一　简介

OpenGL 是用于渲染 2D、3D 矢量图形的跨语言、跨平台的应用程序编程接口 (API), 可用于绘制从简单的图形到复杂的三维景象, 常用于 CAD、虚拟现实、科学可视化程序和电子游戏开发. OpenGL 基本函数库可用来描述图元、属性、几何变换等观察变换等.

光照模型主要用于对象表面某光照位置的颜色计算. 参考网站 [1] 学习 OpenGL，实现 Phong 模型和 Blinn-Phong 光照模型.

注：在 OpenGL-Project/src 目录下，可直接单击 myprogram.exe 运行.

二　基本原理

现实世界的光照是极其复杂的，而且会受到诸多因素的影响，这是我们有限的计算能力所无法模拟的. 因此 OpenGL 的光照使用的是简化的模型，对现实的情况进行近似，这样处理起来会更容易一些，而且看起来也差不多一样.

2.1 Phong 模型

这些光照模型都是基于我们对光的物理特性的理解, 其中一个模型被称为 Phong 模型 (Phong Lighting Model). Phong 模型 [2] 的主要结构由 3 个分量组成：环境 (Ambient)、漫反射 (Diffuse) 和镜面 (Specular) 光照.

- 环境光照：即使在黑暗的情况下，世界上通常也仍然有一些光亮 (月亮、远处的光)，所以物体几乎永远不会是完全黑暗的. 为了模拟这个，我们会使用一个环境光照常量，它永远会给物体一些颜色.
- 漫反射光照：模拟光源对物体的方向性影响. 它是冯氏光照模型中视觉上最显著的分量. 物体的某一部分越是正对着光源，它就会越亮.
- 镜面光照：模拟有光泽物体上面出现的亮点. 镜面光照的颜色相比于物体的颜色会更倾向于光的颜色.

2.2 Blinn-Phong 模型

1977 年，James F. Blinn 在 Phong 模型上加以拓展，引入了 Blinn-Phong 光照模型 [3]. Blinn-Phong 模型与冯氏模型非常相似，但是它对**镜面光照**的处理上有一些不同. Blinn-Phong 模型不再依赖于反射向量，而是采用了所谓的半程向量，即光线与视线夹角一半方向上的一个单位向量. 当半程向量与法线向量越接近时，镜面光分量就越大.

当视线正好与反射向量对齐时，半程向量就会与法线完美契合. 所以当观察者视线越接近于原本反射光线的方向时，镜面高光就会越强.

现在，不论观察者向哪个方向看，半程向量与表面法线之间的夹角都不会超过 90 度 (除非光源在表面以下). 它产生的效果会与冯氏光照有些许不同，但是大部分情况下看起来会更自然一点，特别是低高光的区域. Blinn-Phong 着色模型正是早期固定渲染管线时代时 OpenGL 所采用的光照模型.

Blinn-Phong 模型与 Phong 模型唯一的区别就是，Blinn-Phong 测量的是法线与半程向量之间的夹角，而冯氏模型测量的是观察方向与反射向量间的夹角.

三　核心代码

代码部分参考了网站 [1], 引用了网站上提供的部分工具包.

光照模型计算核心代码如下：

```
void main()
{
    vec3 color = texture(floorTexture, fs_in.TexCoords).rgb;
    // 环境光照
    vec3 ambient = 0.05 * color;
    // 漫反射
    vec3 lightDir = normalize(lightPos - fs_in.FragPos);
    vec3 normal = normalize(fs_in.Normal);
    float diff = max(dot(lightDir, normal), 0.0);
    vec3 diffuse = diff * color;
    // 镜面光照
    vec3 viewDir = normalize(viewPos - fs_in.FragPos);
    vec3 reflectDir = reflect(-lightDir, normal);
    float spec = 0.0;
    if(blinn) // Blinn-Phong 模型
    {
        vec3 halfwayDir = normalize(lightDir + viewDir);
        spec = pow(max(dot(normal, halfwayDir), 0.0), 16.0);
    }
    else      // Phong 模型
    {
        vec3 reflectDir = reflect(-lightDir, normal);
        spec = pow(max(dot(viewDir, reflectDir), 0.0), 8.0);
    }
    vec3 specular = vec3(0.3) * spec; //假设明亮白光
    FragColor = vec4(ambient + diffuse + specular, 1.0);
}
```

四　实验结果

运行项目：Run Without Debugging(VS Code).

Phong 模型：

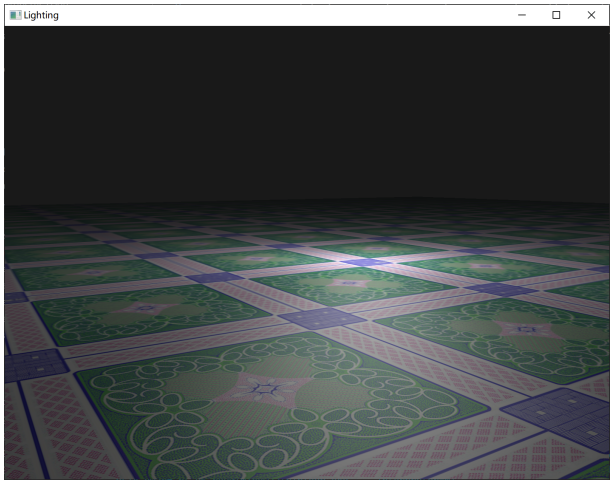

图 4-1: Phong 模型

Blinn-Phong 模型：

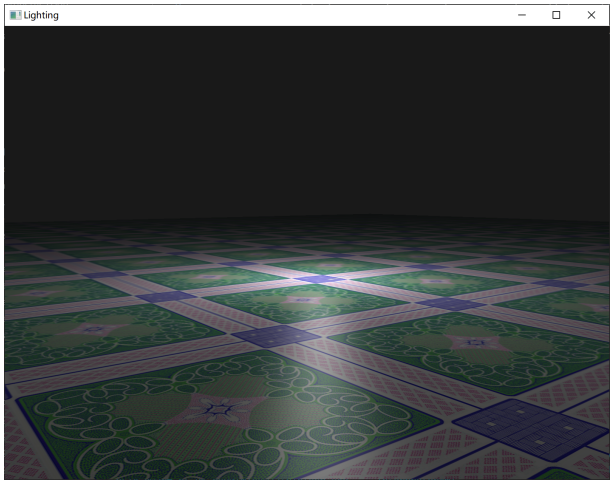

图 4-2: Blinn-Phong 模型

注：按下 F1 可切换 Phong 模型与 Blinn-Phong 光照模型.

参考文献

[1] Learn OpenGL[OL].(2021-7-15)[2021-7-15].https://learnopengl.com/.

[2] Lafortune E P, Willems Y. Using the modified Phong reflectance model for physically based rendering[J]. CW Reports, 1994: 19-19.

[3] Gotanda Y. Beyond a simple physically based Blinn-Phong model in real-time[C]//ACM SIGGRAPH. 2012: 10.