浙江大学实验报告

课程名称：计算机图形学 指导老师：_____成绩：_____
实验名称：GLUT 程序设计 实验类型：基础实验 同组学生姓名：_____

装

订

线

一、实验目的和要求

学会配置 GLUT 开发库并使用 Visual Studio C++开发 OpenGL 程序。

二、实验内容

在 Windows 系统中，配置 GLUT 库：解压并打开文件夹 glut.zip，取出 glut.h，glut32.lib，glut32.dll。之后有两种配置方式，一是将以上 3 个文件分别放在系统盘的相应目录下；二是针对具体项目（本次实验给定项目 Ex1）进行配置。

开发 OpenGL 程序：编译运行项目 Ex1，确认无误后修改代码生成以下图形：

三、主要仪器设备

Visual Studio 2017
Glut 压缩包
Ex1 工程

四、操作方法和实验步骤

（一） 配置 OpenGL 库

1. 为 VS2017 安装 C/C++组件，安装 NuGet 管理包
2. 下载 glut.rar，寻找网络资源
   ① 将 glut.h 放到 D:\Program Files (x86)\Microsoft Visual Studio\2017\Professional\VC\Tools\MSVC\14.16.27023\include\gl 下，
   ② 将 glut.lib,glut32.lib 放到 D:\Program Files (x86)\Microsoft Visual Studio\2017\Professional\VC\Tools\MSVC\14.16.27023\lib\x86 下
   ③ 将 glut.dll,glut32.dll 放到 C:\Windows\SysWOW64 下
3. 打开 VS2017，打开项目 Ex1
4. 点击界面上方项目，选择管理 NuGet 程序包，选择浏览，搜索 nupengl，将 nupengl.core 和 nupengl.core.redist 都安装（每次新建项目都要安装一下 nupengl.core 和 nupengl.core.redist）
5. 下载 Libraries.zip 和 glad.c, 解压后放到 D:\Program Files (x86)\Microsoft Visual Studio\2017\Professional 下，打开 VS2017 回到刚才的界面，右击 Ex1,选择属性，点击 VC++目录，将 D:\Program Files (x86)\Microsoft Visual Studio\2017\Professional\Libraries\Include 添加到包含目录下，将 D:\Program Files (x86)\Microsoft Visual Studio\2017\Professional\Libraries\Libs 添加到库目录下，关闭窗口。
6. 再次右击 Ex1，点击添加>现有项，添加 glad.c。
7. 这样 OpenGL 库就配置完成了

（二） 开发 OpenGL 程序，绘制五星红旗

1. 首先对 glut 函数库进行初始化，制定 glutCreateWindow 函数将要创建的窗口显示模式（RGB 模式，双缓冲），设置窗口位置、大小和标题，注册绘制回调函数

```
1. glutInit(&argc, argv);  //对 glut 函数库进行初始化
2. glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);  //指定 glutCreateWindow 函数将要创建的窗口显示模式 RGB 模式 双缓冲
3. glutInitWindowPosition(100, 100);  //设置窗口位置，窗口的左上角相对于整个屏幕的位置
4. glutInitWindowSize(400, 400);  //设置窗口大小（有可能被其它窗口覆盖）
5. glutCreateWindow("五星红旗");  //设置窗口标题
6. glutDisplayFunc(display);  //注册绘制回调函数，指定当窗口内容需要重绘时要调用的函数
```

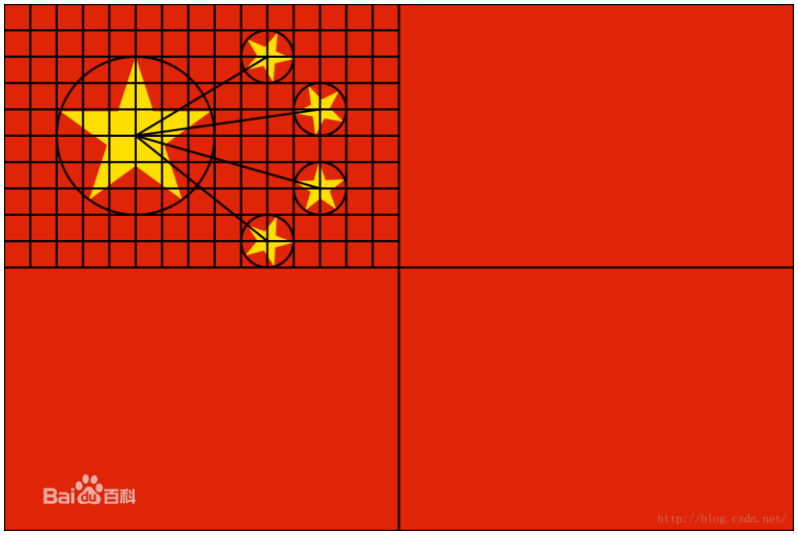
2. 绘制红旗

```
1. //绘制红旗
2. glColor3f(1, 0, 0);  //设置红色
3. glBegin(GL_QUADS);
4. glVertex2f(-0.9, 0.6);
5. glVertex2f(0.9, 0.6);
6. glVertex2f(0.9, -0.6);
7. glVertex2f(-0.9, -0.6);
8. glEnd();
```

3. 绘制星星

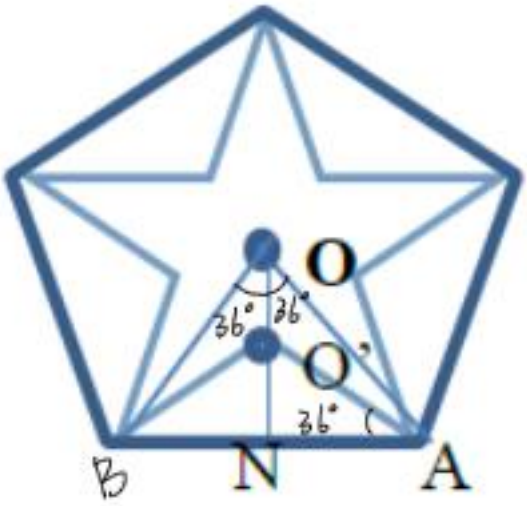
```
1. //绘制五角星
2. DrawStar(-0.60, 0.30, -0.60, 0.48);
3. DrawStar(-0.30, 0.48, -0.24, 0.48);
4. DrawStar(-0.18, 0.36, -0.24, 0.36);
5. DrawStar(-0.18, 0.18, -0.18, 0.24);
6. DrawStar(-0.30, 0.06, -0.24, 0.06);
```

五、实验数据记录处理及实验原理分析

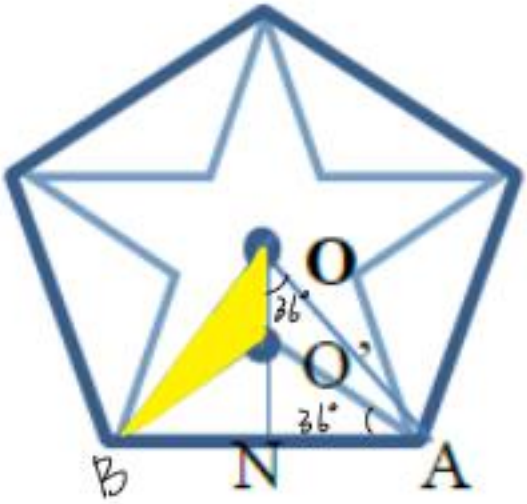


画红旗的方法比较简单（长宽比例 3：2），这里主要说明一下画五角星的数据处理过程和原理分析。

1. 根据上图可以看出每个星星的具体位置，由此可以看出每个五角星外点、内点和中心的坐标。大五角星的中心在网格上五下五，左五右十的位置，以 3 为半径画圆，外点均匀分布在圆上，其它四个五角星的中心分别在上二下八、左十右五，上四下六、左十二右三，上七下三，左十二右三，上九下一、左十右五的位置，以 1 为半径画圆，外点均匀分布在圆上。
2. 分别找出每个五角星的一个外点坐标，即可知道五角星中心到外点的距离为 OA，而五角星中心到内角的距离 OO’即 OO’ = OA*cos36°-OA*sin36°*tan36° = OA*cos72°/cos36° 。

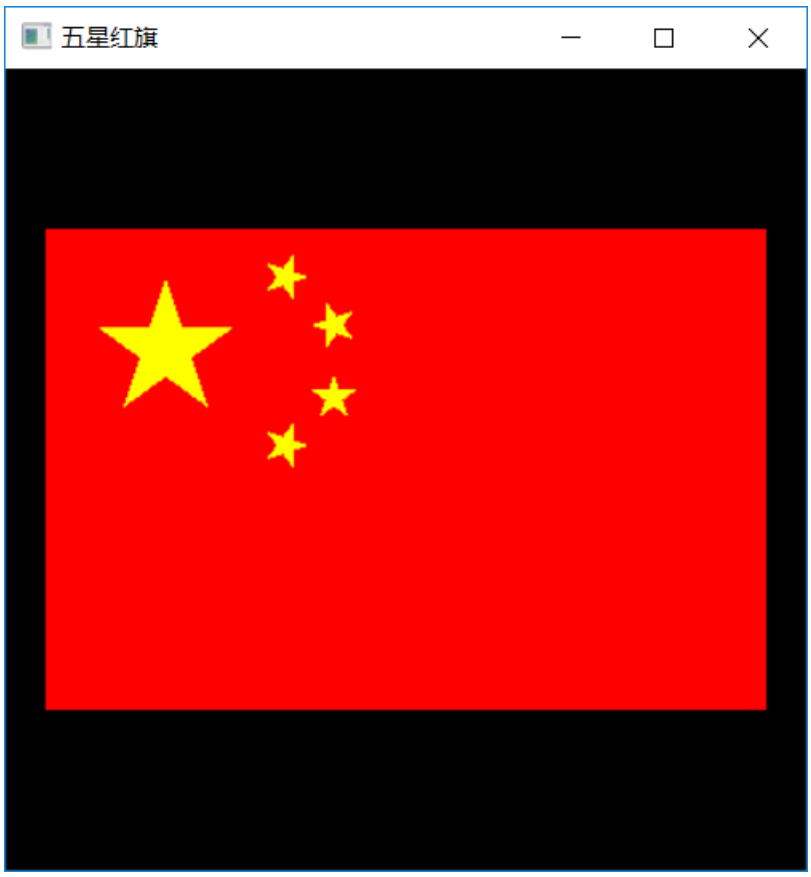


3. 最终绘制十个黄色的全等三角形拼凑成五角星（如图所示）



六、实验结果与分析

实验结果如图，比较成功地绘制出了五星红旗。



七、讨论、心得

万事开头难，刚开始接触 OpenGL 的时候认为这是一个非常抽象的东西，对算法和需要做些什么没有什么概念，但是摸索着配置好 OpenGL 库，上手去画这个五星红旗之后会对程序和规范性有更深一步的理解，其中也出现多次 GLfloat 和 glut 打错的现象，在不断的改进之后熟练度也有了一定的提升，通过本次实验我对图形学有了一定的基础认识。

八、源代码

```
1.  #define GLUT_DISABLE_ATEXIT_HACK
2.  #include<stdio.h>
3.  #include<math.h>
4.  #include<gl/glut.h>
5.
6.  const GLfloat PI = 3.1415926536f;  //定义圆周率
7.
8.  void display();  //绘制五星红旗
9.  void DrawStar(GLfloat cx, GLfloat cy, GLfloat mx, GLfloat my);  //其中 cx，cy 为五角星中心的横坐标和纵坐标，mx，my 为五角星其中一个顶点的坐标
10.
11. int main(int argc, char *argv[])
12. {
13.     glutInit(&argc, argv);  //对 glut 函数库进行初始化
14.     glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);  //指定 glutCreateWindow 函数将要创建的窗口显示模式 RGB 模式 双缓冲
```

```
    glutInitWindowPosition(100, 100);  //设置窗口位置，窗口的左上角相对于整个屏幕的位置
    glutInitWindowSize(400, 400);  //设置窗口大小（有可能被其它窗口覆盖）
    glutCreateWindow("五星红旗");  //设置窗口标题
    glutDisplayFunc(display);  //注册绘制回调函数，指定当窗口内容需要重绘时要调用的函数
    glutMainLoop();  //glut 事件处理循环
    return 0;

}

void DrawStar(GLfloat cx, GLfloat cy, GLfloat mx, GLfloat my)  //其中 cx，cy 为五角星中心的横坐标和纵坐标，mx，my 为五角星其中一个顶点的坐标
{
    int i;
    const GLfloat sin_72 = sin(PI * 0.4), cos_72 = cos(PI * 0.4);  //计算得到 sin72°，cos72°的值
    const GLfloat sin_36 = sin(PI * 0.2), cos_36 = cos(PI * 0.2);  //计算得到 sin36°，cos36°的值
    GLfloat long_length = sqrt((mx - cx) * (mx - cx) + (my - cy) * (my - cy));  //计算得到外点距离五角星中心的距离（半径）
    GLfloat sin_original = (my - cy) / long_length, cos_original = (mx - cx) / long_length;  //计算得到（mx,my）的初始点极坐标的 sin，cos 值
    GLfloat short_length = long_length * cos_72 / cos_36;  //计算得到内点距离五角星中心的距离（半径）

    GLfloat point[11][2];  //用来储存点的坐标
    GLfloat sin_temp, cos_temp;

    point[0][0] = mx, point[0][1] = my;
    point[10][0] = cx, point[10][1] = cy;

    //将外点的坐标存入数组中，存入偶数下标
    for (i = 1; i <= 4; i++) {
        sin_temp = sin_original * cos_72 + cos_original * sin_72;  //计算出旋转后的外点极坐标 sin 值
        cos_temp = cos_original * cos_72 - sin_original * sin_72;  //计算出旋转后的外点极坐标 sin 值
        point[2 * i][0] = cx + long_length * cos_temp;  //计算出旋转后的外点横坐标
        point[2 * i][1] = cy + long_length * sin_temp;  //计算出旋转后的外点纵坐标
        sin_original = sin_temp;  //改变原始点的 sin 值
        cos_original = cos_temp;  //改变原始点的 cos 值
    }

    sin_original = (my - cy) / long_length;
    cos_original = (mx - cx) / long_length;
```

```
51.
52.     //将内点的坐标存入数组中，存入奇数下标
53.     for (i = 1; i <= 5; i++) {
54.         if (i == 1) {
55.             sin_temp = sin_original * cos_36 + cos_original * sin_36;  //计算出旋转后的内点极坐标 sin 值
56.             cos_temp = cos_original * cos_36 - sin_original * sin_36;  //计算出旋转后的内点极坐标 sin 值
57.         }
58.         else {
59.             sin_temp = sin_original * cos_72 + cos_original * sin_72;  //计算出旋转后的内点极坐标 sin 值
60.             cos_temp = cos_original * cos_72 - sin_original * sin_72;  //计算出旋转后的内点极坐标 sin 值
61.         }
62.         point[2 * i - 1][0] = cx + short_length * cos_temp;  //计算出旋转后的内点横坐标
63.         point[2 * i - 1][1] = cy + short_length * sin_temp;  //计算出旋转后的内点纵坐标
64.         sin_original = sin_temp;  //改变原始点的 sin 值
65.         cos_original = cos_temp;  //改变原始点的 cos 值
66.     }
67.
68.     //设置黄色
69.     glColor3f(1, 1, 0);
70.
71.     //绘制十个三角形
72.     glBegin(GL_TRIANGLES);
73.     for (i = 0; i < 10; i++) {
74.         glVertex2fv(point[i % 10]);
75.         glVertex2fv(point[(i + 1) % 10]);
76.         glVertex2fv(point[10]);
77.     }
78.     glEnd();
79. }
80.
81. void display()  //绘制五星红旗
82. {
83.     glClear(GL_COLOR_BUFFER_BIT);  //清除颜色缓存
84.
85.     //绘制红旗
86.     glColor3f(1, 0, 0);  //设置红色
87.     glBegin(GL_QUADS);
88.     glVertex2f(-0.9, 0.6);
89.     glVertex2f(0.9, 0.6);
90.     glVertex2f(0.9, -0.6);
```

```
91.     glVertex2f(-0.9, -0.6);
92.     glEnd();
93.
94.     //绘制五角星
95.     DrawStar(-0.60, 0.30, -0.60, 0.48);
96.     DrawStar(-0.30, 0.48, -0.24, 0.48);
97.     DrawStar(-0.18, 0.36, -0.24, 0.36);
98.     DrawStar(-0.18, 0.18, -0.18, 0.24);
99.     DrawStar(-0.30, 0.06, -0.24, 0.06);
100.
101.     glutSwapBuffers();  //交换缓冲区
102. }
```