PJ3-三维场景漫游

罗俊韬 19307110374

一、项目目录与文件说明

目录：`19307110374_罗俊韬`

- `/BaseCode`：在项目提供的基础代码上完成的本次PJ源代码。`/BaseCode`目录下大部分文件与项目提供的基础代码内容相同，仅说明新增的cube loader文件：
  - `/BaseCode/cube.js`：根据任务2.3新增的三维模型loader文件，用于载入彩色立方体。
- `PJ3文档.pdf`：PJ3项目文档。

二、开发及运行环境

- 支持html5和WebGL的Chrome浏览器（版本：112.0.5615.49 x86_64）
- VSCode

三、运行及使用方法

在VSCode IDE中安装Live Server扩展，在VSCode中打开`/BaseCode`目录后，使用Live Server扩展启动本地服务器，然后使用支持html5和WebGL的浏览器打开`localhost:5500/本项目所在目录/BaseCode/3DWalker.html`即可运行（5500为Live Server扩展启动的本地服务器的默认端口）。使用方法参照项目文档中要求实现的键盘交互即可。

四、项目亮点

- **对于ObjectLoader着色器，实现了项目文档要求的Phong Shading的加分项：**

实现Phong Shading后的shader如下：将法向量计算移到Fragment Shader中。

```
initShaders() {
  // Vertex shader program
  let VSHADER_SOURCE = `
      attribute vec4 a_Position;
      attribute vec4 a_Color;
      attribute vec4 a_Normal;
      uniform mat4 u_MvpMatrix;
      uniform mat4 u_ModelMatrix;
      uniform mat4 u_NormalMatrix;
      uniform vec3 u_Color;
      varying vec4 v_Color;
      varying vec3 varu_Color;
      varying vec3 v_Normal;
      varying vec4 vertexPosition;
      void main() {
        gl_Position = u_MvpMatrix * a_Position;
        v_Color = a_Color;
        varu_Color = u_Color;
        v_Normal = normalize(vec3(u_NormalMatrix * a_Normal));
        vertexPosition = u_ModelMatrix * a_Position;
      }`;
```

```
// Fragment shader program
let FSHADER_SOURCE = `
    #ifdef GL_ES
    precision mediump float;
    #endif
    uniform vec3 u_LightColor;
    uniform vec3 u_LightPosition;
    uniform vec3 u_AmbientLight;
    varying vec3 v_Normal;
    varying vec4 v_Color;
    varying vec3 varu_Color;
    uniform vec3 u_LightDirection;
    uniform int u_UseLight;
    varying vec4 vertexPosition;
    void main() {

      float nDotL = max(dot(u_LightDirection, v_Normal), 0.0);
      vec3 u_DiffuseLight = vec3(1.0, 1.0, 1.0);
      vec3 diffuse = u_DiffuseLight * varu_Color * nDotL;
      vec3 ambient = u_AmbientLight * varu_Color;

      if(u_UseLight == 1){
        vec3 normal = normalize(v_Normal);
        vec3 lightDirection = normalize(u_LightPosition - vec3(vertexPosition));
        float nDotL2 = max(dot(lightDirection, normal), 0.0);
        vec3 pointLight = u_LightColor * varu_Color * nDotL2;
        gl_FragColor = vec4(diffuse + ambient + pointLight, v_Color.a);
      }
      else{
        gl_FragColor = vec4(diffuse + ambient, v_Color.a);
      }
    }`;
```

- 任务2.6实现的动画效果：模仿最终效果，实现了一个绕过海宝旋转，高度随着旋转角正弦值改变的动画，代码如下：

```
if (this.entity.objFilePath.indexOf('bird') > 0) {
  // animation
  // get angle by timestamp
  this.elapsed = timestamp - this.lastTime;
  this.lastTime = timestamp;
  // Update the current rotation angle (adjusted by the elapsed time)
  this.angle = this.angle + (this.ANGLE_STEP * this.elapsed) / 1000.0;
  this.angle %= 360;

  // Set the rotation matrix
  this.modelMatrix.setRotate(this.angle, 0, 1, 0);
  this.modelMatrix.translate(1,Math.sin(this.angle * Math.PI/360), 2);
  g_mvpMatrix.concat(this.modelMatrix);
}
```

- 参考项目文档、自学WebGL教材与Sample，成功达到了需要实现的最终效果，没有额外参考网络资料，纯粹独立完成。

五、开发遇到的问题

- 任务2.7——添加移动点光源，遇到了如下两个问题：
  - 需要在Shader中单独为点光源增加一个光源变量，但课本例子中没有。因此一开始并不清楚该如何引入点光源，之后通过尝试解决了该问题。
  - 思考如何将按键'F'事件转换为布尔值，并且传递到render函数中，以此控制使用的shader程序。最后的解决方法是在shader程序中添加了一个int变量，作为布尔值判断使用的shader程序，然后通过render函数新增形参传入。具体代码如下：

```
if(u_UseLight == 1){
  vec3 normal = normalize(v_Normal);
  vec3 lightDirection = normalize(u_LightPosition - vec3(vertexPosition));
  float nDotL2 = max(dot(lightDirection, normal), 0.0);
  vec3 pointLight = u_LightColor * varu_Color * nDotL2;
  gl_FragColor = vec4(diffuse + ambient + pointLight, v_Color.a);
}
else{
  gl_FragColor = vec4(diffuse + ambient, v_Color.a);
}
```

```
render(timestamp, useLight) {
  this.gl.useProgram(this.program);
  this.gl.program = this.program;

  if (this.g_objDoc != null && this.g_objDoc.isMTLComplete()) {
    this.onReadComplete();
  }
  if (!this.g_drawingInfo) return;

  if (this.hasOwnProperty('nextFrame')) {
    this.nextFrame(timestamp);
    this.initPerspective();
  }

  let lightDirection = new Vector3(sceneDirectionLight);
  lightDirection.normalize();
  this.gl.uniform3fv(this.u_LightDirection, lightDirection.elements);
  this.gl.uniform1i(this.u_UseLight, useLight);
```

- 任务2.8——实现场景漫游: 在添加移动时，遇到了配置文件里，Camera的lookat Y值与最终效果不同的问题。通过咨询助教后，修改配置文件解决。

六、项目可能存在的缺陷

项目已实现本次PJ要求的所有基础功能与一项附加功能，目前没有发现明显缺陷。