Proceedings of IEEE TENCON'02

The Animation Forming Mechanism in VR-Based Virtual Laboraory

Dong Yabo Zhu Miaoliang
(Department of Computer Science and Engineering, Zhejiang University, Hangzhou 310027, China)
E-mail: dongyabo@yahoo.com

Abstract: Animation is an important presentation method in the internet-based Virtual Reality Laboratory (iVRLab). Because of the iVRLab's four basic features--3-D form, interactivity, network basing and virtuality, conventional animating method can not meet the requirement. Based on the VRML animating principle and analysis of the iVRLab's requirement, a brand-new animation forming mechanism is presented. An example of liquid surface lifting animation is illustrated to give the more details.

Keywords: Virtual Reality VRML Virtual Experiment Animation

1. Introduction

Recent technological advances on the Internet have enabled a multitude of applications to operate via the World Wide Web. E-Learning is one such application. To this end, various institutions and universities are now offering online courses that students from all over the world can subscribe to and attend. Various teaching methods and tools have appeared, such as Web Courseware, On-Line Answer Machine, Web Classroom, etc.

Virtual experiments have also been developed. Some recent articles have described the design of virtual experiment systems and their uses in academe [5-8]. Another important simulation technology, virtual reality (VR), was introduced by John Bell with Scott Fogler as a powerful new tool in engineering education [9].

We have developed an internet-based Virtual Reality Laboratory (iVRLab) system to aid the undergraduate's engineering experiment. In this system, we have implemented several engineering chemical experiments, e.g., "Measurement of Water's Degree of Hardness".

Animations are always used in virtual experiment to represent all kinds of actions, such as object movement, liquid shaping, coloring, etc., so they are very important representative means in virtual experiment. But traditional animating mechanism takes a long time to design an animation, and the animation data size generated is too large to be transferred in the Internet[1-3]. So a new animating mechanism is needed in iVRLab to meet its four basic features--3-D form, interactivity, network basing and virtuality.

In the next section, we will give further details of the basic features of iVRLab system. Section 3 describes the requirements of animation in iVRLab. The detailed animating mechanism is introduced in Section 4, followed by a liquid shaping animation as an example in Section 5.The final section gives a conclusion.

2. Basic Features of iVRLab

Educating students in engineering and related scientific fields is made difficult by the complex ideas and phenomena that are hard to demonstrate by conventional methods. A virtual experiment faces the same problem. In order to help the students understand and master the virtual experiments, they must be presented in **3-D form**, where they can observe any object from any point of view and any angle. This important feature of WBVL not only helps the students accomplish their objectives, but it also aids in holding their interest.

When performing experiments, students run the apparatus, observe phenomena, record data and complete a report of the experimental. This means that they must interact with the experiment at every stage. For this reason, **interactivity** is another important feature of WBVL. Each virtual experiment is capable of presenting different reactions to differing input bye the students. This interactivity helps the students feel that they were actually doing the experiment.

At times the students cannot access a laboratory and the only way they can conduct an experiment is through the Internet, so **network basing** is also important. With network-based virtual experiments, experiments are not limited to the laboratory environment.

Virtuality, another feature of WBVL, is a kind of virtual experiments based on simulation and are called Simulation Experiments (SE). In some of the articles mentioned previously, another kind of virtual experiment, called Remote Control Experiment (RCE), is described in which the students control actual experimental instruments via the Internet. This is not feasible in all cases, however, since some experiments use apparatus

0-7803-7490-8/02/$17.00©2002 IEEE.

that cannot be remotely controlled, or they take a long time to complete, or they are expensive. RCE also limit the number of students who can participate. SE has no such limitations and is a more realistic form for virtual experiments.

3. Basic Requirements of Animations in iVRLab

According to the four features of iVRLab, four basic requirements are put forward to the animations:

1. Reality

For the virtual experiment is virtual reality based, reality is also a basic requirement for the animations.

2. Real Time

Nobody can endure discontinuous animations. Also user should get response immediately after he sends a command. With limited computational and image processing abilities of PC, the algorithm to creating the animation should be as simple as possible.

3. Small code and data size

iVRLab delivers experiment curricula through the Internet, so the data of the animation must be short for the sake of reducing the transfer time. Modeling according to the kinematics and dynamics analysis to the animating object, Key Frame technology is used to implement animation in iVRLab.

4. Reusability

Because a multitude of animations are used in iVRLab, reusability is emphasized to reduce the code size. A good animating mechanism can reach this requirement.

4. Animating Mechanism

The Script node is an important VRML[4] node used to implement the animating mechanism in iVRLab. Each Script node has associated programming language code that is executed to carry out the node's function.

The differences of most animations are their mathematical models, while their animation calculating parts are almost the same. So if we abstracted these parts, a compact and highly reusable scheme would be implemented. According to this idea, we designed an animating mechanism as shown in Fig. 1.

Each part in Fig. 1 will be explained below.

1. User Interface

Through the user interface, user can interact with the scene to control the start/stop of the animations, change movement direction, and so on.

2. Shape Object

The shape Object is an animating entity. For example, in liquid shaping animation, the shape object is the shaping liquid in a container. Usually the shape object is composed of some geometric shapes that have one or more variable attributes. During animating, the shape object requests data, then changes its attributes based on the response from the generator to create the animation.

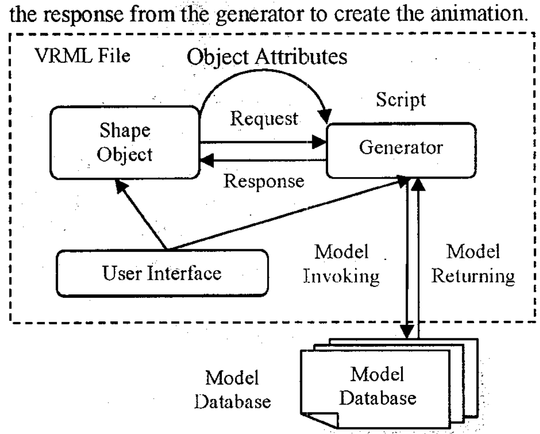

Fig. 1 Structural Architecture of The Animating Mechanism

3. Generator

The generator is the most important part in this animating mechanism. The generator is composed of a calculating module and an interface module as shown in Fig. 2.

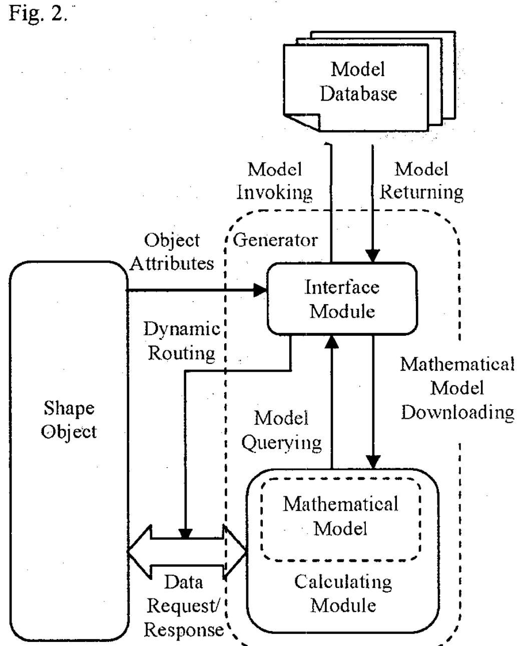

Fig. 2 Structure of The Generator

The calculating module is responsible for real-time calculating of the animation data. Requested by the shape object, it works out the data needed according to the animation mathematical to update the animation model, and sends data back to the shape object. If there has corresponding animation mathematical model, calculating module will query interface module for the model.

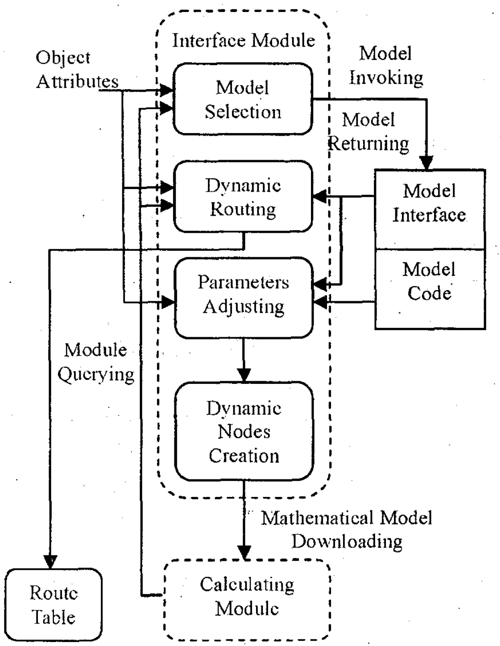

Fig. 3 Structure of The Interface Module

The interface module is responsible for mounting a suitable mathematical model into the calculating module and creating a right routing path to enable correct data transferring between the calculating module and the shape object. Its structure is shown in Fig. 3.

On receiving a model-querying message from the calculating module, the interface module checks the message and determines which model to be invoked, then sends a model-invoking message to the model database.

After getting a mathematical model returned from the model database, the interface module will adjust some of the model's parameters to fit the object attributes, and this model will be downloaded to the calculating module. At the same time some dynamic routes, which are used to transfer commands and data between the calculating module and the shape object, are created by the interface module according to the model interface and the shape object attributes.

4. Model Database

The model database stores mathematical models of all kinds of animations. The models are coded using PROTO nodes. There are two kinds of interfaces in a model, named attribute interface and data interface. The attribute interfaces provide parameters and attributes of the model for the interface module to adjust. The data interfaces define quantities and types of input/output data in a model.

5. Example of Liquid's Surface Lifting Animation

Liquid's surface lifting animation is used to simulate the process of streaming liquid into a container. In this animation, liquid's surface in a container moves horizontally up or down smoothly. Liquid's shape is fit to the container's edge, which is usually an irregular shape.

The container in this example is a conical flask. We choose an Extrusion node as the base of the shape object. The shape object attributes are shown in Table 1.

Table 1 List of The Shape Object Attributes

Name	Data Type	I/O Type	Explanation
evaluator	SFNode	eventOut	Model to calculate animation
tick	SFTime	eventOut	Time tick to update animation
r	SFFloat	eventOut	Radius of cross section
dest_spine	MFVec3f	eventOut	Destination spine points
dest_scale	MFVec2f	eventOut	Scale on each destination spine
orign_spine	MFVec3f	eventOut	Original spine points
orign_spine	MFVec2f	eventOut	Scale on each original spine points
set_crossSection	MFVec2f	eventIn	Set cross section
set_spine	MFVec3f	eventIn	Set spine points
set_scale	MFVec2f	eventIn	Set scale

In Table 1, evaluator attribute is the mathematical model type in the model database. This attribute specifies which model is used in the animation. On a tick being sent out, the generator calculates a set of animation data using the mathematical model and then sends them back to the shape object to display the next frame of animation. R defines radius of the shape object's bottom cross section. Dest_spine and dest_scale define the final object shape at the end of animation. Similarly, orign_spine and orign_scale define the original shape at begining of animation. Set_crossSection is an eventIn to change cross section. Similarly, set_spine and set_scale eventIns are separately used to set spine and scale of the shape object.

The mathematical model of liquid's surface lifting animation is Extrusion_Spine_Point, i.e., value of evaluator attributer of the shape object is node Extrusion_Spine_Point. Interface of this model is shown in Table 2.

Table 2 Interface of Extrusion_Spine_Point Model

Name	Data Type	I/O Type	Explanation
r	SFFloat	eventIn	Radius of cross section
dest_spine	MFVec3f	eventIn	Destination spine points
dest_scale	MFVec2f	eventIn	Scale on each destination spine
orign_spine	MFVec3f	eventIn	Original spine points
orign_spine	MFVec2f	eventIn	Scale on each original spine points
tick	SFTime	eventIn	Time tick to update animation
set_crossSection	MFVec2f	eventOut	Set cross section
set_spine	MFVec3f	eventOut	Set spine points
set_scale	MFVec2f	eventOut	Set scale

Most items in the interface of Extrusion_Spine_Point model is correspondence to the items in the shape object attributes. Note that the first five items are categorized to the attribute interface, while the last four items the data interface.

On dynamic routing, a routing table shown in Table 3 is created. $AO represents the animation shape object, while $Ev represents the mathematical model node mounted dynamically in the generator. Five prior routes are responsible for transferring animation parameters. When $AO sends a time tick to $Ev using the sixth route, a frame of the animation is calculated and sent back to $AO via the last three routes.

Table 3 Dynamic Routing Table

Route $AO.r To $Ev.r
Route $AO.dest_spine To $Ev.dest_spine
Route $AO.dest_scale To $Ev.dest_scale
Route $AO.orign_spine To $Ev.orign_spine
Route $AO.orign_scale To $Ev.orign_scale
Route $AO.tick To $Ev.tick
Route $Ev.set_crossSection To $AO.set_crossSection
Route $Ev.set_spine To $AO.set_spine
Route $Evaluator.set_scale To $AnimObject.set_scale

6. Conclusion

Fig. 4 is the screen copies of the liquid's surface lifting animation.

We have presented a mechanism which can be used to implement many animations used in iVRLab. This mechanism can improve reusability and reduce code size while keeping the reality of animations. It also reduces the difficulty of animation developing by decoupling the animation calculating module and the animation mathematical model. It is a very useful tool in the design of iVRLab.

(a) (b) (c) (d)

Fig. 4 Screen Copies of Liquid's Surface Lifting Animation

References

[1] I. J. Palmer, *Dynamic Behaviours for Computer Animation: the Use of Java*, Computer Animation '97, 1997, pp. 151–156.

[2] F. S. Tamiosso, A. B. Raposo, L. P. Magalhacs, and I. L. M. Ricarte, *Building Interactive Animations using VRML and Java*, Computer Graphics and Image Processing, 1997. Proceedings, 1997, pp.42–48.

[3] C.Elcacho, A.Schafer, R.Dorner, and V.Luckas, *Performing 3D Scene and Animation Authoring Tasks Efficiently: An Innovative Approach*, Computer Graphics International, 1998. Proceedings, 1998, pp. 242–244.

[4] International Organization for Standardization, *The Virtual Reality Modeling Language*, April 1997. ISO/IEC DIS 14772-1, founded at http://www.web3d.org/technicalinfo/specifications/vrml97/index.htm.

[5] Apkarian, J., and Dawes, A., *Interactive control education with virtual presence on the Web*, Proceedings of American Control Conference, 2000, Volume: 6, pp. 3985–3990.

[6] Schmid, C., and Ali, A., *A Web-based system for control engineering education*, Proceedings of American Control Conference, 2000, Volume: 5, pp. 3463–3467.

[7] Dongil Shin, En Sup Yoon, Sang Jin Park, and Euy Soo Lee, *Web-based interactive virtual laboratory system for unit operations and process systems engineering education*, Computers and Chemical Engineering, 2000, 24, pp. 1381–1385.

[8] Ball, J., and Patrick, K., *Learning about heat transfer-"Oh, I see" experiences*, 29th Annual Frontiers in Education Conference, 1999, Volume: 2, pp. 12C5/1-12C5/6.

[9] John T. Bell and H. Scott Fogler, *Vicher: A Prototype Virtual Reality Based Educational Module for Chemical Reaction Engineering*, Computer Applications in Engineering Education, 4(4), 1996.