In Proceedings of ICCE/SchoolNet 2001—Ninth International Conference on Computers in Education, Seoul, South Korea, pp. 295–302. ICCE/SchoolNet 2001.

Supporting user extensibility in a networked virtual reality environment

Chi Wan Lim[*], Yam San Chee[**] and Chit Meng Hooi[***]
School of Computing, National University of Singapore
3, Science Drive 2, Singapore 117543
Tel: +65-874-8090
Fax: +65-779-4580
[*]Email: limchiwa@comp.nus.edu.sg
[**]Email: cheeys@comp.nus.edu.sg
[***]Email: hooicm@comp.nus.edu.sg

Abstract

C-VISions is a networked multi-user 3D virtual environment that allows users to interact with each other and learn experientially and collaboratively in simulation and articulation-oriented virtual worlds. The capabilities of virtual worlds can be more fully exploited if the freedom to create new worlds is given to the users. This paper describes the development of a graphical user interface (GUI) that allows users to do this. In the context of our research, the intended users are mainly secondary school students currently in the process of acquiring new scientific knowledge. In a virtual world scenario where the virtual environment mimics the real world environment in terms of physics phenomena, users are given a free hand in constructing science experiments in whatever ways they wish. Such a learning environment allows users to freely construct and test their personal scientific hypotheses. By relying on graphical tools that aid the visualization of experiment results, the learning process is further facilitated and the learning environment can help students overcome misconceptions and enhance deep understanding of scientific principles.

Keywords: virtual reality, authoring tools, world building, science learning.

1 Introduction

Virtual worlds have long been created and used in various forms of applications, chiefly in areas like gaming and education. Other commercial usage of virtual worlds includes architectural walkthrough, virtual showrooms, and online virtual shops. Virtual worlds have the ability to transmit information in a 3D space, thus providing a greater sense of realism compared to 2-D, flat information space, such as hypermedia text. However, virtual worlds are known to be computationally expensive with respect to both hardware and software.

A virtual world has in it an environment that has a set of laws that govern the relationship between different objects that reside within it. These laws, which include physical laws, are mostly a copy or mirror of physical laws in the real world. A virtual world also provides an interface between the environment and the user who acts though his avatar. This interface usually contains a 3-D representation of the virtual world that allows the user to explore the world, communicate with other online users, and exert his influence on the environment.

The computational cost of running a virtual world has limited its capabilities in trying to mimic the real world. Very often, the virtual worlds can only try to imitate certain aspects of the real world. For example, gaming virtual worlds often place considerable emphasis on the graphical aspects of the environment, such as texture mapping and rendering speed, while the objects shown in the world have very specific usage or very little significance attached to them.

In educational virtual world applications, however, the emphasis shifts to the objects that inhabit the worlds. Generally, the variety of the type of objects increases. Properties and behaviors are contained within the objects. Complex objects have special properties inherent within them while sharing more general properties like weight or mass with other simpler objects. Generally, objects in these worlds can be arranged in a hierarchical structure.

The attraction of virtual worlds lies in their ability to simulate phenomena in real life while providing users with the ability to vary certain object attributes. This kind of usage has found its niche in the education field, where the conduct of experiments is an essential component. The main characteristic of experiments is to establish the relationship between certain variables. However, with the presence of factors such as friction and air-resistance in the domain of physics, the underlying scientific relationship might be clouded to an untrained eye.

With the foregoing in mind, the acquisition of scientific knowledge can be more readily achieved if users are allowed to conduct experiments with virtual objects in a virtual world setting and to observe the outcome of the experiments. The virtual world will realistically simulate physical world conditions. Object attributes and variables can be easily varied or controlled by the user, something that, in real life, might not be possible to the typical user. For instance, having ten balls, each with a different weight, roll down a slope might be physically manageable for a user, but having to measure the results of the experiment as well, would be difficult in practice, given the constraints on an average user. However, the use of virtual world technology makes this possible.

In Section 2 of this paper we introduce the main virtual world system, C-VISions. In Section 3, we describe the world builder tool, VEditor. In Section 4, we outline how VEditor can be used to design virtual world experiments. In Section 5, we illustrate the connection between VEditor's architecture and it's usefulness in helping a user to construct and understand science experiments. Section 6 concludes the paper.

2 C-VISions

C-VISions is a virtual reality research project developed by the Learning Environments and Learning Science Laboratory in the National University of Singapore. It focuses on the creation of a networked multi-user 3D virtual environment where students from any part of the world can learn experientially and collaboratively in simulation- and articulation-oriented virtual worlds.

These worlds are accessed via a virtual world browser that renders the world in a 3-D view. In these worlds, science experiments are set up, and users, represented in the form of avatars, are allowed to traverse through the world and explore both the world itself as well as other science experiments that have been set up beforehand. Users are able to navigate through the world, viewing it in either a first-person or third-person perspective. Interaction among users in the world can occur either through a text-based chat facility or an audio-based chat.

The object hierarchy in C-VISions is coded within Vtalk. Vtalk has been designed and engineered using object oriented methodology. It is coded in Java, thus allowing the objects to be readily extensible and maintainable. New object classes can be easily incorporated into the object hierarchy by sub classing, and newly constructed objects can be readily used by C-VISions.

A graphics browser that has been developed using Java3D performs the rendering of the virtual world. A separate network component handles the transferring of information among the connected clients. These three main components form the backbone of C-VISions.

2.1 Virtual Objects

Virtual objects, known in C-VISions as VObjects, possess certain physical attributes encapsulated within them. These attributes provide the basis for the experiments hosted in the virtual worlds. These physical attributes can be altered and changed by users. The attributes can even be modified to generate novel objects that have no real world counterparts. For example, a cannonball can be given a rubber texture, turning it effectively into a bouncing cannonball.

In the experiments that occur in the virtual worlds, the attribute values of the VObjects are used in the computation of behaviors and in the resolution of forces. This flexibility allows the creation of highly dynamic user controlled experiments. The types of experiments possible are limited only by the creative invention of the users.

3 VEditor

VEditor has been developed as a C-VISions complementary tool to allow end users to create new virtual worlds by reusing existing system objects. VEditor presents a GUI that allows users to build virtual worlds that can be readily used by C-VISions, without users having to work with Java, Vtalk, or VBrowser. The graphical representation of VObjects takes the form of 3D VRML models. VEditor provides the functionalities to manipulate these 3D VRML models in a 3D graphical window using planar navigational tools.

The interface of VEditor consists of: (1) a 3D graphical panel displaying a 3D view of the current world, (2) two 2D panels showing a representation of the 3D world in 2D format, (3) a navigation panel to assist the user in navigating through the virtual world, and (4) tabbed panels providing different details concerned with the hierarchical structure of the VObjects. (See Figure 1).

The next section will discuss the various implementation and interfacing details concerning VEditor.

3.1 3D Graphical Window

The main focus of every virtual reality interface is the graphical panel rendering the 3D virtual world. In VEditor, the 3D graphical panel not only acts as a window through which the user can view the virtual world, but it also has navigational functions and other features embedded within it.

Direct Drag and Drop

User interaction by drag and drop has become fairly standard in window-based computer interface operations. VEditor allows users to add new VObjects into the world using drag and drop interaction. VEditor has a tabbed panel that contains all available VObjects that can be used in virtual worlds. Users can click on the image of a VObject from the tabbed panel and drag it across to the graphical panel and drop the object, effectively adding that object to the virtual world.

Traversing through the Virtual World

The mouse serves as an input device that is used to navigate though the virtual world. This approach to user navigation is seen in many virtual reality applications including ActiveWorlds^{TM} and also Internet Space Builder from Parallel Graphics^{TM}.

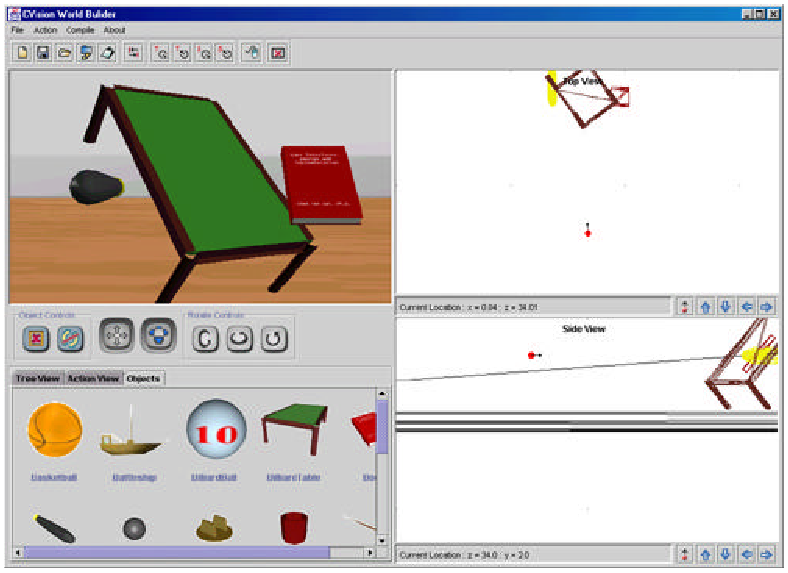


Figure 1: Overall View of VEditor

Manipulation of VObjects

1) Rotational manipulation

The user can rotate the virtual world object in three directions: the x-axis, the y-axis, and the z-axis. These axes are relative to the virtual world object itself, not the axes of the virtual world. To rotate the desired virtual world object, the user has to first select the object, by double clicking on it. Upon selection, a gray rectangular box will encase the virtual world object, signifying that the object has been selected. (See Figure 2)

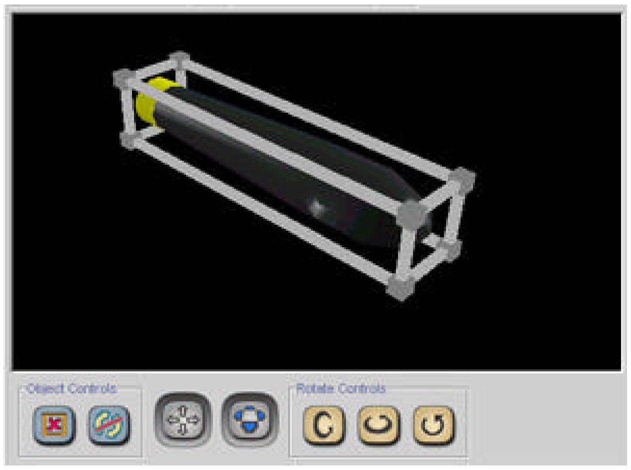


Figure 2. Encapsulation of a virtual world object by a rectangular box.

2) Hierarchical manipulation

A virtual world object can be the "parent" or "child" of another virtual world object. When an object is the "child" of another object (the "parent"), its location and orientation will be relative to the parent object. This parent-child structure is very similar to the scene structure of Java3D, and it supports the construction and manipulation of the Java3D scene graph.

3.2 2D Graphical Panels

Multiple views of a 3D virtual world aid in the visual understanding and construction of the world. Java3D (Nadeau 1999) is used to render the 3D view of the world. By passing the VRML models to Java3D and specifying the location coordinates of the object, the visual representation is rendered in the 3D space. In the 2D view, the wireframe rendering must be developed separately, and this is described below.

Generating Wireframes Models of Virtual World Objects

The original virtual world object models are in VRML format. These VRML files are converted into a scene graph in Java3D^{TM}. The wireframe models of the virtual world objects are be generated by dissecting the scene graphs from the VRML model. (See Figure 3)

Changing the Location of the Virtual World Object

The manipulation of the virtual world object in the 3D graphical window is limited only to the rotation of the objects. It is difficult to change the location of the virtual world object visually because the positioning tool (mouse or keyboard) is in 2D. Hence, the object positioning manipulation is enabled by using a combination of two 2D windows, each of which projects a different 2D view of the world.

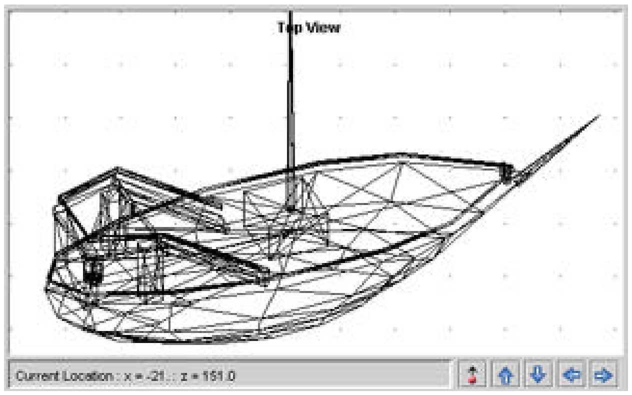


Figure 3. Rendering of the wire frame model on the 2D panel.

3.3 Navigation Panel

The navigation panel is used in combination with the 3D graphical panel. The navigation panel is divided into three different sections: object control, navigation control, and rotate controls.

Object Control has two buttons in it, *Delete* and *Detach*, and these are used in conjunction with the 3D graphical panel. By selecting a virtual world object in the 3D graphical window, the user can choose either to remove the object from the virtual world, or if the object has a parent object, detach it from the parent object.

Navigation Control has two buttons contained within it. One button allows the user to control the alteration of the orientation of the viewpoint. The other button allows the users' point of view to be shifted in four directions; up, down, leftwards, and rightwards (without changing the view orientation).

Rotation Control provides three buttons to allow rotation of the virtual world objects in any of the x-, y-, and z-axes.

3.4 Tabbed Pane Controls

Because C-VISions is essentially an experiment simulation application, it must implement some mechanisms to set the attributes and behavior of the virtual world object. There are three panels in the tabbed pane -- the tree view, the action view and the object panel -- to facilitate this.

The *Tree View* pane displays a hierarchical view of the object in the VWorld. This panel also allows the user modify the object's hierarchy. Virtual world objects can be detached from their parent object and moved to different parts of the object tree.

This *Action View* pane shows a list of actions currently defined by the user. From this pane, the users can make alteration to the existing actions that have been created, or create entirely new actions. Other than setting the attributes, the user can also use the behaviors that are already defined in C-VISions. Such behaviors simulate natural behaviors found in real life, such as *move* and *freefall.*

An action in the context of this project refers to an invocation of a change of attribute in either one or a group of virtual world objects.

The O*bject View* panel contains all the virtual world objects that can be utilized by users for the construction of their own simulations. Users can select and drag the virtual world object from the object panel and drop them into the 3D graphical panel for reuse.

4 Experiment Design

The main functionality of VEditor lies in its experiment designing capability. The previous section of this paper described the object manipulation capabilities of VEditor. These capabilities are tools to help users to visualize and construct their own experiments. Simple actions such as placing an object on a table might be an easy task in real life, but to be able to do the same in virtual space requires a carefully designed user interface.

With the VEditor functions that we have created, users are empowered to design their own experiments and to deploy them in C-VISions.

4.1 Specifying a Virtual Object's Attributes

Experiments are often concerned with establishing the relationship between different variables. For instance, the weight of an object against the speed of the object rolling down a slope.

VEditor possesses the functionality to allow the attribute values of each of the virtual world objects to be changed. These attribute values are taken into account when computing the dynamic behavior of objects during the running of the experiment. C-VISions does not predefine "canned" results; rather, behaviors are computed and rendered on the fly using underlying simulation parameters. This approach allows a broad range of dynamic behaviors to be produced.

4.2 Designing Experiments

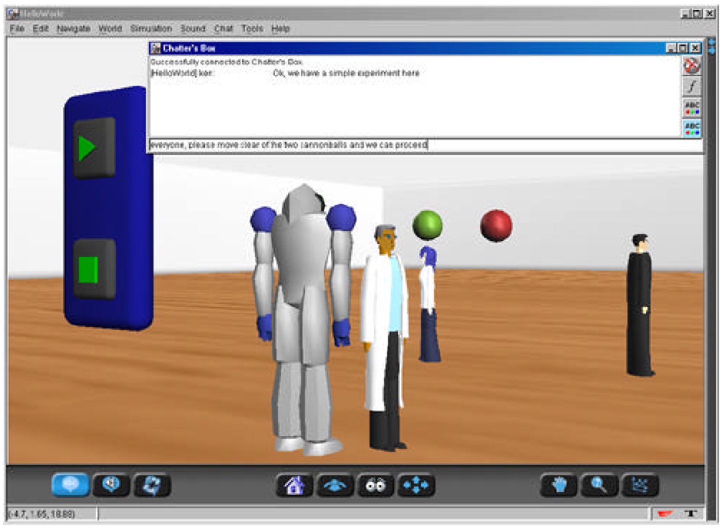
Experiments created using VEditor execute on the basis of triggered events. A specialized virtual object allows users to control experiments. This virtual object, called *SimulationControl*, has a *Start* and a *Reset* button on it. These buttons effectively start and reset the experiments, allowing the user to repeat the experiments.

Experiment phenomena are mimicked by changing one or more attribute values. For instance, to get a ball to start rolling, a velocity is applied to the ball by pressing the *Start* button of the *SimulationControl*. However, to have to setup an experiment entirely in terms of discrete attribute value settings may be somewhat difficult for a secondary school user. Therefore, behaviors are created in C-VISions to make the designing of experiments more user friendly. In order to specify that an object should drop to the floor under the force of gravity, the user needs only to specify the *free fall* behavior, rather than having to specify a downward acceleration attribute value.

Attributes changes can be combined to produce more complex effects. A forward velocity change, coupled with a "free-fall" behavior will result in the simulation of an object that has been thrown forward. Together with the ability to place multiple *SimulationControl* virtual objects in the world, where each object manipulates a different aspect of the experiment, the combinations possible are limitless. The freedom to readily combine multiple behaviors to form more complex ones allows users to freely devise experiments to test their hypotheses of physics phenomena.

5 Usefulness to the Learning Process

In addition to rendering realistic simulation results, C-VISions also allows users to reflect on observed simulation behavior by the process of visualization tools. Graphs can be plotted for object behavior based on attributes of the virtual objects that the user has selected. Taking the experiment of throwing a ball (using the application of free fall behavior with a forward velocity) as an example, the trajectory of the ball can be graphed according to various parameters, such as horizontal velocity, displacement and acceleration. Such data visualization allows users to relate the observations produced by the simulation to the actual mathematical data.



Successfully connected to Chatter's Box
[HelloWorld] ker: Ok, we have a simple experiment here
everyone, please move clear of the two cannonballs and we can proceed

Figure 4: Screen shot of a newly created experiment.

Continuing with the example of the thrown ball, the user will observe that the ball will move in the forward direction, bouncing when it hits the ground, and finally coming to rest at some distance away. With the visualization tools, the student will thus come to realize that it was the change in the vertical velocity that caused the ball to stop, rather than a change in the horizontal velocity. (See Figure 4)

Such information visualization tools allow users to delve into the deeper components underlying an observed simulation. Repeated experimentation will aid students in the conceptualization of scientific concepts. C-VISions also supports text-based and audio chat between users that is connected to the server. By virtue of the fact that C-VISions is a networked virtual reality system, many users can view and share the experience of common experiments at the same time. Every user's perception of the experiment might be different. Through the use of these interaction facilities, ideas and viewpoints can be freely exchanged and constructively discussed.

6 Conclusion

In this paper, we have explained how C-VISions serves as a shared virtual world environment to facilitate experiential, collaborative learning between distributed users. We have also provided a detailed description of VEditor, a complementary tool that allows users to build their own virtual world simulations by reusing world objects previously developed for the C-VISions system. The VEditor tool also empowers end users of C-VISions by allowing them to create a wide, almost limitless, set of rich simulations without the need to engage in any program coding. In this way, they can create simulations to test out their own specific hypotheses and experiment continually with "what if" changes to the object attributes and behaviors in the simulation world. Deeper concept understanding is facilitated by the provision of visualization tools that represent the observed simulation behavior. Support for network-based communication with other learners spread throughout the globe provides a channel for peer-to-peer collaborative learning and community knowledge building.

7 References

Activeworlds.com, Inc (2000). http://www.activeworlds.com

Cable, L. P. G. (1999) Drag and Drop subsystem for Java^{TM} Foundation Classes. Version 0.96. http://java.sun.com/products/javabeans/glasgow/index.html

Fosnot, C. (1996) *Constructivism: Theory, Perspectives, and Practice*. NY: Teachers College Press.

Khoo, Y. B., Chee Y. S. (2000) "Designing extensible simulation-oriented collaborative virtual learning environments" In proceedings of ICCE/ICCAI 2000-Eighth International Conference on Computers in Education/International Conference on Computer-Assisted Instruction, Taipei, Taiwan, pp. 222-230.

Learning Environment and Learning Science Laboratory (2001) C-VISions, Collaborative Virtual Interactive Simulations. http://cvis.ddns.comp.nus.edu.sg

Lim, K. P. E. (1997) VR for Learning, National University of Singapore, Department of Information Systems and Computer Science.

Nadeau, D. D. (1999) Building virtual worlds with VRML, *IEEE Computer Graphics and Applications*, March pp 18-29.

Paelke V. (2000) System Design of Interactive Illustration Techniques for User Guidance in Virtual Environments. IEEE VR, New Brunswick, NJ, March, pp 207-214.

Parallel Graphics (2000) Internet World Builder. http://www.parallelgraphics.com/products/isb

Singhal, S. & Zyda, M. (1999) *Networked Virtual Environments*, NY: ACM Press.

Sun Microsystems, Inc. The Java3D^{TM} API specifications. J3D version 1.2. (2000). http://www.java.sun.com/products/java-media/3D/download.html