A Virtual Control Room with an Embedded, Interactive Nuclear Reactor Simulator

Stefano Markidis and Rizwan-uddin

Department of Nuclear, Plasma, and Radiological Engineering
University of Illinois, Urbana, IL 61801, USA
s.markidis@gmail.com, rizwan@uiuc.edu

Abstract – *The use of virtual nuclear control room can be an effective and powerful tool for training personnel working in the nuclear power plants. Operators could experience and simulate the functioning of the plant, even in critical situations, without being in a real power plant or running any risk. 3D models can be exported to Virtual Reality formats and then displayed in the Virtual Reality environment providing an immersive 3D experience. However, two major limitations of this approach are that 3D models exhibit static textures, and they are not fully interactive and therefore cannot be used effectively in training personnel. In this paper we first describe a possible solution for embedding the output of a computer application in a 3D virtual scene, coupling real-world applications and VR systems. The VR system reported here grabs the output of an application running on an X server; creates a texture with the output and then displays it on a screen or a wall in the virtual reality environment. We then propose a simple model for providing interaction between the user in the VR system and the running simulator. This approach is based on the use of internet-based application that can be commanded by a laptop or tablet-pc added to the virtual environment.*

I. INTRODUCTION

Virtual Reality (VR) systems may be used to help design control rooms and to train personnel in (virtual) nuclear power plants. Using dedicated software, it is possible to create realistic virtual control rooms and other parts of a nuclear power plant. Software packages, like CREATE developed by Halden Virtual Reality Center (HVRC), allow rapid prototyping of 3D virtual models [1]. These models can be visualized and *experienced* in immersive environments of virtual reality systems like CAVE [2], CUBE [3] and VisBOX [4], and used for evaluation, testing and training purposes. Hence, realistic interactions with 3D models can be simulated to optimize design choices, and to make ergonomic estimations [5].

It is important to make a distinction between the functionalities of the "developer's tools" (like CREATE) that display their results on 2D monitors, and the functionalities available in the 3D immersive environment of a virtual reality system like CAVE, CUBE and VisBOX. It is currently possible to select objects and move them interactively in the developer's tools like CREATE. With some effort this capability can also be translated to the virtual reality platforms like VisBOX. This functionality in the VR environment will help speed up the prototyping and testing procedures. A second limitation is that equipment indications and controls on the panels and computer monitors of the virtual control room in the developer's tools as well as in virtual reality immersive environment are static textures, and interactivity is not often supported. It is this second limitation that has been addressed in this work. Thus, a nuclear reactor simulator has been embedded in the virtual control room. The virtual environment can therefore now display the (live) results of a nuclear reactor simulator as well as allow interactivity to operate that simulator.

In this paper we describe the preliminary integration of a generic application, like a nuclear reactor simulator, in a VR system. Moreover, implementation of interactive control of the application in the VR environment using a WEB-based interface is also described. The hardware equipment and software used in the development of this project is given. Finally the results of a nuclear reactor simulator embedded in a virtual nuclear reactor control room are shown.

II. VIRTUAL NUCLEAR CONTROL ROOM

Given a static virtual control room, goal of an interactive, virtual simulator in the VR environment can be broken down into two sub-goals: display of contents of a live and dynamically evolving computer window on a flat surface in the virtual model; and allowing equivalent of mouse-actions in the virtual environment providing interaction with the virtual environment and triggering events.

II.A. Embedded Application in Nuclear Control Room

To display the results of the reactor simulator in the virtual environment, a generic capability has been developed to display the content of any window of choice in a computer monitor on a flat surface of choice in the virtual environment. Window of choice may be displaying the content of a WEB browser, graphical output of a simulator, or a movie. Realizing that results of almost any application (a simulator output, a movie, etc) can be displayed in a WEB browser, and to take advantage of the standardization and portability that accompanies WEB-based tools, specific application developed here is focused on displaying the content of a web browser window in the virtual environment. One can open any page of choice in the web browser and hence display it in the virtual environment. Hence, the machine running the VR application must also be connected to the web and must have a WEB browser.

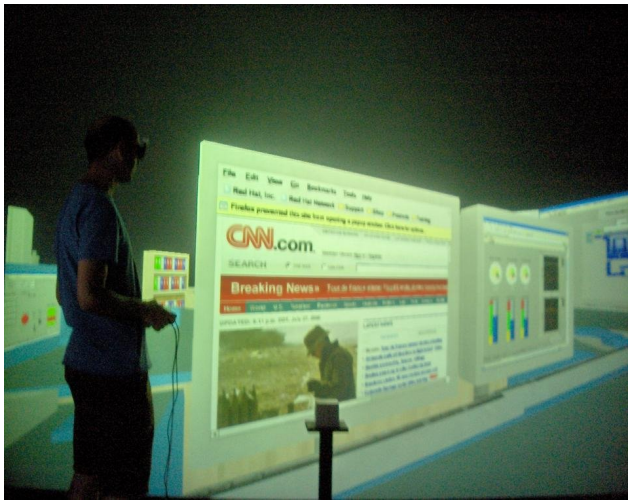

Fig. 1. The CNN WEB-page displayed in the virtual control room, as an example of output of a generic application running in the virtual reality environment.

The WEB browser runs on an X server, called Xvfb, that instead of rendering the image to the standard display renders off screen to a memory area. The VR application captures the graphical output of the WEB browser running on the X virtual server as a bitmap and converts it to an OpenGL texture. The texture is then placed on a specified polygon and displayed in the virtual control room. At each graphical callback, the image of the WEB browser on the X virtual server is captured, converted to texture and mapped to the selected flat surface in the virtual control room. This idea of employing an X virtual server and creating a texture of an application for the virtual system was originally formulated by Belleman, who developed the *XiVE* library [6].

Two different computers—one to run the simulator and the other for the virtual reality application—are used. Figure 2 shows a schematic diagram outlining the interaction between the two computers and the software applications. The simulator runs on a dedicated machine and a WEB-server on the same machine broadcasts the results to the world-wide-web. Two applications run on the virtual reality (second) computer: a WEB browser, that receives and displays the results of the simulator via the internet, and the VR application that renders the virtual control room and tracks the position of the VR user.

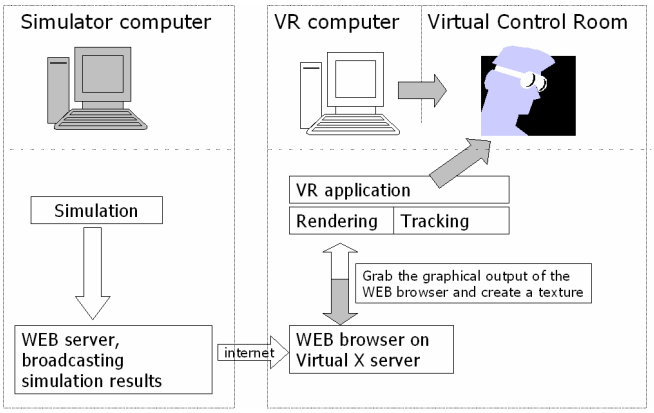

Fig. 2. Schematic diagram of a nuclear reactor simulator embedded in a virtual control room.

II.B. Interactive Nuclear Control Room

The fact that the output of the simulator is webcasted facilitates the interaction between the user in the virtual environment and the window with the dynamic output, and provides a simple mechanism for interactivity. Instead of direct interaction between a user device such as joystick, specialized glove, etc, and the simulator window, it is possible to introduce in the virtual environment one or more laptops, or tablet PCs connected to the WEB server through a wireless Internet connection [7,8]. The tablet PC can display the same front end of the simulator in a WEB browser –with interactive control buttons- as shown in the virtual control room, and hence can be used to trigger events. In this approach the *mouse action* is easily simulated in the VR environment by using a laptop computer or a tablet-pc connected to the Internet. This approach is much more natural than alternatives in which either a joystick is used in conjunction with a ray piercing the virtual button, or approaches in which a specialized glove is used by the operator to manipulate virtual buttons. Besides, it removes the complexity of programming the direct interaction between the user and the window.

II.C. Hardware and Software

The Virtual Reality Lab in the Department of Nuclear, Plasma and Radiological Engineering at the University of Illinois at Urbana-Champaign has a VisBOX immersive projection system [4]. The VR system employs one back projection 12'x 9' display screen (single wall), stereoscopic rendering, specialized audio, a magnetically tracked joystick and a wireless head tracker. See Figures 3 and 4. The computer running the VR system has an Intel Xeon 3.06 GHZ processor with 2 GByte of RAM. Large RAM is necessary in this project because the X virtual server, used in this project for the output of the application, writes to RAM instead of the memory of the graphics card.

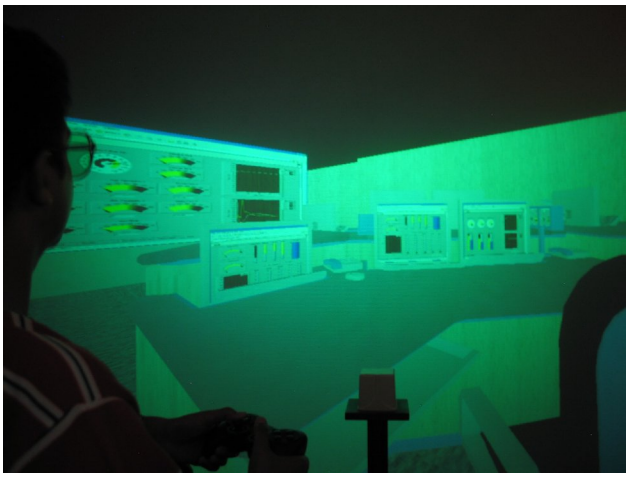

Figure 3. Virtual Reality Laboratory in the Department of Nuclear, Plasma, Radiological Engineering at the University of Illinois in Urbana-Champaign showing a virtual model of a nuclear reactor control room.

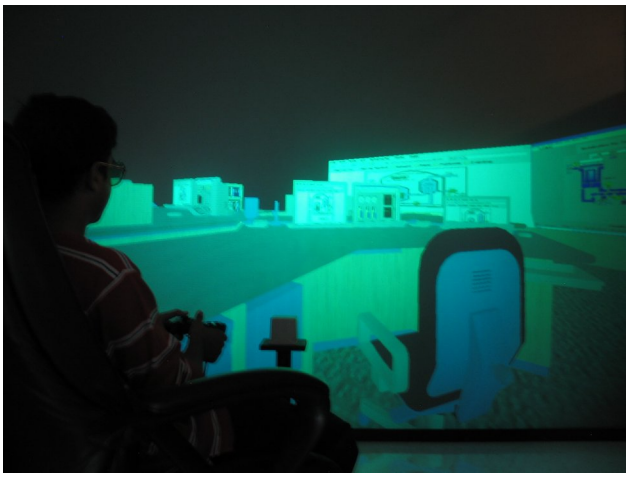

Figure 4. Another view of the virtual control room.

The operating system running VisBOX is Linux Red Hat Fedora Core. We modified a 3D, virtual model of a nuclear control room developed using CREATE, by HVRC to host the output of our nuclear simulator. The 3D model is saved in the VRML format with CREATE and then converted to an OpenGL/C file using *PolyTrans* [9]. The software for capturing the graphical output of the application running on the X virtual server is written in C (gcc compiler) using X11 and OpenGL libraries and is largely inspired by the *XiVE* library and *xwd* application [6, 10]. The source codes of both applications are available on the WEB. Moreover the *XiVE* code also allows a mechanism to use the tracked joystick to interact with the dynamic window. However, in our VR system the interaction with the simulator is via the tablet PC and an internet connection.

The choice of the language *C* is primarily dictated by the fact that all the libraries for virtual reality applications in our group have been developed in *C*. Moreover, the X11 libraries are easily accessible from *C/C++*.. However, we recognize that *Java* coupled with *Java3D* is likely to be a better programming platform for a project of this kind. CREATE and other advanced simulation packages, like *CartaBlanca*, are increasingly been written in Java. [1,11] Besides, the development in Java of WEB-servers and graphical interfaces is faster than in other programming languages

III. RESULTS

To demonstrate the capability to embed a nuclear simulator in a virtual control room, a WEB-based nuclear simulator was chosen. This would, for example, allow the display of results of the simulator running anywhere in the world as long as it is capable of webcasting its results over the World Wide Web. In a parallel development at the University of Illinois [12], a simulator-like front end has been added to the widely used RELAP5 code [13]. Taking advantage of the capabilities of LabVIEW, the RELAP results are graphically displayed and, more importantly, can be viewed in a web browser. If permitted, the remote viewer of RELAP5 output can also interact by, for example, initiating a scram by clicking on the scram button in the WEB browser [12]. It is this recently added capability to RELAP5 that is used to test the framework to embed a simulator in the virtual control room.

In Figures 5 and 6 two views of a virtual control room with the embedded nuclear reactor simulator are shown. Color-coded void fraction distributiondisplayed on the virtual screen can be seen in Fig. 5. The monitor on the right is showing the pressure level in the steam generators

and the user's laptop is also showing the same results. The large screen on the right in Fig.6 is the main control window in the LabVIEW-based RELAP5 nuclear reactor simulator. It shows the red scram button, as well as other features to select different display modes, and interactive control features. The WEB browser in the PC in the front of the user in the virtual environment is also showing the same "page". Hence the user can mimic, for example, a scram by simply tapping (on a tablet PC) or clicking on the scram button on his pc. A click on the scram button initiates the scram sequence in the simulator, and consequently all the virtual screens showing the results of the simulator, which are being continuously updated, also display the results of the scram.

Currently, the performance measured by the speed with which the simulator results are displayed in the virtual control room, is acceptable. However, large amount of data that needs to be frequently updated in the virtual control room will push the limits of the current procedure. We are currently working on improving the performance by employing techniques to faster capture the graphical output of the X virtual server. However, it is also possible to improve the performance simply by reducing the size of the window displayed in the virtual environment and therefore the memory used for bitmapping the output of the applications.

IV. CONCLUSIONS

We proposed a general methodology to embed an application output in a 3D immersive environment and a simple model to provide interactive control of the application using a WEB-based GUI. An implementation and the results of these two basic strategies for coupling a nuclear reactor simulator and a 3D VR model have been shown. The future improvements for this project could be a new design for the software package and a better organization of the code.

We believe that the embedding of a interactive nuclear reactor simulator in a VR environment is another step toward the use of virtual experience of a control room to not only test the control room design features, but also to train reactor operators on a virtual simulator.

ACKNOWLEDGMENTS

The authors would like to acknowledge Paul Rajlich of VisBOX for the important and continuous help during the development of this project. This work was supported in part by a DOE INIE grant and a DOE NEER grant.

REFERENCES

1. HALDEN VIRTUAL REALITY CENTER Web page, http://www.ife.no/laboratories/hvrc1/, (2006).
2. C. CRUZ-NEIRA, D. SANDIN, T.DEFANTI, *Virtual Reality: The Design and Implementation of the CAVE®*. Proceedings of SIGGRAPH 93 Computer Graphics Conference, ACM SIGGRAPH, (1993).
3. "THE CUBE" WEB page, http://www.isl.uiuc.edu/Labs/room_b650.htm, (2006).
4. VISBOX INC. WEB page, http://www.visbox.com/, (2006).
5. RIZWAN-UDDIN, N. KARENCEVIC, S. TIKVES, *Virtual Reality at the service of GEN-IV, and V, and ...*, Proceedings of ICAPP-2003, (2003).
6. XiVE WEB page, http://staff.science.uva.nl/~robbel/XiVE/, (2006).
7. V.E. WHISKER, A.J. BARATTA, T.S. SHAW, J. W. WINTERS, J.A. CLELLAND, F.T. JOHNSON, *Simulating Nuclear Power Plant Maintenance Activities Using Immersive Virtual Environments*, Proceedings of ICAPP-2004, (2004)
8. IOWA STATE UNIVERSITY VESUITE WEB page, http://www.vesuite.org/, 2006.
9. POLYTRANS WEB page, http://www.okino.com/conv/conv.htm, (2006).
10. xwd.c code WEB page, http://stuff.mit.edu/afs/sipb/user/qjb/source/hacks/xwd/xwd.c, (2006).
11. W.B. VANDERHEYDEN, E. D. DENDY, N.T. PADIAL-COLLINS, *CartaBlanca— a pure-Java, component-based systems simulation tool for coupled non-linear physics on unstructured grids*, Proceedings of the 2001 joint ACM-ISCOPE conference on Java Grande, (2001).
12. K.D. KIM, RIZWAN-UDDIN, *A web-based nuclear simulator using RELAP5 and LabVIEW*, manuscript in preparation
13. IDAHO NATIONAL ENGINEERING & ENVIRONMENTAL LABORATORY (INEEL), *RELAP5 manual revisions 2.2 and 2.3*, http://www.inel.gov/relap5/r5manuals.htm, (2006).

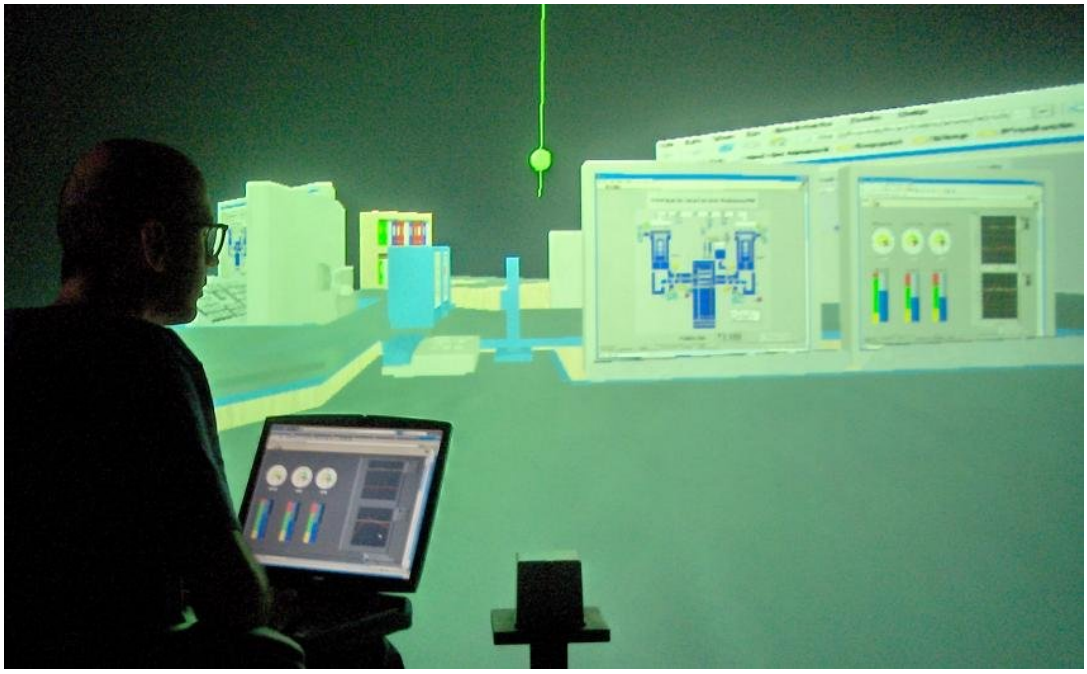

Figure 5. Picture of the virtual control room. One of the screen displays color-coded void fraction distribution in the reactor

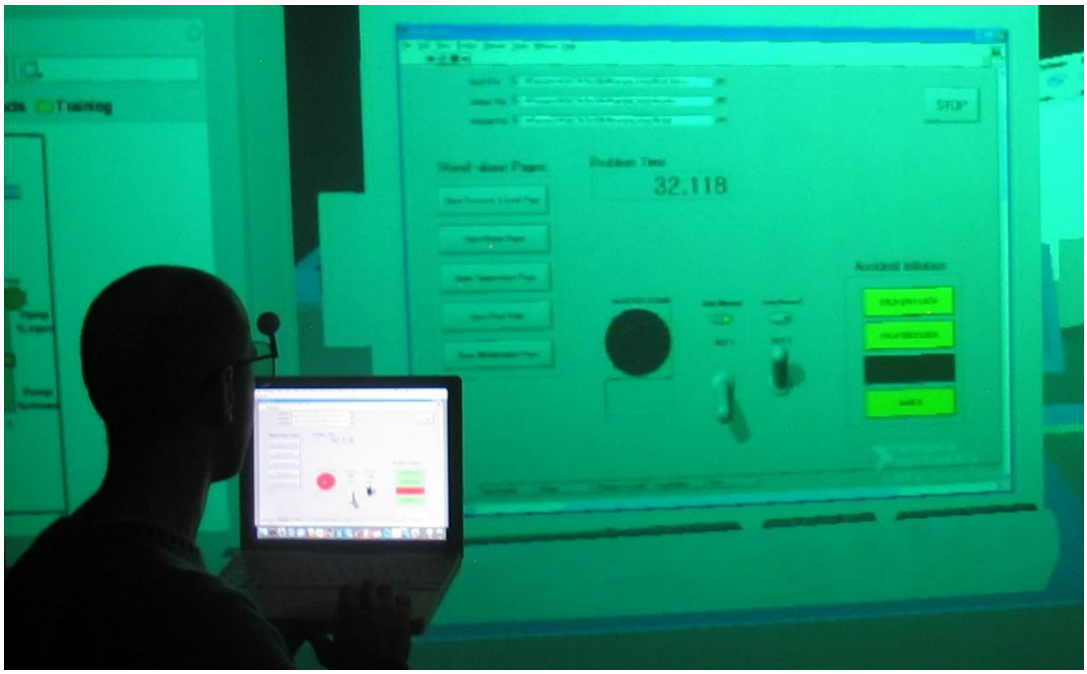

Figure 6. Another view of the virtual control room. The red button on the virtual monitor if (virtually) pressed will scram the reactor.