# 计算机图形学实验报告

沈俞霖

2017 年 3 月 27 日

**成员** 软件 51 沈俞霖
**学号** 2151601013
**提交日期** 2017 年 3 月 27 日
**联系电话** 13679119978

# 目录

# 1 几何图形的绘制

## 1.1 实验目的

利用 OpenGL 绘制简单的几何图形

## 1.2 实验内容

绘制点、直线（实线和虚线）、多边形（区分正反面）、多边形镂空

## 1.3 实验代码

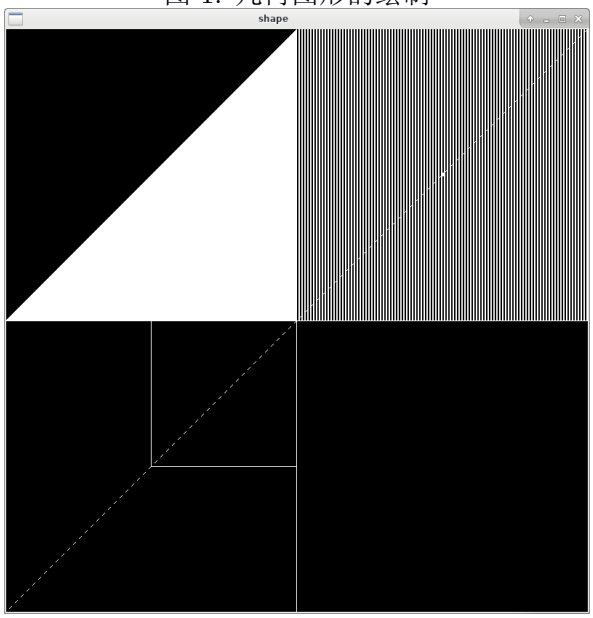
```
#include <GL/glut.h>
#include <cstring>

GLubyte mask[128];

void initMask() {
  std::memset(mask, 0b00110011, sizeof(mask));
}

void myDisplay() {
  // Initialize canvas
  glClear(GL_COLOR_BUFFER_BIT);

  // Points
  // Draw point at (0.5, 0.5)
  glPointSize(5.0f);
  glBegin(GL_POINTS);
    glVertex2f(0.5f, 0.5f);
  glEnd();

  // Lines
  // Draw solid lines from (-1, 0) to (1, 0) and from (0, -1)
  ↪  to (0, 1)
```

```
23    glBegin(GL_LINES);
24      glVertex2f(-1.0f, 0.0f);
25      glVertex2f(1.0f, 0.0f);
26      glVertex2f(0.0f, -1.0f);
27      glVertex2f(0.0f, 1.0f);
28    glEnd();
29    // Draw stipple lines from (-1, -1) to (1, 1)
30    glEnable(GL_LINE_STIPPLE);
31    glLineStipple(4, 0xAAAA);
32    glBegin(GL_LINES);
33      glVertex2f(-1.0f, -1.0f);
34      glVertex2f(1.0f, 1.0f);
35    glEnd();
36    glDisable(GL_LINE_STIPPLE);
37
38    // Polygons
39    // Initialize fill methods
40    glPolygonMode(GL_FRONT, GL_FILL);
41    glPolygonMode(GL_BACK, GL_LINE);
42    glFrontFace(GL_CCW);
43    // Draw a triangle at (0, 0), (0, 1), (-1, 0), we see its
      ↪  front face
44    glBegin(GL_POLYGON);
45      glVertex2f(0.0f, 0.0f);
46      glVertex2f(0.0f, 1.0f);
47      glVertex2f(-1.0f, 0.0f);
48    glEnd();
49    // Draw a rectangle at (0, 0), (-0.5, -0.5), (0, -0.5),
      ↪  (-0.5, 0), we see its back face
50    glBegin(GL_POLYGON);
51      glVertex2f(0.0f, 0.0f);
52      glVertex2f(0.0f, -0.5f);
53      glVertex2f(-0.5f, -0.5f);
```

```
54      glVertex2f(-0.5f, 0.0f);
55    glEnd();
56    // Stipple polygon
57    glEnable(GL_POLYGON_STIPPLE);
58    glPolygonStipple(mask);
59    glBegin(GL_POLYGON);
60      glVertex2f(0.0f, 0.0f);
61      glVertex2f(1.0f, 0.0f);
62      glVertex2f(1.0f, 1.0f);
63      glVertex2f(0.0f, 1.0f);
64    glEnd();
65    glDisable(GL_POLYGON_STIPPLE);
66
67    // Flush to output
68    glFlush();
69  }
70
71  int main(int argc, char *argv[]){
72    glutInit(&argc, argv);
73    glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
74    glutInitWindowPosition(100, 100);
75    glutInitWindowSize(800, 800);
76    glutCreateWindow("shape");
77    glutDisplayFunc(&myDisplay);
78
79    initMask();
80
81    glutMainLoop();
82    return 0;
83  }
```

### 1.4 实验结果

图 1: 几何图形的绘制




### 1.5 小结

通过这次实验，我学会了初始化 OPENGL 和使用 OPENGL 绘制简单的 2D 图形，初步认识了 OPENGL1.0 的状态机模型

## 2 着色方式

### 2.1 实验目的

掌握 OpenGL 两种颜色模型：RGBA 模型和颜色索引模型

### 2.2 实验内容

- 利用 RGBA 模型对多边形进行着色
- 利用颜色索引模型对多边形进行着色
- 分别设置平滑和单色两种方式绘制多边形，观察两种方式的区别。

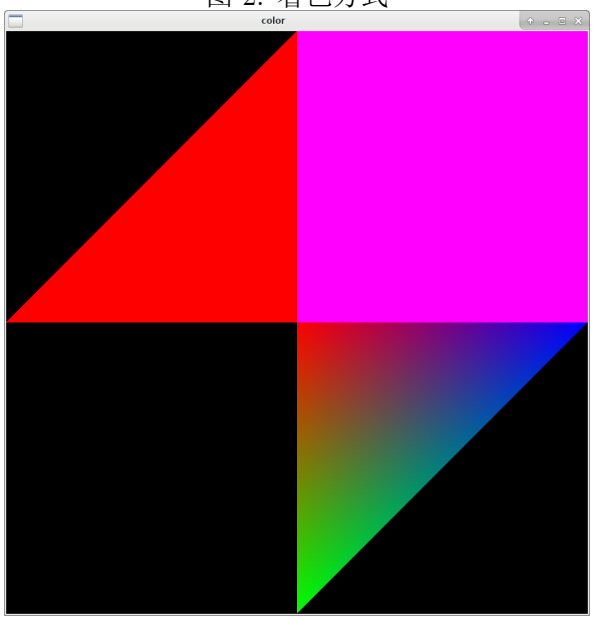
## 2.3 实验代码

```
#include <GL/glut.h>

void myDisplay() {
  // Initialize canvas
  glClear(GL_COLOR_BUFFER_BIT);

  // Color a polygon with RGBA color
  glColor3f(1.0f, 0.0f, 1.0f);
  glRectf(0.0f, 0.0f, 1.0f, 1.0f);

  // Color a polygon with different color smoothing model
  // GL_SMOOTH model
  glShadeModel(GL_SMOOTH);
  glBegin(GL_POLYGON);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex2f(0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex2f(0.0f, -1.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex2f(1.0f, 0.0f);
  glEnd();
  // GL_FLAT model
  glShadeModel(GL_FLAT);
  glBegin(GL_POLYGON);
    glColor3f(1.0f, 0.0f, 0.0f);
    glVertex2f(0.0f, 0.0f);
    glColor3f(0.0f, 1.0f, 0.0f);
    glVertex2f(0.0f, 1.0f);
    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex2f(-1.0f, 0.0f);
  glEnd();

```

```
33    // Flush to output
34    glFlush();
35  }
36
37  int main(int argc, char *argv[]){
38    glutInit(&argc, argv);
39    glutInitDisplayMode(GLUT_RGBA | GLUT_SINGLE);
40    glutInitWindowPosition(100, 100);
41    glutInitWindowSize(800, 800);
42    glutCreateWindow("color");
43    glutDisplayFunc(&myDisplay);
44    glutMainLoop();
45    return 0;
46  }
```

## 2.4 实验结果

图 2: 着色方式



### 2.5 小结

通过这次实验,我巩固了计算机内颜色表示的知识,初步认识了 OPENGL 的颜色模型的概念和使用方法

# 3 三维场景的绘制

### 3.1 实验目的

掌握 OpenGL 的模型变换和视图变换、投影变换与视口变换，熟悉矩阵堆栈的操作。

### 3.2 实验内容

绘制太阳、地球与月亮，采用 OpenGL 各种变换，绘制它们之间的正确三维关系。

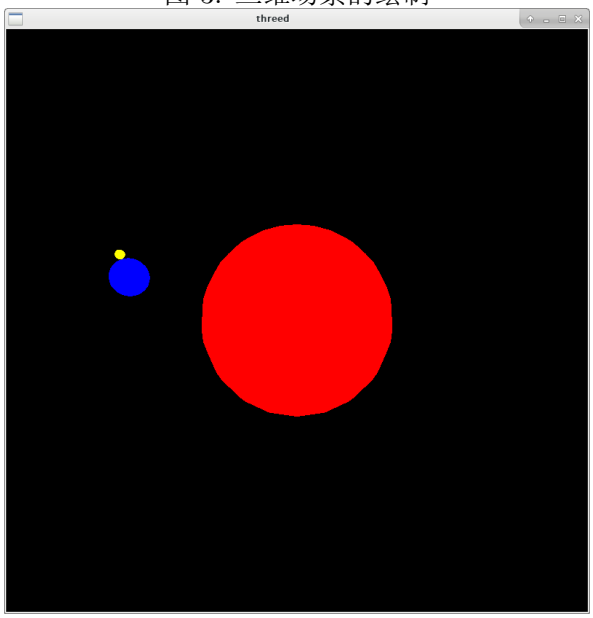
### 3.3 实验代码

```
#include <GL/glut.h>

static int day = 200;
void myDisplay(void)
{
  glEnable(GL_DEPTH_TEST);
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(75, 1, 1, 400000000);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  gluLookAt(0, -200000000, 200000000, 0, 0, 0, 0, 0, 1);

  // Draw sun
  glColor3f(1.0f, 0.0f, 0.0f);
```

```
18    glutSolidSphere(69600000, 20, 20);
19    // Draw earth
20    glColor3f(0.0f, 0.0f, 1.0f);
21    glRotatef(day/360.0*360.0, 0.0f, 0.0f, -1.0f);
22    glTranslatef(150000000, 0.0f, 0.0f);
23    glutSolidSphere(15945000, 20, 20);
24    // Draw moon
25    glColor3f(1.0f, 1.0f, 0.0f);
26    glRotatef(day/30.0*360.0 - day/360.0*360.0, 0.0f, 0.0f,
      ↪  -1.0f);
27    glTranslatef(38000000, 0.0f, 0.0f);
28    glutSolidSphere(4345000, 20, 20);
29
30    glFlush();
31  }
32
33  int main(int argc, char *argv[]){
34    glutInit(&argc, argv);
35    glutInitDisplayMode(GLUT_RGBA | GLUT_SINGLE);
36    glutInitWindowPosition(100, 100);
37    glutInitWindowSize(800, 800);
38    glutCreateWindow("threed");
39    glutDisplayFunc(&myDisplay);
40    glutMainLoop();
41    return 0;
42  }
```

### 3.4 实验结果

图 3: 三维场景的绘制




### 3.5 小结

通过这次实验，我学会了计算机图形学常用的 3D 视图变换的具体实现，学会了使用 OPENGL 编写 3D 程序

## 4 纹理映射

### 4.1 小组成员

- 软件 51，沈俞霖，学号 2151601013
- 软件 51，李南辰，学号 2151601010

### 4.2 实验目的

掌握 OpenGL 的纹理映射功能，绘制真实感图形。

## 4.3 实验内容

采用绘制圆球和纹理映射的方式，实现地球仪的绘制，并加入光照、纹理等信息，使得所绘制的地球仪显得比较逼真。

## 4.4 实验代码

```
// ---------------------------------
// main.cpp 主程序
// Created by sylxjtu on 2016/11/17.
// ---------------------------------

#include <GL/glew.h>
#include <GL/glut.h>
#include "TextureImage.h"

// 定义初始角度，角速度，角加速度，FPS
double angle = 0;
double speed = 5;
double accelerate = 0;
const double FPS = 60;

// 定义贴图对象
TextureImage t;

void display();

// 初始化绘图环境
void init() {
    // 载入贴图文件
    t.loadFile("earth.bmp", 1024, 1024);
    t.loadTexture();
    // 初始化光照
    GLfloat light_position[] = {1.0, 0.3, 0.0, 0.0};
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

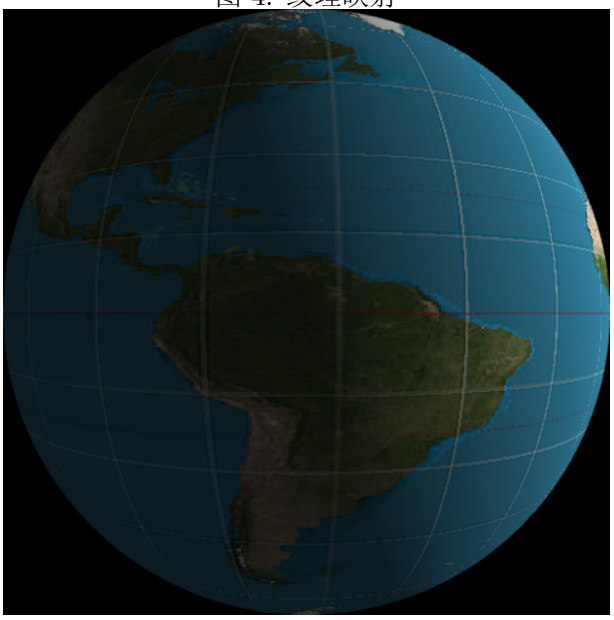
```
29
30      // 启用颜色、深度测试、光照
31      glEnable(GL_COLOR_MATERIAL);
32      glEnable(GL_NORMALIZE);
33      glEnable(GL_DEPTH_TEST);
34      glEnable(GL_LIGHTING);
35      glEnable(GL_LIGHT0);
36  }
37
38  // 动画函数
39  void idle() {
40      // 改变旋转角
41      speed += accelerate / FPS;
42      angle += speed / FPS;
43      display();
44  }
45
46  // 画地球函数
47  void drawSphere(int radius) {
48      // 配置贴图
49      glEnable(GL_TEXTURE_2D);
50      glBindTexture(GL_TEXTURE_2D, t.textureID);
51      glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
        ↪  GL_NEAREST);
52      glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
        ↪  GL_NEAREST);
53      // 画球体
54      GLUquadricObj *sphere = gluNewQuadric();
55      gluQuadricDrawStyle(sphere, GLU_FILL);
56      gluQuadricTexture(sphere, TRUE);
57      gluQuadricNormals(sphere, GLU_SMOOTH);
58      gluSphere(sphere, radius, 1024, 1024);
59  }
```

```
60
61 // 显示函数
62 void display() {
63     // 清空缓冲区
64     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
65     // 初始化视角
66     glMatrixMode(GL_PROJECTION);
67     glLoadIdentity();
68     gluPerspective(60, 1, 1, 100);
69     // 初始化视点
70     glMatrixMode(GL_MODELVIEW);
71     glLoadIdentity();
72     gluLookAt(0, -10, 0, 0, 0, 0, 0, 0, 1);
73     // 旋转球体
74     glRotated(angle, 0, 0, 1);
75     // 画球
76     drawSphere(5);
77     // 显示
78     glutSwapBuffers();
79     glFlush();
80 }
81
82 int main(int argc, char **argv) {
83     // 初始化显示
84     glutInit(&argc, argv);
85     glutInitDisplayMode(GL_DOUBLE | GL_RGB | GL_DEPTH);
86     glutInitWindowPosition(100, 100);
87     glutInitWindowSize(500, 500);
88     glutCreateWindow("Title");
89     init();
90     glutDisplayFunc(display);
91     glutIdleFunc(idle);
92     // 进入主循环
```

```
93      glutMainLoop();
94  }
```

## 4.5 实验结果

图 4: 纹理映射



## 4.6 小结

通过这次实验我锻炼了我的综合能力，学会了使用 OPENGL 编写中等规模程序的方法和技巧