Visually Immersive Theater with CaveUT 2.5

CaveUT is a set of tools and methods which modifies the game UT2004 to display a single virtual world across multiple monitors or projections. With CaveUT the educator can arrange these views in any configuration around the user/audience, usually to produce a single panoramic view of the virtual world. CaveUT is free and open source, and UT2004 is cheap and runs on standard Windows PCs. CaveUT has been available since 2001, used in a variety of educational projects. In colleges, CaveUT has been used for teaching the basics of VR technology, perception, and 3D representations. In museums, hospitals, and research settings CaveUT supports interactive VR environments. The new CaveUT (version 2.5) offers greatly improved separation of movement and selection/firing, control of viewpoint via a UDP connection, easier setup, support for multi-projector dome displays, and other new features. It is available to the public at http://publicvr.org/CaveUT.

Introduction

CaveUT 2.5 is a set of modifications for Unreal® Tournament 2004, which supports low-cost, multi-screen composite displays (http://en.wikipedia.org/wiki/Ut2004). Each screen can be a computer monitor or a projection produced by a standard digital projector, and the user can arrange up to 32 screens in any orientation to the viewer. Each screen acts as a "window" into the same 3D virtual world, and if they are conjoined, they form a composite immersive display. For example, this CaveUT display (Figure 1) uses four projection areas to surround the user, an idea similar to the CAVE™ (Cruz-Neira, 1993). A separate computer (a PC running Microsoft® Windows®) drives each projector, and another computer acts as the operator console. All the PCs are connected via a standard LAN.

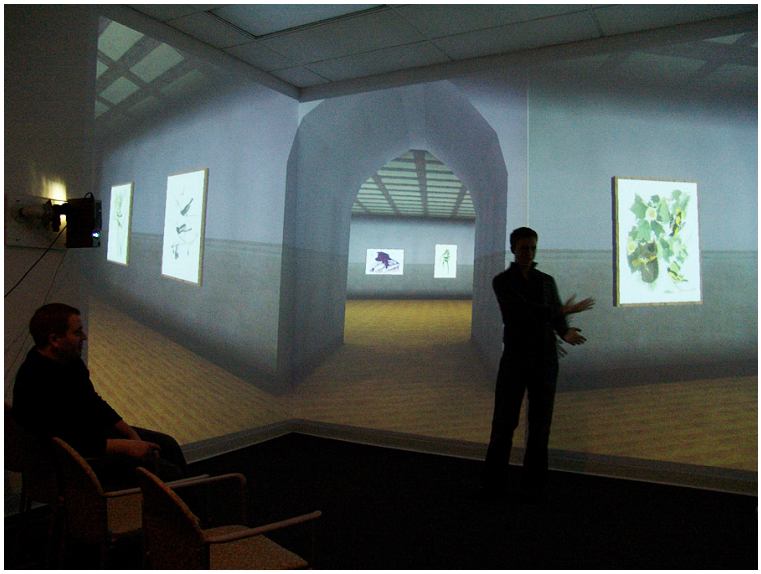

Figure 1: Graduate student, Lyle Seethaller shows his Virtual Audubon Museum in the Virtual Theater at the School of Information Sciences, University of Pittsburgh.

CaveUT supports most virtual worlds and applications built for UT2004, which is based on version 2.5 of Unreal Engine™ by Epic Games. CaveUT is free to the public and is open-source. UT2004 remains a commercially sold game that is inexpensive and readily available through online retailers. It comes with a free authoring tool, UnrealEd, which can take input from most available 3D modeling, texture, sound, and animation packages, such as 3ds Maxx and Adobe® Photoshop®.

CaveUT is an open-source freeware project distributed and maintained by PublicVR, which mediates all contributions (see credits). PublicVR also distributes several helper tools. VRGL, a modified OpenGL® library, introduces off-axis, flat perspective and spherical warping effects essential for composite displays and dome projections. VrpnUT allows an external program to move the primary user viewpoint via a UDP connection. "Logger" records where the user goes and what the user selects. For software and documentation, go to http://publicvr.org.

This is the first paper written about CaveUT in five years, and it announces the release of the latest version (2.5), which provides greatly improved performance, new features, and new or improved companion tools. Here, we will introduce its educational uses, describe its context, list contributor credits, and detail its current features.

Educational (and Other) Uses

The kind of visually immersive virtual reality possible with CaveUT has a long history of educational uses. It allows students to interact with representations of things they could never see or reach in the real world because those things are microscopic (atoms), macroscopic (the solar system), no longer existing (ancient temples), inaccessible (the interior of a living body), or otherwise "impossible" (a world with alternative physics).

For example, Gates of Horus is a learning game based on a virtual Egyptian temple (Jacobson, 2009). The same temple is the subject of public tours at the Carnegie Museum of Natural History in Pittsburgh in its all-digital partial dome display, the Earth Theater. The Earth Theater also has other shows developed for the museum using CaveUT and VRGL, most notably OvirapTour (Figure 2) by Mechanimal (http://www.mechanimal.net/) and the Virtual Seneca Village. All of these shows are thematically tied to one of the museum's physical collections (Jacobson, 2008).

Figure 2: The OvirapTour virtual exhibit, a show in the CMNH's Earth Theater

Most educational uses of CaveUT have been "Constructionist," where students learn by doing. In the CaveRC display at Roanoke College (http://cs.roanoke.edu/~hughes/CaveRC/CaveRC.html), students learned how to represent significant historical structures in visually immersive 3D environments, a process of interpretation, construction, and walkthrough experiences. In 2006, Mary Hart's high-functioning autistic students at LaRoche College, PA, learned computer graphics technology by constructing a two-walled "V-Cave" following the instructions in the CaveUT documentation. In 2007, they made their own virtual environments. In his Master's thesis, Song Hoon Lee used CaveUT to create virtual tours of buildings, with important environmental engineering information about each toured building made visible to the user (Lee, 2008). Charles Hendon modified VRGL for his own Master's thesis to create an interactive tool for the user to warp a projected UT2004 image so that it appears flat, even though it is projected onto an uneven surface (Hendon, 2008).

Other applications are more complex. The Balance NAVE Automatic Virtual Environment (BNAVE) at the Medical Virtual Reality Center of the University of Pittsburgh uses CaveUT to diagnose and treat balance disorders by

moving people through virtual environments constructed by (low-cost) student interns who use the UT2004 authoring tools. (See Figure 3 for a view of the BNAVE.) The Intelligent Virtual Environments (IVE) lab at Teesside University took an older version of CaveUT and created its own complex branch version for advanced interactive narrative systems (http://ive.scm.tees.ac.uk/). There are many other projects, too numerous to list here.

Figure 3: The BNAVE at the Medical Virtual Reality Center

Context

When CaveUT was first released, the hardware needed for most immersive displays was very expensive. There were a few notable exceptions, where dedicated "garage VR" practitioners used low-cost commercial equipment, as CaveUT does (Pape, 2004). However, *all* the immersive display solutions required a high level of skill to produce content for them, and the quality of the graphics was poor compared to game engines of the time. CaveUT allowed users to construct simple virtual worlds and (comparatively) low-cost displays quickly and easily without knowledge of programming.

Today, low-cost Head-Mounted Displays (HMDs) offer reasonable performance (Saenz, 2010) and portable inflatable domes (www.e-planetarium.com). There are also new 3D software platforms which are free (www.blender.org) or low-cost (http://unity3d.com) and which can be adapted to immersive displays. For example, Paul Bourke has released free software which allows the Unity and Blender display in dome and curved-screen displays (http://local.wasp.uwa.edu.au/~pbourke/miscellaneous/domemirror/UnityiDome/). A Google Scholar search on "Low-Cost CAVE" identifies many dozens of projects where researchers and hobbyists constructed low-cost CAVE-like displays using standard PCs and projectors and a variety of self-made software. Doing any of this requires significant skill, however.

Nevertheless, CaveUT is still lower in cost and easier to use than most solutions and will be useful for years to come for educators whose intended projects match its capabilities.

New Features

Improved Performance and Screen Synchronization

Older versions of CaveUT had a significant problem with a lack of synchronization between screen views. This problem resulted from the adaptive networking code built into UT2004 when it was being optimized for game-play among small groups over the Internet. It was never intended for the update rate of 30 frames per second that is required to keep discrepancies between the views below the level of visual perception. Because the Unreal Engine code is proprietary, we were unable to change the networking code. To do so would make it impossible to keep CaveUT an open-source tool.

CaveUT 2.5 forces the UT2004 server to update the state of the world to its clients at 30 frames per second to keep the multiple views synchronized. However, this leads to conflict with UT2004's built-in movement interpolation algorithms. In a normal multiplayer game over the internet, the algorithms make each individual's apparent navigation and interaction with the virtual world appear smooth, even though network updates are often slow and unreliable. These algorithms can be a problem for CaveUT, because, on each machine of the composite display, they move the user's apparent viewpoint in slightly different ways, especially when the user is rotating his or her view. It is possible to turn the algorithms off, but that leads to other complications. Instead, CaveUT 2.5 overrides these algorithms by forcing position updates for every frame. This solution works reasonably well as long as the user is traveling (translating and/or rotating) within a certain range of speeds. Updates begin to look jerky if the user moves too slowly in the virtual environment. They also become jerky if a movement is corrective, but that is much less of a problem. CaveUT 2.5 provides a control for tuning the speed-smoothing algorithm. The application designer must adjust this value based on the desired range of travel speeds and, to a lesser extent, on the performance of the computers in his or her display.

The solution is not optimal, but it does work reasonably well for most applications and preserves the open-source nature of the CaveUT modifications. It also preserves the ability of UT2004 to handle users connecting via the Internet, making it possible for a CaveUT user to interact with players who do not have CaveUT displays.

Object Replication

CaveUT 1.2 and older versions showed player avatars and world geometry properly synchronized across multiple displays, but a single object was displayed in a slightly different location on each screen. Beginning with Versions 2.0, this problem has been solved for most types of objects. The only exceptions are objects that are created client-side, such as particle emitters for special effects. The problem is particularly visible for objects with randomized movement trajectories, rotations or animations. In some cases, sprite-based objects, which are two-dimensional billboards and not actually three-dimensional, may show seams between two screens. For these types of objects, level designers and script programmers may have to create custom variants that implement proper synchronization between multiple view port clients.

Better Support for Multiplayer Environments

All versions of CaveUT have multiple individual screens that show the view as seen by each player in a multiplayer game. This allows the "CaveUT Player" to see other players in the composite display as they exist and interact in the virtual environment. Each one of those players connects through a standard Windows operating system. In older versions, the CaveUT player had to be the player on a "Listen Server" to which all the other players could connect. This is still the default configuration for CaveUT installations because it provides the best performance. However, it is easy to configure each view (each screen) of the composite display to fixate on any player. It is therefore possible to have multiple linked composite displays, to switch the composite display's view to the view as seen by another player, or to use a dedicated server instead of a Listen Server.

Clean Code Base

Over the course of six years, the CaveUT code base had been modified by half a dozen programmers, patching bugs and adding features. In 2009, the code was completely rewritten to reduce its size by more than fifty percent and to make it more stable and readable.

The new code implements the most important core functions of CaveUT in a much simpler module, called the CaveUT Mutator. The mutator handles screen rotation and the networking code that governs view synchronization and object replication. It also contains the parameters that govern which player becomes the CaveUT user, who has access to the multiple views in a multiplayer game. Note that the term *CaveUT* refers to a collection of tools and modifications for UT2004, not a single piece of software, although all of CaveUT is based on the functions in the CaveUT Mutator.

The CaveUT Mutator is intended to be used with other modifications for UT2004 by an author/user who wants to create a VR application. An open-source author could simply expand the code in the CaveUT mutator to include the functions that author wants, but we strongly advise against doing this. The purpose of the "mutator" facility for modifying the UT2004 game is to encapsulate several functions in a clean self-sufficient piece of code. Therefore,

we provide two sample mutators, CaveUT_BasicExample and CaveUT_CursorExample, which any authors can use as the basis for their own applications. Each of these two sample mutators is a self-contained module designed to work seamlessly with the CaveUT mutator.

Configuration Menu

Beginning with version 2.0, an in-game popup menu allows the application author to change the CaveUT parameters (e.g., screen rotation) without having to edit the CaveUT.ini file. The menu toggles on and off when the author presses a specific key on the keyboard. The key to press is defined in User.ini, where the application author can easily change which key is defined.

CaveUT_BasicExample and CaveUT_CursorExample both implement a simplified and improved version of the menu. The new menu makes startup and fine-tuning much easier – the installer simply inputs the rotation parameters needed for a screen and uses the keyboard fine-tuning functions to align individual views. Note that parameters for VRGL (described below) still require editing a separate file.

View Location Offsets

CaveUT 2.5 enables an application author to offset the camera viewpoint shown on the composite display from the actual location of the CaveUT player. For example, changing the camera viewpoint is an easy way to make the CaveUT player feel taller in the virtual environment. Setting the camera off to one side or in front of the player enables the player to look through walls. *It is very important never to use the offset parameters to try to synchronize the composite display.* Use the view rotation parameters for fine-tuning, *not* the view offsets.

Separating Selection/Firing from Movement with the Cross-Screen Cursor

CaveUTCursorExample implements a user-controlled cursor which travels across the entire composite display. It is actually a two-dimensional object traversing a tiny theoretical sphere centered on the user's viewpoint, so it is always in front of anything else in the virtual world. The user can employ the in-game menu to restrict the cursor's horizontal and vertical range in relation to the user's viewpoint. Alternatively, the user can simply allow the cursor to go off-screen. In some applications this would be appropriate.

In the unmodified UT2004 game, the user can fire only at things in the center of his or her view. This works reasonably well with a single monitor, but requires the user to rotate the view a lot more than he or she otherwise would. In a panoramic display, like a CAVE or a dome, this design becomes impractical. CaveUT_CursorExample enables the user to fire along a straight line from the user's viewpoint to the cursor and beyond. Figure 4 shows the two-walled V-Cave displaying one of the virtual environments that UT2004 provides. The user fires the lightning gun, and the blast hits the virtual wall behind the cursor. Importantly, the user can move in one direction and fire in another. This works best in a display that surrounds the user more fully, such as a dome.

Figure 4: Using the Cross-Screen Cursor to direct firing.

Authors of nonviolent applications can program objects to do something other than blow up. For example, in the educational game, Gates of Horus, every time a student clicks on a feature of an Egyptian temple, the virtual priest explains the feature (Jacobson, 2009). We used Gates of Horus in a learning experiment in the Earth Theater at the Carnegie Museum of Natural History (Jacobson, 2010).

Support for Curved Displays (e.g. Domes)

VRGL is the modified OpenGL library which CaveUT has always used to introduce flat-screen perspective corrections needed for multiscreen composite displays. It can be used independently from CaveUT and has its own version number. VRGL 2.5 (the latest version) has fairly detailed support for producing a window onto the virtual world which is not flat, but is instead a section of a sphere or a cylinder. This is perfect for moderately curved displays (Figure 5) and for multi-projector dome displays such as the Earth Theater (Figure 2). Its projection model uses only a single virtual camera, which limits the warping (horizontal, vertical, and radial) to approximately 125° before the center of the image begins to degrade noticeably.

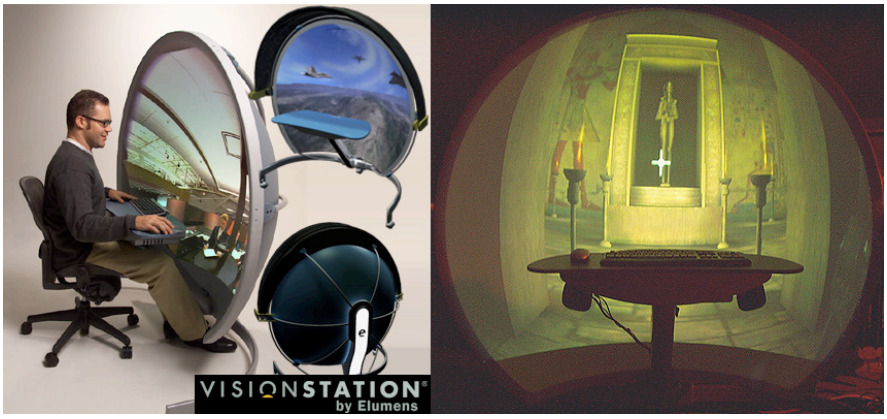

Figure 5: Virtual Temple in the Vision Station

Support for Edge Blending and Color Adjustment

VRGL 2.5 uses a configuration file to support direct color adjustment and brightness control, especially at the edges of view. This is especially useful when using multiple projectors to create a single contiguous image. The application designer can dim the overlapping edges to minimize the visibility of seams between the views.

Control of the User Viewpoint via UDP

VrpnUT 2.5 is a UT2004 mutator which allows an external program on the same computer or LAN to control the movement of a player using a UDP connection. If CaveUT is installed and the composite display shows that player's point of view, the external program can effectively control the (virtual) movement and facing of the entire composite display.

VrpnUT has been used by the Medical Virtual Reality Center (MVRC) at http://www.mvrc.pitt.edu/ and by the US Army Natick Soldier Research, Development, and Engineering Center (NSRDEC) to communicate with a working treadmill attached to the CaveUT display. A device on the treadmill reads the treadmill's motion and feeds the data to a computer on the display's LAN, which in turn generates the UDP packets that command the view to move (Figure 6). After the movement is properly scaled, the effect for the user is to be able to move in the virtual environment by walking in the physical world, creating a greater sense of continuity between them. As the name suggests, we envision VrpnUT as a client for virtual reality peripherals network (VRPN), a freeware facility for interfacing with VR peripherals in a standardized way.

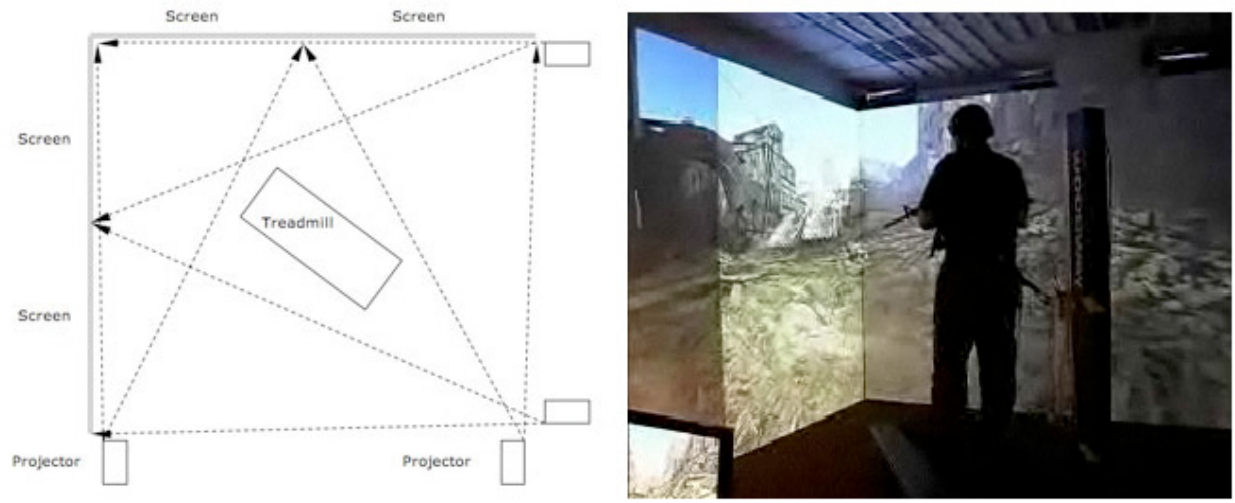

Figure 6: The Dismounted Warfighter Virtual Testbed at the US Army Natick Soldier Research, Development, and Engineering Center in Massachusetts.

Stereographic Displays Are Possible with CaveUT

The CaveUT project does not support stereo specifically, but it is possible to build a stereographic CaveUT display by correctly using hardware. For example, researchers at the Chinese University of Hong Kong built a *stereographic* display with CaveUT in conjunction with multiple dual-projector stacks intended for experiments in shared telepresence. (See http://www.ine.cuhk.edu.hk/I2/web/page4.html).

Future Directions

UT2004 and Unreal Engine 2.5 are now six years old and generally considered obsolete. Nevertheless, they still work and have a lot of development tools and resources. CaveUT 2.5 is still useful as a learning tool for students and educators just starting with low-budget virtual-reality applications and for developers whose needs it readily satisfies. PublicVR will continue to support version 2.5 of CaveUT with documentation and bug-fixes. We seek support to build a new CaveUT for Epic Games' Unreal Development Kit, an authoring environment based on Unreal Engine Three (http://udn.epicgames.com/Three/DevelopmentKitHome.html).

Credits

CaveUT was first developed at the University of Pittsburgh's Medical Virtual Reality Center (MVRC) in 2001 (http://www.mvrc.pitt.edu). Dr. Michael Lewis sponsored the effort under his AFOSR contract F49640-01-1-0542 through the Department of Information Science at the University of Pittsburgh (PITT). Dr Mark S. Redfern and Dr. Joseph M. Furman sponsored the effort under NIH P30 Grant DC05205 through the Department of Otolaryngology at the University of Pittsburgh. Technical work was by Jeffrey Jacobson (design, OpenGL programming, testing) and Zimmy Hwang (Unreal Script Programming). Patrick J. Sparto and Leigh Mahoney (MVRC) helped with testing.

In 2002 Willem de Jonge provided the math code for view rotation, and MVRC generously allowed the use of its cave-like display for testing and development. Later, Dr. Ken Sochats's Visual Information Systems Center (VISC), Lewis's Usability Lab (USL), and the Department of Information Science sponsored construction of the Virtual Theater, a CaveUT-based display we used for development. Joe Manjlovich ported this version of CaveUT to Linux. The Intelligent Virtual Environments Lab at the University of Teesside took this version of CaveUT and developed its own highly advanced immersive narrative system.

CaveUT 1.2: Under Jeffrey Jacobson's management (and design and testing) in 2003 and 2004, CaveUT was ported to UT2003. Willem de Jonge wrote VRGL, a modified OpenGL library which provides the perspective effects; Joe Manjlovich wrote the game code for CaveUT; and Mike Lambert improved it. John de Weese ported CaveUT to

UT2004. Jeffrey Jacobson wrote extensive documentation, and Aaron Allston edited it. Demiurge Studios generously supported Michael Lambert's contribution. Epic Games supported the whole effort by granting the newly constituted PublicVR a read-only license to the UT2003/UT2004 code, which was essential for the creation of VRGL. The Carnegie Museum of Natural History also installed CaveUT, providing a third testing and development site. Its staff, led by Kerry Handron, were quite helpful.

CaveUT 2.0 was never officially released, but was available through PublicVR forums and was informally supported. Jonathan Hagewood at Psyonix, Inc. greatly improved the performance of CaveUT, wrote logging functions for CaveUT, and created a means of controlling the viewpoint via a UDP connection. The MVRC funded this work. Jeremy Carson contributed many usability improvements and bug-fixes.

CaveUT 2.5 is the current version. Gerke Max Preussner at Virtual Heroes, Inc., completely rewrote the code base and structure. Jeffrey Jacobson oversaw the interaction design, documentation, testing, and requirements. Development cost was paid from PublicVR internal funds with a matching contribution of engineering time from Virtual Heroes, Inc.

Our thanks to Richard Graefe for his excellent help with editing this paper.

References

Augustyn, J. S., Brunyé, T. T., Rock, K., Hepfinger, L., and Mahoney, C. R. (2008). Evaluating camouflage effectiveness using virtual reality. Proceedings of the 52nd Annual Meeting of the Human Factors and Ergonomics Society.

Baričević, D., Dujmić, H. and Šarić, M. (2007). "QAVE – a Gamebased Immersive Virtual Reality System", Proceedings of the 18th International Conference on Information and Intelligent Systems, pp. 357, Varaždin, Croatia, 12 -14 September, 2007.

Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A. (1993). Surround-screen projection-based virtual reality: the design and implementation of the CAVE. Proceedings of the 20th annual conference on computer graphics and interactive techniques. ACM (1993) 135-142.

Henden, C., Champion, E., Muhlberger, R., and Jacobson, J. (2008). Sharing the Magic Circle with Spatially Inclusive Games. SIGGRAPH Asia, December, 2008

Henden, C., Champion, E., Muhlberger, R., and Jacobson, J. (2008b). A Surround Display Warp-Mesh Utility to Enhance Player Engagement. International Conference on Entertainment Computing, September, 2008, Pittsburgh, PA.

Jacobson, J. and Handron K. (2008). Dome Displays for Educational Games and Activities in the Museum and on the Road, International Conference on Entertainment Computing, September, 2008, Pittsburgh, PA.

Jacobson, J., Handron, K., and Holden, L. (2009). Narrative and Content Combine in a Learning Game for Virtual Heritage. Computer Applications in Archaeology, Williamsburg, VA.

Jacobson, J. (2010). Digital Dome Versus Desktop Computer in a Learning Game for Religious Architecture. Annual Meeting of the American Educational Research Association (AERA), Denver, CO, April, 2010.

Lee, S. L. and Akin, O. (2005). The Virtual-Augmented-Reality Environment for Building Commission: Case Study Proceedings of ICEBO 2005 Conference, Pittsburgh, PA, USA, October 15-17 with Sang Hoon Lee.

Pape, D., Roussou, M., and Anstey, J. (2004). VR for Public Consumption (Workshop), *IEEE Virtual Reality,* Monterrey, CA, 2004, http://resumbrae.com/vr04/

Saenz, A. (2010). http://singularityhub.com/2010/01/06/retail-head-mounted-displays-getting-better-vuzix-wrap-920