计算机图形学实验报告

实验四：光照

姓　　名：　叶雨静

学　　号：　36720222204019

院　　系：　生命科学学院

专　　业：　生物技术

（计算机科学辅修）

年　　级：　2022 级

指导教师：　陈中贵

2024 年 04 月 17 日

一、实验名称

实验四 光照

二、实验任务

（1）修改程序以使其包括两个位于不同位置的位置光蓝光红光
（2）对每个光的漫反射和镜面反射分量简单相加或加权求和（结果不超出光照值上限）

三、实验环境及工具

Windows 版本：Windows 11 23H2
Visual Studio 2022

四、核心代码

1.设置两个光源的初始位置

```
glm::vec3 initialLightLoc = glm::vec3(4.0f, 8.0f, 2.0f);
glm::vec3 initialLightLoc2 = glm::vec3(-4.0f, -8.0f, 2.0f);
```

2.设置环境光、红光蓝光的环境光、漫反射光、高光

```
float globalAmbient[4] = { 0.3f, 0.3f, 0.3f, 1.0f };
// red light
float lightAmbient[4] = { 1.0f, 0.1f, 0.05f, 1.0f };
float lightDiffuse[4] = { 1.0f, 0.1f, 0.05f, 1.0f};
float lightSpecular[4] = { 1.0f, 0.8f, 0.8f, 0.9f };

// blue light
float lightAmbient2[4] = { 0.05f, 0.1f, 1.0f, 1.0f };
float lightDiffuse2[4] = { 0.05f, 0.1f, 1.0f, 1.0f };
float lightSpecular2[4] = { 0.8f, 0.8f, 1.0f, 0.9f };
```

3.修改材质为银

```
// silver material
float* matAmb = Utils::silverAmbient();
float* matDif = Utils::silverDiffuse();
float* matSpe = Utils::silverSpecular();
float matShi = Utils::silverShininess();
```

4.将光源的位置从世界坐标转换到视图坐标系中

```
transformed = glm::vec3(vMatrix * glm::vec4(currentLightPos, 1.0));
lightPos[0] = transformed.x;
lightPos[1] = transformed.y;
lightPos[2] = transformed.z;

transformed = glm::vec3(vMatrix * glm::vec4(currentLightPos2, 1.0));
lightPos2[0] = transformed.x;
lightPos2[1] = transformed.y;
lightPos2[2] = transformed.z;
```

5.从着色器中获取光源和材质的位置

```
// get the locations of the light and material fields in the shader
globalAmbLoc = glGetUniformLocation(renderingProgram, "globalAmbient");

ambLoc = glGetUniformLocation(renderingProgram, "light.ambient");
diffLoc = glGetUniformLocation(renderingProgram, "light.diffuse");
specLoc = glGetUniformLocation(renderingProgram, "light.specular");
posLoc = glGetUniformLocation(renderingProgram, "light.position");

ambLoc2 = glGetUniformLocation(renderingProgram, "light2.ambient");
diffLoc2 = glGetUniformLocation(renderingProgram, "light2.diffuse");
specLoc2 = glGetUniformLocation(renderingProgram, "light2.specular");
posLoc2 = glGetUniformLocation(renderingProgram, "light2.position");

mambLoc = glGetUniformLocation(renderingProgram, "material.ambient");
mdiffLoc = glGetUniformLocation(renderingProgram, "material.diffuse");
mspecLoc = glGetUniformLocation(renderingProgram, "material.specular");
mshiLoc = glGetUniformLocation(renderingProgram, "material.shininess");
```

6.设置统一变量

```
//  set the uniform light and material values in the shader
glProgramUniform4fv(renderingProgram, globalAmbLoc, 1, globalAmbient);
glProgramUniform4fv(renderingProgram, ambLoc, 1, lightAmbient);
glProgramUniform4fv(renderingProgram, diffLoc, 1, lightDiffuse);
glProgramUniform4fv(renderingProgram, specLoc, 1, lightSpecular);
glProgramUniform3fv(renderingProgram, posLoc, 1, lightPos);

glProgramUniform4fv(renderingProgram, ambLoc2, 1, lightAmbient2);
glProgramUniform4fv(renderingProgram, diffLoc2, 1, lightDiffuse2);
glProgramUniform4fv(renderingProgram, specLoc2, 1, lightSpecular2);
glProgramUniform3fv(renderingProgram, posLoc2, 1, lightPos2);

glProgramUniform4fv(renderingProgram, mambLoc, 1, matAmb);
glProgramUniform4fv(renderingProgram, mdiffLoc, 1, matDif);
glProgramUniform4fv(renderingProgram, mspecLoc, 1, matSpe);
glProgramUniform1f(renderingProgram, mshiLoc, matShi);
```

7.实现光源位置的改变

定义随时间变化的参数 amt，50.0f 设置移动的速度。rMat 矩阵沿 z 轴旋转。对光源位置应用 rMat 矩阵，实现光源沿 z 轴旋转。

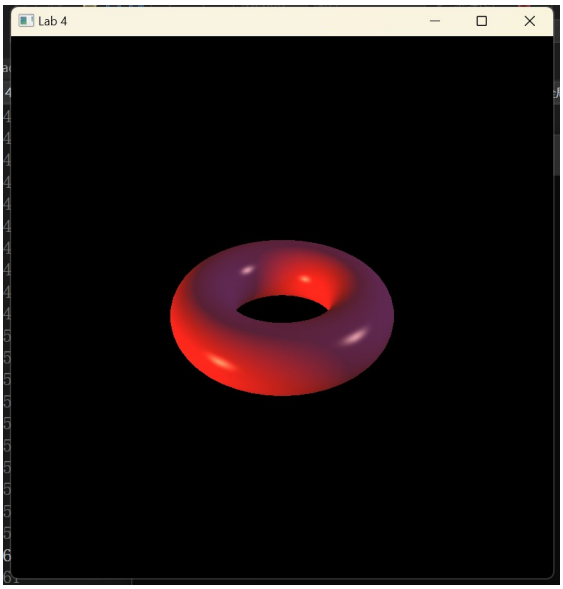
```
currentLightPos = glm::vec3(initialLightLoc.x, initialLightLoc.y, initialLightLoc.z);
currentLightPos2 = glm::vec3(initialLightLoc2.x, initialLightLoc2.y, initialLightLoc2.z);

amt = currentTime * 50.0f;

rMat = glm::rotate(glm::mat4(1.0f), toRadians(amt), glm::vec3(0.0f, 0.0f, 1.0f));

currentLightPos = glm::vec3(rMat * glm::vec4(currentLightPos, 1.0f));
currentLightPos2 = glm::vec3(rMat * glm::vec4(currentLightPos2, 1.0f));
```

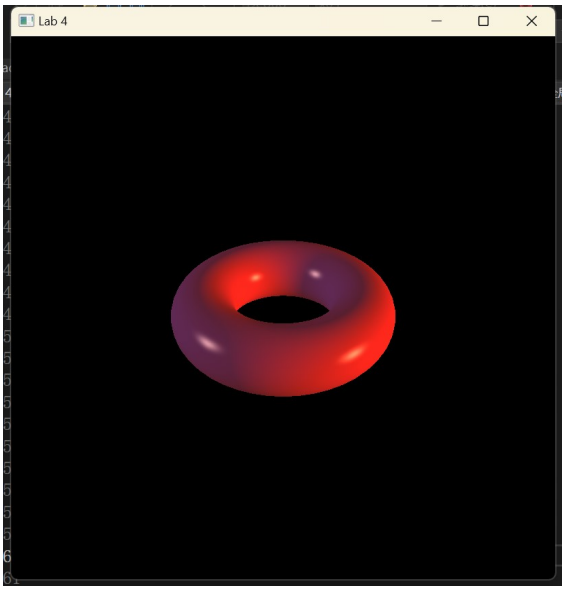
8.在片段着色器中对环境光、漫反射光、高光进行计算，对最终简单相加的结果进行截断，使结果在 0.0 到 1.0 之间。

```
// compute ADS contributions (per pixel):
vec3 ambient = ((globalAmbient * material.ambient) + (light.ambient * material.ambient) + (light2.ambient * material.ambient)).xyz;

vec3 diffuse = light.diffuse.xyz * material.diffuse.xyz * max(cosTheta,0.0);
vec3 diffuse2 = light2.diffuse.xyz * material.diffuse.xyz * max(cosTheta2,0.0);

vec3 specular = light.specular.xyz * material.specular.xyz * pow(max(cosPhi,0.0), material.shininess);
vec3 specular2 = light2.specular.xyz * material.specular.xyz * pow(max(cosPhi2,0.0), material.shininess);

vec3 finalColor = ambient + diffuse + specular + diffuse2 + specular2;

fragColor = vec4(clamp(finalColor, 0.0, 1.0), 1.0);
```

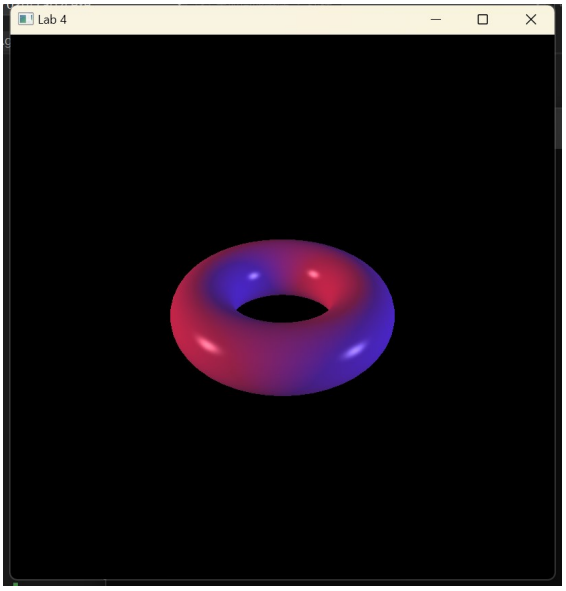
五、运行结果

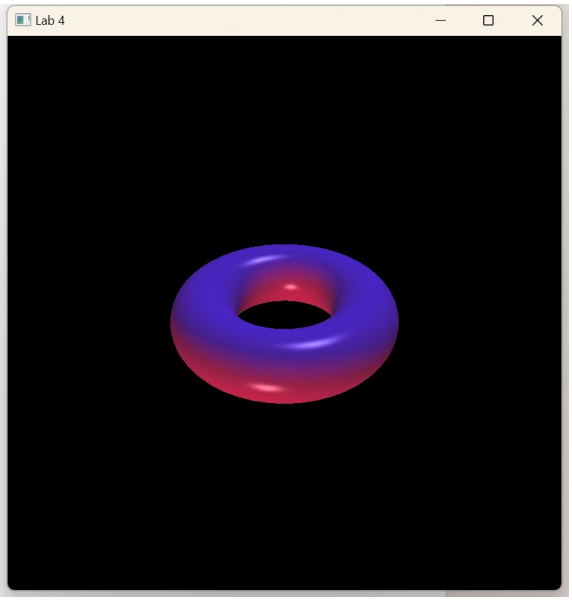
Gold:





Silver:





六、分析讨论

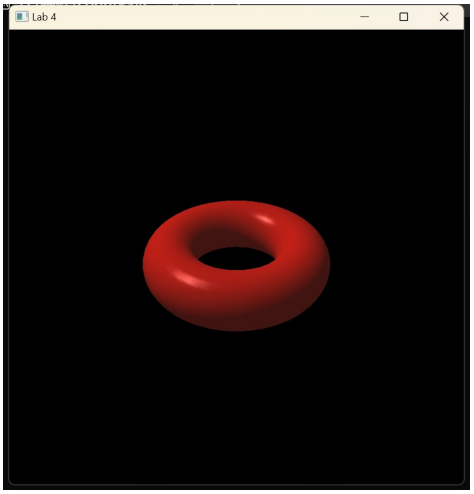
1.

材质	环境光RGBA 漫反射RGBA 反射RGBA	光泽度
黄金	0.2473, 0.1995, 0.0745, 1.0 0.7516, 0.6065, 0.2265, 1.0 0.6283, 0.5558, 0.3661, 1.0	51.200
银	0.1923, 0.1923, 0.1923, 1.0 0.5075, 0.5075, 0.5075, 1.0 0.5083, 0.5083, 0.5083, 1.0	51.200

银材质颜色均衡，表现灯光效果更好；
黄金材质颜色偏向红色，蓝色光源在圆环上呈现紫色。

2.

可以看出 phong 着色高光更加平滑，Gouraud 着色高光处有明显闪烁。



Gouraud 着色效果