**《计算机图形学》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **22计算机科学与技术（卓越）01** | | | **姓名** |  |
| **实验题目** | 实验三 光线与三角形相交 | | | | | |
| **实验时间** | **2025/4/14** | | **实验地点** | **DS3305** | | |
| **实验成绩** |  | | **实验性质** | **□验证性** √**设计性 □综合性** | | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | | |
| 一、实验目的  1. 掌握使用光线追踪来渲染图像的方法  2. 实现光线的生成和光线与三角的相交两个部分，锻炼编程实践能力 | | | | | | |
| 二、实验项目内容  • 光线生成：正确实现光线生成部分，并且能够看到图像中的两个球体。  • 光线与三角形相交：正确实现了 Moller-Trumbore 算法，并且能够看到图像中的地面。 | | | | | | |
| 三、实验过程或算法（源程序）  本次实验需要填的部分就是把相机发出的光线坐标变换到物体所在的世界坐标和利用上课讲过的公式计算射线与三角形的交点。  3.1 CastRay:  CastRay是真正利用光线弹射着色的函数。其作用就是计算从摄像机出发，穿过每个像素击中的物体表面那个点的颜色。如果没有击中物体，就按背景颜色处理。整个场景设置了光线最多弹射5次，所以在函数一开始，会看一下是不是超过第五次弹射了。如果是直接返回。  场景中的物体被设置了三种不同的材质属性，第一种透明的（REFLECTION\_AND\_REFRACTION），第二种镜面的（REFLECTION），第三种粗糙的（DIFFUSE\_AND\_GLOSSY）。当光线打在透明物体和镜面物体上时，由于这两种物体没有颜色，我们看到的颜色都是光线穿过它或者被反射到其他物体上的颜色，所以需要再次调用castRay函数，看一下下一次的反射或者折射的光线能不能打到有颜色的物体上。当光线打在粗糙的物体上，我们需要在光线和物体的交点上（hitpoint），看一下能不能看到光源（从hitpoint发射一条连接光源的线，看一下和场景里面的物体有没有交点，如果没有或者有交点但是交点在光线身后，则可以看到光源，即被光线照射）。如果能看到光源，则用Phong模型计算光照颜色，反之则在阴影内，变成黑色。另外，光线打在粗糙的物体上后，不再做弹射。  castRay函数中的菲涅尔函数是为了计算光线照射到一个物体后的反射与折射的比例。  3.2 Render：  把raster space中的像素坐标转换到world space中的坐标。利用上课所讲的公式带入即可：     1. *// generate primary ray direction* 2. float x = (2 \* ((i + 0.5) / scene.width) - 1) \* scale \* imageAspectRatio; 3. float y = (1 - 2 \* ((j + 0.5) / scene.height)) \* scale;   3.3 rayTriangleIntersect   1. 射线与三角形相交，利用课上讲的MT算法公式带入即可： bool rayTriangleIntersect(const Vector3f& v0, const Vector3f& v1, const Vector3f& v2, const Vector3f& orig, 2. const Vector3f& dir, float& tnear, float& u, float& v) 3. { 4. *// TODO: Implement this function that tests whether the triangle* 5. *// that's specified bt v0, v1 and v2 intersects with the ray (whose* 6. *// origin is \*orig\* and direction is \*dir\*)* 7. *// Also don't forget to update tnear, u and v.* 8. Vector3f E1 = v1 - v0, E2 = v2 - v0, S = orig - v0; 9. Vector3f S1 = crossProduct(dir, E2), S2 = crossProduct(S, E1); 10. float S1E1 = dotProduct(S1, E1); 11. tnear = 1.0 / S1E1 \* dotProduct(S2, E2); 12. u = 1.0 / S1E1 \* dotProduct(S1, S); 13. v = 1.0 / S1E1 \* dotProduct(S2, dir); 14. return tnear > 0 && u >= 0 && u <= 1 && v >= 0 && v <= 1 && 1 - u - v >= 0 && 1 - u - v <= 1; 15. } | | | | | | |
| 四、实验结果及分析和（或）源程序调试过程  4.1经过运行代码可以成功得到以下图案，证明实验结果正确：    4.2源程序调试过程：  1.代码出现报错：   1. 错误 C2039 "optional": 不是 "std" 的成员   **解决方法：**实验在Visual Studio上运行的，std::optional 是 C++17 引入的，我没开启 C++17 标准。属性 → C/C++ → 语言 → C++语言标准，设置为 /std:c++17。  2.代码出现报错：   1. C4996 'fopen': This function or variable may be unsafe. Consider using fopen\_s instead. To disable deprecation, use \_CRT\_SECURE\_NO\_WARNINGS. See online help for details. test   解决方法：VS 编译器（Visual Studio）在使用老的 C 标准函数（比如 fopen）时的一种安全警告。通过定义宏 \_CRT\_SECURE\_NO\_WARNINGS 来禁用这个警告。 | | | | | | |