Research of Dynamically Interactive Simulation Based on VR-and-JavaScript

Wu Xu, *Lingling Kong*, *Jin He*,and *Yali Gao**

School of Electronic and Information Engineering, Yunnan Minzu University, Kunming, China

*Corresponding author: *Yali Gao*, yali75@126.com

Abstract

Along with the rapid development of Internet Multimedia techniques it is increasingly popular in experimental teaching to base virtual, digital, dynamic interactive simulation on the computer network. This paper presents a virtual model built by modeling software using VRML and JavaScript combining instances of virtual experiment of automatic control. Additionally, the experiment model optimizes transmission of vast amount of data on the Internet by data optimization and compression. Furthermore, we discussed the implementation of dynamic interaction in virtual reality (VR) techniques by using JavaScript.

Keywords*: VR; JavaScript; dynamic interaction*

1 Introduction

Virtual Reality (VR) came to the public's attention in the late 20th century. It draws upon many information technologies including Digital Image Processing, Computer Graphics, Multimedia Technology, Sensor Technology etc. VR is widely adopted by medical science, aeronautics and astronautics, industry, entertainment, education, military, and so on. It uses virtual reality modeling language (VRML) to build a virtual world in a network environment. VRML is a modeling language used to create scene model of the real world. It is a standard file format for representing 3-dimensional (3D) interactive vector graphics. VRML is independent of the platform, web-based and object oriented. It is common to embed multimedia content e.g. text, images, audio and video in VRML files, as well as Java, ECMAScript etc. applications.

VRML applications are based on the Internet to build 3D multimedia interactive environment. It works as the fundamentals of VR application. VRML is embedded in HTML web pages and sent to the client when the web browser requests the page. The web browser parses and calculates the VRML text to generate the virtual scene dynamically. For instance, the web browser parses a piece of VRML text content which describing a globe and draw a 3-dimentional globe on the screen. Silicon Graphics International Corp. released VRML specification 2.0 in 1996, which made it practical to simulate reality on the Internet using VRML. Since then, VRML interpreter has been embedded in most popular web browsers. For those web browsers' old versions, VRML plugins can be installed to view VRML scenes [1].

2 Environment Modelling and Optimization

The extension of a VRML file is .wrl. The essential elements of a VRML file are shape nodes and fields, which form the main body of the file. The definition of shape nodes constructs virtual VRML space. VRML provides a set of modeling components for simple geometric bodies e.g. Box Node, Cylinder Node, Sphere Node etc. which can be used in programming to construct basic 3D models. The simplest way of making 3D design is to edit description text file with a text editor like programming, which is simple and convenient [2]. The problems of this method are lack of intuition and steep demand of space imaginary abilities of the designer thus it is inefficient. One of the design software can be adopted is 3DS Max, which is dedicated to modelling, animation, simulation and rendering. We can use it along with VRML to proceed 3D modeling and animation by building 3D models in 3DS Max, exporting as .wrl files, modifying with Vrml Pad and get the models we need [3]. The experiment model is illustrated in Fig. 1.

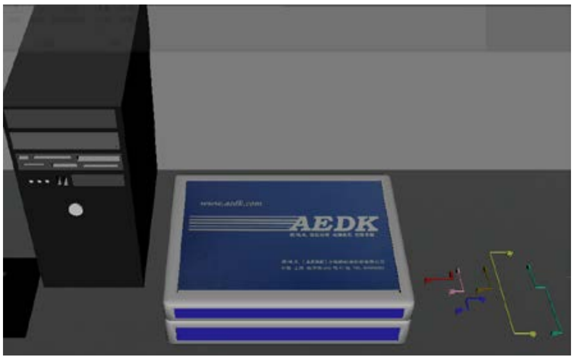

Fig. 1- Experiment Model

The goal of creating VRML virtual reality scene models is to present them on the Internet. We publish models to web servers for visitors to view and download. The web server sets MIME (Multi-Purpose Internet Mail Extensions) type of the model data so web browsers can identify. Since VRML virtual scene data is transferred via network we must optimize it in order to meet the computer network scenario [4]. The methods commonly adopted include:

2.1 Removing Redundant Data.

Upon conversion from 3D model files to VRML files redundant data e.g. unnecessary decimal digits of data, spaces and empty lines are kept in the target file which increases the file size. When it becomes complex enough of the virtual scene, the data size of the VRML file grows significantly so that it is difficult to transfer it via the network. It is necessary to optimize the data under the premise that the virtual scene be presented properly in web browsers. The methods of optimization include rounding numbers down, solidating data accuracy and eliminating redundant, minimizing comments and spaces in VRML files.

2.2 VRML File Compression

VRML is an open and normative language thus the file size is relatively large. We should compress the files before publishing, then the compressed VRML scene files can be viewed in web browsers. Compression software that commonly used includes Win-GZ, GZIP, and Internet Model Optimizer etc. We can optimize and compress VRML files with VrmlPad. If we compress a VRML scene file to 40% of its original file size and view it in a browser we will realize that there is no difference between them. If we open a compressed VRML file with a text editor, what we will witness are unrecognizable characters which maintain secrecy of the work to some extent.

3 Implementation of Interaction of Virtual Scenes

In VRML we observe the result of interaction by watching animations. Actually, animations describe the status changes of scenes that vary with time.

There are some nodes in VRML which are used to do detection and perception that can be used to improve dynamic interactivity between the user and the virtual environment. In order to enforce control of scenes and implement the interaction and complex functionalities, it is insufficient to just rely on VRML embedded nodes. For instance, in order to simulate the trail of a ball that moves regularly by cosine function in the X direction with location interpolation method, we will have to calculate a great deal of coordinates at correspondent time points, which are cluttered and tedious. Furthermore, not all the controls can be implemented by purely adopting nodes. It is necessary to introduce scripts and programming to VRML in order to extend the functionality.

This can be achieved by using Script nodes. Script nodes consist of script applications programming in Java, JavaScript or VRML Script. We use Script nodes to generate scene animation, do logic control of the virtual scene, manipulate the layer structure of the scene and interact with the browser. In Java, we can use the Script abstract class of vrml. node package. By inheriting Script class, a Java application can connect to Script nodes and gain control of other VRML nodes [5]. First, we define field values of a Script node. Then we set Route to connect other VRML nodes to the Script node. While an event is emitted by a node that is connected to the Script node, the event will be transmitted to the evet In field of the Script node, the Script node then find the Java class specified in the url field.

The initialize method of the Java class will be called first – for initialization. In the initialize method we obtain the value of event In or field of the Script node, or return a value in the Java object back to the event Out field of the Script node. If the Java application wants to modify nodes of the scene it will manipulate Script nodes as well. Upon defining a Script node we should modify the event Out field to field and field Type to SF Node.

The most important code of the Script node in the automation control experiment:

```
DEF
go Script {
field SFNode cd2 USE cd2
field SFNode cd3 USE cd3
……
eventIn SFTime caidan1
eventIn SFTime caidan2
……
field SFInt32 d0
field SFNode cd1 USE cd1
field SFNode cmr2 USE Camera02
……
}
```

After combining these powerful nodes and routes, we can implement the interactive virtual experiment by adopting script application containing Touch Sensor, VRML Script and

JavaScript as well as Route statements. The event-driven structure of the interactive virtual experiment is represented by Fig. 2.

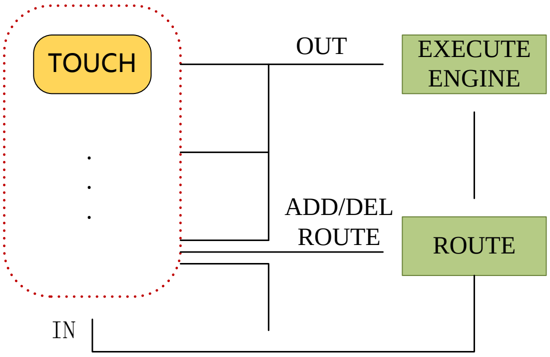

Fig. 2 - Interaction between nodes

All user-triggered events are controlled by Script nodes, i.e. the user's input is converted by the script code and output to specific objects under application control to achieve the control of the interaction [6].

Let us examine the interaction process by studying a real automation control virtual experiment. In this example we use Touch Sensor nodes to sense mouse clicks. Touch Sensor is a shape to create a touch sensor in the virtual space. It is used to detect the movement of the operator and convert the movement into proper output to trigger animation. For example, if the operator moves the mouse upon the sensible experiment box the cursor will change to hand shape. Now if the left key of the mouse is clicked and released, the node is triggered to open the box. This is shown in Fig. 3.

Fig. 3 - The opening and shutting of the experiment box

When the observer releases the mouse key, the node outputs a FALSE in it is Over field (event Out out-event). The absolute time is also output in the touch Time field (event Out out-event). The touch Time field (event Out out-event) is routed to the set-start Time event of the Touch Sensor node, thus the open/shut action of the experiment box is executed. A new invent is generated and correspondent animation is played. The process is shown in Fig. 4.

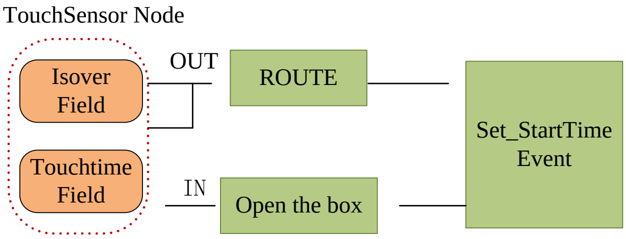

Fig. 4 - Clicking Interaction

The experiment procedure is controlled by controlling the state of the user interface Touch sensor. On a specific step of the experiment the specific Touch Sensor is activated and other Touch Sensors are deactivated so that the user has to finish the current experiment thoroughly before entering the next step. For instance, we use the Proximity Sensor node in the experiment to prompt the operation process with a pop-up menu. The user must finish compulsory activities (i.e. to activate the specific Touch Sensor) to make the virtual scene move on and prompt the operations of the next step. This is shown in Fig. 5 and Fig. 6.

Fig. 5 - Application Entry Point Window

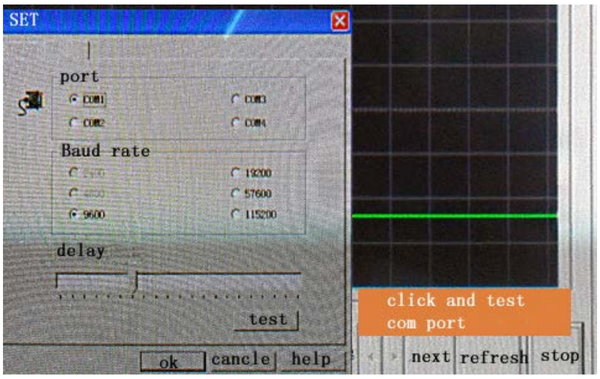

Fig. 6 - Experiment Operation Window

The key code is shown below:

DEF cdlb1 1 Transform {

……

DEF jjcgq1 ProximitySensor # ProximitySensor

……

DEF cdan3 Transform {

……

url "desk1.png" # Open the application

… …

url "testcom.png" # Single-click Testing

……}

4 Conclusions

The paper mainly introduces the studying and implementation of virtual digital simulated dynamic interactive experiments with VRML and JavaScript. By studying its two important elements, adopting third-party software and extended JavaScript nodes we compensate for its weakness and virtualize the environment that is more close to the reality. The operators can perform computer-human interaction to inspire innovation and creativity. Virtual reality technologies not only accelerate the development of remote education but also have the potential in various scientific areas. We try to lay a solid foundation for other potential usage based on the techniques from this paper.

Acknowledgment

This work was financially supported by the teaching reform projects of Yunnan Minzu University 2014" Research of Experiment of remote virtual wireless sensor network (WSN) Based on VRML and X3D" and Yunnan ethnic affairs research projects 2013 "the Research of virtual digital representation and repair of Yunnan YI lacquer ware and drum" and Virtual Digital Protection Yunnan Ethnic Villages Features of Yunnan Ethnic Affairs Commission of Minority Cultures Rescue Protection Projects, July 2015 and Research on related technology of pattern recognition based on the Yunnan national herbs (No.61365007).

References

1. *Xu Wu,* Virtual Reality Technology. Zhejiang University Press, Hangzhou, 2013.
2. *Xu Wu, and Shao Jian-long,* The application of the VRML in virtual city scene, Modern information technology theory and application. Vol. 1, China University of Science and Technology Press, 2002.
3. *Dai Jun, and Shen Jian-hua,* Research and implementation of virtual laboratory based on VRML, Ship Electronic Engineering. **4** (2009) 21 - 24.
4. *Bao Hai-tao,* The application of virtual reality technology in the automobile modeling design, Equipment Manufacturing Technology Magazine. **10** (2009) 33-38.
5. *Ye Yan Qing, Shao Jian Long, and Nian Xiao,* Interactive Virtual Reality Modeling based on VRML, J. System Simulation. **18 (10)** (2006) 2827–2831.
6. *Shao Jian-long, Ye Yan-qing, and Xu Wu,* The realization of the function of VRML virtual scene in Java programming, Computers and Applied Chemistry. **21(4)** (2004) 641 -644.

