An Auto-Stereoscopic VRML Viewer for 3D Data on the World Wide Web

Shinji Uchiyama, Hiroyuki Yamamoto, and Hideyuki Tamura
Mixed Reality Systems Laboratory Inc.
6-145 Hanasakicho, Nishi-ku, Yokohama 220-0022, Japan

Abstract. This paper describes an auto-stereoscopic VRML viewer for a stereoscopic display without any glasses called "Rear Cross Lenticular 3D display (RCL3D display)." By presenting a virtual environment directly on the 3D display, people feel the depth of environment just as the physical model. In the "net-surfing," this viewer automatically generates a pair of stereoscopic images from VRML2.0 data on WWW sites and composes them into one image for the 3D display. Operations such as the walk-through and interaction in a virtual environment are provided. In VRML2.0 data, information to generate the stereoscopic images, such as a base line length and a convergence angle, is not represented. Thus, this viewer provides a way to dynamically adjust such parameters by the operator in order to get natural depth feeling.

1. Introduction

The field of 3D computer graphics is rapidly becoming more popular. Recently, VRML (Virtual Reality Modeling language [1]) has been standardized. VRML has become popular as the language for creating 3D spaces on the internet and WWW. To display a 3D space described with VRML2.0, a VRML viewer is necessary. Popular VRML viewers [2][3] can render a 3D space, however, they ultimately only display a 2D view using a normal monitor.

On the other hand, investigations of stereoscopic display are promising in which an observer can feel the depth of space. We have developed a stereoscopic display called "Rear Cross Lenticular 3D (RCL3D)" display, in which an observer can feel the depth of objects without any special eye-glasses [4]. We have further applied this display to the WWW environment and developed a VRML viewer through which an observer can directly see 3D images described with VRML. This paper describes the method for generating 3D view, setting and adjusting stereoscopic parameters, and functions as a VRML2.0 viewer.

2. Principle of Rear Cross Lenticular 3D Display

This section briefly describes our 3D display without any eye-glasses used as a display unit for our VRML viewer. Refer to the reference [4] for details about this display.

The RCL3D display has particular configuration of two lenticular lenses (lenticular H and lenticular V) and a checkered pattern mask between a TFT liquid crystal (LC) panel and back light. The lenticular V separates an image into left and right eyes by changing the direction of illumination from the back light for each scanning line. The lenticular H prevents generation of crosstalk by collecting light from the opening of the checkered pattern mask onto one of the scanning lines, and enlarging three dimensional viewing area in vertical direction by making the light from the opening diffused to an observer.

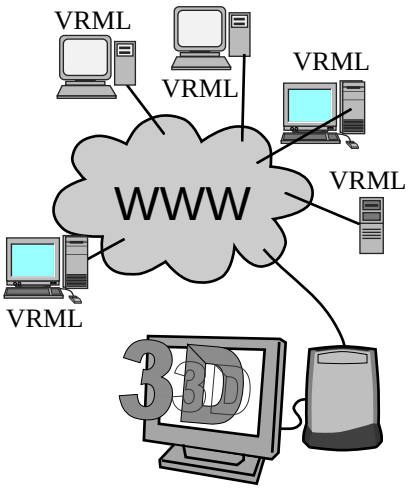


Figure 1 VRML Viewer for the WWW

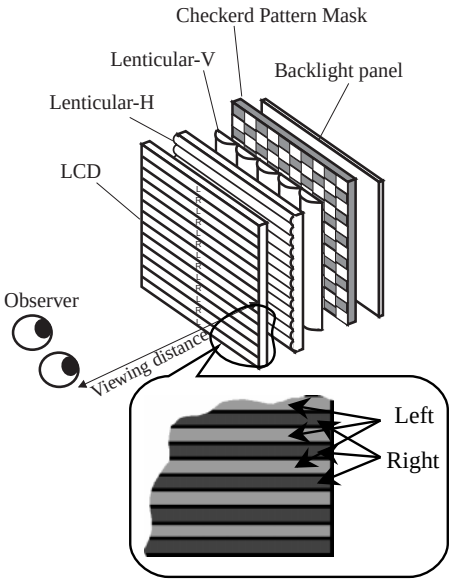


Figure 2 Configuration of RCL3D Display

This display casts an image composed from a pair of stereoscopic images for left and right eyes by alternating each image onto the scanning lines so that they consist horizontal stripe pattern on the composed image. This configuration gives us several advantages such as color separation due to vertical stripes of the LC panel, which is often observed in the ordinary lenticular method using vertical stripe image, is not generated, and having compatibility with the Field Sequential Stereoscopic Method that is standard for the LC shutter glasses.

3. Rendering and Composing Stereoscopic Images

3.1 Generation and Composition of Left and Right Images for 3D View

Rendering of stereoscopic images is accomplished by using usual CG technique to render images viewed from two points of eyes (Figure 3). However, we have to compose one striped image from these two images in order to generate a 3D view of virtual space.

The easiest way of the composition is to render left and right images separately and then composing one image after cutting out these images into many stripes. This method requires operations to read two rendered images from the frame buffer and then write the composed image into other area of the frame buffer before displaying 3D image. Thus, time to render two images, to read images from the buffer, to compose and to write into the buffer and resource of frame buffer memory at least for two pages of images are required. This method is unsuitable for real-time rendering, since the data transfer between memories takes considerable time.

In order to reduce the time and required memory, we have adopted a stencil buffer (Figure 4). The stencil buffer is a part of frame buffer for rendering and having flagging function for each pixel in order to control whether to update the buffer value.

In our method, a right eye image is written onto the frame buffer, and then alternating mask pattern, 0 and 1 for every other scanning lines is written onto the stencil buffer. By overwriting the left eye image onto this frame buffer, the striped image can be acquired, since the scanning lines masked by the stencil buffer are not updated. In this method, the time to

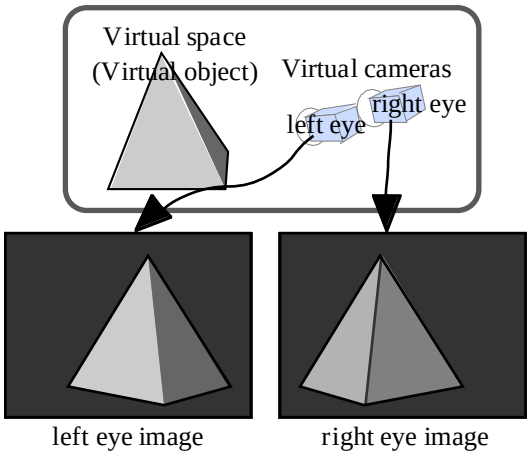


Figure 3 Generating Images for Left and Right

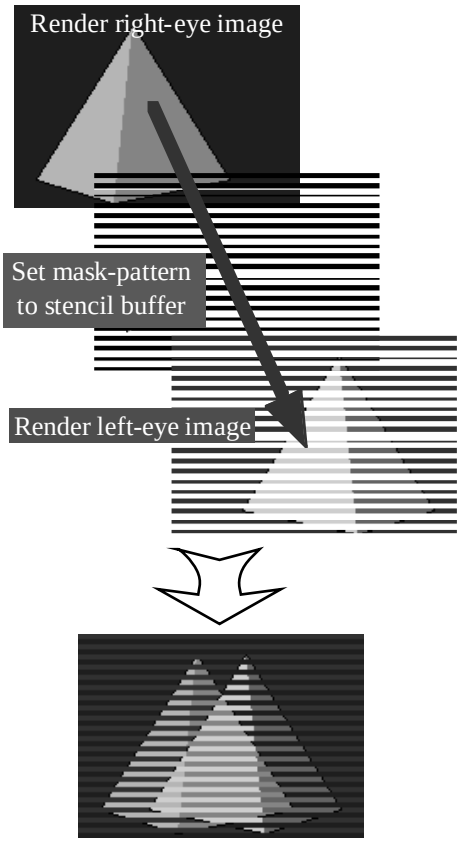


Figure 4 Composing Left and Right Eye Images by Using Stencil Buffer

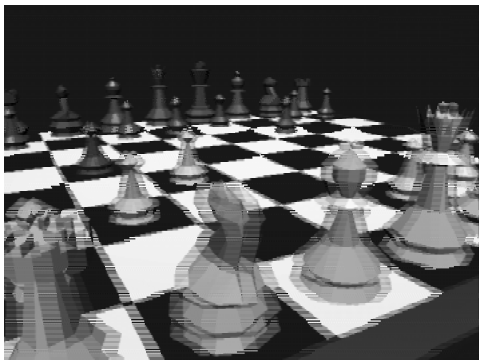


Figure 5 Example of Composed Image

generate the alternating image can be reduced as about twice as of the time to generate an ordinary image for a 2D monitor. Figure 5 shows the result of rendering a VRML data. An observer can feel the depth of the scene when the data is displayed onto the RCL3D display. By using this method, we can render about 20 frames of alternating images per second with a SGI's graphic workstation O2 (CPU : R10000 / 175MHz). It is quite fast comparing to the result using the same computer and the same data but without stencil buffer, which is almost three frames per second.

3.2 Resizing Viewer Window

Generally, the 3D display uses full field of display area in order to obtain the broadest viewing angle. However, our viewer is implemented with a function to set the size and position of the display window, since we assume this viewer is used in 'net-surfing' with a WWW browser. This requires that the user can change the size of the window or its layout. And our viewer is realized as the helper application for the WWW browser, and can be started from the WWW browser.

For the 3D display, some special managements must be taken even for such simple problems of resizing and repositioning of the window. Since the scanning lines of the 3D display are fixed on the LC panel, the lines for the right eye must always be for the right eye image. On the other hand, generation of a 3D image is performed for each window. This means that the top line of the window may be for the right eye or may be for the left eye. If the left eye image is cast to the line for right eye and the right eye image is cast to the line for left eye, the observer cannot grasp the 3D feeling. For this reason, our viewer calculates

number of scanning lines from the top of the display to the top of the window, and exchange the display line for the left and right eye images as needed. This can be accomplished by exchanging 0 and 1 of the mask pattern of the stencil buffer shown in Figure 4.

3.3 3D Mouse Cursor

By using VRML2.0, some kinds of interaction, such as to make the observer possible to manipulate an object, can be written. Since VRML2.0 supports this kind of functionality, our viewer has a function to move an object in the 3D image using a mouse. In this case, how to display a mouse pointer becomes a problem. The mouse pointer in the ordinary windowing system is geometrically fixed on the LC panel. When an observer can feel the depth of a scene in the window of our viewer, he/she can merge the right and left images, but cannot merge the mouse cursor as a single pointer. In our viewer, the mouse pointer is displayed as a 3D object and moving along the surface of the rendered space, as shown in Figure 6.

4. Adjusting Stereoscopic Parameters

4.1 Problems on Viewing VRML Data as 3D

In order to experience virtual space with two dimensional display, we have to render the space after setting up a position, direction, moving speed and angle of the view point in the virtual space. These conditions affect how an observer feels he/she is in that space. The parameters other than the moving speed can be written in a VRML data.

On the other hand, in order to experience virtual space with three dimensional display, we have to set up some other parameters such as the distance between left and right eyes (base-line-length) and its convergence angle before rendering stereoscopic images. These

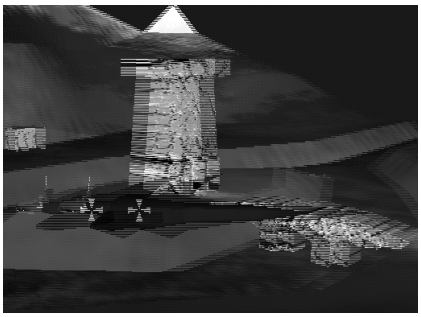


Figure 6 3D Pointer

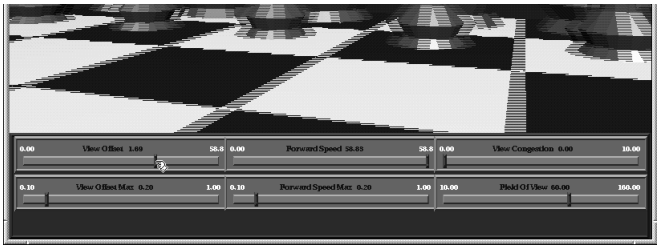
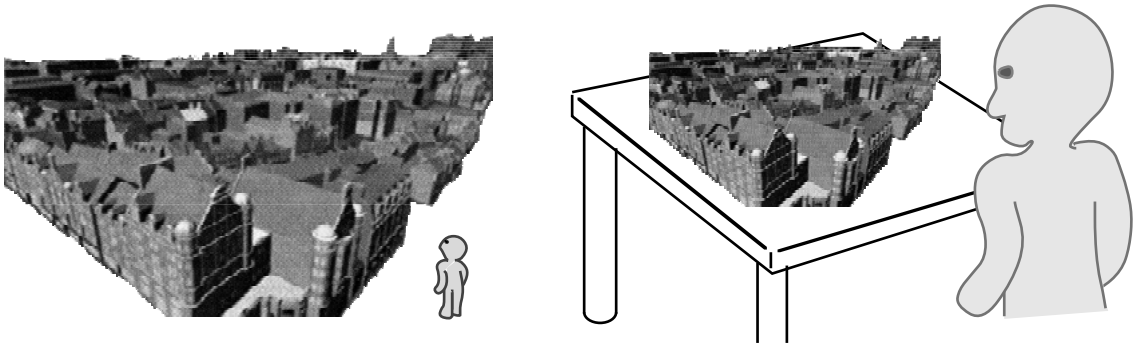


Figure 8 GUI for adjusting Parameters



(a) Case of Actual Town (b) Case of MiniatureTown on a Desktop

Figure 7 Size of Virtual Space and Observer

conditions about 3D view (3D view parameters) greatly affect the feel of depth in the virtual environment. For example, just by changing these parameters, an observer can feel the same virtual space data of a town as an actual town (Figure 7(a)) or models on a desktop (Figure 7(b)). The level of feel is decided by the intention of the designer and the intention of the observer.

(1) Intention of the Designer

According to the standard of VRML2.0, the virtual space must be written using metric scale in VRML2.0. Therefore, the designer can consider how his/her object be felt by an observer while setting the base-line-length as a human average of 65 mm and the convergence angle as 0 degree. However, actual many VRML data are based on the other scales such a normalized coordinate, since they don't have their own actual size in the physical world. Therefore, it is impossible to generate only one set of 3D view parameters for all the VRML data. And also, the standard of VRML2.0 is not considered for displaying on a 3D display and there is no way to describe 3D view parameters in the data.

(2) Intention of the Observer

Suppose that an observer seeing large scaled virtual space of a town decides to come nearer to one building in order to see its details. In this case, it becomes impossible to merge left and right eye images from a certain point of the movement of the observer, and the observer looses 3D feel of the building. As this example, the 3D feel of the observer is also decided based on the data itself and the intention of the observer.

4.2 Adjustment of 3D View Parameters

By considering the above facts, we have decided to calculate initial values for the 3D view parameters from the VRML2.0 data and to provide a way to alter these parameters according to the intention of an observer so that the observer is presented 3D feel he/she want.

The initial value of the base-line-length is obtained from a value calculated based on the whole size of the virtual space by multiplying some adequate constant, and the convergence angle is 0. This is not an ideal method, since the size of the virtual space cannot be used as a base to decide the size of an observer, if the virtual space expresses whole city.

An observer of the virtual space can manually change these 3D view parameters. In our viewer, certain key operation opens an operation panel shown in Figure 8. The two slide bars at the left of the panel are for the base-line-length, the two slide bars at the middle are for speed of walk-through, the upper right bar is for convergence angle. The observer can also change the viewing angle by the slide bar at the lower right of the panel.

5. Implementation of VRML2.0 Functions

Since the purpose of our viewer is to directly read VRML data in the Internet, communication with HTTP is inevitable. VRML is not only a format to describe static geometrical shapes but a language to express virtual space in which an observer can perform some interactive operation. Thus, we can describe both static geometrical shapes in a static space and dynamically changing or interactive spaces. Therefore a viewer for VRML2.0 must perform their functions to give an observer satisfactory display and experience.

Actually, our viewer implements various functions that can be written in VRML2.0. Table 1 shows main functions of VRML2.0 that can be performed in our viewer.

Table 1 Inplemented Functions

Functions required to the VRML2.0 viewer	Implemented	Description
Communication	Data acquisition with HTTP	Acquires VRML data and other related data such as image data for texture from the WWW.
	URL embedding	Makes it possible to embed URLs anywhere in the VRML2.0 data and to set hyperlinks to other sites. Also makes it possible to designate partial space images and texture images which consists the space using URLs.
Dynamic scene expression	Interpolator	Expresses key frame animation or gradual color transition.
Interaction	Sensor node	Acquires events from interaction with an observer in order to make it possible to open a door with mouse click or play some sound when an observer come near to some objects.
Propagation of events	Routing function	Propagates events generated at the sensor nodes to various objects in the space in order to make it possible to change the state of space according to events.

6. Conclusion

By applying our viewer onto the WWW browser, an observer can directly see virtual 3D spaces in the Internet in the three dimensional way.

Since there is no way to describe 3D view parameters in VRML2.0, our viewer uses expedient way to decide initial values and provides a way to change these parameters manually. Some methods are required in VRML to describe intention of a designer if it is used as a language to describe virtual spaces displayed in three dimensional way. Though it is, the way to change parameters is also required in order to reflect intention of observers.

Our viewer has been designed for the Internet, but it can also be used in an intranet without any modification. For example, you can view 3D CAD data to confirm its final shape.

We have already reported a method to construct a space where geometrical models and ray-space data co-exist [5][6]. Since the method renders a space based on the VRML, it is easy to apply our viewer to this method.

References

[1] http://www.vrml.org/

[2] http://www.cosmosoftware.com/

[3] http://www.pc.sony.co.jp/int3d/cpb.htm

[4] H. Morishima, *et al.*: “Rear cross lenticular 3D display without eyeglasses,” Proc. IS&T/SPIE's 10 th Annual Symposium, Vol.3295, pp.193-222, 1998.

[5] S. Uchiyama, *et al.*: “Building a cyberspace by mixing real data based on ray-space theory with computer graphics models,” Proc. 3D Image Conference '96, pp.13-18, 1996 (In Japanese).

[6] S. Uchiyama, *et al.*: “Collaborative CyberMirage: a shared cyberspace with mixed reality,” Proc. VSMM'97, pp.9-17, 1997.