Journal of Theoretical and Applied Information Technology
10th March 2013. Vol. 49 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

REALISTIC RENDERING IN WEBGL

[1]LI YU, [2]XIAOMENG MAO
[1]Assoc Prof., School of Information Technology, Shanghai Jianqiao University, Shanghai
[2]Lecturer, School of Information Technology, Shanghai Jianqiao University, Shanghai
E-mail: [1]yuli110@126.com , [2]maomxm@163.com

ABSTRACT

HTML5 introduces many new features, in which the Canvas can use WebGL technology to achieve 3D graphics rendering. This paper presents a method to render virtual models in WebGL on browser platform. We use Vertex Shader and Fragment Shader programming to achieve realistic graphics, including Phong illumination model, Phong shading and texture mapping. The advantages of this method are 1) The browser support WebGL natively, so there is no need to install any plug-ins, and it is cross-platform; 2) 3D scene roaming supported (pan, zoom and rotate).

Keywords: *WebGL, Vertex Shader, Fragment Shader*

1. INTRODUCTION

We can use computers to create vivid scene to gain realistic experience [1]. With the development of Internet and virtual reality technology, creating 3D virtual world and providing real-time interactive performance becomes the urgent needs of people on the Internet. Early 3D virtual reality technology include VRML [2, 3] and Java3D [4, 5], yet most of them need to install plug-ins to browsers, which will arouse disgusts to some users and hinder those who are not professional. Meanwhile, there emerge many algorithms to interact with the virtual scene [6, 7].

WebGL is developed by the Khronos Group. Google, Apple, Mozilla, Opera and other companies and organizations are among the members. WebGL runs on browser, currently mainstream browsers including Google Chrome browser, Opera browser, Firefox and Apple Safari browser all support WebGL. WebGL is a JavaScript API, which allows developers to embed interactive 3D graphics in the browser, what's more, 3D graphics support hardware acceleration, so web developers can make use of graphics card to render 3D scenes and models in the browser more smoothly, to create complex navigation and data visualization [8], and the rendering speed is faster than ray tracing algorithm [9]. The browser supports WebGL natively, so there is no need to install any plug-ins, and the browser as a platform for Internet applications is already multi-platform supported, including mobile platform, desktop computers, smart phones, tablet PCs and smart TV.

2. WEBGL OVERVIEW

2.1. Vertex Shader And Fragment Shader

WebGL uses OpenGL ES 2.0 standard, in which programmable shaders bring greater flexibility and adaptability for GPU programming. In polygon-based graphics system, the geometry pipeline converts objects from 3D coordinate to 2D screen coordinate, each vertex is attached to vertex color, normal vector, texture coordinate and other information; rendering pipeline calculates the appearance of each vertex, firstly geometric objects are decomposed into fragments corresponding to the raster scan line, secondly fragments are filled in the frame buffer to generate images [10].

Vertex Shader (referred to as VS below) can be used to perform traditional vertex-based operations, such as using matrix to transform vertex position, vertex color, and texture coordinates, but we can not generate a new vertex in VS.

Fragment is a pixels collection within a polygon on the same scan line. Fragment Shader (referred to as FS below) is commonly used to deal with scene lighting and related effects, such as bump texture mapping. We will use FS to calculate Phong shading in this paper.

There are three commonly used shader variables. "Attribute" is only used by VS, it carries variable passed to it from the program, such as vertex position array; "Uniform" can be used by both VS and FS, it can not be changed, like a constant, such as the modelview matrix and projection matrix; "Varying" is a variable VS passed to FS, such as

Journal of Theoretical and Applied Information Technology
10th March 2013. Vol. 49 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

gl_Position, which carries transformed vertex coordinates.

2.2. Webgl Rendering Pipeline

Figure 1 is WebGL rendering pipeline. On each call to drawArrays function, WebGL Uniform and Attribute variables are passed to VS, VS calculated results are stored in Varying and passed to FS; FS calculates color of each pixel, and stores it in gl_FragColor Varying variable; lastly, the output is written to Frame Buffer, which is the image on the screen.

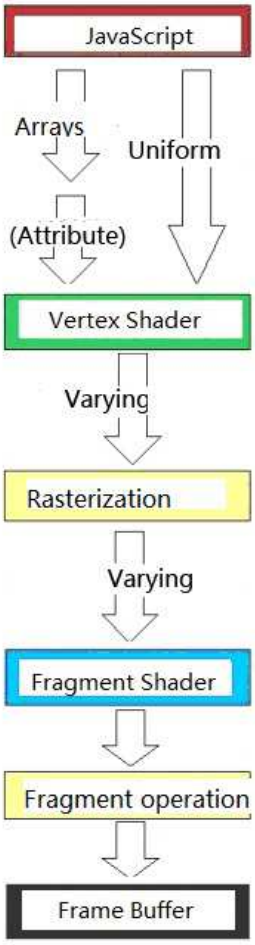

Figure 1. Webgl Rendering Pipeline

3. REALISTIC RENDERING METHOD

One main purpose of computer graphics is to generate pleasing, realistic graphics with computer. To generate photo-like pictures, we need to establish the geometric representation of the scene, and use some kind of illumination model to calculate lighting effects under the hypothetical light source, texture and material properties. In the end, we may hope to interact with the virtual scene freely.

3.1. The Geometric Representation Of The Model

When WebGL read the model, such variables as vertexPositions, vertexNormals, vertexTextureCoords and indices and other information are needed to feed into program. Figure 2 is the wireframe of a model, which displays topology connection clearly. JSON file of the model is as follows:

"vertexPositions" : [-1,-1,2, -1,1,2, 0,2,2, 1,1,2, 1,-1,2, -1,-1,-2, -1,1,-2,0, 2,-2,1, 1,-2,1, -1,-2,-1,-1,-2,-1,1,-2,-1,1,2,-1,-1,2,1,-1,2,1,1,2,1,1,-2,1,-1,-2,-1,1,-2,0,2,-2,0,2,2,-1,1,2,1,1,2,0,2,2,0,2,-2,1,1,-2],

"vertexNormals" : [0.0,0.0,1.0, 0.0,0.0,1.0, 0.0,0.0,1.0, 0.0,0.0,1.0, 0.0,0.0,1.0, 0.0,0.0,-1.0,0.0,0.0,-0.0,0.0,0.0,-1.0,0.0,0.0,-1.0,0.0,0.0,-1.0,-1.0,0.0,0.0,-1.0,0.0,0.0,-1.0,0.0,0.0,-1.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,1.0,0.0,0.0,-0.7,0.7,0.0,-0.7,0.7,0.0,-0.7,0.7,0.0,-0.7,0.7,0.0,0.7,-0.7,0.0,0.7,-0.7,0.0,0.7,-0.7,0.0,0.7,-0.7,0.0],

"vertexTextureCoords" : [0.0,0.0, 0.0,0.66, 0.5,1.0, 1.0,0.66, 1.0,0.0, 0.0,0.0, 0.0,0.66, 0.5,1.0, 1.0,0.66, 1.0,0.0, 0.0,0.0, 0.0,1.0, 1.0,1.0, 1.0,0.0, 0.0,0.0, 0.0,1.0, 1.0,1.0, 1.0,0.0, 0.0, 0.0, 0.0,1.0, 1.0,1.0, 1.0,0.0, 0.0,0.0, 0.0,1.0, 1.0,1.0, 1.0,0.0],

"indices" : [0,1,2, 0,2,3 ,0,3,4, 5,7,6, 5,8,7, 5,9,8, 10,11,12, 10,12,13, 14,15,16, 14,16,17, 18,19,20, 18,20,21, 22,23,24, 22,24,25]

Where "vertexPositions" are x, y and z coordinates of a vertex; "vertexNormals" provide normal of a vertex, which will be used in illumiation step; "vertexTextureCoords" provide s and t parameter in texture mapping step; and "indices" are vertex indices to form a polygon, usually a triangle.

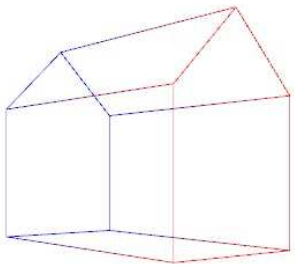

Figure 2. The Wireframe Of A Model

3.2. Phong Illumination Model

WebGL does not support any illumination model directly. This is a defect, but also, it brings more

Journal of Theoretical and Applied Information Technology
10th March 2013. Vol. 49 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

freedom to programmers. We can program in shaders to achieve Phong illumination model. As shown in formula 1, light reflected by non-ideal reflecting surface are composed of ambient reflection component, diffuse reflection component and specular reflection component,

(1)

Where Ka, Kd and Ks are the ambient reflection coefficient, the diffuse reflection coefficient and specular reflection coefficient of the model; Ia is ambient light; Il is spot light source; N is the normal vector, which comes from "vertexNormals"; L is the light source direction; V is the sight direction; R is the specular reflection direction; n is the specular highlight index. Users can set these parameters on the webpage, and then these parameters are passed to WebGL by javascript.

Part of the code of Vertex Shader is as follows:

```
<script id="per-fragment-lighting-vs" type="x-shader/x-vertex"> / / VS
void main (void) {
vPosition = uMVMatrix * vec4 (aVertexPosition, 1.0);
gl_Position = uPMatrix * vPosition;
vTextureCoord = aTextureCoord;
vTransformedNormal = (uNMatrix * vec4 (aVertexNormal, 1.0)). xyz ;
}
</ Script>
```

In Vertex Shader, after projection and modelview transformation, transformed position of every vertex of model is calculated, which is gl_Position. vTextureCoord and vTransformedNormal are transfered to Fragment Shader.

In Fragment Shader, color of every fragment or pixel is calculated, which is gl_FragColor. Fragment Shader is where Phong illumination model is realized. Part of the code of Fragment Shader is as follows:

```
<script id="per-fragment-lighting-fs" type="x-shader/x-fragment"> / / FS
void main (void) {
vec4 textureColor = texture2D (uSampler, vec2 (vTextureCoord.s, vTextureCoord.t));
lightDirection = normalize (uLightPosition - vPosition.xyz) ;/ / point source
vec3 normal = normalize (vTransformedNormal);
float diffuseLightWeight = max (dot (normal, lightDirection), 0.0) ;
vec3 eyeDirection = normalize (-vPosition.xyz) ;/ / point of view is always at (0,0,0)
vec3 reflectionDirection = reflect (-lightDirection, normal);
float specularLightWeight =pow (max (dot (normalize (reflectionDirection), eyeDirection), 0.0), uMaterialShininess) ;/ / specular light
lightWeighting = uAmbientColor + uDiffuseLightColor * uMaterialColor * diffuseLightWeight
+ USpecularLightColor * uSpecularColor * specularLightWeight;
gl_FragColor = vec4 (textureColor.rgb * lightWeighting, textureColor.a);
}
</ Script>
```

3.3. Phong Shading

The illumination model determines the color of a vertex on polygon, while the shading determines the color of a pixel within the polygon. With Phong shading, also known as normal interpolation method, firstly we calculate the normal of the pixel on a fragment by interpolating the normals of polygon vertices, then we use Phong illumination model to calculate the color of the pixel. The bold type in the code above is the pixel's normal, it is a varying variable that VS passed to FS. Figure 3 is an image with lighting and Phong shading. In figure 3, yellow small circle demonstrates specular highlight, and red big circle demonstrates the pleasing effect of Phong shading, with a spot light fairly close to the model.

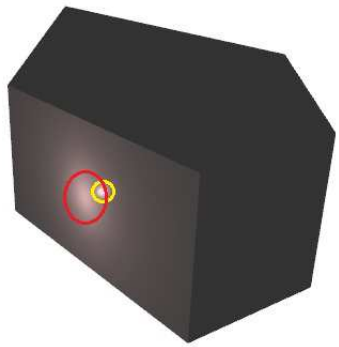

Figure 3. Phong Shading With Lighting

As you know, SMOOTH Shading mode in both OpenGL and DirectX graphics API is just Gouraud Shading (also known as color interpolation), which cannot calculate the shading effect in the red circle in this case. While in Phong shading, since every fragment has its own normal, programmers can generate more refined image with VS and FS.

Journal of Theoretical and Applied Information Technology
10th March 2013. Vol. 49 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.4. Texture Mapping

Texture mapping is a technology to map texture image onto 3D model surfaces. Texture can make a simple geometry to produce a vivid image. Figure 4 is the model with brick texture and lighting, now it looks more realistic. The italics and bold type in the code above is textureColor, which is sampled color from the texture image for each pixel in the fragment, then textureColor is multiplied with the color calculated by illumination, which turns out to be the pixel's gl_FragColor, to be written to frame buffer.

Figure 4. Model With Texture And Lighting

To contrast with figure 4, figure 5 is the same scene with different texture, now it looks more like a wooden house, not a brick one. The fact proves that texture is the most economical way to get photorealistic effect. All we need to do is to load a different image as texture, the geometric representation of the model is the same from the very beginning.

Figure 5. Different Texture

3.5. Scene Interactive

With javascript we can easily achieve 3D scene roaming. In this paper, we use the mouse to control rotation, and PgUp and PgDn on the keyboard to control zoom in and zoom out. Interactive results are shown in Figure 6 and Figure 7.

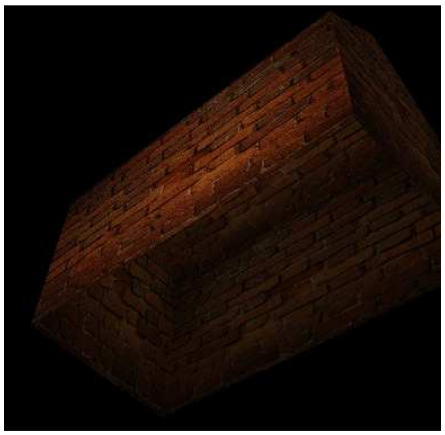

Figure 6. Scene after rotation

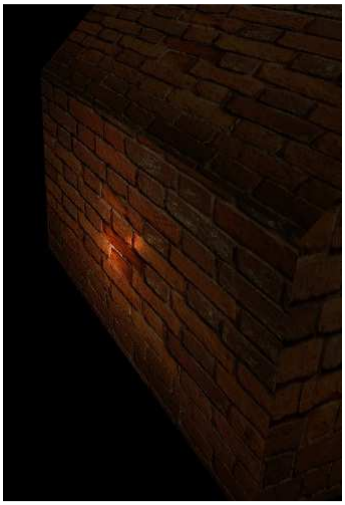

Figure 7. Scene After Zoom In

Firstly, let's analyse mouse interactive. Three varibles are defined and initialized. Three functions are defined to response to mousedown, mouseup and mousemove events. Part of the mouse interactive code is as follows:

```
var mouseDown = false;
var lastMouseX = null;
var lastMouseY = null;
function handleMouseDown(event) {
   mouseDown = true;
   lastMouseX = event.clientX;
   lastMouseY = event.clientY;
}
function handleMouseUp(event) {
   mouseDown = false;
}
function handleMouseMove(event) {
   var newX = event.clientX;
```

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

```
        var newY = event.clientY;
        var deltaX = newX - lastMouseX
        var newRotationMatrix = new okMat4();

    newRotationMatrix.rotY(OAK.SPACE_LOCAL,
deltaX / 10, true);
        var deltaY = newY - lastMouseY;

newRotationMatrix.rotX(OAK.SPACE_LOCAL,
deltaY / 10, true);
        moonRotationMatrix                    =
okMat4Mul(newRotationMatrix,
houseRotationMatrix);
        lastMouseX = newX
        lastMouseY = newY;
      }
```

Secondly, how to realize zoom in and zoom out. Traditionally, we use right key of mouse to control zoom in and zoom out. However, in webpage, right key of mouse is always binded with short cuts already. So keyboard is used instead, which is also a tradition in game playing. Part of the keyboard interactive code is as follows:

```
  var zoom = 0;
     function handleKeys() {
       if (currentlyPressedKeys[33]) {
          zoom-=0.1; // Page Up
       } else if (currentlyPressedKeys[34]) {
          zoom+=0.1; // Page Down
       }
  }
```

4. CONCLUSION

In this paper, we use WebGL to render a virtual house. VS and FS programming are used to achieve realistic graphics, including Phong illumination model, Phong shading and texture mapping, and users can interact with the scene. Since browser support WebGL natively, there is no need to install any plug-ins to enjoy 3D scene, and WebGL is cross-platform. The future work can commence in two ways: 1) to import 3D models generated with 3DMax or others; 2) collision detection and LOD technology to improve real-time rendering performance.

ACKNOWLEDGMENT

This work was partly supported by Shanghai Private Education Research Projects (No. AAM12002), Shanghai Municipal Education Committee Scientific Research Innovation Project(No.11YZ284) and Shanghai Municipal Education Committee Dawn Project (No. AASH0910).

REFRENCES:

[1] Chen Yong, Ma Chunyong, Chen Ge, "VC / OpenGL Virtual Navigation System of Sea Campus", *Computer Aided Design and Computer Graphics*, Vol. 19, No. 2, 2007, pp. 263-267.

[2] Qiu Wei, Zhang Lichen, "Design and Realization of Web3D-based Virtual Scene Roaming", *Micro-computer information*, Vol 23, No 4, 2007, pp. 57-59

[3] Peng Shengze, Ge Wengeng, "Research on Application of 3D Modeling Technique Based on Network in Drawing Education", *Journal of Theoretical and Applied Information Technology*, Vol. 44, No. 2, 2012, pp. 221 – 227.

[4] Yansong Deng, Kaiyu Qin, Guangming Xie, "3-D Space Flight Formation Control for UAVS Based on MAS", *Journal of Theoretical and Applied Information Technology*, Vol. 47, No. 2, 2013, pp. 653 – 659.

[5] He Tonglin, Chang ice, "Java3D Based Virtual Scene", *Computer applications*, Vol. 6, No. 27, 2007, pp. 291-292.

[6] Jin Tonghong, Dou Zhongjiang, "Collision Detection in Large Virtual Scene", *Engineering Graphics*, Vol. 17, No. 1, 2007, pp. 33-36.

[7] Shi Hongbing, Zhang Yibin, "Virtual Scene Path Planning Algorithm for Automatic Roaming", *Computer Aided Design and Computer Graphics*, No. 18, 2006, pp.592-597.

[8] Tan Wenwen, Ding Shiyong, Li Guiying, "Design and Implementation of WebGL and HTML5 Based Web 3D Animation", *Computer Knowledge and Technology*, Vol. 7 , No. 28, 2011, pp. 6981-6983.

[9] Liu Huaxing, Yang Geng, "HTML5 - the Next Generation of Web Development Standards", *Computer Technology and Development*, Vol. 21, No. 8, 2011, pp. 54-58.

[10] Steve Cunningham, *Computer graphics.*, First Edition, Machinery Industry Press (2008)