Collaborative Virtual Environment for Industrial Training and e-Commerce

J.C.OLIVEIRA, X.SHEN AND N.D.GEORGANAS
School of Information Technology and Engineering
Multimedia Communications Research Laboratory
University of Ottawa
161 Louis Pasteur Priv., Ottawa, ON K1S 5J6
CANADA
jauvane|shen|georganas@mcrlab.uottawa.ca http://www.mcrlab.uottawa.ca

Abstract: Collaborative Virtual Environment concepts have been used in many systems in the past few years. Applications of such technology range from military combat simulations to various civilian commercial applications. This paper presents CVE as a medium for Industrial Training and Electronic Commerce.

Key-Words: Collaborative Virtual Environments, Virtual Reality, Collaboration, Java3D, VRML, Multimedia, Industrial Training, E-Commerce.

1 Introduction

Over the past few years, a number of interactive virtual reality (VR) systems have been developed. A Collaborative Virtual Environment (CVE) is a special case of a VR system where the emphasis is more on "collaboration between users" rather than on simulation. CVEs are used for applications such as collaborative design, training, telepresence, and tele-robotics.

We have exploited CVEs for Industrial Tele-Training aiming at reducing expenses and saving time while training remote users/engineers to operate some equipment. We also have been exploiting CVE for Electronic Commerce aiming at providing the user with a much enjoyable experience while shopping online. In both cases the users, represented by avatars, can join a Virtual World and then manipulate and interact with the objects as in the real world. For training purposes the users are expected to perform some tasks under supervision of a trainer (or unattended) while the e-Commerce client would be able to see a model of the products he/she wishes to buy. If questions need to be answered an Intelligent Agent may pop-up and assist the user with detailed information and walk through features of the virtual objects. One can extend the Virtual World to be similar to a real shopping mall, which would facilitate the adaptation of users not used to browser-based interface.

Motivated by these advantages, we have designed and implemented a few prototypes in those two areas, namely an Industrial Teletraining prototype for ATM switching equipment, as well as a Virtual Shopping Mall. Section 2 focuses on the Industrial Training prototype, while section 3 focuses on the e-Commerce one.

2 Industrial Training

Our prototype is a multiuser teletraining application, which allows users, represented by avatars, to learn how to operate on a faulty ATM switch. The avatars repair the switch in steps which precisely reflect those necessary to perform the same actions in the real world. The prototype consists of two general modules: user interface, and network communication. The user interface itself consists of a graphical interface (GUI), a 3D interface (VR), and media interfaces (speech recognition, voice streaming, head tracking).

The upper right area of the interface, which takes the largest part, is the *3D environment*. On the left and below the 3D environment are the *controls* used by the trainees to interact with objects and navigate in the environment. At the top left is the *head-tracking facility*. Below the head-tracking window is a *utility panel* that is used for different purposes as discussed later. There is also a *chat space* where users can exchange textual messages. In order to avoid navigation problems with inexperienced users, it is

possible to view the world from a set of predefined camera views.

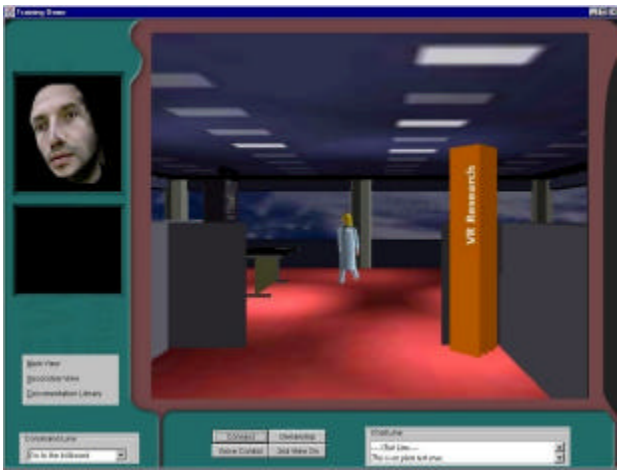

Figure 1. Training Application's Interface

A user is able to approach and verify the operation of the switch and its cards, remove a faulty card and put it on the repair table, and replace it by installing a new card into the switch. Other parties will be able to watch that user's avatar taking such actions. All of the above actions can be performed by directly navigating in the scene and manipulating objects with the mouse or by selecting the action in a menu.

A user can also view video segments showing the correct procedure for performing certain actions. The utility panel is used to display the video clips. If chosen by the trainer, the video will be displayed in every participant's screen. Another use for the utility panel is the *secondary-view* feature, where a user can simultaneously watch the current actions from an alternative point of view.

In addition to the above explained interfaces, the prototype offers voice recognition technology whereby the user simply enters commands by talking into the computer's microphone. In this case, the user may simply say pre-defined commands such as "Go to the table" for the avatar to perform, or may change views by saying "Table Top View", and so on.

2.1. Architecture & Components

Rather than developing the system from scratch, we decided to make use of a wide spectrum of available technologies, reuse software packages, and concentrate our efforts in integrating them. Hence, the system brings together a large number of components and technologies as shown in Figure 3, namely for 2D graphics and interface design, 3D rendering, multi-user communications, voice recognition, audio and video streaming, and head tracking. In this section we will briefly describe the prototype architecture and the role of each component. Such architecture is shown in Figure 2.

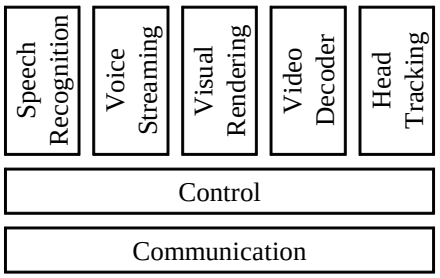

Figure 2. System Architecture

The communication layer is responsible for all exchange of information amongst users. There are three components in this layer: directory server, control communication, and media communication layers. The *directory server* is an entity which holds information about the participants in a session. The *control communication layer* is responsible for all communication with exception of audio data. The directory server has been written in ANSI C and the client side of the control communication in C++ resulting in a Dynamically-Linked Library (DLL) to be loaded at run time. There are IPv4 and IPv6 implementations available. The communication layer decouples all networking issues from the upper layers. The native-platform communication DLL is loaded into the Java environment using the Java Native Interface (JNI). Incoming messages are sent to the Java core via callback of Java methods from the native code.

For the GUI part, we used JavaSwing. Java technology was chosen due to its easy utilization of high-level APIs. Our prototype makes use of VRML objects and the VRML behavior of the objects is converted to Java3D code using VRML to Java3D conversion tools. A view of a world created in this manner is then presented in a Canvas3D. Hot spots are defined in the world in order to allow the user to interact with them in a point-and-click fashion. User actions are sent to the communication layer, which in turn informs the other parties involved in or affected by such actions, creating thus a collaborative environment. Figure 3 shows the various components of the system along with communication among them.

We use the Microsoft Speech API (SAPI) to provide a speaker independent recognition of pre-defined commands described by a SAPI grammar. This way commands such as "go to the table", "pick up the card", are recognized by the SAPI module. The speech recognition module is implemented using ActiveX components and its integration to the prototype is made via another native DLL loaded by JNI.

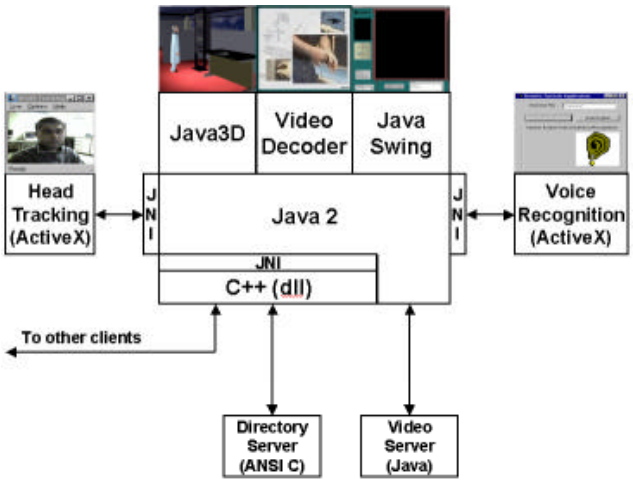

Figure 3. Prototype Modules/Connection

The audio capturing module allows participants to enter into an audio-conferencing session. The module is based on the Microsoft NetMeeting SDK.

The Video Decoder module is an H.263 Video Decoder developed entirely in Java which is able to receive and decode streamed video from its server.

The head-tracking module captures the user's head motion wirelessly by video processing techniques using a simple camera installed on the user's computer. Head movements are sent to the prototype and are used to control the corresponding avatar's head. The head-tracking module is implemented using ActiveX components which generate a series of rotation parameters which are sent to the prototype via a JNI connection through yet another DLL.

The control layer is implicitly included in the other layers and is composed of a set of rules which ensures that all components work together. The coordination of the DLLs, which enable the exchange of data between the Java core of the prototype and the native components, comprises the Control Layer

A later enhancement to this prototype was the possibility to have the same environment in an more immersive fashion. Figure 4 shows the immersive version of this prototype projected in a wall as well as in a computer' screen. The immersive version of the prototype benefits from alternative input methods such as voice recognition and head-tracking previously introduced.

Figure 4. Immersive version of Prototype

This Prototype shows the potential that CVE has for Industrial training.

3 e-Commerce

The project aims at designing and implementing a prototype user interface for electronic commerce in a Distributed Virtual Environment (DVE) on the Internet supported by intelligent agents.

Existing electronic commerce applications only provide the user with a relatively simple browser-based interface to access the available products. Buyers, however, are not provided with the same shopping experience, as they would have in an actual store or shopping mall. With the creation of a virtual shopping mall, simulations of most of the actual shopping environments and user interactions can be achieved. The virtual mall brings together the services and inventories of various vendors. Users can either navigate through the vendors, adding items into a virtual shopping cart, or perform intelligent searches through "user and vendor agents". The user may find items of interest in the normal navigation process or use the search agent without having to navigate through the maze.

A user uses a web browser, logs into the URL of a database of virtual worlds/catalogs and, following proper authentication/access control, retrieves the virtual world of interest realized in VRML97, supported by a free browser plugin (e.g., Cosmo Player). He/she then can use a DVE user interface to navigate the virtual world, where both the users and agents are represented on the screen by 3-D animator avatars (See Figure 5). The VRML-enabled browser

uses Java as its scripting language. Initially, a VRML file is downloaded to the local browser who renders its contents. The script nodes in the VRML scene point either to a local Java script, or to Java scripts on remote servers. Scripts are able to manipulate scene graph nodes by generating events that are delivered to the node and change one or more of it properties; for example, its position in the scene, its shape, or one of its material attributes. Obviously, since the scripts are fully functional Java codes, they are not restricted to just changing the scene graph. They can, for example, dynamically generate additional VRML nodes, or locate and add existing VRML to the base scene downloaded in the original VRML file. For communication between a VRML world and its external program (referred to here as a Java applet), an interface between the two called External Authoring Interface (EAI) is needed. This interface defines the set of functions on the VRML browser so that the external environment can perform to affect the virtual world.

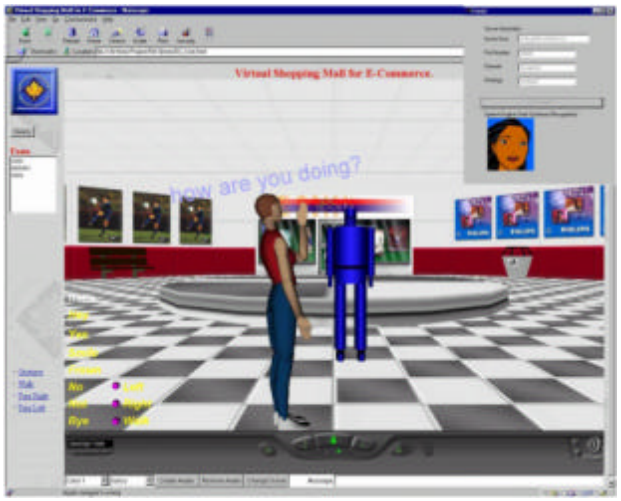

Figure 5. e-Commerce Application's Interface

3.1 Intelligent Agents

In the context of a public virtual mall with multiple independent vendors, our goal is to create a software entity (which we call Sales Agent or SA) that will represent each of these vendors. More specifically, we are interested in enabling the consumer to negotiate the price of what he/she is buying by making a binding offer to the SA. The SA is able to respond to such consumer's offer by making use of decision rules based on private information (such as cost prices and inventory numbers) not available in the public catalogue of the mall. A response consists of either an acceptance or rejection of the offer. In case of acceptance, the user is notified and the transaction is recorded at the consumer's offered price. In case of refusal, the user is also notified and the SA has the liberty of making a counter-offer or not. If a counter-offer is made and accepted by the consumer, the SA records the transaction at the counter-offer price.

Our architecture is that of a Multi-Agent System (MAS). Each vendor in the mall has one instance of a SA. Consumers are represented by User Interface Agents. Each instance of an agent is a separate entity and is running asynchronously in its own process and possibly its own machine (SAs are assumed to be continuously running).

Communication between agents is completed through a message passing system that uses a broker agent with name resolution service to register agents. The broker agent is also considered an agent and run in a predefined machine at a predefined port. It is responsible for registering agents and relaying messages to the appropriate agent. Each SA is identified by the unique name of the vendor it represents (e.g. Sears) and each consumer is identified by the unique user ID provided upon registration in the virtual mall.

The electronic mall prototype also allows the user to communicate with an "intelligent assistant" (IA) using simple voice commands. This assistant interacts with the shopper using voice synthesis and helps him or her use the interface to navigate efficiently in the mall. Voice-enabled agents provide users with the friendly speech-command interface, and also act as a general "help" facility for the user interface, accepting simple voice commands and giving voice responses.

3.2 RTI/HLA

Real-time interactions among entities in the virtual environment are implemented over the Run Time Infrastructure of High Level Architecture (RTI/HLA), an OMG and IEEE standard for distributed simulations. This High Level Architecture (HLA) standard is a general architecture for simulation reuse and interoperability developed by the US Department of Defense.

The HLA standard promotes reuse of simulations and their components. It attempts to specify the general structure of the interfaces between simulations without making specific demands on the implementation of each simulation.

References

[1] J. Leigh, "A Review of Tele-Immersive Applications in the CAVE Research Network", Procedings of the IEEE International Conference on Virtual Reality, Texas, March 1999.

[2] J. C. de Oliveira; S. Shirmohammadi and N. D. Georganas, "Collaborative Virtual Environments Standards: A Performance Evaluation", IEEE DiS-RT'99, Greenbelt, MD, October 1999.

[3] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland "Visually controlled graphics", IEEE Trans. Pattern Analysis and Machine Intelligence, 15(6): pages 602-605, June 1993.

[4] T.J. Broida and R. Chellappa "Estimation of object motion parameters from noisy images", IEEE Trans. Pattern Analysis and Machine Intelligence, 8(1): pages 90-99, January 1986.

[5] D.B. Gennery. "Visual tracking of known 3-dimensional object", Int. J. of Computer Vision, 7(3), pages 243–270, 1992.

[6] A. Azarbayejani and A Pentland "Recursive estimation of motion, structure, and focal length", IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(6), 1995.

[7] K. Shoemake. "Quaternions", Department of Computer and Information Science University of Pennsylvania Philadelphia, PA 19104.

[8] J. Shi, and C. Tomasi, "Good Features to Track", IEEE Conf. on Computer Vision and Pattern Recognition (CVPR94) Seattle, June 1994.

[9] K. Aizawa, T.S. Huang, "Model Based Image Coding: Advanced Video Coding Techniques for very Low Bit-Rate Applications", Proc. IEEE, vol. 3, No. 2, Feb. 1995.

[10] X. Shen, R. Hage and N.D.Georganas, "Agent-aided Collaborative Virtual Environments over HLA/RTI", IEEE DiS-RT'99, Greenbelt, MD, October 1999.

[11] HLA Homepage, URL at http://hla.dmso.mil/hla

[12] E. T. Powell, "The Use of Multicast and Interest Management in DIS and HLA Applications" Proceedings of the 15th DIS Workshop, #96-14-125, September 1996.