《计算机图形学实验》综合实验报告

题目　基于 OpenGL 的三维图形的实现

学　号　20201050298

姓　名　刘钊宇

指导教师　钱文华

日　期　2022.6.21

摘要

渲染是三维计算机图形学中的最重要的研究课题之一，并且在实践领域它与其它技术密切相关。在图形流水线中，渲染是最后一项重要步骤，通过它得到模型与动画最终显示效果。自从二十世纪七十年代以来，随着计算机图形的不断复杂化，渲染也越来越成为一项重要的技术。

本次实验使用 OpenGL 实现三维图形渲染的一部分，设计一个茶壶的三维图形，在渲染过程加入了色彩和光照效果。

关键词

OpenGL;茶壶;反射光；颜色；光照

目录

一、实验背景

渲染是三维计算机图形学中的最重要的研究课题之一，并且在实践领域它与其它技术密切相关。在图形流水线中，渲染是最后一项重要步骤，通过它得到模型与动画最终显示效果。自从二十世纪七十年代以来，随着计算机图形的不断复杂化，渲染也越来越成为一项重要的技术。

渲染的应用领域有:计算机与视频游戏、模拟、电影或者电视特效以及可视化设计，每一种应用都是特性与技术的综合考虑。作为产品来看，现在已经有各种不同的渲染工具产品，有些集成到更大的建模或者动画包中，有些是独立产品，有些是开放源代码的产品。从内部来看，渲染工具都是根据各种学科理论，经过仔细设计的程序，其中有:光学、视觉感知、数学以及软件开发。

二、实验内容

实现三维图形渲染，自定义三维图形，三维图形不能仅仅是简单的茶壶、球体、圆柱体、圆锥体等图形，渲染过程须加入纹理、色彩、光照、阴影、透明等效果，可采用光线跟踪、光照明模型、纹理贴图、纹理映射等算法。评分标准包括实验设计、实验完成情况、报告撰写情况等。

三、开发工具及实现目的

开发工具：Visual C++, OpenGL, Java 等

目的:设计一个茶壶，并添加可以通过控制光照和颜色来改变其观感的代码

四、关键代码及算法理论

下面是一些比较基本的函数解释：

glutInit()用 glut 来初始化 OpenGL

glutInitDisplayMode()这告诉系统我们如何需要一个显示模式。

GLUT_SINGLE 只使用单缓存

GLUT_RGB 当未指明 GLUT-RGBA 或 GLUT-INDEX 时，是默认使用的模式

GLUT_DEPTH 使用深度缓存

glutInitWindowSize(750,750) 设置显示窗口的大小

glutInitWindowPosition(75, 75)用来设置窗口出现的位置。

glutCreateWindow(“茶壶”)出现一个窗口，参数是窗口的标题。

glutMainLoop()主循环，一旦它被调用，OpenGL 就会继续运行

glClear(GL_COLOR_BUFFER_BIT) 擦去之前的图片

glFlush()清空缓冲区

下面是一些可以为本次实验做技术支撑的代码：

GL_AMBIENT (0.2,0.2,0.2,1.0)　材料的环境光颜色

GL_DIFFUSE (0.8,0.8,0.8,1.0)　材料的漫反射光颜色

GL_AMBIENT_AND_DIFFUSE　材料的环境光和漫反射光颜色

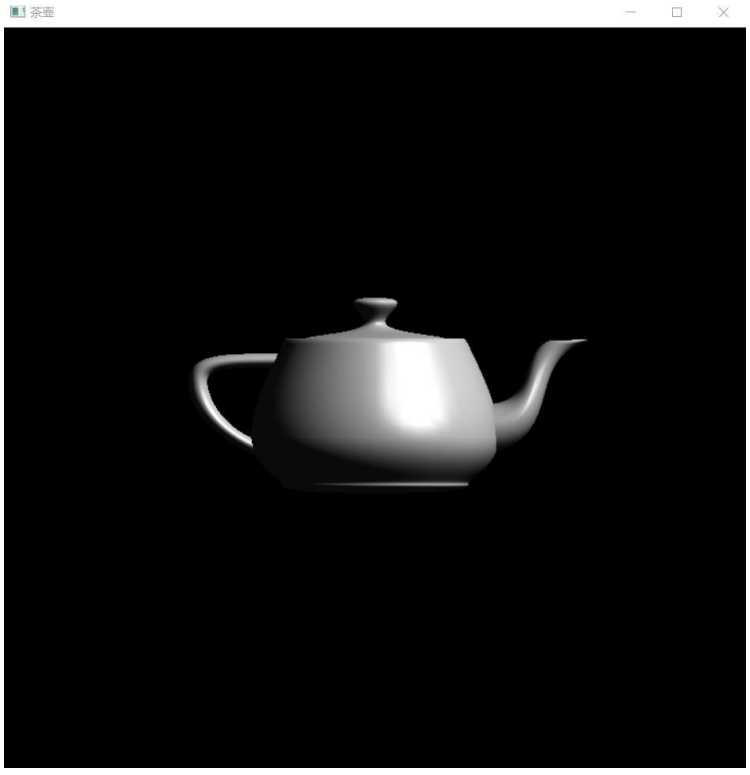
GL_SPECULAR (1.0,1.0,1.0,1.0)　材料的镜面反射光颜色

GL_SHINESS　50.0　镜面指数（光亮度）

GL_EMISSION (0.0,0.0,0.0,1.0)　材料的辐射光颜色

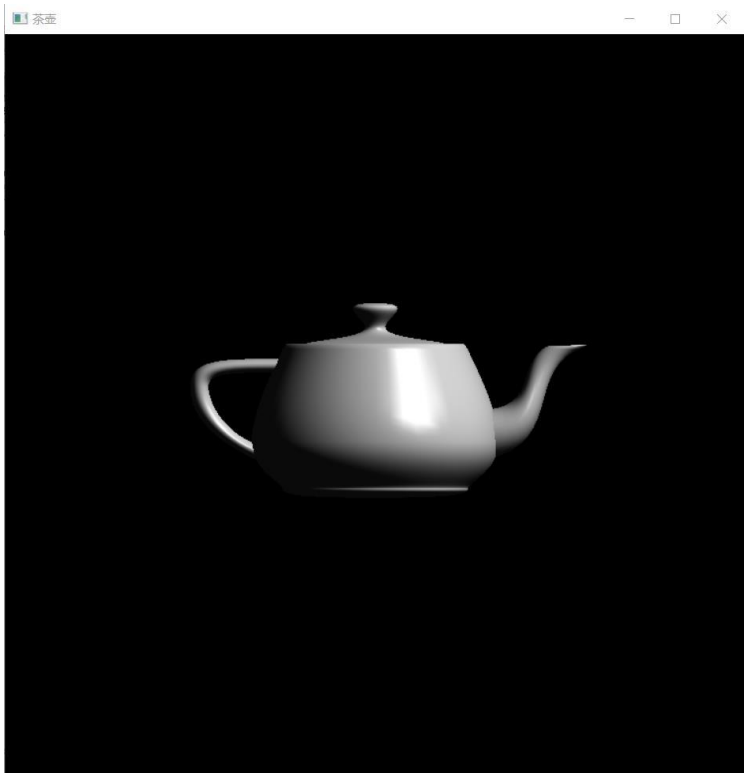
GL_COLOR_INDEXES (0,1,1)　材料的环境光、漫反射光和镜面光颜色

代码运行结果：



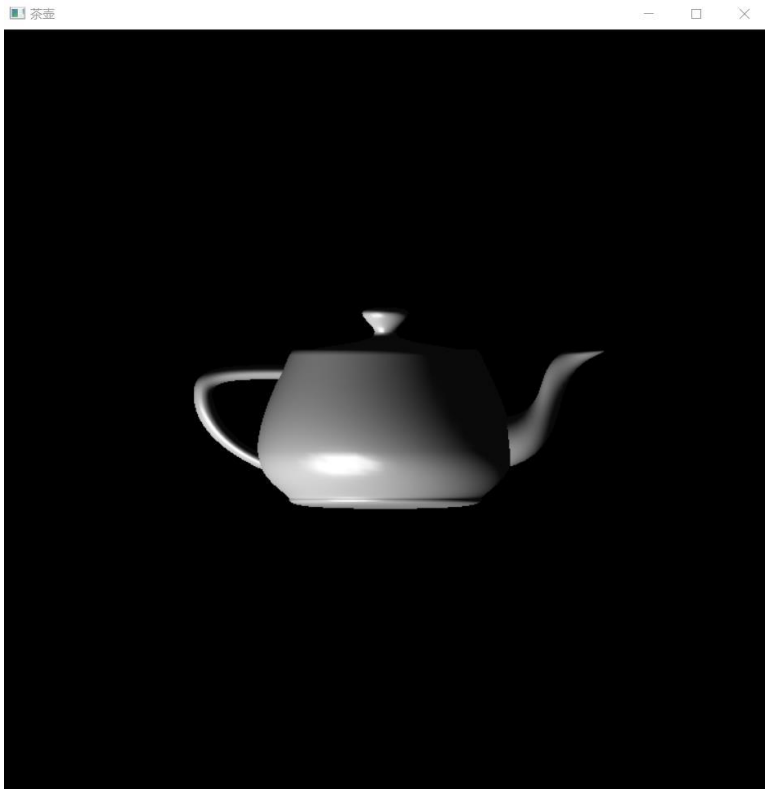
代码中 GLfloat mat_specular[]和 GLfloat mat_shininess[]分别为镜面反射光颜色和镜面指数，可以用来控制光照亮度来改变观感。

下面是分别将代码中的 GLfloat mat_specular[]改为{2.0，2.0，2.0，1.0}和 GLfloat mat_shininess[]改为{100.0}的运行结果：



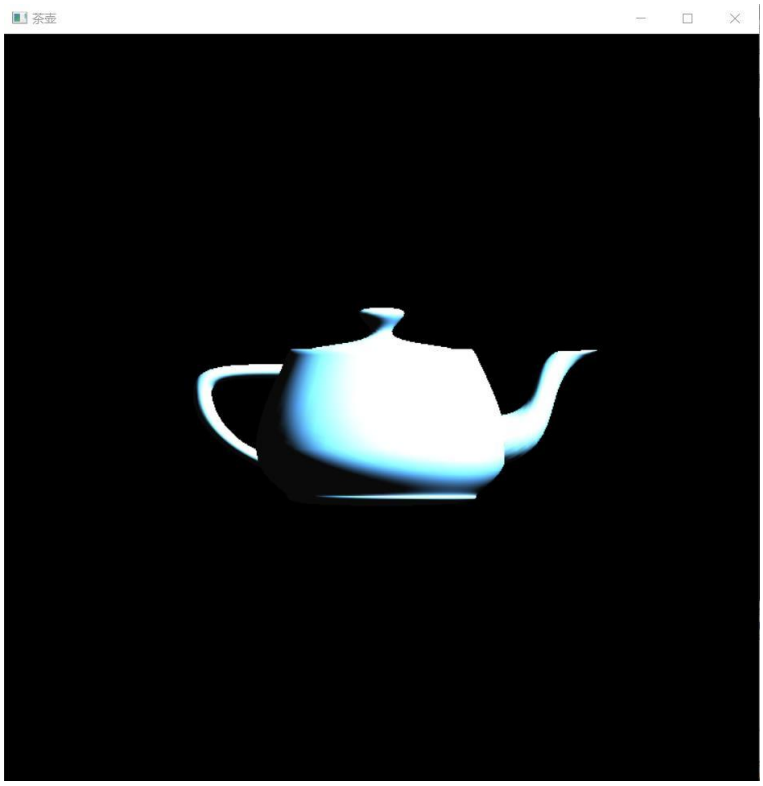
GLfloat light_position[]可以控制光的照射位置

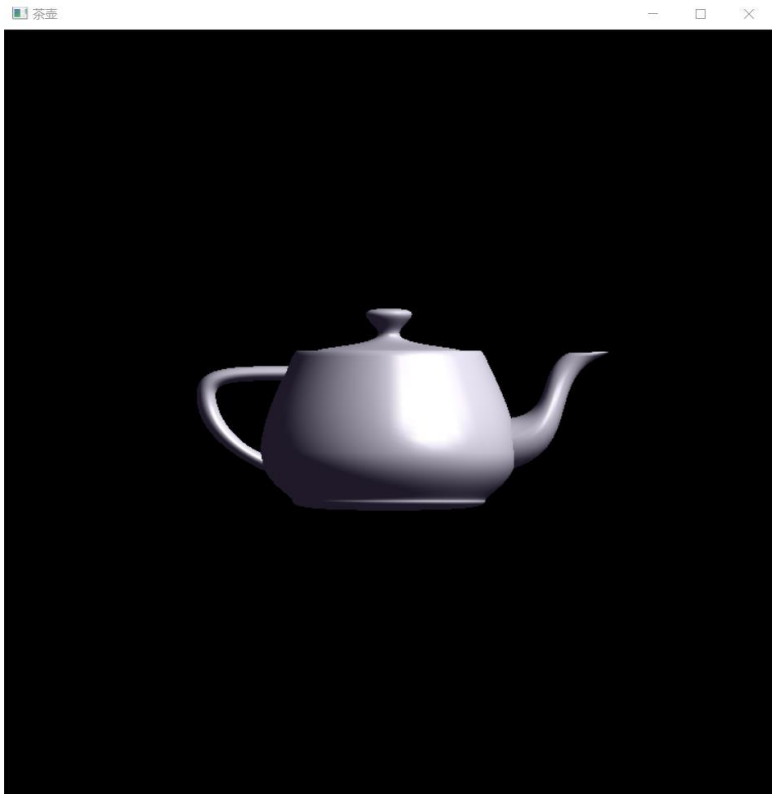
下面是将代码中 GLfloat light_position[]改为{-1.0,-1.0,1.0,0.0}的运行结果：



GLfloat white_light[]和 GLfloat Light_Model_Ambient[]可以更改光的颜色

下面是通过更改 GLfloat white_light[]和 GLfloat Light_Model_Ambient[]的参数得到的实验结果（有很多种，这里只展示两种）：





五、实验心得及小结

本次实验使用 OpenGL 实现三维图形渲染的一部分，设计一个茶壶的三维图形，在渲染过程加入了色彩和光照效果。由于本人能力有限，现阶段只完成了光照和颜色两种渲染效果，以后会继续深入学习。经过一学期的学习，不得不说计算机图形学是一门运用很广泛的学科，无论在学术上还是生活中，虽不像是空气这般生活的必需品，却也犹如鲜花白云这样，为学习生活增添了一丝别样的色彩。

参考文献：

360 百科：渲染

https://baike.so.com/doc/5275166-5509241.html

OpenGL 中常用的 GLUT 函数

https://blog.csdn.net/so_geili/article/details/53958960

《OpenGL 编程指南》示例笔记（1）--渲染光照球体

https://blog.csdn.net/iteye_5736/article/details/81886440

附录：

本次实验代码：

```
#include<stdlib.h>
#include<math.h>
#include<windows.h>
#define GLUT_DISABLE_ATEXIT_HACK
#include<gl/glut.h>

//绘制茶壶
//自定义初始化 opengl 函数
void init(void)
{
    //材质反光性设置
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };  //镜面反射参数
    GLfloat mat_shininess[] = { 50.0 };               //高光指数
    GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };
    GLfloat white_light[] = { 1.0, 1.0, 1.0, 1.0 };   //灯位置(1,1,1),
最后 1-开关
    GLfloat Light_Model_Ambient[] = { 0.2, 0.2, 0.2, 1.0 }; //环境光
参数

    glClearColor(0.0, 0.0, 0.0, 0.0);  //背景色
    glShadeModel(GL_SMOOTH);           //多变性填充模式

    //材质属性
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);

    //灯光设置
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
    glLightfv(GL_LIGHT0, GL_DIFFUSE, white_light);   //散射光属性
    glLightfv(GL_LIGHT0, GL_SPECULAR, white_light);  //镜面反射光
    glLightModelfv(GL_LIGHT_MODEL_AMBIENT, Light_Model_Ambient);  //
环境光参数

    glEnable(GL_LIGHTING);   //开关:使用光
    glEnable(GL_LIGHT0);     //打开 0#灯
    glEnable(GL_DEPTH_TEST); //打开深度测试
}

void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```

```
    glutSolidTeapot(0.5);
    glFlush();   //glSwapBuffers();
}

void reshape(int w, int h)
{
    glViewport(0, 0, (GLsizei)w, (GLsizei)h);

    //设置投影参数
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    //正交投影
    if (w <= h)
        glOrtho(-1.5, 1.5, -1.5*(GLfloat)h / (GLfloat)w, 1.5*(GLfloat)h
/ (GLfloat)w, -10.0, 10.0);
    else
        glOrtho(-1.5*(GLfloat)w / (GLfloat)h, 1.5*(GLfloat)w /
(GLfloat)h, -1.5, 1.5, -10.0, 10.0);

    //设置模型参数--几何体参数
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(750, 750);
    glutInitWindowPosition(75, 75);
    glutCreateWindow("茶壶");
    init();
    glutDisplayFunc(display);
    glutReshapeFunc(reshape);
    glutMainLoop();
    return 0;
}
```