Advanced Engineering Forum *Online: 2012-09-26*
ISSN: 2234-991X, Vols. 6-7, pp 1000-1003
doi:10.4028/www.scientific.net/AEF.6-7.1000

Bezier Surfaces and Texture Mapping Using Java 3D

Xinrui Gao

Information Science Technology Institute of Hainan University, Haikou 570228, China

xr_gao2002@yahoo.com.cn

Key words: Bezier surfaces, Texture mapping, Java 3D.

Abstract. By using Bezier surface matrix formula and the algorithms of texture mapping, the texture mapping onto Bezier surface and its control points net were tested. The texture mapping for Bezier surface model that is composed of six Bezier surfaces was tested too. From the testing examples, it is concluded that these texture mapping algorithms are reliable. All algorithms were implemented by Java and Java 3D.

Introduction

Texture mapping is to map a 2D image on a plane or a surface and it makes the surface or plane look colorful and real. The texture mapping looks like dressing one surface with a colorful cloth. In texture mapping, 2D coordinates of one image are mapped on the corresponding 3D coordinates of one surface, like Fig 1. The algorithms of texture mapping are complex. So, in general, texture mapping are used for plane, cylinder, or sphere etc. Texture mapping for Bezier surface is very complex. In following sections, we discuss the Texture mapping of Bezier surface using Java and Java 3D. It uses the Java 3D's basic functions.

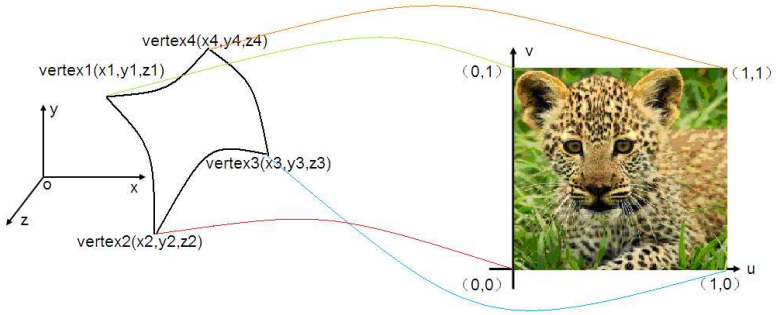

Fig 1. Texture mapping

Three order Bezier surface formulas

Following Eq. 1 is Bezier surface definition. If giving (m+1)*(n+1) control points Pij (i=1, …m, j=1, … n), we could produce one patch of Bezier surface which is m order in the u parameter direction and n order in the v parameter direction. Its formula is as following.

$$S(u,v)=\sum_{i=0}^{n}\sum_{j=0}^{m}P_{ij}B_{i,n}(u)B_{j,m}(v) \qquad 0\le u,v\le 1, \qquad (1)$$

Among Bezier surfaces, the three order Bezier surface is widely used. Each patch of a three order Bezier surface has 16 control points. By adjusting the control points of the Bezier surface, we could change its shape. The formula 1 is hard to use. But its matrix formula is easy to use. The following is three orders Bezier surface matrix formula as Eq. 2.

$$C(u,v)=UMPM^{T}V^{T}$$

$$U=\begin{bmatrix}1 & u & u^{2} & u^{3}\end{bmatrix}$$

$$V=\begin{bmatrix}1 & v & v^{2} & v^{3}\end{bmatrix}$$

$$M=\begin{bmatrix}1 & 0 & 0 & 0\\ -3 & 3 & 0 & 0\\ 3 & -6 & 3 & 0\\ -1 & 3 & -3 & 1\end{bmatrix}$$

$$P=\begin{bmatrix}P_{0,0} & P_{0,1} & P_{0,2} & P_{0,3}\\ P_{1,0} & P_{1,1} & P_{1,2} & P_{1,3}\\ P_{2,0} & P_{2,1} & P_{2,2} & P_{2,3}\\ P_{3,0} & P_{3,1} & P_{3,2} & P_{3,3}\end{bmatrix}$$

$$0\le u\le 1,\quad 0\le v\le 1 \qquad (2)$$

The u and v are two parameters. Pij is the control points. UT and VT are transposed matrices U and V.

If we change the coordinates of the control points, the surface shape will be changed. Following are two examples as Fig 2 and Fig 3. Fig 2 is the original control points and its Bezier surface. But Fig 3 is the changed control points and its Bezier surface. The two Bezier surfaces are different.

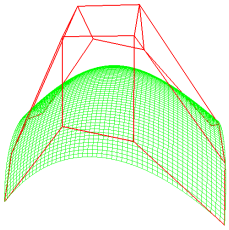

Fig 2. The original control points

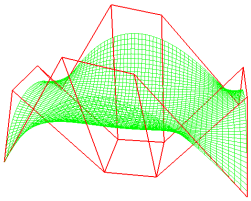

Fig 3. The changed control points

Three order Bezier surface texture mapping

Before texture mapping, we divide u, v parameters of the image and u, v parameters of the three order Bezier surface in the same way, and match the Bezier surface division points with the image division points, like Fig 4. For the three order Bezier surface, we calculate every division point's x, y, z coordinates, normal, and its texture coordinate. These are needed for texture mapping. The magnification filter function is used when the pixel being rendered maps to an area less than or equal to one texel. The minification filter function is used when the pixel which is being rendered maps to an area greater than one texel.

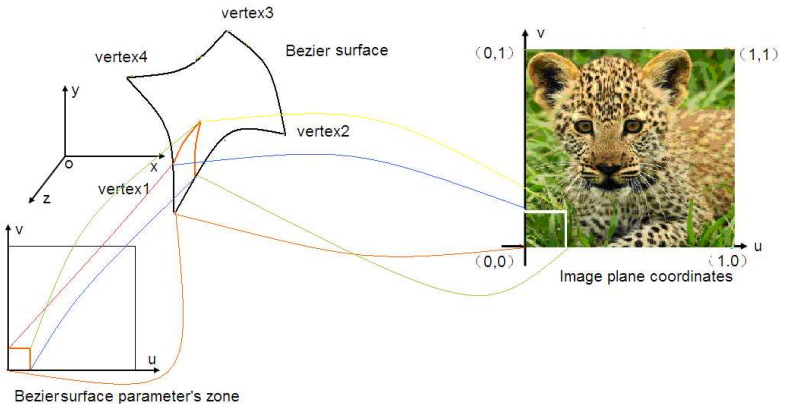

Fig 4. Three order Bezier surface's u, v parameters and the image's u, v parameters

This function is implemented by Java and Java 3D programming. Following is the part of this program. The classes about textures in Java 3D such as TextureLoader, ImageComponent2D, Texture2D, and TexCoord2f are used.

```
Appearance app = new Appearance();
TextureLoader loader=new TextureLoader("maotouying.jpg",2, this);
ImageComponent2D image=loader.getImage();
Texture2D texture=new Texture2D(Texture.BASE_LEVEL,Texture.RGBA,
image.getWidth(),image.getHeight());
texture.setImage(0, image);
texture.setEnable(true);
//set The magnification filter function and The minification filter function
texture.setMagFilter(Texture.BASE_LEVEL_LINEAR);
texture.setMinFilter(Texture.BASE_LEVEL_LINEAR);
app.setTexture(texture);
…….
//set order number of points on Bazier surface
BezierQuadsurfaceface.setCoordinate(c, A);
BezierQuadsurfaceface.setCoordinate(c+1, B);
BezierQuadsurfaceface.setCoordinate(c+2, C);
BezierQuadsurfaceface.setCoordinate(c+3, D);
//set the normal of each point
BezierQuadsurfaceface.setNormal(c, n);
BezierQuadsurfaceface.setNormal(c+1, n);
BezierQuadsurfaceface.setNormal(c+2, n);
BezierQuadsurfaceface.setNormal(c+3, n);
//set the texture coordinates of each point
TexCoord2f texCoords=new TexCoord2f(i*1.f/n0,1.f-j*1.f/n0);
BezierQuadsurfaceface.setTextureCoordinate(0,c,texCoords);
texCoords=new TexCoord2f((i+1)*1.f/n0,1.f-j*1.f/n0);
BezierQuadsurfaceface.setTextureCoordinate(0,c+1,texCoords);
texCoords=new TexCoord2f((i+1)*1.f/n0,1.f-(j+1)*1.f/n0);
BezierQuadsurfaceface.setTextureCoordinate(0,c+2,texCoords);
texCoords=new TexCoord2f(i*1.f/n0,1.f-(j+1)*1.f/n0);
BezierQuadsurfaceface.setTextureCoordinate(0,c+3,texCoords);
```

Following figures are the results of this program. Fig 5 is the original image. Fig 6 is the front and the back of this Bezier surface which is texture mapped.

Fig 5. Original image

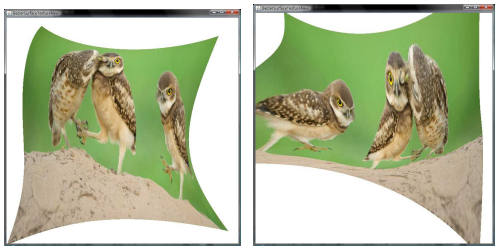

Fig 6. The front and the back of texture mapped Bezier surface

In following example, this algorithm of texture mapping is extended to use in the control points surface of this Bezier surface. Fig 7 is the original image. Fig 8 is the control points' surface of this Bezier surface. Fig 9 is the front and the back of this texture mapped surface. Though this kind of surface is not one plane and not smooth, this texture mapping algorithm is reliable.

Fig 7. Image

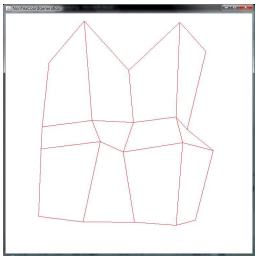

Fig 8.Control points net

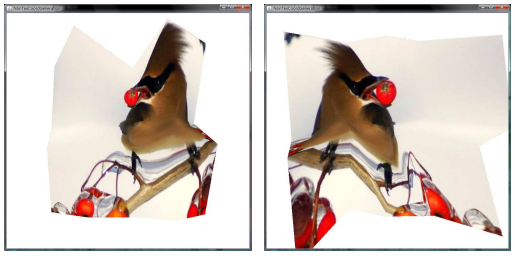

Fig9. The front and the back of Control points

Following example is for the Bezier surface model which contains six three order Bezier surfaces. Fig 10 is the regular model and its' texture mapping. Fig 11 is the twisted Bezier surface model in which two end control points of the three order Bezier surface are merged together. Fig 11 is the texture mapping result and this kind of texture mapping is normal.

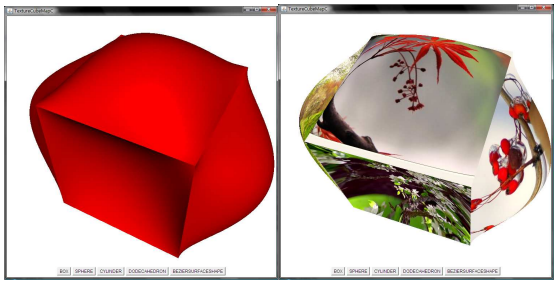

Fig 10. Bezier surface model

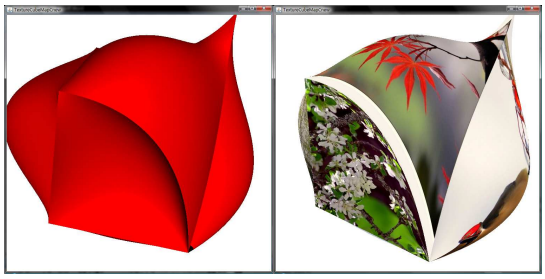

Fig 11. Twisted Bezier surface model

Conclusions

From above texture mapping examples, it is concluded that these texture mapping algorithms are reliable.

Acknowledgement

Acknowledgements for the support of Ph D Scientific Research Foundation of Hainan University, China.

References

[1] Java, Java 3D and Computer Geometry Design in Chenese [M].Gao Xinrui.Publishing House of Electronics Industry (PHEI) of China.2007.8

[2] Computer Graphics using Java 3D and Java 3D [M]. Hong Zhang,Y. Daniel Liang. Prentice Hall.2006.12. New Jersey, USA.

[3] http://www.j3d.org/

[4] http://java.sun.com/developer/onlineTraining/java3d/

[5] http://www.javagaming.org