International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:12, 2007

Virtual Scene based on VRML and Java

Hui-jun Ren, and Da-kun ZHang

Abstract**—**VRML(The virtual reality modeling language) is a standard language used to build up 3D virtualized models. The quick development of internet technology and computer manipulation has promoted the commercialization of reality virtualization. VRML, thereof, is expected to be the most effective framework of building up virtual reality. This article has studied plans to build virtualized scenes based on the technology of virtual reality and Java programe, and introduced how to execute real-time data transactions of VRML file and Java programe by applying Script Node, in doing so we have the VRML interactivity being strengthened.

Keywords**—**VRML; Java; Virtual scene; Script.

I. Introduction

WITH the development of computer, virtual reality technology has involved a wide range of sectors. The virtual reality modeling language (VRML), a text language used to quickly build virtual worlds incorporating 3D shapes, light sources, fog, animation, and even sound effects. Each virtual world is described by one or more VRML world files named with a.wrl filename extension and delivered across the Web with the "model/world" multipurpose Internet mail extension (MIME) type. [1]

II. The Building Up of Basic Models of Virtualized Scene

We are developing virtualized scenes with an internet background; therefore, VRML is favored to build up 3D models. VRML files describe scenes in the form of ACSII text as what HTML does. This feature guarantees its universal adaptability to various platforms and smooth transaction of data, so it is conveniently utilized to virtualize reality on the internet. The building up of construction models is one important part of 3D visualization, whereas the basic geometry modes are the foundation of VRML that nearly any complicated scene can not be realized without them. As for real objects, the basic elements are shape, texture and outlook. In VRML, the most important node to build models is Shape, with which VRML creates and controls modeling shape, texture and outlook. According to different outlook characteristics of constructions in the scene, we mainly apply the following modeling methods and their combinations.

(1) Box Node Method

As value of geometry field on node Shape, Box is the basic modeling node of VRML with the following forms of expression:

Authors are with the School of Computer Technology and Automation, Tianjin Polytechnic University, Tianjin, 300160, China.

```
Shape{
    appearance Appearance{ }
      Box{Size {length height width}}
```

This method is featured by its quickness and low required of data amount; it is fit for the situation of modeling regular shaped and simply structured constructions.

(2) Coordinate Node Method

The basic models are the most popularly used in building up scenes, but in the reality, objects are not always of regular shapes. Those scenes made by the basic models can not truly reflect reality. Although sometimes we may combine the basic geometric models together instead, which will overload us eventually. Coordinate node is to solve this problem by defining the point, line, and surface in the virtualized scenes. The following is its form of expression:

```
Coordinate {point [ ]}
```

In the expression, "Point" designates one or a group of x、y、z coordinate system. It is Multi-domain 3D vector of values, which means to list up a 3D coordinate system.

(3) Extrusion Node Method [2]

Extrusion modeling includes the following elements: Lofting graphics、Lofting Path、Lofting direction. The nature of this method is to press lofting graphics toward lofting direction. By way of lofting Path in order to cause scale and thereby to make up a complicated special object; the cross-cut dimension of the scaled object is similar to the lofting graphics. Together with Box node, Extrusion node is used in geometry field of Shape node. The form of expression of Extrusion node is:

```
Extrusion{
    #field MFVec2f    crossSection
    [
    1.0  1.0
    1.0  -1.0
    -1.0 -1.0
    -1.0 -1.0
    1.0 1.0
    ]
    #field MFVec2f    spine
    [
    0.0  0.0 0.0
    0.0  1.0 0.0
    ]
    #field MFVec2f    scale 1.0 1.0
    #field MFRotation orientation 0.0 0.0 1.0 0.0
    #field SFBool    beginCap    TRUE
    #field SFBool    endCap    TRUE
    #field SFBool    ccw    TRUE
    #field SFBool    solid    TRUE
    #field SFBool    convex    TRUE
    #field SFFloat    creaseAngle    0.0
```

```
#eventIn MFVec3f     set_spine
#eventIn MFVec2f     set_ crossSection
#eventIn MFVec2f     set_ scale
#eventIn MFRotation   set_ orientation
  }
```

III. Setting Up of Virtualized Models Outlook

In order to get super realistic effects of virtualized images, we can use different textures and also texture mapping. Texture mapping is a bitmap used to improve the quality of rendering by sticking shapes to certain modeling surface without changing geometrical shape or adding polygons. This method can be applied in modeling outer wall, floor and grassland, etc.VRML supports such image modes as jpeg image mode, gif image mode, png image mode and mpeg video mode.

Texture mapping shapes images onto the surface of models via texture field. Texture is one field of Appearance node, its field value is Single-node domain values. Three textures can be finished under texture background:

```
ImageTexture{
        #exposedField    MFSring   url   [ ]
        #field           SFBool    repeatS TRUE
        #field           SFBool    repeatT TRUE
            }
PixelTexture{
        #exposedField     SFImagel   image   0 0 0
        #field            SFBool     repeatS TRUE
        #field            SFBool     repeatT TRUE
            }
MovieTexture{
        #exposedField     SFBool   loop   FALSE
        #exposedField     SFFloat  speed  1.0
        #exposedField     SFTime   startTime    0
        #exposedField     SFTime   stopTime     0
        #exposedField     MFString  url    [ ]
        #field            SFBool    repeatS TRUE
        #field            SFBool    repeatT TRUE
        #eventOut         SFTime    duration_changed
        #eventOut         SFBool    isActive
            }
```

TABLE I
The Common Result of Start time Stop time Speed and Loop

Loop domain values	The relation of startTime stopTime speed and loop	Result
TRUE	stopTime≤startTime	No loop
TRUE	startTime < stopTime	Stop at the stopTime
FALSH	stopTime≤startTime	Broadcast one cycle
FALSH	startTime < (startTime+duration/speed)≤stopTime	Broadcast one cycle
FALSH	startTime < stopTime < (startTime+duration/speed)	Stop at the stopTime but is not a cycle

IV. Establishing of Environment of Virtualized Scene

Generally, scenes are set up with no background except supplied by the screen, whose “headlight” only adds rigid background color and lacks of enough light resource. Obviously this background can not create a vivid scene nor fully meet the needs of reality.

In the case of VRML, Background Node is the tool to create backgrounds that is more vivid and supportive in viewing screen scenes.

```
Background {
     #exposedField   MFColor   groundColor   [ ]
     #exposedField   MFFloat   groundAngle   [ ]
     #exposedField   MFColor   skyColor   [0 0 0 ]
     #exposedField   MFFloat   skyAngle   [ ]
     #exposedString   MFColor   frontUrl   “ ”
     #exposedString   MFColor   backUrl   “ ”
     #exposedString   MFColor   rightUrl   “ ”
     #exposedString   MFColor   leftUrl   “ ”
     #exposedString   MFColor   topUrl   “ ”
     #exposedString   MFColor   bottomUrl   “ ”
}
```

Being similar to the real world, VRML has different types of light source nodes: point light source, parallel light source and aggregation light source. These light sources have different application environment according to their lighting effect, so we are supposed to choose the one that fits the requirement most.

V. Animation Effect of Virtualized Object

Scenes we see most oftenly are static, actually we see the contrary in the real world. Therefore, we need to vary the shapes, colors and relative location of images virtualized in our virtualized environment. This calls for the animation effect which can reflect those dynamic changes. In designing this animation, time test sensor node controls the time of movement, Insert node (which defines the states and outlook of models) controls state of movement and outlook movement along with the passage of time.

```
TimeSensor{
        #eventOut   SFBool   isActive
```

```
 #eventOut    SFTime    time
 #eventOut    SFTime    cycleTime
#eventOut    SFFloat    fraction_changed
#exposedField    SFTime    cycleInterval    1.0
#exposedField    SFBool     enabled      TRUE
#exposedField    SFTime    startTime       0
#exposedField    SFTime    stopTime       0
#exposedField    SFBool    loop      FALSH
}
```

Use the basic modeling nodes and texture nodes to build a virtual scene, as shown in Fig. 1:

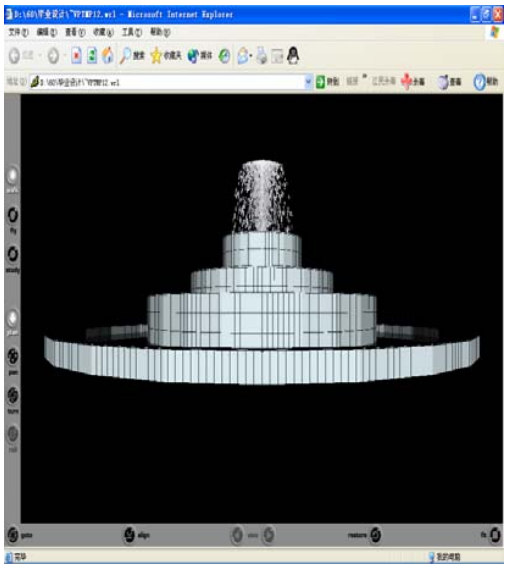

Fig. 1 A virtual scene

VI. Implementation of Interactive Function of Virtualized Objects

For a VRML scene, even though it is vivid to use animation effect, viewers take in message passively. It would be better for viewers being able to control scenes. As a programming language characterized by hardware and software being unrelated, Java complemented VRML. The former is for programe designing in internet environment whereas the latter is for building virtualized reality. As VRML is not strong in its interactivity, Java complemented VRML by way of using Script node which connects VRML with Java within VRML environment.

(1)Send the case to Script Node as manuscript through EventIn.

(2)Manipulate the manuscript with corresponding Java procedure.

(3)Send back the manipulation outcome to VRML envirnonment by EventOutand finally animation or interactivity is completed.

Nowaday this method is most commonly used to to realize interactivity by joining VRML and Java together. [3]

In VRML, field、EventIn、EventOut of Script Node are all available for their corresponding Script category in Java, but access to three of them are different: Field can be both read and written, eventOut is write only, eventIn is read only. There are three accesses to the Field、eventIn and eventOut of Script node:

getField(String fieldName) : Reture to a Script node field named fieldName,then VRML field is shifted to Java field.

getEventIn(String eventlnName) : Return to Script node EventIn named EventInName, then VRML eventIn is shifted to Java eventIn.

getEventOut(String eventOutName) : Return to Script Node EventOut named EventInName, then VRML eventIn is shifted to Java eventOut.

Once you have get other nodes (this node can be available only if it is one Field of Script Node) in VRML in the way we have introduced above, then any eventIn and eventOut of this node are permitted to be visited by Java procedure. The following are the ways to complement this via certain nodes in Java:

For a VRML scene, even though it is vivid to use animation effect, viewers take in message passively. It would be better for viewers being able to control scenes. As a programming language characterized by hardware and software being unrelated, Java complemented VRML. The former is for program designing in internet environment whereas the latter is for building virtualized reality. As VRML is not strong in its interactivity, Java complemented VRML by way of using script Node which connects VRML with Java within VRML environment.

(1)Send the case to Script Node as manuscript through EventIn.

(2)Manipulate the manuscript with corresponding Java procedure.

(3)Send back the manipulation outcome to VRML envirnonment by EventOutand finally animation or interactivity is completed.

Nowaday this method is most commonly used to realize interactivity by joining VRML and Java together. [3]

In VRML, field、EventIn、EventOut of Script node are all available for their corresponding Script category in Java, but access to three of them are different: Field can be both read and written, eventOut is write only, eventIn is read only. There are three accesses to the Field、eventIn and eventOut of Script node:

getField(String fieldName) : Return to a Script node field named fieldName, then VRML field is shifted to Java field.

getEventIn(String eventlnName) : Return to Script Node EventIn named EventInName, then VRML eventIn is shifted to Java eventIn.

getEventOut(String eventOutName) : Return to Script node EventOut named EventInName, then VRML eventIn is shifted to Java eventOut.

Once you have get other nodes (this node can be available only if it is one Field of Script node) in VRML in the way we have introduced above, then any eventIn and eventOut of this node are permitted to be visited by Java procedure. The

following are the ways to complement this via certain nodes in Java:

getEventIn(String eventInName) : Return to a eventIn named eventInName of this node.

getEventOut(String eventOutName) : Return to a eventOut named eventOutName of this node.

getExposedField(string fieldName) : Return to a exposedField named fieldName of this node.

The field value will be manipulated in Java and rewritten in the original VRML scene only when the data is writable (For reference, see Reference Books [4]).This can be done by the following ways:

setValue(value) : converse Java type value to the corresponding VRML type value, and set it up in VRML.

setlValue(int index , value) : converse Java type value to the corresponding VRML type value, and set it up to be an element numbered index(the first element is numbered 0)

addVahe(value) : converse Java type value to the corresponding VRML type value, which is to be added behind the last element.

insertValue(int index , value) : converse Java type value to the corresponding VRML type value, which is to be inserted into the element numbered index.

delete(int index) : Delete value of the element numbered index of the targeted object.

clear() : Erase all elements of the targeted object.

Notice: Methods of set1Value() 、 addValue() 、 insertValue()、 delete()and clear() are only valid for the multiple value fields (i.e. MF type).

Implementation of interactive function of virtualized object, as shown in Fig. 2:

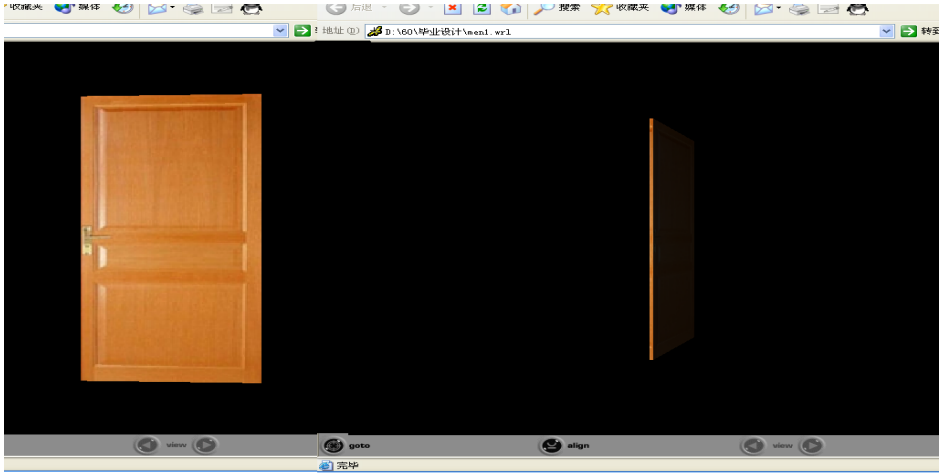

Fig. 2 Java control the door open or close

VII. Conclusion

This article elaborates the key process of using VRML as a tool to build virtualized scenes and the varied methods to interact VRML and Java. VRML turns the internet to be a real broad 3D space with its strong capability to construct 3D models and its interactivity. In spite of its limitation, VRML keeps developing by involving technologies of Java3D and XML in the new product of X3D, which backs up the latest streaming and rendering technology. Therefore, VRML has become the mainstream of 3D world and shows great prospects of further application.

References

[1] Nadeau,David R.(San Diego Supercomputer Cent) 《Building virtual worlds with VRML 》.Source: IEEE Computer Graphics and Applications, v 19, n 2, Mar-Apr,1999, p 18-29.

[2] Jjian-hua Wan,Hong-xia ZHeng,Hui Jin,CHang Feng.《3D landscapes modeling in the virtual campus based on VRML》.Computer Applications and Software Vo1.21,No.7July 2004.

[3] Hong-zhen Xu,Shu-min ZHou,Tang Bin.《Interaction and Its Application of VRML and Java》 Computer and Modernization No.11 2003.

[4] LSO / IEC 14772—1 : 1997 , VRML97 International Standard[S]

[5] Xiao-qiang Hu.《Thetechnology and its application Of Virtual reality》.Beijing: Higher Education Press, May 2004

HuiJun Ren received Bachelor degree in Computer application technology in Tianjin Polytechnic University in China. Now she is studying towards Master degree in Computer application technology in Tianjin Polytechnic University in China. Her interest is in the field of Virtual reality.

DaKun Zhang is an professor of Computer Technology and Automation of Tianjin Polytechnic University. Her research direction is portfolio algorithm design 、virtual reality technology and geographic information system.