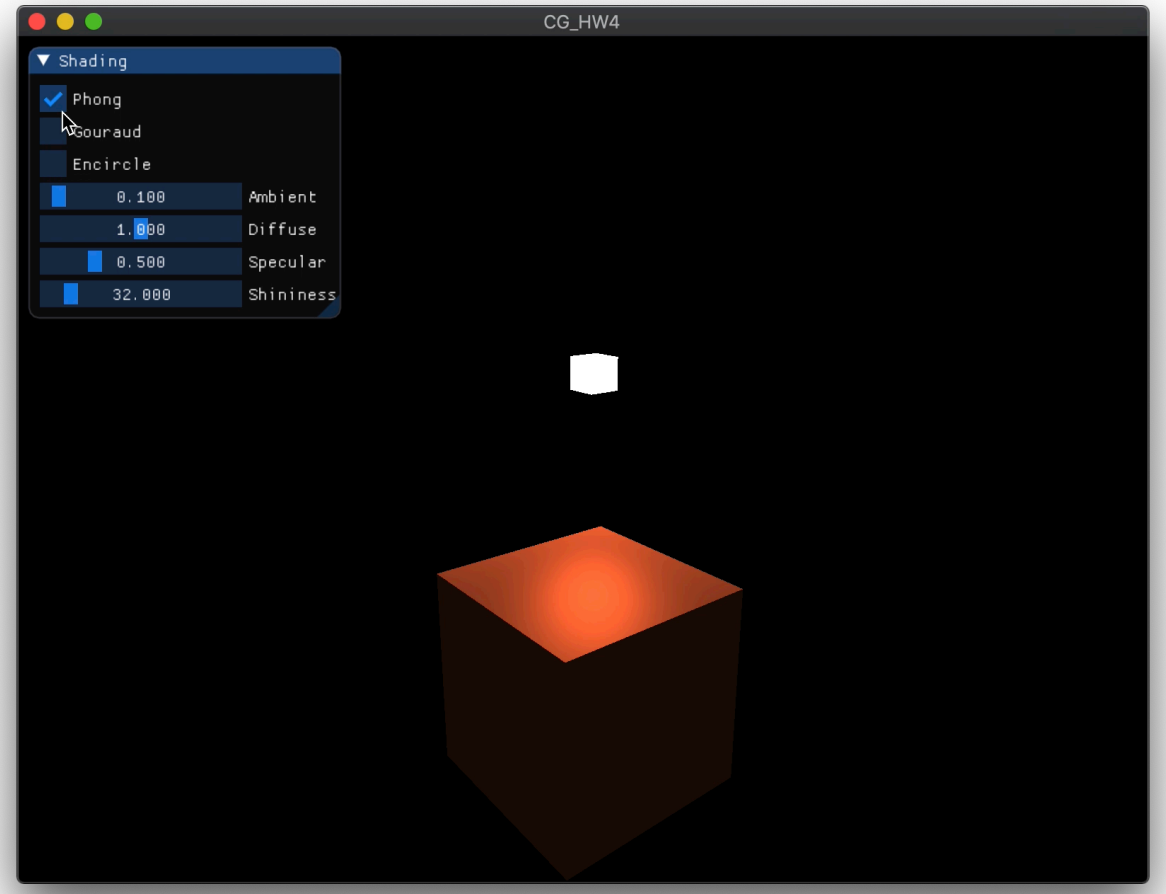
张吉祺 16340286

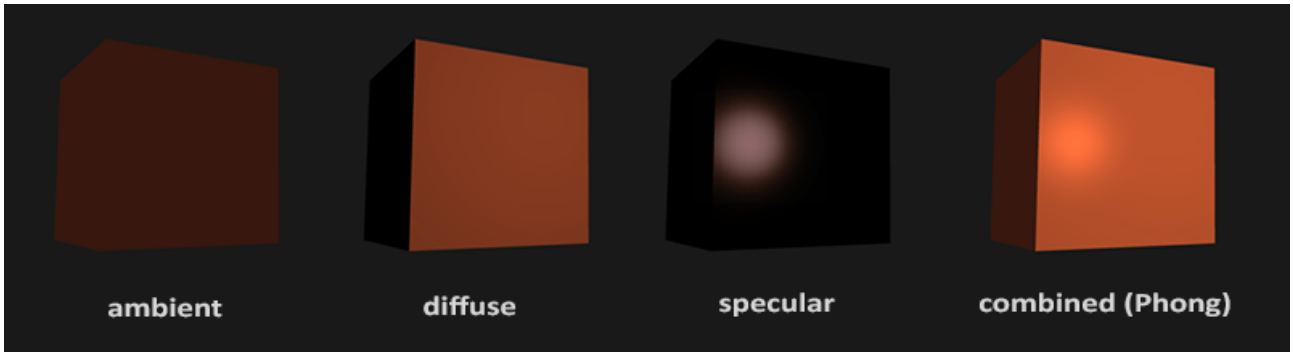
Basic:

1. 实现Phong Shading：



实现原理：

冯氏光照模型(Phong Lighting Model)由3个分量组成：环境(Ambient)、漫反射(Diffuse)和镜面(Specular)光照。



环境光照(Ambient Lighting)：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，是简化的全局照明模型，用光的颜色乘以一个很小的常量环境因子，再乘以物体的颜色，然后将最终结果作为

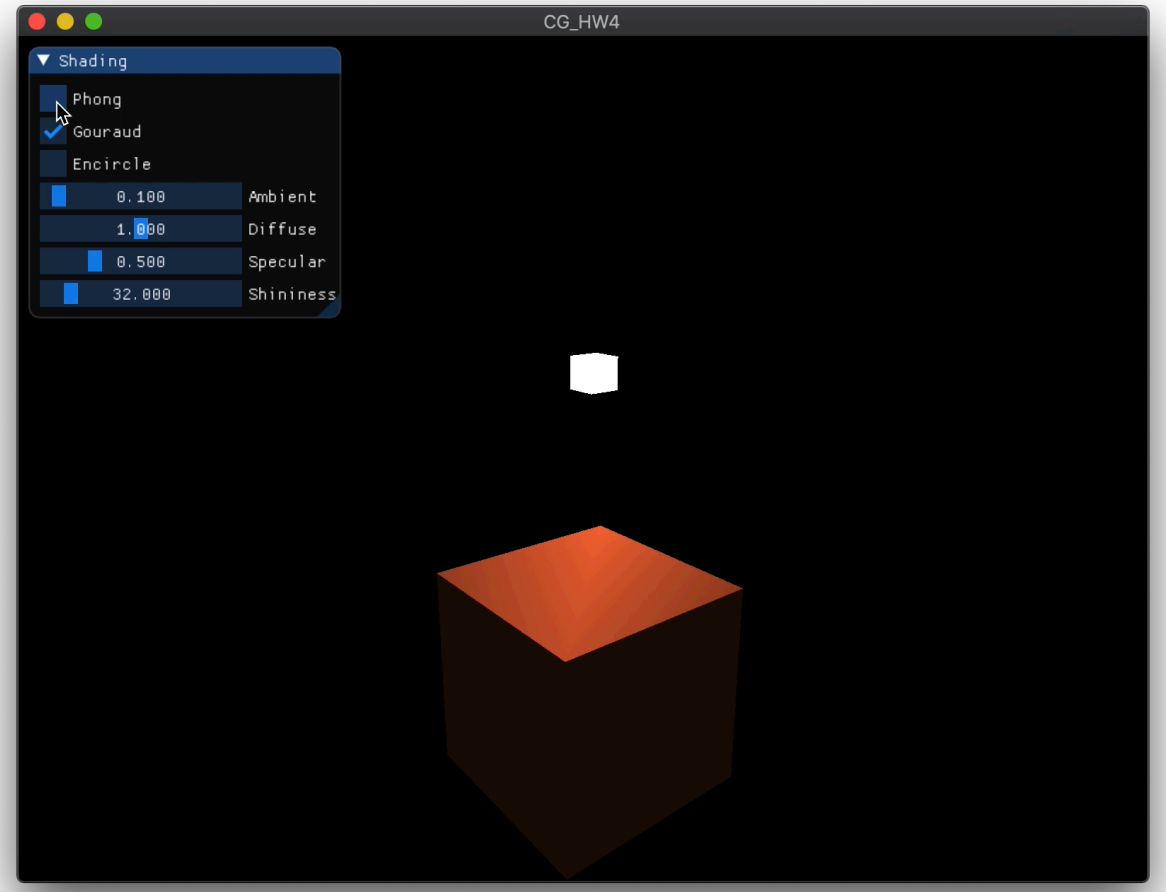
片段的颜色。
漫反射光照(Diffuse Lighting)：模拟光源对物体的方向性影响(Directional Impact)。它是冯氏光照模型中视觉上最显著的分量。测量这个光线是以什么角度接触到这个片段，如果光线垂直于物体表面，这束光对物体的影响最大。使用法向量测量光线和片段的角度（点乘），再乘以光的颜色和物体的颜色。
镜面光照(Specular Lighting)：模拟有光泽物体上面出现的亮点。镜面光照的颜色相比于物体的颜色会更倾向于光的颜色。通过反射法向量周围光的方向来计算反射向量。然后计算反射向量和视线方向（使用观察者世界空间位置和片段的位置来计算）的角度差，如果夹角越小，那么镜面光的影响就会越大，这里计算视线方向与反射方向的点乘然后取它的反光度次幂，再乘以光的颜色和物体的颜色。

在片段着色器实现Phong光照模型为Phong Shading。

对于顶点着色器，设置`in`变量`aPos`获取顶点坐标，`in`变量`aNormal`获取顶点法向量，`uniform`变量`model`、`view`、`projection`获取对应矩阵，`out`变量`FragPos`为顶点的世界坐标（`model * aPos`），`out`变量`Normal`为顶点法向量`transpose(inverse(model))`（法线矩阵防止缩放影响法向量方向） `* aNormal`，剪裁坐标`gl_Position`为`projection * view * FragPos`。

对于片段着色器，设置`in`变量`FragPos`，`Normal`得到顶点着色器对应的`out`变量，`uniform`变量`lightPos`为光照世界坐标，`lightColor`为光照颜色，`objectColor`为物体每个像素共同的颜色，`viewPos`为摄影机的世界坐标，`ambientStrength`为环境光照强度，`diffuseStrength`为漫反射光照强度，`specularStrength`为镜面光照强度，`shininess`为反光度，`out`变量`FragColor`为像素的最终的颜色，为`(ambient + diffuse + specular) * objectColor`，其中环境光照`ambient`为`ambientStrength * lightColor`，漫反射光照`diffuse`为`diffuseStrength * diff`（归一化后的`Normal`乘以归一化后的光线方向反方向`lightPos - FragPos`） `* lightColor`，镜面光照`specular`为`specularStrength * spec`（归一化后的视线方向`viewPos - FragPos`乘以归一化后的光线方向关于法向量的对称的结果的`shininess`次幂） `* lightColor`。

2. 实现**Gouraud Shading**:



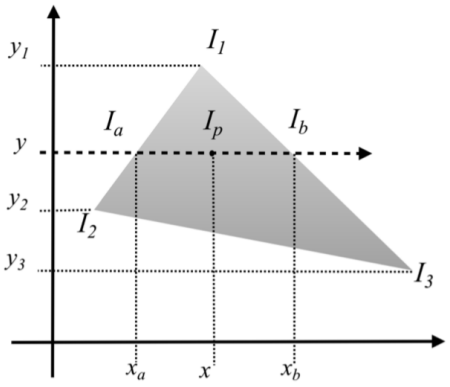
实现原理:

与Phong Shading不同的是，这是一种插值方法，只计算顶点的法向量并使用Phong光照模型计算顶点的光强，其内部多边形的点的光强通过在边或扫描线上对顶点插值$n = \frac{n_1 + n_2 + n_3 + n_4}{|n_1 + n_2 + n_3 + n_4|}$计算。

$$I_a = I_1 + (I_2 - I_1)\frac{y - y_1}{y_2 - y_1}$$

$$I_b = I_1 + (I_3 - I_1)\frac{y - y_1}{y_3 - y_1}$$

$$I_p = I_a + (I_b - I_a)\frac{x - x_a}{x_b - x_a}$$

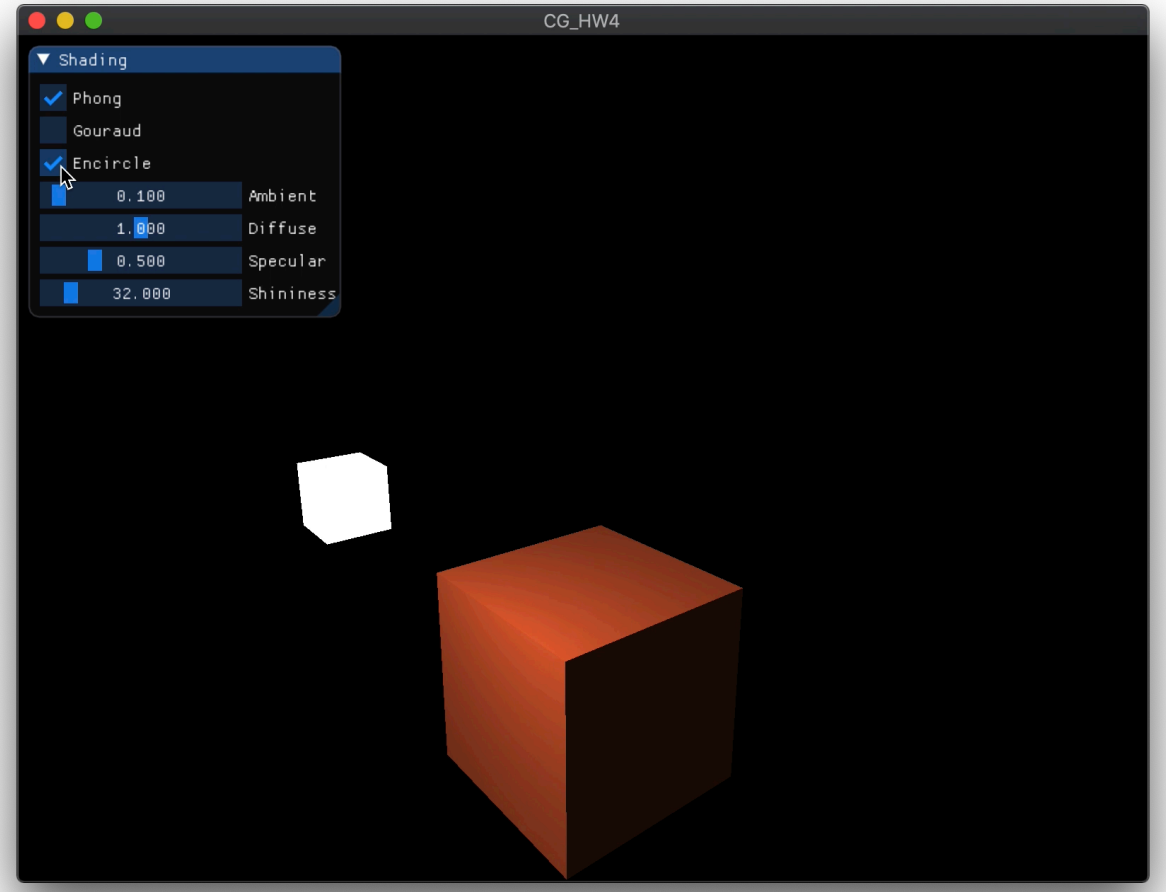


在顶点着色器实现Phong光照模型为Gouraud Shading。将上面片段着色器中的颜色计算移到顶点着色器中，片段着色器只需要将`FragColor`赋值为顶点着色器传来的颜色。

张吉祺 16340286

Bonus:

1. 当前光源为静止状态，尝试使光源在场景中来回移动，光照效果实时更改。



通过在渲染循环中改变光源位置`glm::vec3 lightPos`使光源移动，为了实现光源水平方向做半径为`radius`的圆周运动，在渲染循环中`lightPos`的`y`分量不变，而`x`，`z`分量随时间变化，分别为`sin(glfwGetTime()) * radius`，`cos(glfwGetTime()) * radius`。