Innovative Technology for Computing Professionals
Computer
The flagship magazine of the IEEE Computer Society

Entertainment Computing

Game Engine Virtual Reality with CaveUT

(pdf) printer friendly version

Jeffrey Jacobson and Michael Lewis, University of Pittsburgh

CaveUT, an open source freeware project, uses game technology to make immersive projection-based virtual reality affordable and accessible.

Relatively simple, the current public release of CaveUT works well for low-cost displays. The most advanced version, which will be publicly available in the fall of 2005, supports real-time spatial tracking and stereographic imaging. It is currently installed and working in the SAS-Cube, a CAVE-like display.

Computer games with the most advanced simulation and graphics usually employ a *game engine,* a commercially available software package that handles basic functions. For example, the first-person shooter *Unreal Tournament* for the PC employs the Unreal Engine to provide richly detailed graphics, high-speed processing performance, a built-in physics engine, a scripting language interpreter, and robust networking for shared environments.

CaveUT modifies *Unreal Tournament* to let it display in multiscreen enclosures suitable for immersive virtual reality applications. VR applications developed with CaveUT inherit all the Unreal Engine's capabilities along with *Unreal Tournament*'s authoring support, open source code, content library, and large user community.

ORIGIN AND DEVELOPMENT

In 1990, researchers developed the original Cave Automatic Virtual Environment, a partial cube approximately three meters per side, with each wall functioning as a rear-projection screen illuminated by a projector. A mainframe drives all of the CAVE's projectors, displaying a contiguous visual image across all screens to produce a virtual landscape. Stereographic imaging makes the virtual objects look more three-dimensional, while real-time spatial tracking lets users interact with the objects and navigate the space.

Throughout the 1990s, researchers developed many VR applications for the CAVE and similar displays. They either programmed these applications directly, starting from OpenGL or a similar graphics library, or wrote them using advanced authoring kits. Despite the process's difficulty and expense, it usually produced applications with poor and often primitive graphics, low performance, and limited networking functionality.

In 1997, Paul Radjlich produced a version of *Quake* for CAVE that inherited the game's authoring support, networking, and other features. Unfortunately, Cave-Quake could not benefit from *Quake*'s game engine, which was PC-based.

By 2000, the game industry had driven significant advances in graphics hardware for the PC, and CAVE owners began replacing their mainframes with PC networks thanks to the PC's low cost and increasing graphics power. Further, the leading first-person shooters, *Quake* and *Unreal Tournament*, surpassed the traditional CAVE-based applications in graphics quality, performance, animation, and networking. These games also had respectable authoring support, built-in physics, partially open source code, a large base of existing content, and an active developer community.

Despite these developments, the games still lacked the ability to perform real-time shape generation for scientific visualization applications. Their interface and animation support also proved limiting for applications using a different paradigm.

Thus, in 2000 we decided to adapt *Unreal Tournament* to the BNAVE, a PC-based CAVE-like display. With Michael Lewis's support and guidance, Jeffrey Jacobson and Zimmy Hwang invested a year of careful study, then solved the multiscreen display problem by inserting just six lines into the game's open source code.

We packaged this as the first version of CaveUT, which we made into a freeware project to attract collaborators. Since then, CaveUT has become far more capable and complex through independent code contributions.

Today, most software development for CAVE still uses the traditional VR authoring tools, which have improved considerably but remain expensive. However, the game engines have maintained their list of advantages and increased their lead in graphics quality and performance, thanks to massive funding by a game industry that has grown larger than Hollywood.

Already, many developers in science and industry take advantage of game technology for other uses, with developers basing more and more immersive VR applications on games. Meanwhile, CaveUT has attracted a growing list of collaborators, which should lengthen with the release of the project's stereographic code.

HOW IT WORKS

A multiscreen display based on CaveUT requires a server computer that connects by a standard LAN to several client computers. Each client drives one screen of the display, usually a projection screen illuminated by a digital projector.

The operator begins a multiplayer game of *Unreal Tournament* with one normal player on the server and one spectator player on each of the clients. Each spectator's view duplicates the view seen by the player on the server. On each client, the CaveUT code rotates the view—the screen's window—into the virtual world so that each screen shows its part of the composite view. Figure 1 shows the view of one screen turned 45 degrees to the left, while the other screen has been turned 45 degrees to the right.

To handle perspective correction, CaveUT employs Willem de Jonge's VRGL, an OpenGL library modified for VR applications. For an

installation with no head tracking, the user must specify a single ideal viewing location for the whole display. As long as the user's head stays at or very near this point, the view will remain unified and undistorted. If the installation does have a head tracker, CaveUT can correct the perspective in real time to effectively follow the user's movement.

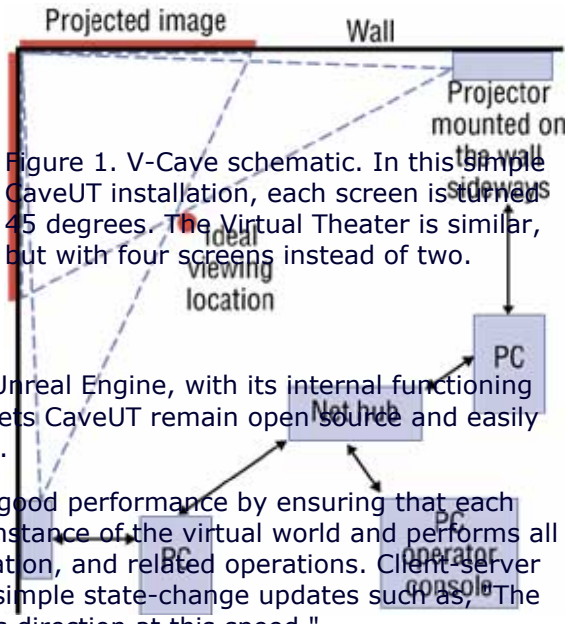

Figure 1. V-Cave schematic. In this simple CaveUT installation, each screen is turned 45 degrees. The Virtual Theater is similar, but with four screens instead of two.

This arrangement uses an unmodified Unreal Engine, with its internal functioning unaffected by CaveUT and VRGL. This lets CaveUT remain open source and easily upgradable to each new engine version.

Unreal Tournament provides CaveUT's good performance by ensuring that each machine's copy maintains a complete instance of the virtual world and performs all its own graphics rendering, physics, animation, and related operations. Client-server communication is confined to fast and simple state-change updates such as, "The player is now here and is moving in this direction at this speed."

CaveUT can support at least 32 independent view screens for a single application, in any configuration, and with multiple real players. For example, developers could configure a six-walled enclosure with all its views centered on the first user, two more four-walled enclosures with views that center on the second and third users, and single-desktop arrangements for eight more users. All 11 users could share a single virtual environment. Some could be students, others instructors, and yet others acting as intelligent scenery. In this scenario, only imagination and budget limit the possibilities.

MINIMUM REQUIREMENTS

A monoscopic nontracking CaveUT display can be configured cheaply from common office equipment and simple hardware. Front-projection screens can consist of any clean, white surface, while rear-projection screens can be made from any translucent material. Each screen requires one common DLP or LCD projector driven by a standard personal computer with a good video card and an installed copy of *Unreal Tournament,* which currently retails for around $20.

An additional computer hosts the server and acts as the operator console. The server connects to the clients through an Ethernet switch and the appropriate cables. CaveUT/VRGL is currently limited to OpenGL and the Windows OS, although porting it to Linux/Unix/MAC-OSX would not be difficult. Configuring the Virtual Theater shown in Figure 2 cost only $25,000 and required no special

programming.

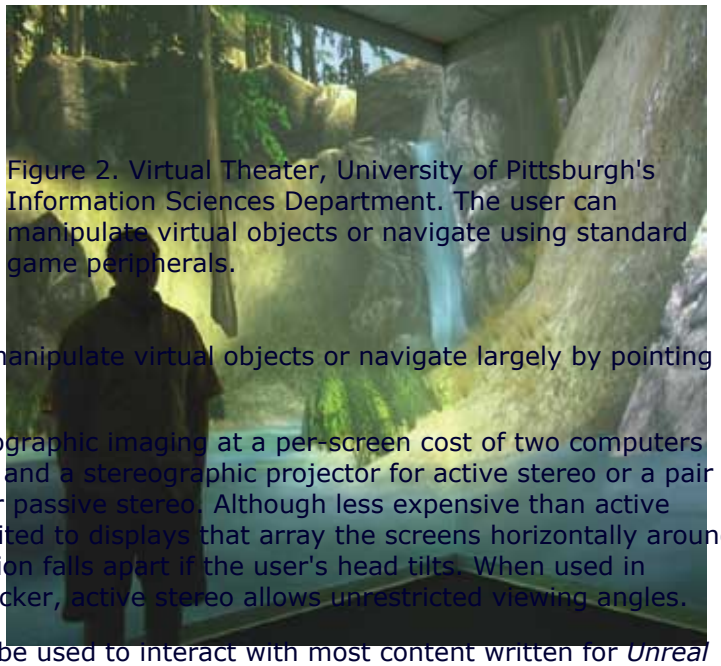

Figure 2. Virtual Theater, University of Pittsburgh's Information Sciences Department. The user can manipulate virtual objects or navigate using standard game peripherals.

With CaveUT, the installation developer can upgrade steadily from an inexpensive display to a fully capable CAVE-like interface. The display supports spatial trackers useful for head-tracking and advanced controls. For example, the user can manipulate virtual objects or navigate largely by pointing with the tracked controller.

CaveUT also supports stereographic imaging at a per-screen cost of two computers with specialized video cards and a stereographic projector for active stereo or a pair of monoscopic projectors for passive stereo. Although less expensive than active stereo, passive stereo is limited to displays that array the screens horizontally around the viewer because the illusion falls apart if the user's head tilts. When used in combination with a head tracker, active stereo allows unrestricted viewing angles.

A CaveUT installation could be used to interact with most content written for *Unreal Tournament* and most applications built on the game. The highly localized code changes that CaveUT introduces are unlikely to conflict with the *UT*-based application's code changes. This is an important advantage because the large community of *Unreal Tournament* gamers and researchers produces a great deal of artistic content, animation, and code modifications for the game engine. CaveUT developers can use most of this material, in pieces or in whole applications, and benefit from the *Unreal* development community's support and cooperation.

CURRENT PROJECTS

Several projects already use CaveUT, including the following:

- The ALTERNE project's artists use CaveUT for interactive art installations, storytelling, and information visualization.
- Researchers at the University of Pittsburgh's Visual Information Sciences Center use CaveUT to prototype systems for first-responder emergency training, virtual museums, way-finding applications, and architectural planning.
- Researchers at the University of Pittsburgh's Usability Lab (ULAB) are developing CaveUT projects for robotic simulations and virtual archeology.
- Stagecraft designers at the University of Southern California's Institute for Creative Technologies use CaveUT for interface prototyping.
- A low-cost portable CaveUT display has been demonstrated at conferences such as CHI 2002, HFES 2002, VR 2003, and I3D 2003.
- The artists at Elumenati are helping to develop CaveUT's close cousin, DomeUT, for applications in all-digital dome displays.
- Military researchers are using CaveUT as an immersive interface for their Unreal-Engine-based training simulators.

This list will continue to expand because CaveUT's low initial cost makes it accessible and attractive to students, educators, artists, developers, gamers, and small businesses working in a wide range of disciplines.

Along with the powers of the Unreal Engine, CaveUT also inherits its biases. The engine can support large virtual environments, but it works best with relatively small worlds that are rich in detail and activity. For example, the engine can support a dozen or so amazingly detailed and lifelike humanoid agents more efficiently than a large number of simple ones. All game engines work with pre-defined objects and

cannot use a data stream to continuously generate visual effects. However, they can be used to display effects transmitted to the engine from other software or generated from data in advance. CaveUT also inherits the advantages and limitations of *Unreal Tournament,* but it could easily be adapted to any game based on the Unreal Engine. Similarly, CaveUT could be adapted for other graphics systems to provide more options for the developer.

Nevertheless, CaveUT now supports a wide range of applications. It continues to develop through contributions of all kinds, such as the following:

- Marc Le Renard and Jean Lugrin will release their stereographic extension for CaveUT this fall, and Le Renard will also release his spatial tracking for CaveUT then.
- Demiurge Studios is adapting VRGL—and therefore DomeUT/ CaveUT—for use with multiprojector curved screen and partial-sphere displays.
- Ivor Diosi is working on a method for streaming video from an external source onto a surface in the virtual environment.
- Jacobson and Demiurge are developing optimal ways to use commercially available game peripherals in an immersive display.

All these new features will require documentation, tutorials, testing, and extension to a variety of platforms, especially the low-cost ones. We always welcome new users and collaborators as we build up the CaveUT user community.

Jeffrey Jacobson is a graduate student and Michael Lewis is an associate professor, both in the University of Pittsburgh's Information Sciences Department. Contact Jacobson at jeff@planetjeff.net and Lewis at ml@sis.pitt.edu.

From the April 2005 issue of *Computer*.