Design of 3D Scene Visual Communication Modeling Based on Virtual Reality Graphics Rendering Framework

Jintao Zheng, Qian Liu*
Shandong Xiehe University, Jinan, Shandong 250107, China

*Corresponding author: liuqian5202@aliyun.com

Abstract. Virtual reality is a kind of computer simulation system which can create virtual environment. People can interact with the three-dimensional environment generated by computer. In this paper, the real-time computer graphics technology, three-dimensional modeling technology and binocular stereo vision technology in virtual reality technology are studied, and the binocular stereo vision optical system is designed. This paper studies the visual communication modeling design of 3D scene under the framework of virtual reality graphics rendering. In this paper, we design and make a three-dimensional model, and develop a virtual indoor scene application. In this paper, the 3D model is used to build the scene, and several commonly used rendering technologies such as texture mapping, material, lighting calculation, transparent effect and shadow calculation in unity engine are used to render the scene. The experimental results show that this method can improve the design efficiency and the accuracy of design matching.

Keywords: Virtual Reality, Binocular Vision, Graphics Rendering, Visual Communication.

1. Introduction

In the era of rapid development of science and technology, virtual reality technology is also developing rapidly, and has been applied in many industries, such as military simulation [1], car driving [2-3], 3D game entertainment, 3D film production, virtual teaching and so on. Because the virtual environment or scene of the application of virtual reality technology is mainly generated and controlled by the computer, the realization process is closely related to graphic image design technology and visual communication design technology. Therefore, for different real environments, we need to design different virtual scenes to express different contents [4].

Nowadays, machine vision, artificial intelligence and other technologies are developing rapidly. Artificial intelligence can be compared to a thing that creates and receives perception. Then it can be simulated and trained in the virtual reality environment. With the increasingly complete interactive tools and means, the behavior of machines will gradually converge with that of human beings [5]. It is conceivable that the creation, rendering and visual communication of 3D graphics play an important role in this virtual reality environment.

As we all know, the Internet, especially the mobile Internet, has greatly changed human behavior. Facebook's Zuckerberg believes that virtual reality will become the next generation of social tools.
Virtual reality devices or application products may become the next mobile device substitutes for mobile phones. For the current virtual reality and artificial intelligence (especially weak artificial intelligence) applications, their application scenarios are also related to entertainment, so after long-term development, we have reason to believe that people will spend the same entertainment time on experiencing virtual reality and artificial intelligence.

Even in other fields, virtual reality also has a wide range of use scenarios and application prospects, such as the display and evaluation of internal scenes such as aircraft cabin and factory workshop; a virtual scene teaching system suitable for multiple people can build a different model to teach students in accordance with their aptitude. It is suitable for industrial design, industrial scene roaming and other immersive scene simulation; the online education and entertainment application of virtual reality can make the education mode more real, vivid and interesting, and also make life full of more fun.

It can be predicted that virtual reality and artificial intelligence will lead the next wave of technological changes in the next few decades. For virtual reality, the content of graphics modeling and rendering and the creation of virtual scene are also its core content, and the significance of its research and development is huge.

2. Research on graphics rendering pipeline

Rendering pipeline is a conceptual model in computer graphics. It describes what steps a graphics system needs to perform to render a 3D scene to a 2D screen. The main content of this chapter is to introduce the key technologies in some stages of graphics rendering pipeline, and analyze the 3D graphics algorithm applied in the stage.

2.1. Rendering pipeline

Generally speaking, rendering pipeline is a series of data processing procedures, and converts the application data to the final rendered image. The rendering process is shown in Figure 1. First, set the vertex and attribute data of the geometry in the application client, and then input these data into a series of shader stages for processing. The output content of an element is used as the input content of the next stage, and finally get the image that can be rendered to the 2D screen. Then the rendering pipeline can be divided into several main stages: vertex processing, rasterization, slice processing and output integration.

In the stage of vertex processing, all kinds of corresponding processing, such as conversion operation, will be performed on the vertices and primitives stored in the buffer. The rasterization stage is to transfer the updated primitives to the rasterized cells after cutting, and convert each primitives into a set of pieces. Here, the slice element is defined as a set of data, which is the pixels that can be placed in the frame buffer, but these pixels may also be finally eliminated, and the pixels in the color buffer are updated. The color buffer is defined as a memory space, which stores the pixels displayed on the screen. In the chip processing stage, the chip test is mainly carried out, and then the color value of the chip is determined by the chip shader and various operations. In the output merging stage, the pixels in the color buffer are compared or merged to update the color values of the pixels.
2.2. Vertex processing and 3D observation

Vertex processing and 3D observation perform various 3D geometric transformation operations for each input vertex stored in the vertex buffer, and the vertex processing phase presents a programmable state. The transformation operation based on vertex processing can transform 3D objects from object space to clipping space. Figure 2 shows the flow of transformation pipeline.

![Figure 2 Processing flow of transformation pipeline](image)

The local coordinate system is defined as the coordinate system centered on the object. Each object has a local coordinate system. It can also be considered that each object is defined in its own object space, and multiple objects can be integrated into one world space. If the form of coordinate transformation is [6-8]:

\[
\begin{align*}
    x' &= a_{xx}x + a_{xy}y + a_{xz}z + b_x \\
    y' &= a_{yx}x + a_{yy}y + a_{yz}z + b_y \\
    z' &= a_{zx}x + a_{zy}y + a_{zz}z + b_z
\end{align*}
\]

(1)

\(x', y'\) and \(z'\) are the coordinates obtained by linear transformation of the original coordinates \(x, y\) and \(z\), which is called affine transformation. Translation, rotation, scaling, reflection and cutting are special cases of affine transformation. Any affine transformation can always be expressed as a combination of these five transformations. The transformation process in Figure 2 is also an example of affine transformation.

In the representation of three-dimensional homogeneous coordinates, the three-dimensional translation of coordinate position can be expressed by the following matrix form:

\[
\begin{bmatrix}
    x' \\
    y' \\
    z'
\end{bmatrix} =
\begin{bmatrix}
    1 & 0 & 0 & t_x \\
    0 & 1 & 0 & t_y \\
    0 & 0 & 1 & t_z
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
    z
\end{bmatrix}
\]

(2)

In the 3D scene environment, the model object can be translated by translating the vertex coordinates of the model object.

For 3D rotation operation, the corresponding rotation axis needs to be defined. Firstly, the three-dimensional z-axis rotation equation can be obtained by formula (3), and then the coordinate form is shown in formula (4).

\[
\begin{align*}
    x' &= x\cos\theta - y\sin\theta \\
    y' &= x\sin\theta + y\cos\theta \\
    z' &= z
\end{align*}
\]

(3)
Where $\theta$ is the angle of rotation.

The rotation formula about the other two axes can be obtained by the cyclic substitution of the coordinate parameters $x$, $y$ and $z$ in equation (3):

$$x \rightarrow y \rightarrow z \rightarrow x$$ (5)

Therefore, using equation (5), we can obtain the transformation formulas of rotation around x-axis and y-axis respectively:

$$y' = y\cos \theta - z\sin \theta$$
$$z' = z\sin \theta + z\cos \theta$$ (6)
$$x' = x$$
$$z' = z\cos \theta - x\sin \theta$$
$$x' = z\sin \theta + x\cos \theta$$ (7)
$$y' = y$$

Three dimensional scaling can be represented by the following matrix form:

$$\begin{bmatrix}
    x' \\
    y' \\
    z' \\
1
\end{bmatrix} = \begin{bmatrix}
    s_x & 0 & 0 & 0 \\
    0 & s_y & 0 & 0 \\
    0 & 0 & s_z & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
    x \\
    y \\
    z \\
1
\end{bmatrix}$$ (8)

### 2.3. Slice processing and output merging

After the rasterization operation, a set of fragments has been set for each element, and each fragment has a position aligned with the pixel grid. And the slice element has depth, color, normal and texture coordinates. These attribute values are obtained by linear interpolation during scan conversion. These attribute values can be used to determine the color value of each slice in the slice processing stage, which plays an important role in the quality of the final image.

Texture is a kind of image data that can be applied to the triangle elements in the scene. Texture fills the image data to the surface of the model through the texture unit. Texture mapping needs the corresponding relationship between texture and object surface, that is, addressing the texture unit through the texture coordinates of the entity. In short, each vertex on the primitive is assigned a texture coordinate, which can refer to the texture pixels in different positions of the image to carry out texture mapping on the model surface, as shown in Figure 3.

![Figure 3 Texture mapping diagram](image)
As can be seen from the figure, \((u, v)\) are texture coordinates, \((s, t)\) are texture pixel position coordinates. After texture mapping, the surface of the model can be mapped. However, the texture unit does not correspond to the pixels on the screen, so we need to stretch or shrink the texture map to make the texture map match the model surface. Then, the process of calculating the output color of a slice based on a stretched texture map is called texture filtering. There are two filtering methods: nearest neighbor filtering and linear filtering. Nearest neighbor filtering is the most simple and fast method. It is a nearest point sampling scheme based on magnifying operation. However, this scheme is feasible only when the texture map size is close to the pixel size. Linear filtering can get better effect than nearest neighbor filtering. It applies the weighted average value of texture units around the texture coordinate to the texture coordinate. It is similar to linear interpolation in scan conversion. Bilinear interpolation can get better effect and is used more often.

Under normal circumstances, the range of texture coordinates is specified in the range of 0.0 - 1.0, which can form a mapping relationship with the texture units in the texture map. When the texture coordinates are out of range, the texture can be processed in surround mode. Surround mode is divided into repeat and intercept. In repeat mode, small texture images are repeatedly pasted on the surface of the model. In the interception mode, when a large texture is processed, the texture map can be intercepted. The wrapping mode of texture has a great influence on how to filter the edge of texture map. In the case of nearest neighbor filtering, surround mode does not work because the texture coordinates are aligned with the texture cells.

3. 3D modeling
Three dimensional objects in the scene are usually generated by three-dimensional modeling software, and can also be generated by other different technologies. In this paper, with the help of the 3D modeling and rendering software of 3ds max, and according to the representation method of 3D objects, the 3D model of indoor scene is created and designed.

3.1. Grid modeling and model design
Mesh modeling can directly edit points, edges, faces, polygons and elements [38]. By editing these primitives, fine processing of object model can be realized, so as to establish the required shape model. Sometimes not all objects in the scene need to be accurately modeled, so mesh modeling is the first choice.

When we create or select an object, we can choose to edit the mesh. The edit mesh is completed by entering each sub object to edit and modify. Corresponding to different secondary objects, 3ds Max provides different editing modes, including vertex mode, edge mode and face mode. Vertex mode focuses on changing the relative position of each control vertex to achieve modeling, while edge mode focuses on meeting the needs of mesh surface modeling. Using vertex pattern and edge pattern to assist surface pattern modeling is the most commonly used method of mesh modeling.

Design idea: firstly, create a chamfered cuboid, select the vertex mode of the Editable Mesh, then set the attenuation coefficient to 150 mm, shrinkage coefficient to -0.01 and expansion coefficient to 0.55, finally select the vertices of the cuboid surface in all directions, stretch or extrude them, and gradually form the pillow model.

Because the surface is defined by the control vertices, when optimizing the model, the vertices can be created and deleted at the exact position. The created vertex becomes a part of the object and is a basic element of creating a new face. Deleting a vertex can quickly remove some unnecessary meshes, that is, all the faces sharing the vertex are deleted.

3.2. Patch modeling and model design
Patch modeling is to make a smooth surface with multiple patches. Patch refers to the abbreviation of Bézier surface. In 3ds max, patches are divided into quadrilateral patches and triangular patches. Patch object is an independent model type, which is defined by Bézier spline curve. The vertex of patch object is the endpoint control point of Bézier spline curve, and the vector handle of patch object
is the tangent corresponding to the spline vertex. The biggest advantage of patch modeling is that it can produce a shape with smooth surface and consistent with the object contour with little detail.

The editing operations of patch modeling include vertex mode, edge mode, patch mode and element mode. Vertex layer is the main layer of patch modeling, because the vector handle can only be accessed through vertex layer. Obviously different from the mesh vertex, adjusting the vertex and its vector handle on the patch will have a great impact on the surface of the patch, which is also the feature of patch modeling. In vertex mode, the transformation of the vector handle directly affects the curvilinearness of the two sides sharing the point. Adding patches is the main function of edge mode. Patch mode is mainly used to subdivide and stretch patches. In element mode, it is mainly to complete the process of merging other patch objects, and control the mesh density of the whole patch object to get a better view or rendering effect.

Design idea:
1) Create two cuboids, one of which is used as the bed board, and carry out the Boolean operation of the compound object in the lower part of the cuboid to design the effect of the bed leg. The other is used as the back plate of the head of the bed, and a bend modifier is added to bend it.
2) Select the patch grid and create a quadrilateral patch as the sheet model, then place it on the top of the mattress, set the mesh smooth, select the control vertices on both sides of the patch to move downward along the y-axis direction, and select the control points on the surface above the patch to move upward to realize the concave convex feeling.
3) Each model and pillow model are combined to get a bed model.

4. Principle of stereo vision
Stereo vision is often used to refer to the perception of depth and three-dimensional structure based on visual information, which is obtained by human eyes, also known as binocular stereo vision. Because the human eyes are located on different sides of the face, binocular vision will produce two different images, which are processed differently in the visual cortex of the brain, resulting in depth perception. When observing a real three-dimensional scene with both eyes, different images can be presented to each eye through the lens in HMD. In this case, the perception of depth is also called "stereo depth".

Stereoscopic view is the core of interactive vision in VR system, through which people can experience the realistic virtual world. Stereo images are generated in pairs. For a pair of different angles of the same scene, they correspond to a person's visual angle of observing the object itself. Stereo rendering is a two-dimensional projection technology for displaying discrete three-dimensional sampled data sets. Stereo vision is an important binocular vision ability for people. It gives people a direct depth perception, which can be stimulated by presenting a stereo view to each eye from different perspectives. Roughly speaking, animals and people with binocular superimposed light fields have this ability. Stereo vision can also be considered as a linear intersection problem. When light is emitted from the light source, a small part of the light will be reflected on the object and rebound towards the human eye. A weaker beam of light enters the pupil of the human eye, where it ends its journey. At this time, there are two information needs to be processed: the position of the light source entering the eye, and the information of the light shining on the retina of the eye. The problem with the visual cortex and the processing part of the brain is to determine where the light comes from. To solve this problem, we need to determine the direction of light in the eyes. Suppose the light enters the eye in a straight line, and the visual cortex traces the light in the opposite direction. It's like this for both eyes, and it determines the focus of the two lines. If the light does move in a straight line, then the intersection is where the light is reflected from the object. By solving the problem of crossing the lines of each pair of light reaching both eyes, our brain will reconstruct the depth image we see. So before we consider how the lens works, we can easily project the virtual object onto the display, and then fill the corresponding pixel array with the color value of the object. The light emitted by the pixels on the screen will be projected into the eyes.

Although the basic principle can be well understood through the intersection of lines, it is not used in a typical rendering application. A typical grid based application uses a projection matrix to project
geometric objects, present the image on the display, and then map it to the eyes. There are two common methods to realize projection, one of which is the "toe in" method. As shown in Figure 4, in this kind of projection, the cameras of the left eye and the right eye are pointed to a single focus. The image created by this method will have a sense of three-dimensional, but the vertical parallax it introduces will increase the degree of discomfort. The vertical parallax introduced increases from the center of the projection plane and becomes more obvious with the increase of camera aperture.

![Figure 4 Projection schematic diagram of Toe-in method](image)

5. Conclusion
For the establishment and rendering of 3D scene, it is necessary to understand the rendering pipeline and the process of each stage. This paper first introduces the flow of graphics pipeline as a whole, and briefly describes the concept of shaders used in each stage. Then we mainly study and analyze the 3D graphics algorithm in the vertex processing stage and 3D observation, and discuss the space conversion of objects in 3D scene. Finally, the techniques used in the rendering pipeline's element processing stage are studied and briefly introduced. By studying the whole rendering pipeline, we can fully understand the rendering process of 3D scene.

Acknowledgements
The research was financially supported by “2019 schools First-rate course”.

References
[1] Suzuki, Q. A corpus-based research toward creating teaching system for medical/nursing English: an analysis of academic papers on clinical surgery. Journal of the Royal Society Interface, 2010, 7 suppl 5(49):S501-S502.
[2] Yang, N. D. Developing a web-based learning portfolio system for English teaching and assessment. Iee transactions on power delivery, 2006, 7;2(2):729-734.
[3] Ono Y , Ishihara M , Yamashiro M . Blended Instruction Utilizing Mobile Tools in English Teaching at Colleges of Technology. Electrical Engineering in Japan, 2015, 192(2):1-11.
[4] Rivo M L , Keller D R , Teherani A , et al. Practicing effectively in today's health system: teaching systems-based care.. Family Medicine, 2004, 36 Suppl:S63-7.
[5] Shen J , Joynt G M , Critchley L A H , et al. Survey of current status of intensive care teaching in English-speaking medical schools.. Critical Care Medicine, 2003, 31(1):293.
[6] Carolina Patricia Aires, Hugo F N , Rosalen P L , et al. Teaching of bioethics in dental graduate programs in Brazil. Brazilian Oral Research, 2006, 20(4):285.
[7] Geng, Mitchell, Zhu. Teaching Industrial Ecology at Dalian University of Technology. Journal of Industrial Ecology, 2010, 13(6):978-989.
[8] Bussabamintra Chaluaisaeng. An evaluation of the first year live English teaching materials in diploma level [reading for information I and aural and oral communication at work] for technical and commercial students. International Journal of Radiation Biology, 1995, 68(6):631-45.