**实 验 报 告 二**

**（2021-2022学年第一学期）**

**计算机图形学**

**（Computer Graphics）**

|  |  |  |
| --- | --- | --- |
| 学生姓名：陈卓文 | 学号：201936380215 | 年级，班级：19级2班 |
| 成绩： | 任课教师签名： | 日期：2021/10/25 |
| 实验题目：OpenGL中的光照功能 | | |
| 目录：   1. 实验内容描述，使用模型定义顶点的属性信息：法向量。描述材质信息 2. 实验功能算法描述，即光照是如何建立、如何定义，描述其原理 3. 实验shader程序描述，即vertex shader和fragment shader的程序代码及说明 4. 实验结果，有光照的场景模型实验结果图 5. 心得体会 | | |
| 实验报告：   1. 实验内容描述，使用模型定义顶点的属性信息：法向量。描述材质信息   这次试验使用的模型是“Utah teapot”。法向量在vertex shader中计算，材质未知。   1. 实验功能算法描述，即光照是如何建立、如何定义，描述其原理   这次实验使用的光照模型是“Phong shading”。在Phong shading中，光照分为，环境光（ambient），漫反射（diffuse），镜面反射（specular）。  环境光是每个点相同的，都有的基础光照。  漫反射依据入射角度，计算出物体反射光线的亮度。  镜面反射依据出射角度和摄像机观测的角度，计算出反射高亮的亮度。  将三者强度相加，就是最后计算出的光照模型     1. 实验shader程序描述，即vertex shader和fragment shader的程序代码及说明   vertexShader :  varying vec3 Normal;  varying vec3 Position;  void main() {  Normal = normalize(normalMatrix \* normal);  Position = vec3(modelViewMatrix \* vec4(position, 1.0));  gl\_Position = projectionMatrix \* modelViewMatrix \* vec4(position, 1.0);  }  fragmentShader :  varying vec3 Normal;  varying vec3 Position;  uniform vec3 Ka;  uniform vec3 Kd;  uniform vec3 Ks;  uniform vec4 LightPosition;  uniform vec3 LightIntensity;  uniform float Shininess;  vec3 phong() {  vec3 n = normalize(Normal);  vec3 s = normalize(vec3(LightPosition) - Position);  vec3 v = normalize(vec3(-Position));  vec3 r = reflect(-s, n);  vec3 ambient = Ka;  vec3 diffuse = Kd \* max(dot(s, n), 0.0);  vec3 specular = Ks \* pow(max(dot(r, v), 0.0), Shininess);  return LightIntensity \* (ambient + diffuse + specular);  }  void main() {  gl\_FragColor = vec4(phong(), 1.0);  }  VS 中计算表面的法向量，FS中依据Phong模型计算像素点的光照。   1. 实验结果，有光照的场景模型实验结果图      1. 心得体会   这次实验中，我学会了如何使用shader 进行模型的光照计算，了解到了绘制一个平时计算机中习以为常的模型，图片是多么的困难。 | | |