行政院國家科學委員會專題研究計畫成果報告

分散式物件導向環境下 3D 原型物件之研究(II)
A Study of 3D Models Objects under Distributed Object Environment (II)

計畫編號：NSC 90-2213-E-002-067
執行期限：90 年 8 月 1 日至 91 年 7 月 31 日
主持人：李秀惠　國立台灣大學資訊工程學系
共同主持人：林一鵬　國立台灣大學資訊工程學系

一、中文摘要

本計劃主要是研究如何簡單的經由一般的瀏覽器就能存取到伺服器上的三度空間之虛擬世界，並借由一些自訂的網路協定及良好的使用者界面，而達到多人互動的功能。

整個計劃可以分成三大部份，首先是瀏覽器的部份，我們必需提供一套簡易的繪圖引擎程式，它可以直接在瀏覽器上執行，並會負責將所接收到的虛擬世界呈現在使用者面前。接下來是伺服器的部份，我們必需在伺服器上提供一個簡單的 parser，它會負責將所接收到的 X3D（Extensible 3D Specification）檔案加以分析，其主要就是利用 XML（Extensible Markup Language）的語法來描述 VRML 97 （Virtual Reality Modeling Language）的語法，並將我們所需要的內容分類處理，以利後續工作的處理。最後是網路連結的部份，首先在伺服器上開一個 port，讓使用者都可以透過瀏覽器連上來，接著參考網路虛擬實境（Networked virtual environment）中的分散式互動模擬協定（Distributed Interactive Simulation Protocol）來處理虛擬世界中物體的運動狀態，並加以適當改良，使其更適用於我們的系統。

關鍵詞：VRML, X3D, XML, 網路虛擬實境。

Abstract

The purpose of this project is to research that how to access the three-dimensional virtual world on the server through the generally browser simply. It also allows multiple users to interact through the network protocol and good user interface.

Briefly, the whole project could be divided into three parts. The first part is the browser. We must provide a simple render engine program that could be executed on the browser. And it could display the virtual world in front of the users.

The second part is the server. We must provide a simple parser on the server. The parser could parse the X3D (Extensible 3D Specification) files, and classify the data that we need. X3D express the geometry and behavior capabilities of the Virtual Reality Modeling Language (VRML 2.0) using the Extensible Markup Language (XML).

The last part is the connection of the network. We must open a port on the server, and let users could get connection through the browser. Then we refer to the Distributed Interactive Simulation Protocol of the Networked virtual environment to process the object's motion state in the virtual world, and improve it according to our system.

Keywords: VRML, X3D, XML, Networked Virtual Environment

二、緣由與目的

Our goal is to improve the VRML2.0 and meet the needs of the Virtual Reality (Look real, Feels real, Smells real, Move realistically). So, we adopt our approach as below:

(a) We design a rendering engine using the JDK1.1.x, and write a java applet using the rendering engine as our client program. Thus, people can browse the virtual world through the general browser.
(b) We provide a friendly user interface, people can interact with the virtual models through this interface.
(c) To speed up the download time, we first filter the data of the virtual world on the server, and transfer the filtered data to the client.
(d) We provide a protocol to enable the function of the Multi-user.

The overall framework of our system is shown below:

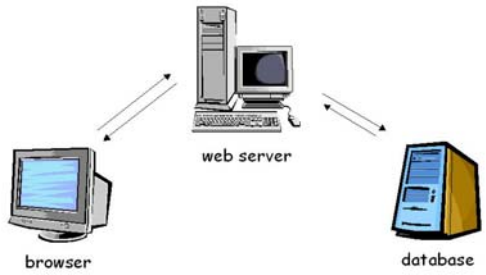

Figure 1. Our System Framework

三、結果與討論

1. Server Program Implementation

Our server program is run on the web server, and use the socket to connect with the applet. At the first, we open a server socket and give it a suitable port to wait for connection.

```
ServerSocket listen_socket = new ServerSocket(port);
```

Secondly, we supply the function of the Multi-user. A multi-thread program is set up to reach our needs. Each connection of the client was dealt by an individual thread.

(a) Using jdbc driver to connect with the database

All the 3D models are put on the database. We use the jdbc driver as a channel between the server program and the database. We have several functions below:

```
Connection con = DriverManager.getConnection(
                    url, user, pass);
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery(query);
```

(b) Dealing with the 3D models that were returned by the database

We adopt the specification of the X3D to represent our virtual world. Our server program must have the ability to deal with the X3D data. X3D uses the grammar of the XML to describe the virtual world. We adopt the jaxp (Java API for XML Parsing) to deal with the X3D data. The codes are shown below:

```
DocumentBuilderFactory docBuilderFactory =
        DocumentBuilderFactory.newInstance();
DocumentBuilder docBuilder =
        docBuilderFactory.newDocumentBuilder();
Document doc =
         docBuilder.parse(new File (FileName));
Element element = doc.getDocumentElement();
```

2. Client Program Implementation

To write a simple applet that can connect to the server using the socket.

```
Socket s = new Socket(host,port);
```

Providing a simple rendering engine

If we want the virtual world looks real, a simple rendering engine must be provided. It can present the virtual world that is got from the server. There are two important components below:

(a) A simple projection

We can decide the projection-coordinates of the objects according to their coordinates in the virtual world, the coordinates of the user, and the position of the monitor.

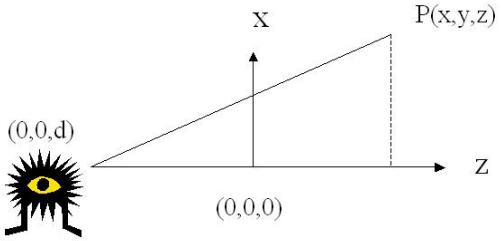

Figure 2. Projection Component

Using the projection function, we can transfer the coordinate system between the virtual world and the user world.

(b) Multiple shading functions

To produce the image according to the result of the projection, we have the following shading functions: wire frame, Flat shading, Gouraud shading, Phong shading.

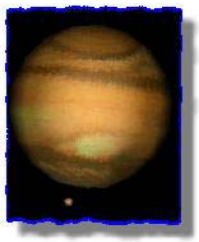

Figure 3. Gouraud Shading

To produce several simple models like triangle, rectangle, cube, circle, ball, and etc. All the objects can be divided into several triangles. We can produce the complicated objects using the simple objects. We also past some real image on our objects, it make the objects look real.

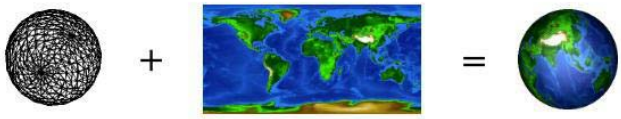

Figure 4. Mapping Method

Animation on the X3D

It is noted that the animation on the VRML 2.0 was made up with the three basic nodes (Sensor, Interpolator, Route). We can get the different animations according the mutual relations among the three nodes. Though the animation on the X3D is similar to the VRML2.0, there exists some difference between these two specifications.

(a) Sensor

Sensor is used to detect the change of environment.

in VRML 2.0:

```
DEF Clock1 TimeSensor
{
cycleInterval 2.0
loop TRUE
}
```

in X3D:

```
<TimeSensor DEF="Clock1"
 cycleInterval="2.0"
 loop="TRUE" />
```

(b) Interpolator

Interpolator is used to describe the change of environment:

in VRML 2.0:

```
DEF PlanetPath1
OrientationInterpolator
{
    key [0.0   0.50   1.0]
   keyValue
   [
     0.0 0.0 1.0 0.0,
     0.0 0.0 1.0 3.14,
      0.0 0.0 1.0 6.28
   ]
}
```

in X3D:

```
<OrientationInterpolator
DEF="PlanetPath1"
key="0.0 0.50 1.0"
keyValue=   "0.0 0.0 1.0 0.0,
             0.0 0.0 1.0 3.14,
             0.0 0.0 1.0 6.28"/>
```

(c) Route

Route is used to transfer the message.

in VRML 2.0:

```
ROUTE node1.eventOut TO
        node2.eventIn
```

in X3D:

```
<Route fromNode=node1
fromField=eventOut
toNode=node2
toField=eventIn />
```

3. A Simple Networked Virtual Environment (Net-VE)

We have established a networked virtual environment. It allows many people to browse our virtual world. The whole framework was below:

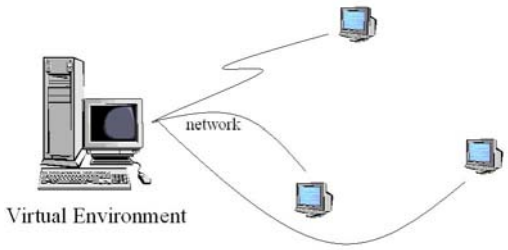

Figure 5. Net-VE Framework

People can connect to our web server through the general browser, and download a java applet. Using that applet, people can connect to our multi-thread Net-VE server and start the journey of the virtual world. When the main thread receives a client's request, it produces another thread to communicate between client and server.

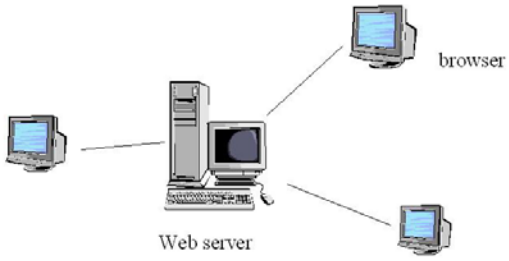

Figure 6. Using a Browser to Enter the Virtual World

Since we control the message centralized, all the messages must transfer through the server. We use a Vector to record the connections, and create a thread to monitor the states of the connections. Using this method we can update online lists dynamically.

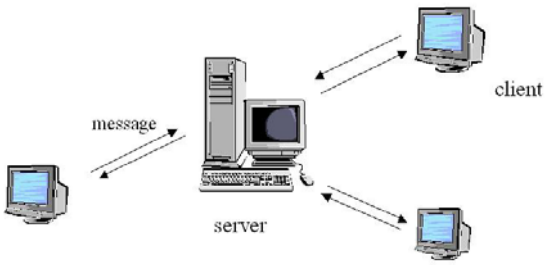

Figure 7. Centralized Control

We adopt the Dead Reckoning Algorithm to speculate the unknown state of the object, and improve it to reach our needs. On the server side, we record the client's old states and speculate client's next state. If the difference between the real state received from client and the speculated state is over our threshold, we send the real state to all the online users. We don't need to transfer all the data to the users, so it can reduce the overhead of our network. On the client side, we use a thread to monitor all the other objects and use Dead Reckoning Algorithm to speculate their state.

In this project, we adopt the X3D approach to

establish a networked virtual environment. It allows many people to browse our virtual world. The users can browse the virtual world through the general browser. In the future, we permit the people to save their states in the virtual world. Therefore, at the next login, people can load the previous states in the virtual world.

四、計畫成果自評

本研究之內容與原計畫極為相符，並已達成預期目標，研究成果的學術或應用價值頗高，亦極適合在學術期刊發表。

五、參考文獻

[1] http://www.web3d.org
[2] http://www.w3c.org
[3] http://www.javasoft.com
[4] http://www/javaworld.com
[5] Foley, Van Dam, Feiner, Hughes, and Phillips, *Introduction to Computer Graphics,* Addison-Wesley, 1990.
[6] David Flanagan, *JAVA in a Nutshell*, O'REILLY, 1997.
[7] Tom Myers, and Alexander Nakhimovsky, *Professional Java XML Programming with Servlets and JSP*, Wrox Press Ltd., 2000.
[8] Patrick Naughton, and Herbert Schildt, *The Complete Reference Java 2,* The McGraw-Hill Companies, Inc., 1999.
[9] Sandeep Singhal, and Michael Zyda, *Networked Virtual Environments*, Addison-Wesley, 1999.