IMPLEMENTING VIRTUAL REALITY MODELING LANGUAGE (VRML) TO CONVEY SIMULATION INFORMATION

Robert Anschuetz (Veridian), Charlie Jones (Veridian),
Mike Garnsey (STRICOM), Paul Dumanoir (STRICOM)

Veridian
3452 Lake Lynda Drive, Suite 215
Orlando, FL 32817
407/658-0044
{anschuer|jonesc}@orlando.veda.com

STRICOM
12350 Research Parkway
Orlando, FL 32826-3275
407/380-4340
{Mike_Garnsey|Paul_Dumanoir}@stricom.army.mil

ABSTRACT

The U.S. Army's Simulation Training and Instrumentation Command (STRICOM) has sponsored an initial effort to evaluate the applicability of utilizing the Virtual Reality Modeling Language (VRML) technology in the Modeling and Simulation (M&S) community. VRML is a platform-independent, graphical programming language used to generate images across the World Wide Web. In a sense, VRML is a three- dimensional extension to the HyperText Markup Language (HTML). Several M&S applications have been demonstrated thus far, with the most ambitious being a VRML stealth viewer that combines the technologies of VRML, HTML, Java, C++, DIS, HLA, and client/server communication.

AUTHOR BIOGRAPHIES

Mr. Robert Anschuetz is a senior software engineer for Veridian and is the lead engineer for the VRML-Enabled M&S Information Dissemination R&D project. Mr. Anschuetz holds a Master of Science Degree in Mathematics from the University of Central Florida and a Bachelor of Science Degree in Computer Science from Eastern Michigan University. Mr. Anschuetz has over 10 years experience in military simulation programs.

Mr. Charlie Jones is a visual systems engineer for Veridian and is currently assigned to the VRML-Enabled M&S Information Dissemination R&D project. Mr. Jones holds a Bachelor of Science Degree in Computer Science from the University of Central Florida and a Bachelor of Science Degree in Electronics Engineering Technology from the University of Southern Mississippi. Mr. Jones also has four years of military experience with the United States Navy.

Mr. Mike Garnsey is a principle investigator in the Synthetic Environment & Technology Management Division at STRICOM and is the government project manager on the VRML–Enabled M&S Information Dissemination R&D project. Mr. Garnsey has over seven years of experience in military simulation networking, protocol, and security architecture implementation; simulation system visualization research; and intelligent simulation agent/actor technologies.

Mr. Paul Dumanoir is a systems engineer at the STRICOM Engineering Directorate, Modeling & Simulation Technology Division. He is currently the DISAF / DWN project engineer. Prior to his involvement with DWN, he worked as software and systems engineer on the CCTT and WARSIM programs. His interests include dismounted infantry computer generated forces and ModSAF. Mr. Dumanoir earned a Bachelor of Science Degree in Electrical Engineering from the University of South Alabama and a Master of Science Degree in Computer Systems from the University of Central Florida.

IMPLEMENTING VIRTUAL REALITY MODELING LANGUAGE (VRML) TO CONVEY SIMULATION INFORMATION

INTRODUCTION

The U.S. Army's Simulation Training and Instrumentation Command (STRICOM) and Veridian are exploring potential applications of the Virtual Reality Modeling Language (VRML) for use in the Modeling and Simulation (M&S) domain. This paper covers the current ongoing Feasibility Analysis Study (FAS) concentrating on the use of VRML in three areas: (1) as a tool to enhance the visualization capabilities of Modular Semi-Automated Forces (ModSAF), (2) as an aid to navigating the Functional Description of the Battlespace (FDB) internet-hosted database, and (3) as a generic platform-independent distributed simulation stealth viewer.

VRML BACKGROUND

VRML is an open graphical modeling language that is designed to depict three-dimensional environments across the World Wide Web. Much like HyperText Markup Language (HTML) documents, VRML files are human-readable, interpreted source code viewed using a VRML browser. VRML browsers are typically implemented as plug-ins to Internet HTML viewers such as Netscape Navigator or Microsoft Internet Explorer.

Since VRML is not compiled or specifically tailored to a specific computer or graphics board, it is not as fast as other graphical APIs such as OpenGL, Direct3D, or Performer. However, where VRML shines is in platform independence and in distributed graphics processing over the Internet. VRML code is easy to write and has powerful graphics building blocks built into the language's keyword list.

M&S VRML STUDY -- OVERVIEW

STRICOM and Veridian staff members have focused on the following M&S areas to explore VRML's applicability in regards to being: (1) a tool to enhance the visualization capabilities of ModSAF, (2) an aid to navigating the FDB internet-hosted database, and (3) a generic platform-independent distributed simulation stealth viewer.

1st Focus Area: VRML-Enhanced ModSAF

VRML was used to enhance the prototype work that Nations had done in providing HTML links to the ModSAF application. The HTML approach extends the functionality of ModSAF so that when a user selects an entity (initially limited to an AH-64D helicopter or an M1A1 tank) in ModSAF's plan view display, a hyper-linked page describing the entity is displayed within an Internet HTML browser on the same workstation. The obvious VRML extension to this prototype effort was to allow the user the ability to view the entity in three dimensions. Indeed, this capability was explored and somewhat implemented, but some difficulties occurred in the process.

Three-dimensional VRML models of the M1A1 and AH64 vehicles were created, and the appropriate links were made from the HTML pages generated for those entities. However, the prototype ModSAF version that supported hyperlinking to HTML pages only worked on computers running the Linux operating system. Unfortunately, sufficient VRML browsers had not yet been developed for Linux at the time of the effort. So, while the VRML concept could be demonstrated offline on a separate PC with VRML browser software, an integrated VRML-ModSAF demonstration capability was not fully realized.

Several other ideas have been discussed for future ModSAF work, including finishing the implementation of the hyper-linked three-dimensional entity information, providing a three-dimensional courseware package for the ModSAF user and developer, and providing three-dimensional previews of entity movements.

2nd Focus Area: VRML-Enhanced FDB

VRML was also used to enhance the FDB system with three-dimensional support. The FDB is a repository for simulation developers to access simulation information such as documents, algorithms, and vehicle specifications. The FDB is accessible through password control at **Error! Bookmark not defined.**.

Some of the data currently residing on the FDB cannot be easily represented in the two-dimensional world of HTML. Using a three-dimensional representation of the data enables the user to visualize several objects of the FDB much more intuitively. Demonstrating key aspects of the FDB with VRML focuses more on a user-based, rather than content-based, approach

distribution. For example, the FDB contains a document repository that represents several categories of documents. The document repository has been represented in VRML as a "virtual library" with bookshelves housing different types of documents. The user is able to traverse down the isles of the library and select a document much like a person would in the real world.

Several other VRML applications for the FDB will be explored, including allowing the user to preview a terrain database before downloading, enabling the user to view a vehicle and its parts in three-dimensions, and providing a three-dimensional tutorial of the FDB's purpose and content.

3rd Focus Area: VRML Stealth Viewer

The rest of this paper will be devoted to the discussion of the development of a DIS/HLA compliant VRML stealth viewer. The impetus for developing this VRML stealth viewer was to determine and demonstrate the strengths and weaknesses of VRML as a three-dimensional visualization tool for distributed simulation exercises. Based on insights gained from the first two focus areas, the project team set out to develop a VRML stealth that would minimally provide the user the ability to navigate around a simulation database, view live entity updates, and allow the user to hyperlink from certain features or objects within the database to areas which would provide additional information. Most importantly, the computer that ran the VRML stealth application would not have to be pre-loaded with any simulation software or databases, and could be any computer platform outfitted with an Internet browser and VRML plug-in.

But why use VRML? As indicated earlier, VRML is not intended for dynamic rendering of complex "out-the-window" views at high frame update rates. However, even with the current generation of non-optimized VRML browsers, three-dimensional navigation through fairly large and complex VRML virtual environments is achievable with marginal but adequate performance on personal computers. Additionally, VRML provides compelling support for attaching or 'hyper-linking' augmented content to any arbitrary three-dimensional object or feature in the displayed view. Finally, while features such as platform independence, three-dimensional graphics, and internet-friendly user interface are not the unique domain of VRML, it is important to note that such capabilities are realized in VRML in an open, standards-based modeling language framework versus a Java, c/c++, etc. programming language implementation.

That said, the culmination of this phase of the project was the live technology demonstration involving a link from the VRML stealth viewer located at STRICOM's Engineering Computer Resource Center (ECRC) to the Advanced Distributed Simulation Technology II (ADST II) Dismounted Warrior Network (DWN) user experiments at Ft. Benning's Land Warrior Test Bed (LWTB) in July, 1998. The DWN user experiments explore the maturity level of the Individual Combatant (IC) class of simulators by networking four virtual IC simulators with the dismounted infantry version of ModSAF called DI-SAF.

Figure 1 shows how the various components were pieced together to form the DWN experiment VRML stealth viewer.

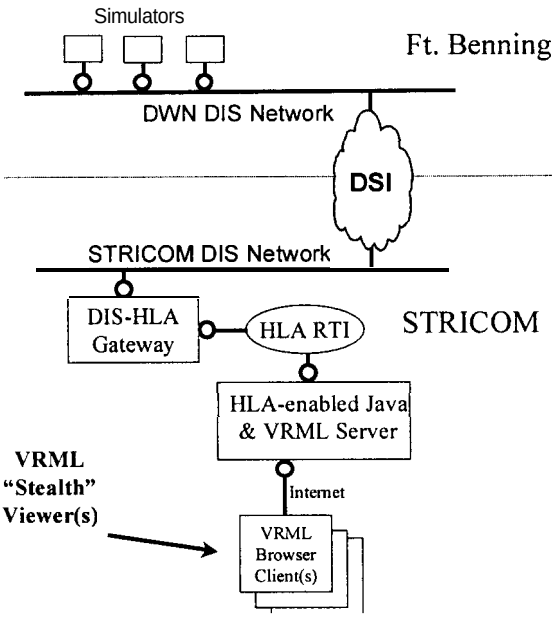

Figure 1. *VRML Stealth Viewer Components*

The DWN simulators communicate using the Distributed Interactive Simulation (DIS) 2.04 protocol to send and receive Entity State, Fire, Detonation, and Collision PDUs. Soldiers operate the four virtual IC simulators forming a fire team, while a Semi Automated Forces (SAF) operator controls three additional fire teams composed of 12 SAF individual combatant entities. During the user experiments, the four fire teams work together to assault and clear a building occupied by a sniper, which is controlled by human-operated desktop simulator. The DWN simulation

12 SAF individual combatant entities. During the user experiments, the four fire teams work together to assault and clear a building occupied by a sniper, which is controlled by human-operated desktop simulator. The DWN simulation takes place on the McKenna Military Operations on Urban Terrain (MOUT) database.

The DIS PDUs from the LWTB DWN experiments were sent to the ECRC lab using the Defense Simulation Internet (DSI) link. The DIS PDUs were then run through an HLA Gateway to convert the PDUs to HLA packets sent to the Run-Time Infrastructure (RTI). An HLA interface module subscribed to data from the RTI, and passed the data to a server. Finally, users were able to connect to the server through an HTML browser with VRML and Java applet support to render the stealth visuals on the client machine.

Database Conversion

The first task involved in writing the VRML stealth was developing a VRML database which could be used in the stealth application. One standard visual database used in military simulation applications is the Multigen .FLT format. A .FLT database of the McKenna MOUT facility already existed, so it was only necessary to find a translator which would take the .FLT format and convert it to .WRL.

The Multigen Creator tool for Windows NT provided such a .FLT to .WRL translation mechanism. The Creator tool allowed the database to be read in as .FLT format, and exported as .WRL. Unfortunately, external references abound in simulation databases (typically, a separate file is used to describe each building and cultural feature in the database), and the VRML export feature in Creator didn't handle this properly. The secret was to convert all external references to embedded references, and save the file out as a giant superset of all the external references. In addition to the main VRML file, texture files were also necessary to form the complete database. In order to speed up initial download time, the textures were cut down in size without loss of detail through a graphical editor.

Once the database file was created, it was tested on a 233 MHz PC using Netscape Navigator and CosmoPlayer plug-in. At this point it was apparent that the VRML database had far too many polygons and used too much texture memory to run anywhere close to the target 5-10 Hz update rate that would be tolerable. So, the .FLT database was loaded back into Multigen Creator and trimmed by more than half down to roughly 1000 polygons. The database was then reconverted to VRML with acceptable results.

The database coordinates of the .FLT and .WRL files are not the true Geocentric coordinates of the McKenna MOUT site. Instead, the database coordinates have the origin at the lower left corner of the database. Therefore, a coordinate conversion was necessary when receiving entities from the DIS entities, which broadcast their coordinates in Geocentric units.

In addition to the database, the moving models that represent the entities in the simulation were also developed in VRML. Several VRML models were constructed to depict ICs in various postures, such as standing, kneeling, crouching, prone, and dead positions. An animated walking model was also developed. Models of generic tanks, planes, and helicopters were also developed to populate the database.

HLA Gateway and RTI

The LWTB, located in Ft. Benning, GA, was networked to STRICOM's ECRC facility using the DSI link. This high-speed connection allows DIS PDUs to rapidly be exchanged between both locations. The DIS data was received at STRICOM and then converted to the HLA protocol using the HLA Gateway application (Version 2.1) developed for STRICOM by the Institute for Simulation and Training (IST). As a side note, HLA was chosen over DIS as the implementation target for the VRML stealth application since the Department of Defense (DoD) is migrating from DIS to HLA for all future simulation systems. The selection of the overarching simulation protocol is a secondary matter, however, since the VRML stealth application actually converts the simulation protocol data to a more streamlined form of data for efficient client/server VRML connectivity (to be discussed shortly).

The HLA gateway uses a standard Federation Object Model (FOM) to convert DIS data to a form recognizable by the HLA RTI. The Gateway provides the transformed DIS data to the RTI as the other federates subscribe to the data. The RTI executive application used for this effort was the RTI 1.0 Release 3 distribution implementing the HLA Interface Specification 1.1 for SGI Irix 6.2 operating system and SGI C++ 7.1 compiler.

VRML Server HLA Interface Module

The HLA interface module for the VRML stealth server application was written using C++ on a Windows NT platform. This HLA interface module makes use of the HLA RTI library calls to subscribe to the data published to the RTI by the HLA Gateway software. The VRML stealth server is a passive federate in that it subscribes to but does not publish any data. As the HLA interface module receives the data from the LWTB DWN systems, it updates its list of active entities and makes the necessary manipulations to the data for communication to the VRML stealth client(s).

Several data-limiting assumptions were made to increase the efficiency of the stealth viewer. For instance, since the focus of the entities in the DWN experiment were with individual combatants, the HLA interface subscribes to the country, category, subcategory, etc. of the various entities, but uses this information only to determine a general entity type – tank, plane, helicopter, or individual combatant. Also, since no simulation management (SIMAN) PDUs were used in the DWN experiments, the SIMAN data was ignored. The primary data that was required from the RTI was the entity identification, entity type, entity position, entity orientation, entity velocity, entity life form state, entity damage state, and firing interactions.

VRML Stealth Viewer Server

The next piece of the puzzle was to write a Java client/server application that allowed users to log into a computer, seamlessly download the McKenna MOUT database and entity models, and bring up the VRML browser to allow the user to navigate the database and see the simulation activity. The general foundation for the client/server application came from Bernie Roehl and Justin Couch's example in Late Nite VRML 2.0 with Java.

The Java central server has two main functions: (1) to allow VRML client users to login to the server and maintain communication, and (2) to transmit data to the VRML clients that are connected to the server.

The first objective was accomplished by accepting standard login data from the clients over a TCP registry port. The client sends a "HELLO" command over the port along with a username and password, which the server authenticates. After login access is granted, the client sends the local UDP ports on which it will receive data updates from the server.

The server receives two different types of data from two different sources, then forwards that data to all of the clients that have connected to the server. The two types of data are entity information and text chat information, received through UDP ports.

The source of the entity information data is from the HLA interface module. This data consists of an array of identification numbers, positions, orientations, velocities, and status flags that provide information on all of the active simulation entities.

The HLA interface module transmits data to the server using the Java Native Interface (JNI). The JNI allows compiled C++ source code to interface with Java compiled byte code. Since native compiled C++ code is much more efficient than Java byte code, the JNI is a mechanism for allowing C++ to be used to generate portions of the source code that are machine specific. The HLA interface module was written specifically for the Windows NT platform, making use of RTI library calls and other NT operating system calls necessary for efficient operation. On the other hand, the VRML stealth server was a web based application, where it made much more sense to use Java. The JNI provided the mechanism to piece the two applications together.

The other source of data to the server is text chat data, which originates from the client applications. It is useful to allow the users that are connected to the server to be able to chat with one another to discuss the distributed simulation session as it is happening. For example, a battlemaster can login to the server and broadcast messages about where the most important activity is taking place, as well as the status of the exercise. The server receives the text chat information and relays it to each of the clients that are connected to the server.

The transmission of both entity information and text chat data to clients is performed using the Real-Time Protocol (RTP). The RTP is a protocol that sits on top of either UDP or TCP and is a very streamlined data packet. The header of the RTP packet consists of a sequence number, a time stamp, a source identification number, and a packet type. The RTP packet data body is simply an array of bytes that follow the header. The bytes can be interpreted in several ways, as long as it agreed upon by both the sender and receiver. In the case of the entity information and text chat data, each is given a different packet type, and each is sent from the server to the client over a

different UDP port, so confusion about the contents is negated. The text chat data is simply a stream of characters. The entity information consists of position (x,y,z), orientation (y,p,r), velocity (x,y,z), and appearance (active status, entity posture, firing status, wounded status, and entity type).

VRML Stealth Viewer Client

The end-user interface to the VRML stealth is the client application that runs under an Internet browser on the user's computer, which can be any machine capable of hosting Internet browser software. The Internet browser must also have a VRML browser plugin installed and be capable of supporting the Java Virtual Machine, version 1.0+, with AWT 1.1.5 and above. In addition, the computer should have a relatively fast Internet connection, plus a fast CPU and graphics card. Most of the testing of the client software was performed on a Pentium II 233 MHz processor with a standard PCI graphics card, installed with Netscape Navigator 4.05 and CosmoPlayer 2.0, with a direct 10 Mbps connection to the Server computer.

The client application is embedded in an HTML file that resides on the same computer that the server is hosted on. The user starts the client application by bringing up an Internet browser and trying to connect to client application on the server host computer. The HTML file contains a reference to the McKenna MOUT database, which also resides on the server host computer. After connecting to the server, the client machine receives the entire VRML file, along with the texture files. The client application HTML file also contains references to a Java applet that is also obtained from the server host computer. The applet controls the text chat and the entity update functions. Once the user is connected to the server, the client begins receiving entity information and text chat data from the server.

Text chat information is received by the client application from the server and posted to the chat window. As the user types a text chat message on the bottom line of the chat application, the chat information will subsequently be sent to the server. The user will receive a reflected update of the message that was typed, and it will appear on the text chatboard.

Entity information that is received by the client is updated on the VRML browser window through the VRML stealth client applet. The client applet communicates to the VRML browser through the External Authoring Interface (EAI). The EAI allows Java applets to send and receive browser information from the VRML scene.

As entities are received from the server, they are added to a local client registry. The local client registry then tries to add the entity to the VRML scene through an EAI call. After the entity is added to the scene, all additional updates received by the applet are then simply passed as updates to the entity in the VRML scene. The VRML application receives all of the moving models from the server machine.

The client applet dead reckons the entities received from the server to conserve the amount of traffic sent across the network. Currently the Java client/server application implements a one-to-many point-to-point broadcast of messages, so bandwidth can become a problem based on the number of clients connected to the server.

Movement through the VRML scene is handled through the standard VRML browser navigation bar. Each VRML browser has a slightly different implementation of the navigation bar, but most support the basic controls of flying, walking, and examining. In addition, several browsers allow the user to toggle on/off collision detection, which is necessary to travel inside buildings.

The VRML browser also allows the user to select various viewpoints from which to view the scene. The database has predetermined viewpoints that correspond to key areas within the database where action is likely to occur, such as the outside of a building, the inside of a building, or at an intersection of two roads. Viewpoints are also attached to each of the entities as they are created.

The user can receive status updates on any of the entities in the simulation. To do this, the user clicks on the entity, and information will be presented in a status window containing the entity identification, position, orientation, and other status information.

Figure 2 shows the information presented to the user through the VRML stealth client application.

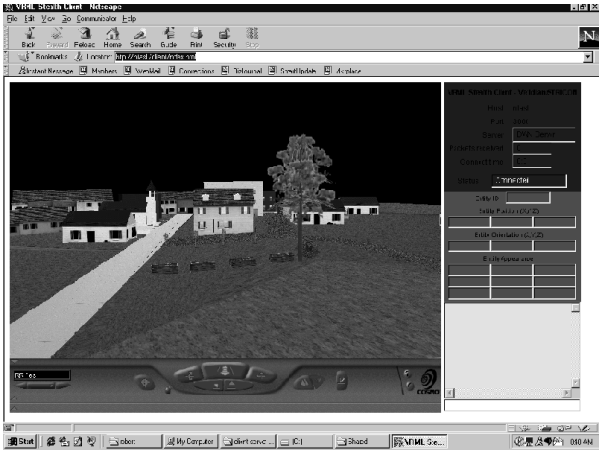

Figure 2. ***VRML Stealth Viewer Client***

FUTURE WORK

Future work will concentrate on expanding the realm of VRML applications within the M&S environment. Additional work might involve converting the VRML stealth into an unsophisticated reconfigurable simulator. All that would need to be added is be a user interface to control the scene movement, i.e., a simple vehicle dynamics model, plus the capability to send position and orientation update information back to the HLA RTI. Simple dynamics could be coded for fixed wing, rotary wing, ground, and individual combatant simulators. The web site **Error! Bookmark not defined.** has been established that provides the latest VRML demonstration software.

BIBLIOGRAPHY

Veda Incorporated. Technical Proposal - VRML - Enabled M&S Information Dissemination R&D, September 26, 1997

ISO/IEC DIS14772-1:1997, The Virtual Reality Modeling Language, December 1997

Roehl, Bernie, et al, Late Night VRML 2.0 with Java, Ziff-Davis Press, 1997