Dynamic Virtual Environment for Multiple Physics Experiments in Higher Education

Fabio Ricardo dos Santos[1,3]
[1]Graz University of Technology
Graz, Austria

Christian Guetl[1,2]
[2]Curtin University of Technology
Perth, WA

Philip H. Bailey[3], V. Judson Harward[3]
[3]Massachusetts Institute of Technology
Cambridge, USA

Abstract—**The transportation of a campus classroom and/or laboratory into a three dimensional virtual representation has changed remote learning, specially in engineering education. Our first collaborative virtual environment, a proof of concept, provides full functionality of one physics experiment, though there are still some performance issues to be resolved.**

The next step for integrating TEALsim and iLabs in Sun's Project Wonderland is porting our system from Wonderland's version 0.4 to 0.5. Our goal is a system redesign in order to support adding flexibility to multiple physics simulations. The performance improvements in Wonderland 0.5 will allow a large number of avatars in our future scenario, where they will be able to run even more physics experiments, through a new 3D user interface.

Keywords-collaborative immersive environment; virtual worlds; Project Wonderland; iLabs; TEALsim; physics experiments, remote labs, visualization.

I. Introduction

In the past few years, especially due to the increasing availability of faster and robust hardware and software, 3D environments have become common technology. Virtual laboratories, scientific visualizations, and some collaborative work approaches are just some of the successful fields of 3D graphics applications.

One essential term is Virtual Environment (VE), which is a computer generated spatial environment, where the stimulation of diverse human senses gives the user a feeling of being immersed. Immersion means how deep the user is emotionally involved within a specific virtual environment [1].

Nowadays, a great research challenge for educational technology professionals is to build technology that not just supports the learning process, but also connect students and educators in a way so they can easily cooperate, even when both parties are geographically spread.

Our first collaborative virtual environment provides full functionality of one physics experiment. This environment results from the integration of internet-accessible physics experiments (iLabs) combined with the TEALsim 3D simulation toolkit in Project Wonderland [2]. Students and educators, represented as avatars, within this environment can remotely control experiment equipment, visualize physics phenomena generated by the experiment and discuss results. This environment was developed following the Technology Enabled Active Learning (TEAL) classroom idea to support social interactions, encourage student's active learning and interest, in an environment that fosters conceptual change [3].

This study explores the process of conversion of the TEALsim simulation package to the rendering engine jMonkeyEngine (JME), which is the graphics engine used in Project Wonderland version 0.5. Additionally, we have built a Wonderland cell class that can dynamically load TEAL simulations into Wonderland by the end of the research project. This will provide a collaborative environment similar to the current 0.4 version, but using the new graphics engine.

Furthermore, we implemented an automatic generation of the simulation's controls in the Wonderland environment either as buttons, sliders and other control elements within the 3D space or as elements in the Heads Up Display. These controls will provide the standard Java event handling model.

Therefore avatars have the ability to directly interact with TEALsim elements, including moving elements and activating sensors. Changes in the environment would be updated in real-time.

Finally, based on the latest research, this paper outlines the future research directions and challenges to overcome.

Figure 1. Close-up view of the 'Force on a Dipole' Experiment in Wonderland 0.4

This research has been partially supported by the Austrian Marshall Plan Foundation.

II. Related Projects

A. iLabs

An iLab stands for an internet-accessible experiment that can be reachable remotely at any time over the internet. Compared to conventional laboratories, iLabs are available 24-7 and are easily shared. Since they provide access to unique remote resources, they are less expensive and complex than conventional labs. Users can access these online laboratories from around the world through a single standard administrative interface. In engineering education, iLabs enrich the scope of experiments students have available to them in the course of their academic careers [4].

B. TEALSim

TEALSim, the TEAL simulation system is designed as a framework for authoring, presenting and controlling simulations in a variety of domains. It was developed by the Technology Enabled Active Learning (TEAL) Project at Massachusetts Institute of Technology (MIT) [3]. Among others, the objectives of this project are to increase student's conceptual and analytical understanding of the nature and dynamics of electromagnetic fields and phenomena, and also foster student's visualization skills.

Figure 2 illustrates an example of how TEALSim is very useful in electromagnetism helping students to visualize and process phenomena. During the Force on a Dipole experiment, TEALsim enables students to see the invisible magnetic field lines, which of course are not visible in real settings. This visualization behaves according to changes in simulation input values made by students, giving them a better understanding of electromagnetic fields. Such visualizations allow students to make abstract ideas concrete. [3]

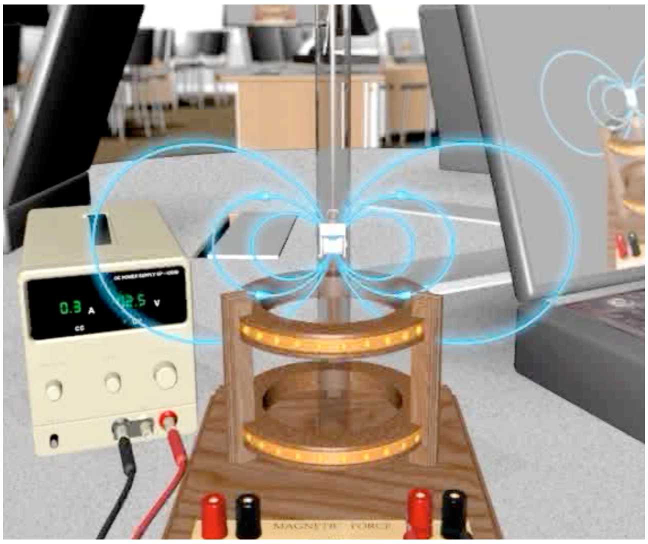

Figure 2. The *TeachSpin*™ apparatus from the Magnetostatics Session.

C. Project Wonderland

Project Wonderland is a toolkit for building 3D virtual worlds, which has been developed by Sun Microsystems Laboratories [5]. Based fully on 100% open source Java technology, Wonderland is extensible so developers and graphic artists can extend its functionalities to create new collaborative 3D virtual worlds. Wonderland also supports a high level of communication via highly immersive audio and enables desktop application sharing, among other features.

In the new redesigned version 0.5 of Wonderland, JMonkey Engine is one of the supported rendering engines. In the Wonderland project, jME helps us to specify 3D objects in a scene [5]. This concerns an object's size, appearance and how they are related to each other. Not just computer graphics operations such as lighting and texture can be done through jME, but also complex techniques like particle systems.

MTGame is a Multi-Threaded Game engine build on top of the jMonkey Engine scene graph [6]. Its main purpose is 3D graphics content processing and rendering. MTGame extends the capabilities of jME offering a fully parallel processing model and a fully featured rendering system, which supports all major rendering techniques. Besides that, three plug-able systems are included: picking and collision, physics and an input system for some input devices. So basically, MTGame is a library for scene graph management and concurrency management.

Animations and other changes, which happen in wonderland scene graph over time, are managed by MTGame. Developers can use MTGame to animate various 3D Object attributes like position, color and lighting. Further information about MTGame will be explained in the next session.

Besides the gain in performance, another important feature of version 0.5 is the possibility of embedded Swing components for user interface development in world. By creating a swing-based interface, we can interact directly with the experiment instead of interacting with a LabVIEW application through VNC viewer.

D. jMonkeyEngine

Due to the lack of graphics engines written in Java, the open source jME was built. jME is an Application Programming Interface (API) dedicated for high performance scene graph based graphics [7]. The feature list of jME is extensive and includes embedded integration of Java Applets and SWT (Standard Widget Toolkit). jME Desktop System is responsible for the rendering of Swing components in jME scenes.

III. Collarborative Virtual Learning Environment

A. Overview

In CLVE students are represented as avatars, through them users communicate with each other, cooperate with other students or educators to solve common physics problems and experience physics experiments. This Virtual Environment extends the real studio physics classroom in many ways in order to support students understanding of physics concepts.

Students are able to interact with a remote experiment through a 2D LabVIEW front panel, for example changing experiment input parameters such as amplitude or frequency. This panel has been presented via the graphical sharing desktop system Virtual Networking Computer (VNC) until now since the LabVIEW application is running remotely in the computer wired to the experiment.

Synchronously, a 3D visualization of the simulation is generated from the input and output values in the experiment equipment. For instance, in the case of the Force on a Dipole Experiment, shown in Figure 1, students are able to compare the generated magnetic field lines with the real physical experiment, which is streamed in real time through a video camera. The network camera is provided by the iLab ServiceBroker.

Due to the lack of support of swing components in the 0.4 Version of Wonderland, a button was developed to start the virtual representation of the experiment and synchronize with the real hardware.

Furthermore, a whiteboard was added to the environment to use for post experiment discussion results. Textchat functionality is also provided. These tools give students the ability to interact with each other during the experiment and discuss results of their work, as it is in a real face-to-face collaboration.

B. TEALsim Architecture

TEALSim was designed according to the principles of the classic design pattern Model-View-Controller [8]. Many classes contribute to the functioning of each of these modules. While the simulation engine represents the Model, the user interface represents the Control. The viewer and renderer constitute the View. A simulation consists of the combination of these three modules with the collection of simulated objects and user interface components specific to the simulation. All simulation elements are JavaBeans and implement the TElement interface. Interfaces provide the basic functionality of the component, leaving the customization for its implementation.

A typical basic Electromagnetic simulation will include a simulation engine, a viewer, which by default is a Java3D viewer, UI elements and other objects being simulated. TSimulation interface describes the requirements for a complete simulation and brings all these components together.

Like any other object in the world, a simulation must implement the TElement interface, which describes the most basic functionality for objects in the world. Graphical elements, control objects and physical objects such as point charge implement this interface. A TSimulation class is responsible for the entire logic in a particular simulation. All the components in the simulation are managed by instances of a class that implements the TFramework interface. SimPlayer, is an implementation of TFramework, that loads a TSimulation object and presents it to the user either as a 3D desktop Java application or applet.

The third basic component is the SimEngine, the simulation engine. Basically, SimEngine performs three actions in a loop. First, it computes variables for the current time step, then updates simulation objects to reflect these new values. Completing the loop SimEngine informs the renderer of any visual changes to the simulation. This corresponds to the main application thread for a TEALsim simulation.

The Viewer represents the rendering engine, displaying 3D simulation elements on the screen. It also manages the user interaction with the rendered images. The interface TViewer defines rendering properties and tasks including camera controls, visual effects and also maintains a list of rendered elements. Each of the rendered elements is responsible for updating the portion of the scenegraph that represents its current position and state. These were considerations for the major architecture components in TEALSim.

C. Incorporating jME in TEALSim

Basically, three TEAL packages should be redesigned: scene graph, render, and geometries.

As Java3D API, jME is also a scene-graph based rendering engine. A scene graph organizes the data in a tree structure, normally spatially. In the case of the jME, scene graph nodes can be called either Nodes or Geometries, depending if we are considering an internal node or a leaf node.

Node's relevant information, like transforms have to be considered during the integration, because Java3D uses different utilities classes. Java3D fundamental types are slightly different than jME ones. These are Matrix,Vector and Quaternion.

Vector3f, Vector2f, Quaternions, Matrices are in jME float because LWJGL only supports float. That is, our double precision values handle the application side and are converted to float at the scene graph level.

Besides other minor changes, we start to substitute Java3D native geometries through jME fundamental shapes: box, cone, cylinder, sphere, tourus.

A factory method is used to facilitate the rendering engine migration. This method just instantiates an object. They are useful here because we can leave the concrete implementation of the conversion to the factory method. In the simulation we didn't know whether to create Java3D or jME instances. Instead we can use "render" and leave the instantiation of implementation to a factory method.

D. MTGame

Although jME is a robust scene graph based rendering engine, it does not provide everything necessary for rendering 3D real time simulation in Wonderland.

MT Game is a full performance Multi-Threaded Game engine, which fulfills this need and also takes advantage of new multicore client systems. In the API hierarchy, MT Game sits on top of the jMonkey engine scene graph and the Wonderland 3D client sits on the top of MT Game.

It extends the JME features by providing a fully parallel processing model and its rendering system supports all major rendering techniques. Besides that, it offers plugable systems including picking and collision resolution and an input system for processing not just mouse and keyboard events but also other devices. In addition, we use MT Game collada model loader to import our models into Wonderland.

MT Game is designed following a component model; this enables us to dynamically add new features to new models with almost no programmer intervention. All data will get into the system through a component. In this model our base object is called the Entity object, which is simply a container object for components that are responsible for the object visual and behavior. These components are created and managed by Managers. The Worldmanager object provides the access to all four Managers in the system: Render Manger, Input Manager, CollisionManager, and Physics.

E. User Interfrace Considerations

As relevant as the graphic engine are the interface controls. Several studies have shown that not well-planned navigation, complex user action models and annoying conclusions normally slow performance in the real world in the same manner that a 3D interface does [9].

This is the reason why we redesign the environment to integrate TEAL in Wonderland. Figure 3 shows the top view of our new environment. We always keep in mind to give the user in the virtual world fast situation awareness through effective overview of the simulation and we are always engaged to provide a meaningful feedback for user actions.

The controversy over 3D versus 2D interfaces is also present in Virtual Worlds. In scientific visualization we consider 3D as necessary to the number of tasks involving continuous variables, surfaces and volumes. However for other applications, a better strategy would be to explore variable relationships in two coordinated diagrams to discover trends, outliers or gaps.

Usability testing is more than essential for successful user interface design. So, for the development of a new prototype, which will integrate all the simulations from the TEAL project, we considered findings of our latest evaluation of the proof-of-concept [10]. In this study, we could compare how effective it is to be immersed in a 3D environment for educational purposes in traditional settings, specially related to understanding physics concepts.

Figure 3. The redesigned Collaborative Virtual Learning Environment in Wonderland integrated with TEAL Simulations

We start by grouping some high level objects together. 3D Simulation visualization, the interface controls and the camera were placed next to each other to provide the students an effective overview of the situation. This organization allows students a rapid visual search of any item. We have noticed that by doing so, students could accomplish their tasks with a reduced numbers of movements and clicks, thus reducing navigation complexity. Figure 4 illustrates this organization.

By the time of the development we were not sure about the implications of implementing either 2D HUD (Heads-up Display) or 3D interfaces in Wonderland cells. On one hand, a 2D version could boost performance and remove clutter to the plan view display. On the other hand, 3DUI can look similar to real world objects leading to a highly immersive environment. Our buttons can appear raised or depressed. Students normally enjoy these interfaces, according to the students preferences documented during the evaluation. In the cognitive domain we believe that students would easily recognize and memorize these objects because they improve spatial memory use [11].

Simulation elements in TEAL can communicate with any other element in the world since all objects must implement TElement interface. Its functionality includes support of Routes and ProperyChangeEvents. In such a manner, simulation objects can communicate with UI components. A property of a simulation object can be manipulated if we connect a UI element such as a slider. In our simulations, the Wonderland swing provided API is wired to the TEAL simulation objects.

Project Wonderland HUD consists of 2D windows, which appear above the 3D scene and they are not shared with other in-world users [5]. They can be either visible or iconified. We developed customs HUDComponents (HUDButtons) using the Java(™) Swing GUI toolkit to control and change some parameters of the simulation. To display the HUD component the user should select the simulation check box menu item. Since the Wonderland client core itself uses Swing to implement HUD, both are local GUI only and not shared among the users.

Parallel to this implementation, we use the Wonderland API to create 3D objects and make them react to user input events. This interface can be manipulated by multiple Wonderland users, instead of HUD, which can only be presented to one user. For the Force on a Dipole we developed a set of buttons to control the amplitude and frequency parameters and to turn on and off the simulation and the coil on the top corner. The great advantage of having 3D interfaces is that these buttons are shared among the other Wonderland clients. By organizing all the buttons next to the experiment, we avoid the unnecessary visual clutter caused by the larger VNC panel.

To define how big the 3D buttons are and how they look, we use the jME API. For the dynamic behavior of the 3D objects, such as the button animations we use MT Game processor objects. In order to display 3D interfaces we create a RenderComponent. Otherwise nothing could be displayed. Buttons can only react to user input when they are “pickable”.

To make them pickable we need to attach to their nodes a collision component. Every time users press the button, the simulation opens a data socket to the lab view application. The TEALsim simulation engine receives the data through the socket channel and sends it back to update simulation objects to reflect these new values. As we mentioned earlier also part of the simulation engine rule is to inform the viewers rendering engine if there is any visual change necessary to be made. Having this 3D interface near the visualization minimizes the number of navigation steps for students to accomplish their tasks in this environment. This also makes the environment more game-like, being dynamic and enjoyable for students and educators.

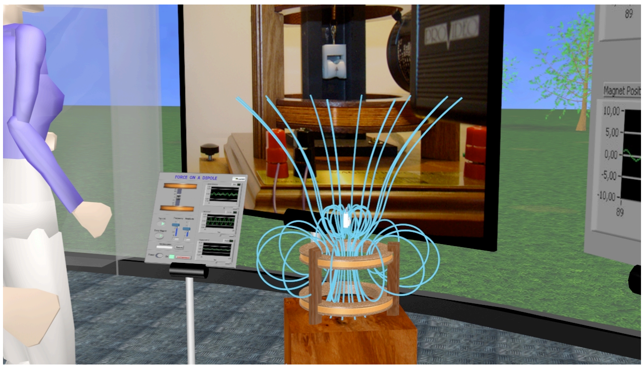

Figure 4. The redesigned Force on a Dipole Experiment in Wonderland

F. *TEALSim & Wonderland Integration*

The integration of TEALsim with Wonderland will not significantly change the process of creating simulations, but will provide Wonderland modules and cells that implement the TEAL framework. These modules will provide simulation engines that run within the Wonderland environment, render engines that execute in the Wonderland client, and manage the communication between the server and clients. In Wonderland a module is similar to a plug-in, just by including the TEALsim module and specifying the top level Java simulation class, it will be in your world.

As we mentioned, the TEALSim rendering engine is implemented as a Java3D based Viewer, which also handles the explicit rendering of the scene. Since Project Wonderland in its version 0.5 uses jME as its main rendering engine, we considered also a redesign of the TealSim 3D viewer. For many reasons, the viewer is tightly coupled to the SimEngine. Every time a simulation object has changed, the simulation engine has to inform the Viewer. Additionally, each of these elements has an associate visual representation, now as jME geometry. In the other way, when the user manipulates a visual object in the viewer, this should report back to the simulation.

IV. Current Status and Future Work

We plan to finish the TEALSim integration into Wonderland at the end of the Summer 2010. A stable version of this module should be running by April 2010. Depending on time and the level of integration of the game engine at this time, we will start to experiment with the NPC avatars or semi-autonomous tutors, making them intelligently react according to data stream.

Additionally, we are involved in the development of 3D interactive games, where students should be able to understand how magnetic forces and fields behave in nature.

V. Conclusion

This research investigates the learning benefit for students and educators by using avatars in virtual worlds for collaboration and educational undertakings. We strongly believe that CLVE plays an important role converging collaborative technologies and tools such as video, graphics and real time simulations.

Such a game like design, keeps students interest even though they are physically remote. Combined with peer cooperation, the real time visualization helps students fully understand the dynamics of electromagnetism [2]. More and more, students and educators are agreeing on the additional value of CLVE as an educational tool, fulfilling students and educators needs for an environment for remote communication and collaboration. Once the students see the same behavior at the same time, it is easier to cooperate on misunderstood concepts. Other than 2D applications, here users could explore our 3D space analyzing TEAL simulations from different locations in the room.

Moreover, the combination of a collaborative learning environment with internet-accessible iLabs is a less expensive solution for educators, because both are based in open source technology and sharing of resources. Some of the experiments and simulations could be very costly in traditional settings. Both are easy to use and are intuitive.

We are also planning to evaluate the different user interfaces approach considering student's cognitive process, relevant features and entertainment. Through such a study we can polish our design and generate new guidelines.

These are the first steps integrating TEAL Simulations in our collaborative virtual learning environment. We will continue to research ways CLVE can increase even more its pedagogical value fostering the learning process. Scientific collaboration may happen in the future primarily in Virtual Environments.

Acknowledgment

We thank the Center for Educational Computing Initiatives (CECI) at MIT for hosting and supporting this research. We also thank the faculty, staff and students involved in the TEAL Project and the team of developers from Sun Microsystems. Special thanks to Mark Bessette for Figures 3 and 4 and Tina Scheucher. This research has been partially supported by the Austrian Marshall Plan Foundation.

REFERENCES

[1] K. Jaa-aro, "Reconsidering the avatar: From user mirror to interacrtion locus" Ph.D. dissertation, KTH Numerical Analysis and Computer Science, Stockholm, Sweden 2004.

[2] B. Scheucher, P. Bailey, C. Guetl, V. Harward, " Collaborative virtual 3D environment for internet-accessible physics experiments" The International Journal on Emerging Technologies in Learning, 2009.

[3] Y.D. Dori, J Belcher, "How does technology-enabled active learning affect undergraduate student's understanding of electromagnetism concepts?" The Journal of the Learning Sciences, 14(2), pp. 243-279., 2005.

[4] V. Judson Harward et al, "iLab: A scalable architecture for sharing online experiments" International Conference on Engineering Education, Gainesville, Florida, 2004.

[5] Project Wonderland (2009)[Online]. Available: https://lg3d-wonderland.dev.java.net

[6] D.Twilleager (2008), "*MTGame Programming Guide*" Sun Microsystem. Available:https://lg3d-wonderland.dev.java.net

[7] jMonkeyEngine (2009) "*jMonkeyEngine User Guide*" Available**:** http://www.jmonkeyengine.com/

[8] J.Belcher, A. Mckinney, P. Bailey, M. Danzinger "TEALSim: A Guide to the Java 3D Software." Massachusetts Institute of Technology, Cambridge, MA, Dec. 2007.

[9] B. Schneiderman "Why not make interfaces better than 3D Reality?" *IEEE Comput. Graph. Appl.* Nov/Dec 2003.

[10] B. Scheucher, J. Belcher, P. Bailey, F. Dos Santos, C. Guetl "Evaluation results of a 3D virtual environment for internet-accessible physics experiments" *Interactive Computer Aided Learning Conference*, Villach, Austria, 2009.

[11] W. Ark et al., "Representation Matters: The effect of 3D objects and a spatial metaphor in a graphical user interface" Proc. Human -Computer Interaction Conf. People and Computers XIII, Springer Verlag, 1998, pp. 209-219.