IMMERSIVE COLLABORATIVE LABORATORY SIMULATIONS USING A GAME ENGINE

Chenghung Chang, Dror Kodman,
Constantin Chassapis, Sven K. Esche,
Department of Mechanical Engineering
Stevens Institute of Technology
Hoboken, NJ 07030

Abstract

This paper discusses the possibility of using a commercial game engine, such as the "Source" engine used in "Half-Life 2", to implement an immersive and collaborative virtual laboratory environment that will enable multiple students to perform educational laboratory experiment simulations. These simulations will involve real-time student interaction through a computer network, and they will benefit the students by stimulating the different modalities of learning, i.e. visual, audio, read/write and kinesthetic.

By using an existing commercial computer game engine, the need for creating from the ground up the components that are combined into an interactive virtual world is eliminated. Instead of expanding extensive resources on developing the underlying visual, audio and logistical system infrastructure, the main attention can be focused on those features that facilitate student learning, such as high levels of interactivity, significant collaboration between the students and their feeling of immersion. It is postulated that such laboratory environments can be tailored so as to actively engage the students, foster their desire to learn more about the experiment and its underlying theories, make them feel like they are essential to carrying out the experiments as a group and to feel comfortable in such virtual environments.

This multi-disciplinary research is being carried out at Stevens Institute of Technology (SIT) with funding from a multi-year grant by the National Science Foundation's Information Technology Research program.[1]

Key words - Laboratory education; virtual laboratory; virtual experiment; collaborative virtual environment; virtual reality; game engine; Source engine.

Introduction

In the computer gaming world, there is a constant demand for increasingly realistic gaming experiences that incorporate high fidelity graphics, real-time physics engines and innovative interactive game playing. By setting the bar so high, a very intense competition in terms of both hardware and software is fueled within the industry, which generated sales of over $7 billion in 2004.[2] The current state of raw computing power, graphics hardware technology and game software complexity have evolved to the point where low-end personal computers have the potential to run and render environments that are encroaching upon the borderlines of reality. Coupled with a publicly available game Software Development Kit (SDK), what results from this development is the ability for users to create their own customized 3-D environments, then have a game engine render them in such a way that the users are enabled to realistically explore the environment via first person perspective and interact with the environment.

Alternate uses of these gaming technologies other than for pure entertainment purposes are emerging and becoming increasingly prevalent. Research is currently being conducted at SIT to incorporate such gaming technologies into educational laboratory experiments.

From a societal point of view, video games and computer games have existed for decades and are becoming an integrated part of our culture, either directly or indirectly on a personal level. Despite existing reservations by some individuals, the majority of youths today is highly accustomed to and knowledgeable about playing electronic

games and regards them in a positive light. Taking advantage of this positive attitude, we can utilize an existing game engine as a means for creating educational tools that will make it easier for students to learn in an engaging manner. It is possible to customize content to better address the needs that arise in connection with the various preferred learning styles of students – visual, audio, read/write and kinesthetic. A multiplayer game mode can be employed to enable multiple students to participate in a collaborative laboratory session. By using electronic gaming technology, we are engaging a generation of learners that are accustomed to using computers and playing computer games.

Game Engines

At present, some of the better known commercial game engines representing the cutting edge of technology in first person perspective graphics are Epic Game's "Unreal" engine,[3] id Software's "DOOM 3" engine[4] and Valve Corporation's "Source" engine[5]. The type of games that these engines were designed for are predominantly "first person shooters", where the user controls the movements and actions of a computer character and the visual display mimics the perspective of what the in-game character would see with his/her own eyes.

Some non-entertainment applications of game engines are immersive first person environmental exploration, social interactions or tactical simulations involving multiple users as well as the creation of animated 3-D movies. The "Unreal" engine for example has been used in the CaveUT project at the University of Pittsburgh, which aims at designing an affordable open space virtual reality experience.[6] A main computer hosts the "Unreal" game with two other client computers in spectator mode functioning as cameras attached to the user controlled avatar.[a] Modifications to the original source code of the "Unreal" game were made that rotate the camera directions such that their respective images can be projected onto white screens oriented 90 degrees from each other. Standing in a strategic location with respect to these projection screens, the physical user experiences a sense of immersion as the screen takes up nearly the user's entire field of vision. Meanwhile, tactical simulations involving multiple users are being investigated by the US Military as training software.[7] In the architecture of their "Unreal Tournament Semi-Automated Force" (UTSAF) software, Epic Game's "Unreal Tournament" game engine and their main simulation servers are interfaced. This enables them to use common PCs in conjunction with "Unreal Tournament" as a low cost replacement to the expensive proprietary equipment they used previously in their simulations.

Game Engines and Educational Laboratories

"Source" Game Engine

"Half-Life 2", the computer game software developed by Valve Corporation, was officially released to the public at the end of 2004. For all individuals who purchase this game, the "Source" SDK is made available as an optional download that enables the purchasers to create and edit new and existing content and to incorporate their own game logic and features. By facilitating the creation of a modification - or Mod as it is called in the gaming industry - to the "Half-Life 2" game, users are permitted to copy all of the original C++ game files into a separate location and then change and recompile the source code.

Usage of "Source" for Developing Virtual Laboratory Experiments

Using the "Source" SDK, it is possible to create a collaborative environment that utilizes part of the existing game engine to handle the delivery of the visual and audio components of the interactive experience. Such a development is currently being undertaken at SIT, where the "Source" SDK is being used to implement a virtual educational laboratory. This laboratory incorporates a virtual space that contains 3-D virtual models of existing physical laboratory setups, which are currently used by students in various laboratory facilities at SIT.

[a] Avatar: instance of the user in a game, i.e. the game character controlled by the user.

Some benefits of utilizing this virtual system are that it:

- addresses different learning modalities (read-write, audio, visual, kinesthetic),
- provides engaging environments for a generation of students accustomed to video games,
- reduces university resources (time, space and equipment) required for operating educational laboratories,
- enables nearly unlimited student access to the laboratory experiments, and
- facilitates the convenient distribution of information on the experiment by directing the students to resources pertaining to the topic at hand.

By utilizing a commercial game engine to create our modified game and the framework on which to build our application, we are able to take advantage of the latest gaming technology. By employing the game engine for delivering the video and audio content as well as the basic underlying game playing interface, we can concentrate with our development work on creating the content for the laboratory simulation and on enhancing the feel of immersion that the students experience. The creation of a feel of immersion depends on achieving:

- a high level of interactivity within the environment, and
- an appropriate architecture of the environment (light, texture, objects, etc.).

Currently, two approaches for implementing laboratory experiment interactivity are being explored. In both scenarios, the students log into the laboratory environment and interact with the experiment simulations. These simulations are either executed internally or externally to the 3-D environment. In the first case, the experiment functionality is programmed "in house" with modifications made to the C++ code of the original game. This modified C++ source code is then recompiled in its entirety to reflect the desired alterations made to the original game, and thus the experimental procedure becomes part of the source code. The second strategy is based on using the game engine as the virtual environment generator. This generator acts like a shell for interfacing a virtual environment with external agents, i.e. by generating input data to be passed on to the agents and by parsing the simulated output data received from the agents. In this approach, a student interacting with an experiment will trigger an application external to the game itself, which is hosted by a server that simulates the experimental procedure.

Modeling of a Realistic Laboratory Space

In order to implement the 3-D virtual laboratory environment in a fashion that fosters the students' feel of immersion by presenting a realistic visualization of a physical laboratory space, three different levels of detail are employed: extended, intermediary and detailed. For example, the detailed level of the laboratory environment would be limited to the experiment located on a desk that the student can walk up to and physically interact with, effecting the input and output of such an experiment. The intermediary level would represent a laboratory room that the student is currently standing in, with windows, desks, proper textures and light mapped onto all surfaces of the room. The extended level of the laboratory environment would include the appearance of a landscape outside of the laboratory windows which gives the student a sense of realism that expands beyond the limited area of accessibility of the room.

Depending on the permissible physical proximity with respect to the user within the game environment, in-game entities will need a certain level of detail for the user to smoothly transition from one perspective to another. Distant buildings can be constructed out of low to medium geometric and textural detail. They can be created using the SDK's software for editing the virtual environment in conjunction with a material texture of relatively lower resolution. On the other hand, in-game entities that will be inspected within close range should be generated using a dedicated 3-D modeling program allowing for higher degree of complexity, and likewise the material texture should be of greater quality.

Modes of Usage

After the underlying "Source" game engine has been initially installed onto a local client computer together with the laboratory module, the laboratory environment can be accessed in either single-player or multiplayer mode.

In single player mode, the students independently access the laboratory experiment simulations by first downloading the necessary files from a university server (see Figure 1). This laboratory approach can be implemented as a complement to the actual hands-on interaction with real laboratory equipment (preparatory training before or extension of traditional laboratory session), or it can serve as a complete substitute to the hands-on laboratory approach. The students are able to run the laboratory simulation whenever they choose to and to progress through the experiment simulation at their own pace.

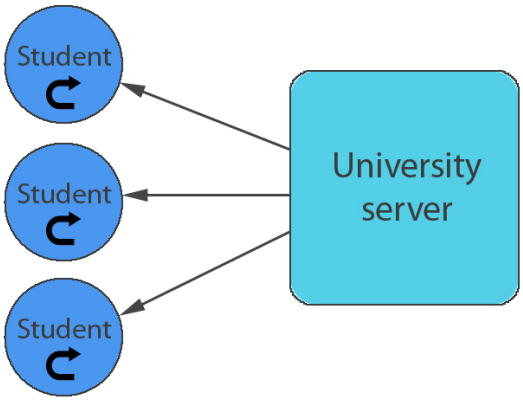

Figure 1: Implementation with local hosting where students download the laboratory environment and interact with the virtual experiments independently.

In multiplayer mode, the individual students – with the assistance of Half-Life 2's networking interface that acts as a relay (see Figure 2) – connect to either a university server or another user's computer that is hosting a virtual laboratory environment. With multiple students logged on at the same time, collaborative sessions are then possible where the laboratory experiment is carried out by students as a group. This is of particular interest as it has been shown in the context of Asynchronous Learning Networks (ALN) that it provides for more effective learning when students can participate and interact as a group.[8] Verbal and text message communications enable the proper coordination of the experimental tasks amongst the students. In addition, visual and audio cues in the form of help text windows and prerecorded narrations can also guide the students in accomplishing the required tasks for completing the laboratory exercise.

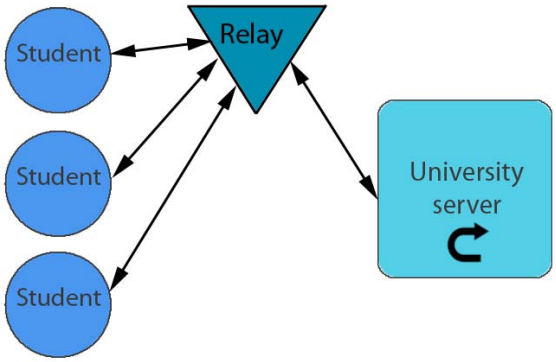

Figure 2: Implementation with remote hosting where a university server enables the interaction of multiple students with the virtual experiments.

Sample Experiment Script

Below, a sample script for an experimental session is given:

- When logging into the laboratory environment, the student is first presented with a static screen describing the essence of the experiment (see Figure 3).
- The students work in groups and can communicate among themselves by means of voice or chat (see Figure 4).
- Each one of the groups sharing a laboratory space has a designated location in the virtual environment, and each student has laboratory instructions available through browsing in his/her personal menu (see Figure 5).
- Once the student gets close to the experiment area, he/she receives a message that instructs him/her to browse through his/her laboratory gear menu and to place the necessary equipment in the designated experiment area.

- The students then proceed together in assembling the experimental setup according to the laboratory instructions, or each student assembles his/her own experimental setup.
- Once the students are done setting up their individual experiments, they launch an input/output menu to carry out the actual experimental procedure.

Some Remarks on the Development of a Game Modification

In order to develop the immersive laboratory simulation, a series of steps has to be taken by the developers, ranging from the physical architectural layout of the environment, the use of texture, light and shadows, to different levels of interactivity and role playing. As a result, a number of complex development tasks can be worked on simultaneously by different groups of developers. The types of work required in developing a game modification, or Mod, fall into the following main categories: developing a conceptual artistic vision, generation of 3-D models and animations, environment mapping, and implementing audio effects.

In order for all the individual elements of the game to form a cohesive whole, an overall artistic direction is needed where conceptual visuals of characters, objects, and the environment are generated and shared among all developers. All parties involved will therefore be proceeding in the same direction with respect to the envisioned shared environment, while being focused on their individual case-specific tasks and responsibilities, thus bringing the vision into an actual playable form.

3-D modelers will be responsible for modeling in-game characters, equipment, and interactive and non-interactive 3D objects. For the generation of the animated 3-D models (i.e. human characters), the development process requires merging a predefined skeletal system (i.e. rigid bone members and joints of motion connecting them) with a visual 3-D mesh (representing the physical appearance). Then, the 3-D visual mesh as well as the bone locations for sequential frames of animation are exported and compiled to be integrated within the game.

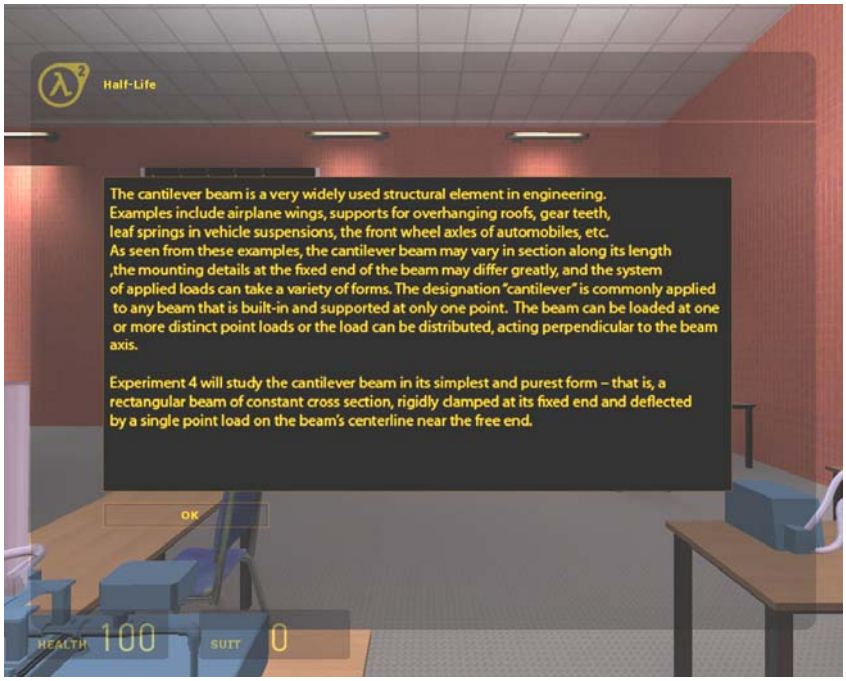

Figure 3: Text appears at the start of the simulation providing introductory information on the laboratory objectives, procedures, and equipment.

Figure 4: Multiple users can interact with each other by chat to collaborate in a laboratory experiment.

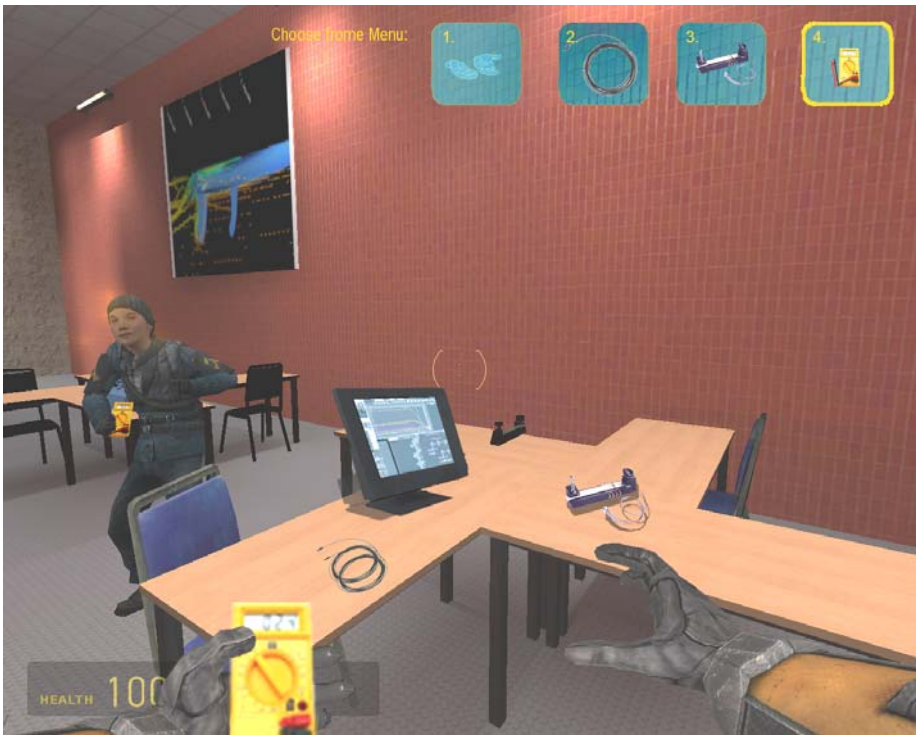

Figure 5: The users will be able to browse through a tool menu located on the screen.

A significant number of modifications will be required in order to add customized interactions that are not built into the game engine itself. This might include imbedding complex state logic of cause and effect into the game and then associating it with a specific entity. In addition, there is also significant potential in modifying and utilizing a hierarchical window system already residing within the "Source" engine that can be displayed as a two-dimensional overlay window or an in-game 3-D interactive panel. This would allow a more immersive experience because all events of the experiment simulation will appear as if they were carried out entirely within the virtual space.

Strategic creation and application of material textures can alter a simple geometry into one that is ostensibly rich in detail and complexity. Different aspects of a material are defined by using the red, green, blue, and alpha (RGBA) channel values of single or multiple raster image files, thus defining visual properties such as base color, transparency, bump mapping[b], and specularity[c]. Also, each texture file has the option of associating physical properties to be used within the game, such as, friction, density, sounds and other special effects.

Effective construction of virtual environments requires intimate knowledge of the library of various built-in game functions. Some of the tasks involved in the development process include creating the physical layout of the environment, placing all 3-D models in their respective locations, setting up cause and effect triggers, ensuring that the lighting conditions are appropriate, and placing of other various in-game entities.

In order to create an engaging interactive scene, a certain degree of audio functionality should be integrated into the simulation. For example, the audio aspect could be the sound effects of machines in motion, audio cues and vocal explanations, and the ambient sound or background music that would suit the environment.

[b] Bump mapping: computer graphics technique where at each pixel, a perturbation to the surface normal of the object being rendered is looked up in a texture map and applied before the illumination calculation is performed.

[c] Specularity: a quality used in 3D rendering programs that gives a surface the illusion of reflectivity.

Conclusions

This paper discusses the use of Valve Corporation's "Source" SDK to create a customized educational laboratory simulation. By using an existing commercial game engine, one can take advantage of the latest computer gaming technology without the need to recreate the underlying functionality of game engines. By appropriately modifying the C++ source code and creating additional visual and audio content, a rich interactive laboratory experience can be designed and implemented. This approach is tailored to a new generation of students, which is accustomed to using computer game technology. This laboratory format has the potential for providing educational laboratory content in both 2-D and 3-D, which will enable students to practice in virtual environments what they learned in class. This is expected to result in a robust and efficient way of reinforcing learning by an experimental component. The existing multiplayer framework also provides the possibility for multiple students to participate in the same virtual laboratory experiment, thus facilitating collaborative work. The laboratory experiment simulation could be employed as a complement to or a complete substitute for traditional hands-on student experiments.

References

1. Esche, S. K., Corter, J. E., Nickerson, J. V. & Chassapis, C. (2003). ITR: An Infrastructure for Designing and Conducting Remote Laboratories, NSF-ITR Grant No. 0326309.

2. Entertainment Software Association (2005). Essential facts about the computer and video game industry, 2005 sales, demographics and usage data, http://www.theESA.com/.

3. "Unreal" game engine by Epic Games, Inc.: http://www.unrealtechnology.com/.

4. "DOOM 3" game engine by id Software: http://www.idsoftware.com/.

5. "Source" game engine by Valve Corporation: http://www.valvesoftware.com/.

6. Jacobson, J. & Lewis, M. (2005). Game engine virtual reality with CaveUT. Computer, Vol. 38, No. 4, pp. 79-82.

7. Manojlovich, J., Prasithsangaree, P., Hughes, S., Chen, J. & Lewis, M. (2003). UTSAF: A multi-agent-based framework for supporting military-based distributed interactive simulations in 3-D virtual environments. Proceedings of the Winter Simulation Conference, December 7-10, 2003, Vol. 1, pp. 960-968.

8. Hiltz, S. R., Coppola, N., Rotter, N., Turoff, M. & Benbunan-Fich, R. (2000). Measuring the importance of collaborative learning for the effectiveness of ALN: A multi-measure, multi-method approach. Journal of Asynchronous Learning Network, Vol. 4, No. 2.

Biographical Information

Chenghung Paul Chang is currently a Research Assistant at Stevens Institute of Technology and a Ph.D. candidate in Mechanical Engineering. He obtained his Master's Degree in Mechanical Engineering from Stevens Institute of Technology in 2005. His research includes alternatives to traditional methods of administering laboratory experiments, including remote experiments and virtual experimental simulations.

Dror Kodman received a Bachelor's degree in Architecture from New Jersey Institute of Technology in 2001. In 2006, he completed his studies for a Master's Degree in Mechanical Engineering at Stevens Institute of Technology. He has professional experience in a variety of design fields such as architecture, exhibit design, as well as new media and graphics design and has research interests in the fields of gaming technology and computational fluid dynamics.

Dr. Sven K. Esche is currently holding a position as Associate Professor of Mechanical Engineering at Stevens Institute of Technology. In 1989, he received an undergraduate degree in Applied Mechanics from Chemnitz University of Technology (Germany). After working for three years at Mercedes Benz AG in Stuttgart Germany, he obtained his M.S and Ph.D degrees in Mechanical Engineering from Ohio State University in 1994 and 1997, respectively. His current research interests include multi-scale modeling of thermo-mechanical processing of metals, integrated product and process design under conditions of uncertainty and risk as well as remote sensing and control of distributed devices with special focus on remote laboratories.

Dr. Constantin Chassapis is a Professor and the Director of the Mechanical Engineering Department at Stevens Institute of Technology. His research interests are in knowledge-based engineering systems; computer-aided design and manufacturing; structure-property modeling and characterization of polymers and polymer composites as well as in remotely controlled distributed systems. He has been an active member of AMSE and SPE, and he has received a best paper award from SPE's Injection Molding Division, the distinguished Assistant Professor Award at Stevens Institute of Technology, an Honorary Master's Degree from Stevens Institute of Technology, and the Tau Beta Pi Academic Excellence Award.