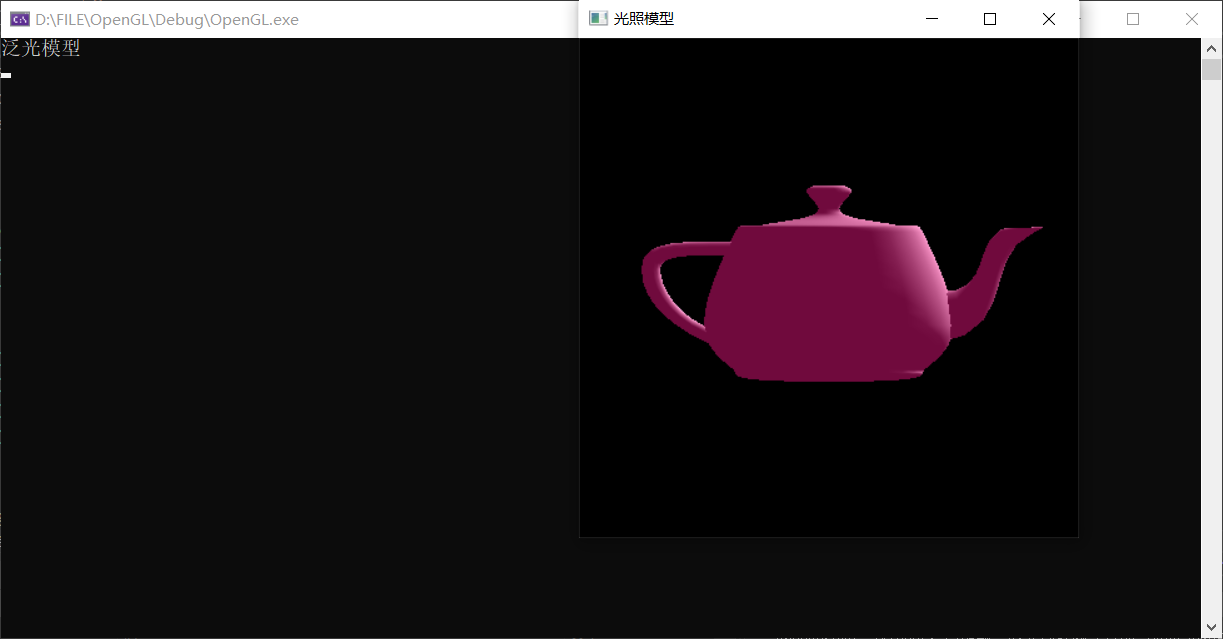
# 计算机图形学第三次作业

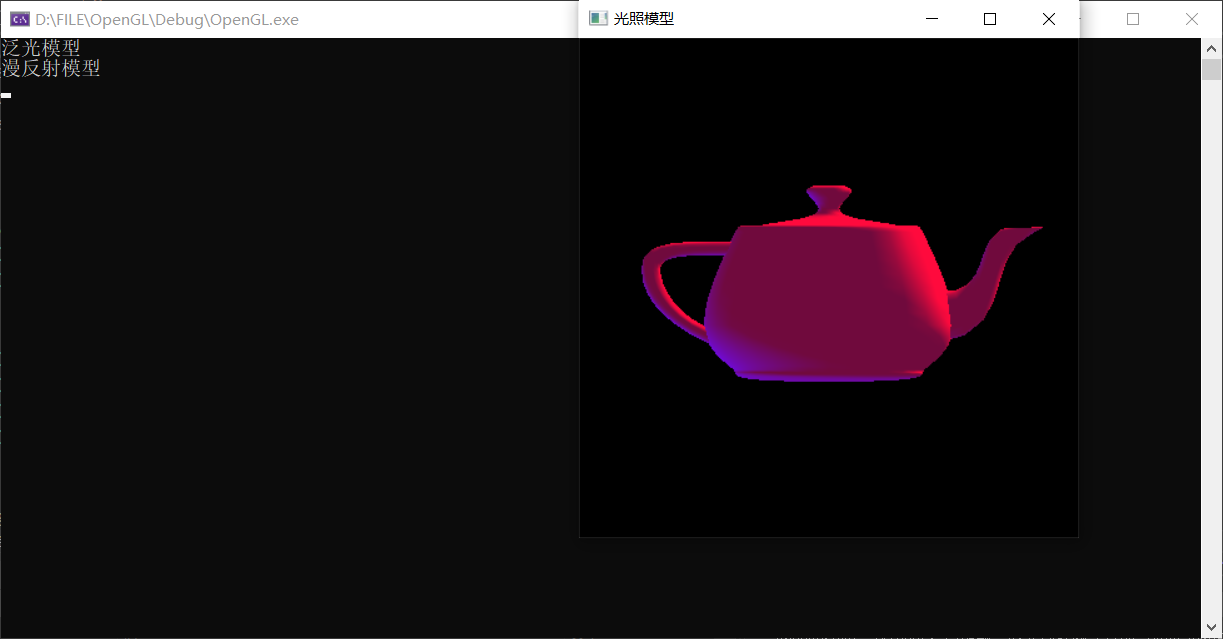
作业名：光照模型 姓名：方腾 学号：20051511

运行结果：

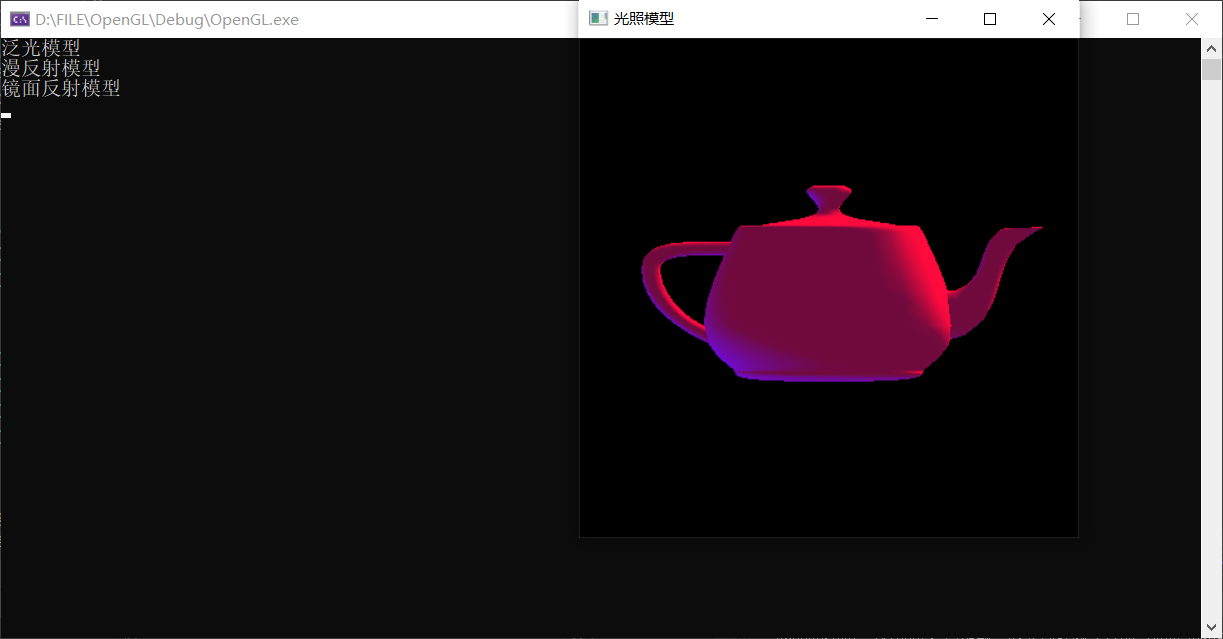
泛光模型：



漫反射模型：



镜面反射模型：



实验实现内容：

1. 生成茶壶模型
2. 通过键盘控制，对生成的茶壶模型进行泛光，漫反射，镜面反射三种不同光照模型的表现。

优缺点分析：

泛光模型：

只是单纯考虑环境光照影响，真实感显示的三维茶壶看起来像单一颜色填充的二维多边形物体。

Lamber（漫反射）模型：

相比于泛光模型，真实感增加，但是其看起来像素不够平滑。

Phong（镜面反射模型）：

相比于Lamber模型，真实感更强，像素也更加平滑。

程序

# include<GL/glut.h>

#include<stdio.h>

int ambient = 0;

int diffuse = 0;

int specular = 0;

// 设置两个不同的点光源（位置、颜色），实现在两个不同点光源照射下不同光照模型的几何体的表现。

void init(void)

{

//光照参数初始化全位于此处

GLfloat mat\_specular[] = { 1.0,1.0,1.0,1.0 }; //镜面反射参数

GLfloat mat\_shininess[] = { 100.0 }; //高光指数

GLfloat light\_1[] = { 2.0,0.0,0.0,0.0 }; //设置光源颜色

GLfloat light\_2[] = { 0.0,0.0,1.0,0.0 }; //设置光源颜色

GLfloat light\_position1[] = { 5.0,5.0,5.0,1.0 }; //点光源位置

GLfloat light\_position2[] = { -5.0,-5.0,5.0,1.0 }; //点光源位置

glClearColor(0.0, 0.0, 0.0, 0.0); //背景色

glShadeModel(GL\_SMOOTH); //多变形填充模式

//材质属性：如何反射光线（材料环境、散射、镜面颜色、光泽度）

glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular); //使用镜面材质颜色

glMaterialfv(GL\_FRONT, GL\_SHININESS, mat\_shininess); //使用光泽度

//第一个光源

glLightfv(GL\_LIGHT0, GL\_POSITION, light\_position1); //定义光源的位置

//第二个光源

glLightfv(GL\_LIGHT1, GL\_POSITION, light\_position2); //定义光源的位置

if (ambient == 1)

{

glLightfv(GL\_LIGHT0, GL\_AMBIENT, light\_1); //定义散射光

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light\_2); //定义散射光

}

if (diffuse==1)

{

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light\_1); //定义散射光

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light\_2); //定义散射光

}

if (specular==1)

{

glLightfv(GL\_LIGHT0, GL\_SPECULAR, light\_1); //定义镜面光

glLightfv(GL\_LIGHT1, GL\_SPECULAR, light\_2); //定义镜面光

}

glEnable(GL\_LIGHTING); //打开光源

glEnable(GL\_LIGHT0); //启动 0 号光源

glEnable(GL\_LIGHT1); //启动 1 号光源

glEnable(GL\_DEPTH\_TEST); //打开深度测试

}

// 生成茶壶模型，通过glutSolidTeapot()直接实现。

void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glNewList(1, GL\_COMPILE);

glColor3f(0.0, 0.0, 1.0);

init();

glutSolidTeapot(0.5);

glEndList();

glCallList(1);

glFlush();

glutSwapBuffers();

}

// 键盘事件：通过选择“1，2，3”选择不同的光照模型。

void mykeyboard(unsigned char key, int x, int y)

{

switch (key)

{

case '1':

ambient = 1;

diffuse = 0;

specular = 0;

printf("泛光模型\n");

break;

case '2':

ambient = 1;

diffuse = 1;

specular = 0;

printf("漫反射模型\n");

break;

case '3':

ambient = 1;

diffuse = 1;

specular = 1;

printf("镜面反射模型\n");

break;

default:

break;

}

glutPostRedisplay();

}

void reshape(GLsizei w, GLsizei h)

{

glViewport(0, 0, w, h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);

glMatrixMode(GL\_MODELVIEW);

}

int main(int argc,char \*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB | GLUT\_DEPTH);

glutInitWindowPosition(0, 0);

glutInitWindowSize(400, 400);

glutCreateWindow("光照模型");

glutKeyboardFunc(mykeyboard);

glutReshapeFunc(reshape);

glutIdleFunc(display);

glutMainLoop();

return 0;

}