《计算机图形学原理实践》

实验报告

|  |  |  |
| --- | --- | --- |
| 实验名称 | ： | 球面Gouruad光照模型 |
|  |  |  |
| 姓 名 | ： |  |
| 学 号 | ： |  |
| 专业班级 | ： |  |
| 实验时间 | ： |  |

西南科技大学计算机科学与技术学院

1. **实验目的**

（1）掌握双线性光强插值模型

（2）掌握双线性颜色插值算法

**二、实验步骤**

**1.实验要求：**

（1）自定义屏幕三维右手坐标系，原点位于客户区中心，x 轴水平向右为正，y 轴垂直向上为正，z 轴指向屏幕内部。

（2）建立三维用户右手坐标系{O;x,y,z}，原点 O 位于客户区中心， x 轴水平向右为正，y 轴垂直向上为正，z 轴指向读者。

（3）绘制体心和坐标系原点重合的球体表面，使用 Z-Buffer 消隐进行消隐。

（4）使用单点光源对球体进行照射生成 Gouraud 光照模型，光源位置位于球体右上方。

（5）背景色设置为RGB(128,0,0)。

（6）使用键盘方向键旋转球体。

（7）使用工具条上的“动画”图标按钮播放或停止球体动画。

**2.算法设计**

1、读入球面的顶点表和面表。

2、循环访问三角形面片和四边形面片的每个顶点，调用光照函数计算所获得的光强。

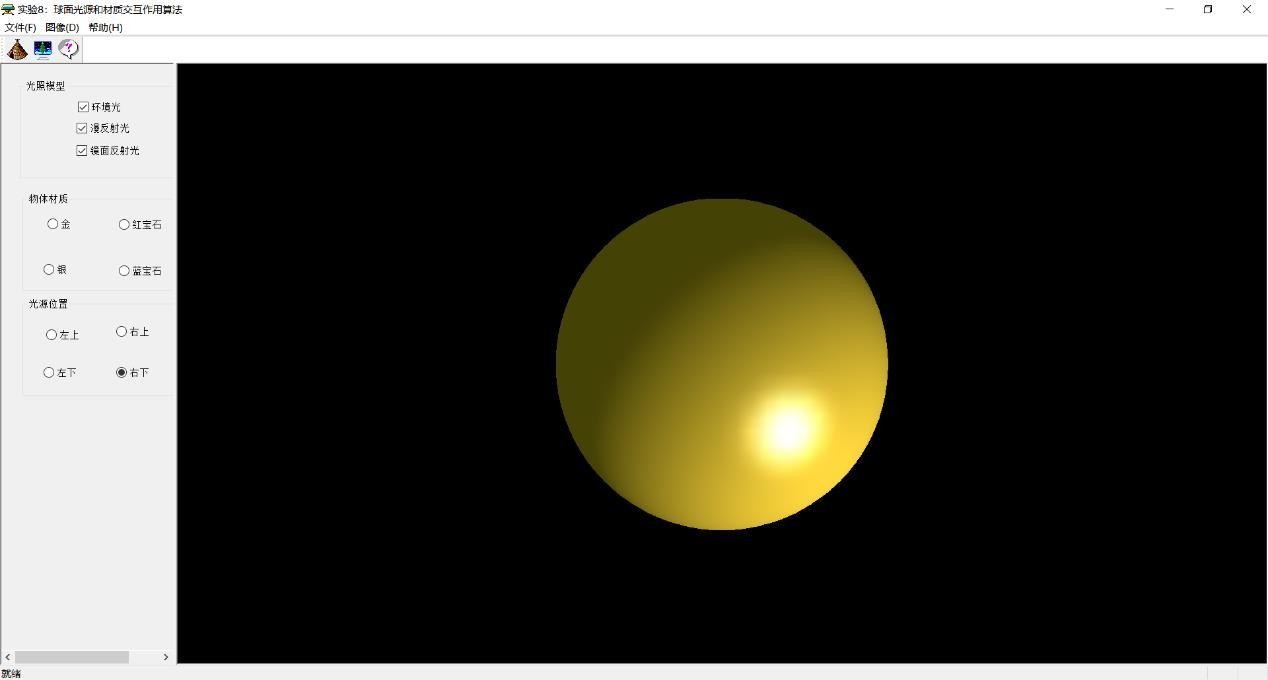
3、使用双缓冲技术绘制球面小面片的二维透视投影图。

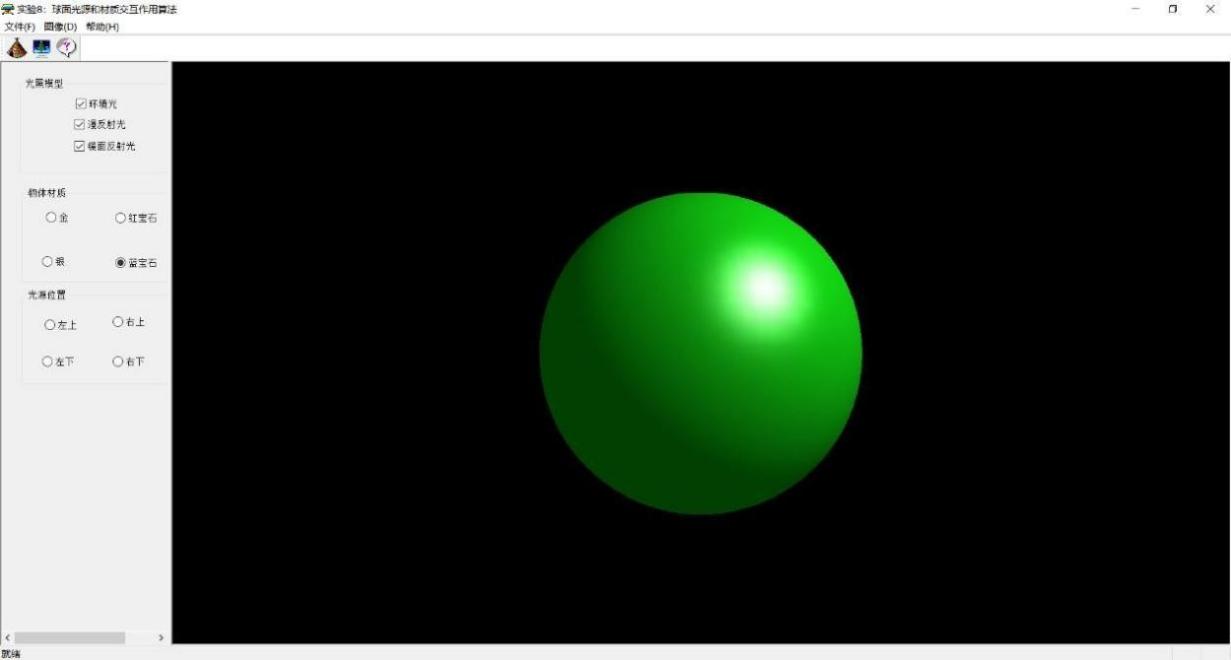
4、使用有效边表算法填充三角形面片和四边形面片。

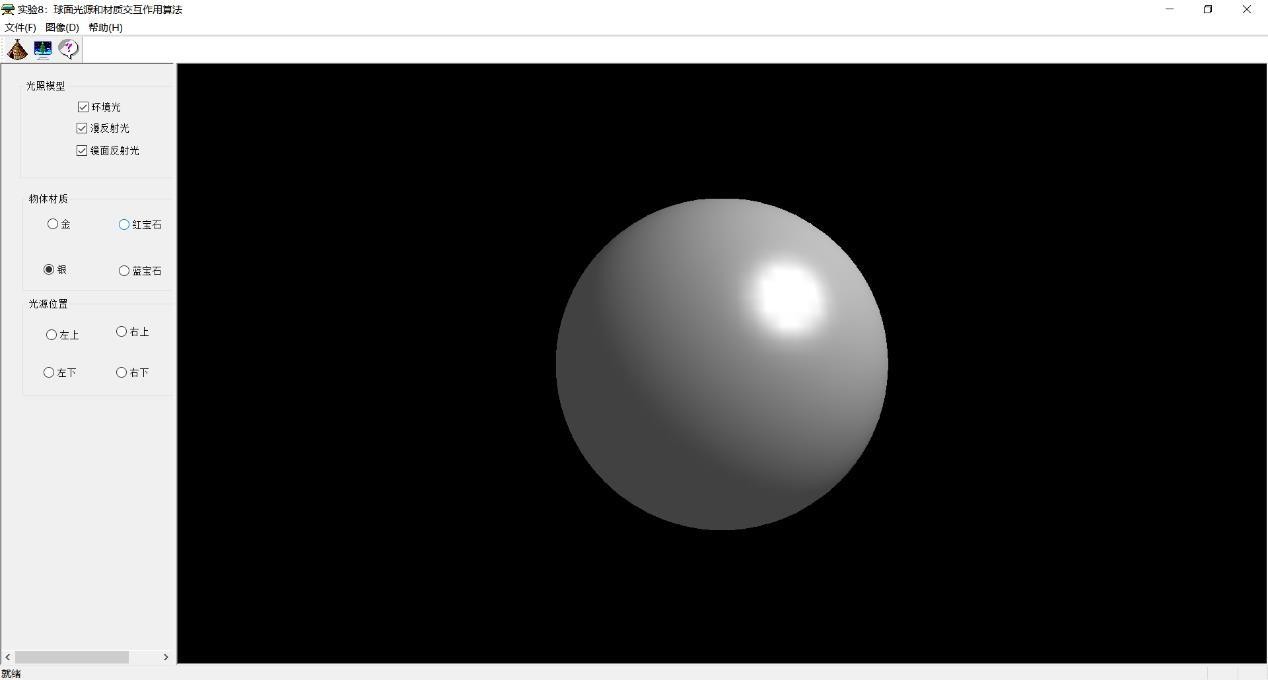
5、使用深度缓冲算法对球面进行动态消隐。

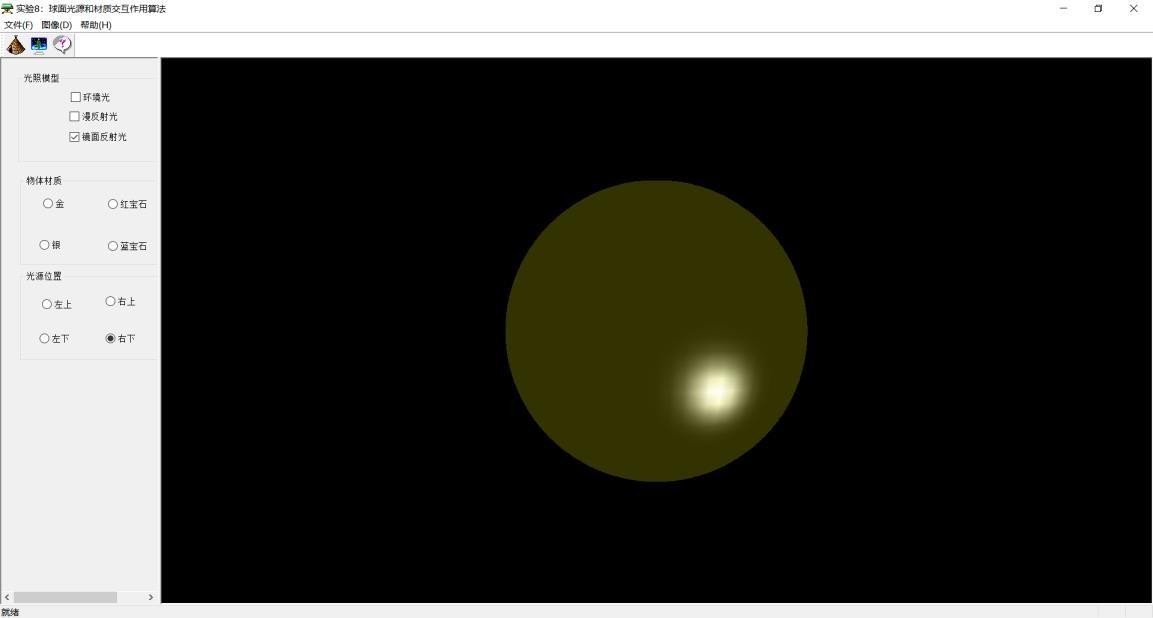
6、使用定时器改变球面的转角生成旋转动画。  
**三、实验结果**

根据不同的光照模型、物质材料和光源位置绘制出了以下不同的球面Gouruad 光照模型：









1. **实验总结**

这是计算机图像学实验的最后一次实验，在本次实验中，通过不断的探索和实践，我掌握了双线性光强插值模型的构造，知道怎么使用ZBuffer消隐算法进行消隐，怎么使用单点光源对球体进行照射生成Gourand光照模型，在这些过程中，我熟悉了光照模型、双线性颜色插值算法以及对有效边表填充算法更加熟练。虽然过程中遇到了许多困难，但在再次阅读教材理解教材后，以及跟同学讨论后，最终通过不断的努力将实验完成了，学会了将理论知识用在实践中，收获良多。八次的实验将我不断地带进新世界的大门，每一次探索都是全新的体验。

**附录：源代码（关键代码）**

void CTestView::PointColor()//计算顶点颜色

{

//TODO

for (int i = 0; i < (N1 - 1) \* N2 + 2; i++)//遍历所有点

{

CVector PNormal(V[i]);//点的位置矢量代麦共烹该点的所有面的平均法矢盘

V[i].c = pLight->Lighting(ViewPoint, V[i], PNormal, pMaterial);//调用光照函数

}

}

void CTestView::DrawObject(CDC \*pDC)//绘制球面

{

//TODO PointColor ();

CZBuffer\* zbuf = new CZBuffer;//申请内存

zbuf->InitDeepBuffer(800, 800, - 1000);//深度初始化CPi3 Point3[3];//南北极顶点数组

CPi3 Point4[4];//球体顶点数组for (int i = 0; i < N1; i++)

{

for (int j = 0; j < N2; j++)

{

CVector VS(V[F[i][j].vI[1]], ViewPoint);//面的视矢量F[i][j].SetFaceNormal(V[F[i][j].vI[0]], V[F[i][j].vI[1]], V[F[i][j].vI[2]]);

if (Dot(VS, F[i][j].fNormal) >= 0)//背面剔除

{

if (F[i][j].vN == 3)//三角形面片

{

for (int m = 0; m < F[i][j].vN; m++)

{

PerProject(V[F[i][j].vI[m]]); Point3[m] = ScreenP;

}

zbuf->SetPoint(Point3, 3);//设置顶点zbuf->CreateBucket();//建立桶表zbuf->CreateEdge();//建立边表

zbuf->Gouraud(pDC);//填充三角形zbuf->ClearMemory();//内存清理

}

else//四边形面片

{

for (int m = 0; m < F[i][j].vN; m++)

{

PerProject(V[F[i][j].vI[m]]); Point4[m] = ScreenP;

}

zbuf->SetPoint(Point4, 4);//设置顶点zbuf->CreateBucket();//建立桶表zbuf->CreateEdge();//建立边表

zbuf->Gouraud(pDC);//填充三角形zbuf->ClearMemory();//内存清理

}

}

}}

delete zbuf; //释放内存

}