will minimize these effects. Through a simple interactive visualization application, we investigated various placement options considering the installation constraints of the physical space. To evaluate the relative merit of various layouts, we calculated coverage for a collection of several hundred models representing architectural designs created by fourth- and fifth-year architecture students.

We used 10 projectors to create the results for this paper, 5 of these projectors have resolution 1400 x 1050 and 5 have resolution 1920 x 1080 (Figure 7). This arrangement has proven to be quite satisfactory in practice; in our system testing, objectionable occlusion artifacts were not observed. The active view volume was designed to provide full surface coverage to 2.5m tall projection surfaces throughout an active volume with diameter of 12m. Our surface texel resolution is approximately one pixel per cm.

2.6. Calibration

System calibration begins with calibrating the intrinsic parameters of the camera and fisheye lens using the model presented by Scaramuzza et al. [21]. Their model accurately calibrates the camera, producing an error of ± 0.36 pixels ($\approx$ 0.42–1.3 cm in the center and edge of the space, respectively). To determine the extrinsic camera parameters, we use a calibration grid of points on the floor and position an infrared LED marker on each grid point. To calibrate the projectors, a photodiode-based sensor is placed on each calibration point, and Gray-coded structured light patterns are

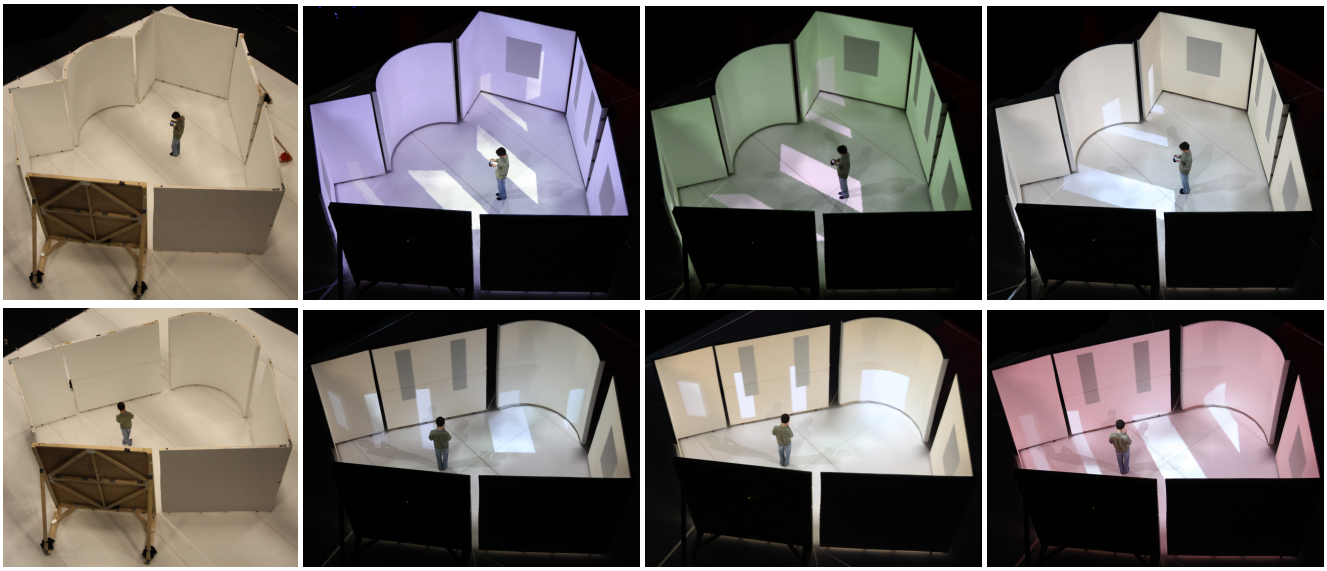


Figure 8. The dynamic projection surfaces can be arranged to sketch out an architectural interior. The projectors display the results of a global illumination daylighting simulation on the surfaces. The user controls other variables of the design including the virtual wall color, placement of windows, and time of day through a handheld wireless interface.

emitted from each projector in turn, allowing the location of the sensor in projector image coordinates to be determined. We collect data for each grid point on both the floor and 2.5m above the floor. Using the resulting 2D-3D correspondences, we apply Tsai's algorithm [24], to yield the intrinsic and extrinsic parameters of each projector.

The accuracy of the calibration is within a pixel throughout most of our viewing volume. When a projection surface is not actively moving, the edge of the projected "paint" usually reaches correctly to the edge of the target surface and does not spill onto the floor. When the projection surface is in motion, in particular when it is abruptly accelerating or decelerating, we do observe some texture sliding, as expected. The physical assembly for some of the projection surfaces is less rigid and these modules do not always maintain their assumed true vertical orientation during motion, which yields geometric estimation errors and projection inaccuracy. This is most apparent on the "L"-shaped walls and could be addressed by improving the structural rigidity of the walls or "shrinking" the target projections when these surfaces are in motion. Alternatively, these shapes could be more robustly scanned and/or tracked with additional cameras or other methods.

3. Applications

Using the basic paradigm of tracked moving projection screens, we envision three general classes of applications that would make use of our system and present a prototype example for each. In the first category of application, typified by architectural visualization, the projection surfaces represent physical entities, such as walls in a building. In the second type of application, projection surfaces are used to interactively explore volumetric data by defining cross-sections. Finally, we developed applications where the moving projection surfaces are used as general purpose user-interface elements.

3.1. Architectural Visualization

In architectural visualization applications, the moving projection surfaces become walls of a proposed interior design; in this way, users may experience an immersive feel for the space as they may walk through it and experience it at full-scale. Significantly, several users may experience the environment simultaneously; for example, an architect and client may collaborate on the design in real-time. This interaction mode is in contrast to virtual reality systems employing head-mounted displays or viewpoint-tracking, where typically only a single user experiences and controls the environment.

The users position a set of large mobile partition walls to sketch the full-scale 3D geometry of their design. The locations of windows, material properties of the virtual surfaces, environmental parameters of the simulation, and a time-lapse animation mode are specified through a handheld wireless input device (Figure 1b). The system interprets the sketch, computes a global illumination solution, and displays the solution directly on the walls (Figure 8).

Accurate global illumination and sharp shadows are important visual factors in architectural daylighting analysis. We use radiosity to compute the indirect illumination and shadow volumes to generate the direct illumination from the sun. Form factors between surface patches must be recom-

Figure 9. Visualization of volumetric data.

puted when the geometry is edited; i.e., when the projection surfaces are moved. Visualizations that involve relighting a fixed scene, such as changing the materials or the time of day, are sufficiently fast for interactive analysis and design (3-5 fps), but still benefit from our projection keyframing.

Although our initial application concentrates on lighting simulations, we envision other architectural simulations and visualizations would benefit from online immersive simulations. For example, spatially-varying heating, cooling, or airflow data could be simulated for a given room and then projected using pseudo-color or isoline techniques, possibly annotated with numerical data. Since the system allows for intuitive geometric design interaction, we expect that these applications would enjoy the same benefits as lighting simulations. Due to the computational complexity of many of these simulations, our projector keyframing method (§2.4) will be necessary to achieve smooth real-time updates as the projection surfaces are moved (Figure 6).

3.2. Volumetric Data

By substituting the 2D textures used for architectural visualization applications with 3D textures, a volumetric visualization system is created (Figure 9). First, a volumetric texture, for example computed tomography or magnetic resonance imaging medical data, is fixed to the coordinate axes. Then, a projection surface can be used to slice through the data, displaying the resulting cross-section on its surface. Similar systems using small-scale surfaces and single projectors have been previously described [10, 9, 23]. The utility of the visualization system is enhanced by the ease with which the cut plane(s) can be moved, and the tangible feedback and interactivity of the system provide an intuitive experience for exploring the volumetric data. We found this system easier to use than manipulating the cut-plane on a traditional monitor, while facilitating multi-user interaction.

Since, in this application, the volumetric dataset is anchored "in space", any number of projection surfaces plus the floor can be swept through it to produce novel views of multiple sections simultaneously (Figure 1c). This mode of use facilitates easier visualization of structures in the dataset, and the tangible interface enhances the experience of interaction. Although our initial applications for this mode are explorations of medical data, any volumetric dataset could be experienced in a similar manner. In fact, the dataset need not be static; we are particularly excited about the prospect of volumetric simulations that can also be affected by movement of the projection surfaces.

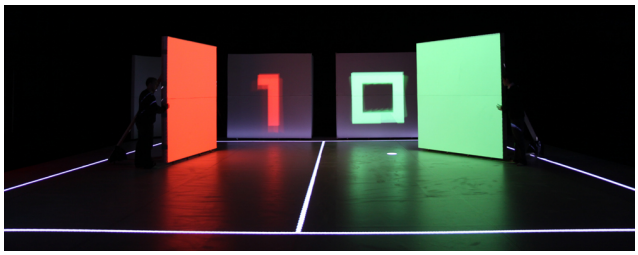


Figure 10. A game of immersive augmented reality ping pong.

3.3. General User Interface Elements

Since the positions of the projection surfaces are tracked in real-time, they can be used as general-purpose user interface elements, replacing mice or joysticks. Information relevant to the application can be projected on the surfaces, and user input taken from the motion of the surfaces.

Using this paradigm, we implemented a ping-pong game in which players manipulate the projection surfaces as physical game paddles (Figure 10). The court lines and a virtual ball are projected on the floor surface, and additional projection surfaces are used to keep track of the current score. The game play is an interesting mix of singles and doubles strategies, with two players working as a team to move a single projection surface (paddle). In our informal presentation of this interface to people outside of our research group, we found little to no instruction was required to use this natural and intuitive user interface.

4. Conclusions and Future Work

We have presented a novel human-scale dynamic projection visualization environment and demonstrated a variety of engaging applications that make use of this system. We found that relatively simple system components can be connected to produce a very usable visualization system, although each component could be enhanced for particular applications. For interactive games, for instance, fast projection surface tracking and high frame rates are of great importance; thus, for these applications, tracking and rendering engines can be optimized. Other applications may require more flexibility in size or shape of projection surfaces; to enable these, only the tracking component need be modified. Similarly, although the common, inexpensive, "gaming-class" video cards used to drive projectors in our current system proved adequate for most visualization tasks, more sophisticated cards which allow genlocking of the video output would provide a more seamless experience.

Acknowledgments

The authors wish to thank Eric Ameres, Mick Bello, and Robert Bovard from Rensselaer's Experimental Media and Performing Arts Center for their technical support of our research. We also thank Bill Bergman for assistance in constructing the projection surfaces. This work was supported by NSF CMMI 0841319, NSF IIS 0845401, and a grant from IBM.

References

[1] M. Ashdown, M. Flagg, R. Sukthankar, and J. M. Rehg. A flexible projector-camera system for multi-planar displays. In *2004 Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, pages 165–172, June 2004. 2

[2] D. Bandyopadhyay, R. Raskar, and H. Fuchs. Dynamic shader lamps: Painting on movable objects. In *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, page 207, Washington, DC, USA, 2001. IEEE Computer Society. 2

[3] O. Bimber, A. Grundhöfer, T. Zeidler, D. Danch, and P. Kapakos. Compensating indirect scattering for immersive and semi-immersive projection displays. In *VR '06: Proceedings of the IEEE conference on Virtual Reality*, pages 151–158, Washington, DC, USA, 2006. IEEE Computer Society. 2

[4] O. Bimber and R. Raskar. Modern approaches to augmented reality, 2007. ACM SIGGRAPH 2007 Course Notes. 1

[5] D. Cotting, M. Naef, M. Gross, and H. Fuchs. Embedding imperceptible patterns into projected images for simultaneous acquisition and display. In *ISMAR '04: Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 100–109, Washington, DC, USA, 2004. IEEE Computer Society. 2

[6] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the cave. In *Proceedings of SIGGRAPH 93*, Computer Graphics Proceedings, Annual Conference Series, pages 135–142, Aug. 1993. 2

[7] M. Gross, S. Würmlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. V. Gool, S. Lang, K. Strehlke, A. V. de Moere, and O. Staadt. blue-c: A spatially immersive display and 3d video portal for telepresence. *ACM Transactions on Graphics*, 22(3):819–827, July 2003. 2

[8] A. Grundhöfer and O. Bimber. Real-time adaptive radiometric compensation. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):97–108, Jan./Feb. 2008. 2

[9] K. Hirota and Y. Saeki. Cross-section projector: Interactive and intuitive presentation of 3d volume data using a handheld screen. In *3D User Interfaces, 2007. 3DUI '07. IEEE Symposium on*, pages –, March 2007. 7

[10] J. Konieczny, C. Shimizu, G. Meyer, and D. Colucci. A handheld flexible display system. In *Visualization, 2005. VIS 05. IEEE*, pages 591–597, Oct. 2005. 7

[11] J. Lee, S. Hudson, and P. Dietz. Hybrid infrared and visible light projection for location tracking. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 57–60, New York, NY, USA, 2007. ACM. 2

[12] J. C. Lee, P. H. Dietz, D. Maynes-Aminzade, and S. E. Hudson. Automatic projector calibration with embedded light sensors. In *ACM Symposium on User Interface Software and Technology (UIST)*, 2004. 2

[13] J. C. Lee, S. E. Hudson, J. W. Summet, and P. H. Dietz. Moveable interactive projected displays using projector based tracking. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 63–72, New York, NY, USA, 2005. ACM. 2

[14] J. C. Lee, S. E. Hudson, and E. Tse. Foldable interactive displays. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 287–290, New York, NY, USA, 2008. ACM. 2

[15] K.-L. Low, G. Welch, A. Lastra, and H. Fuchs. Life-sized projector-based dioramas. In *ACM Symposium on Virtual Reality Software and Technology*, pages 93–101, 2001. 2

[16] A. Majumder and R. Stevens. Color nonuniformity in projection-based displays: Analysis and solutions. *IEEE Transactions on Visualization and Computer Graphics*, 10:177–188, March 2004. 2

[17] C. Pinhanez. The everywhere displays projector: A device to create ubiquitous graphical interfaces. In *Ubiquitous Computing (Ubicomp)*, September 2001. 2

[18] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 179–188, July 1998. 2, 5

[19] R. Raskar, G. Welch, K.-L. Low, and D. Bandyopadhyay. Shader lamps: Animating real objects with image-based illumination. In *Rendering Techniques 2001: 12th Eurographics Workshop on Rendering*, pages 89–102, June 2001. 2

[20] R. Raskar, R. Ziegler, and T. Willwacher. Cartoon dioramas in motion: Apparent motion effects on real objects with image-based illumination. In *International Symposium on Nonphotorealistic Animation and Rendering (NPAR)*, June 2002. 2

[21] D. Scaramuzza, A. Martinelli, and R. Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *ICVS '06: Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*, page 45, Washington, DC, USA, 2006. IEEE Computer Society. 5

[22] Y. Sheng, T. C. Yapo, and B. Cutler. Global illumination compensation for spatially augmented reality. *Computer Graphics Forum: Eurographics Conference*, 29, 2010. 2

[23] M. Spindler, S. Stellmach, and R. Dachselt. PaperLens: Advanced magic lens interaction above the tabletop. In *Proceedings of ACM International Conference on Interactive Tabletops and Surfaces*, Nov. 2009. 7

[24] R. Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 364–374, 1986. 6