专业：数字媒体技术
姓名：杨锐
学号：3180101941
日期：2020/3/2
地点：家

浙江大学实验报告

课程名称：计算机图形学 指导老师：唐敏 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：无

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、实验内容和原理

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

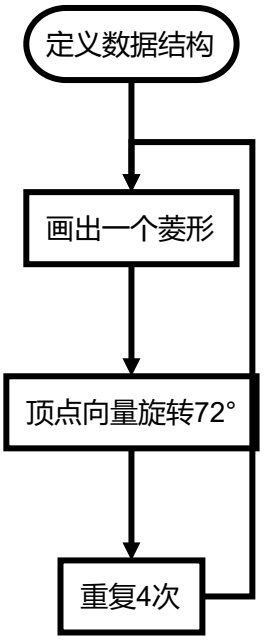
Visual Studio C++2008
Glut 压缩包
Ex1 工程

四、操作方法和实验步骤

0.实验前思考：

这次实验是在已有的红色背景下画出五个五角星组成五星红旗，这五个五角星位置不一，有大有小。

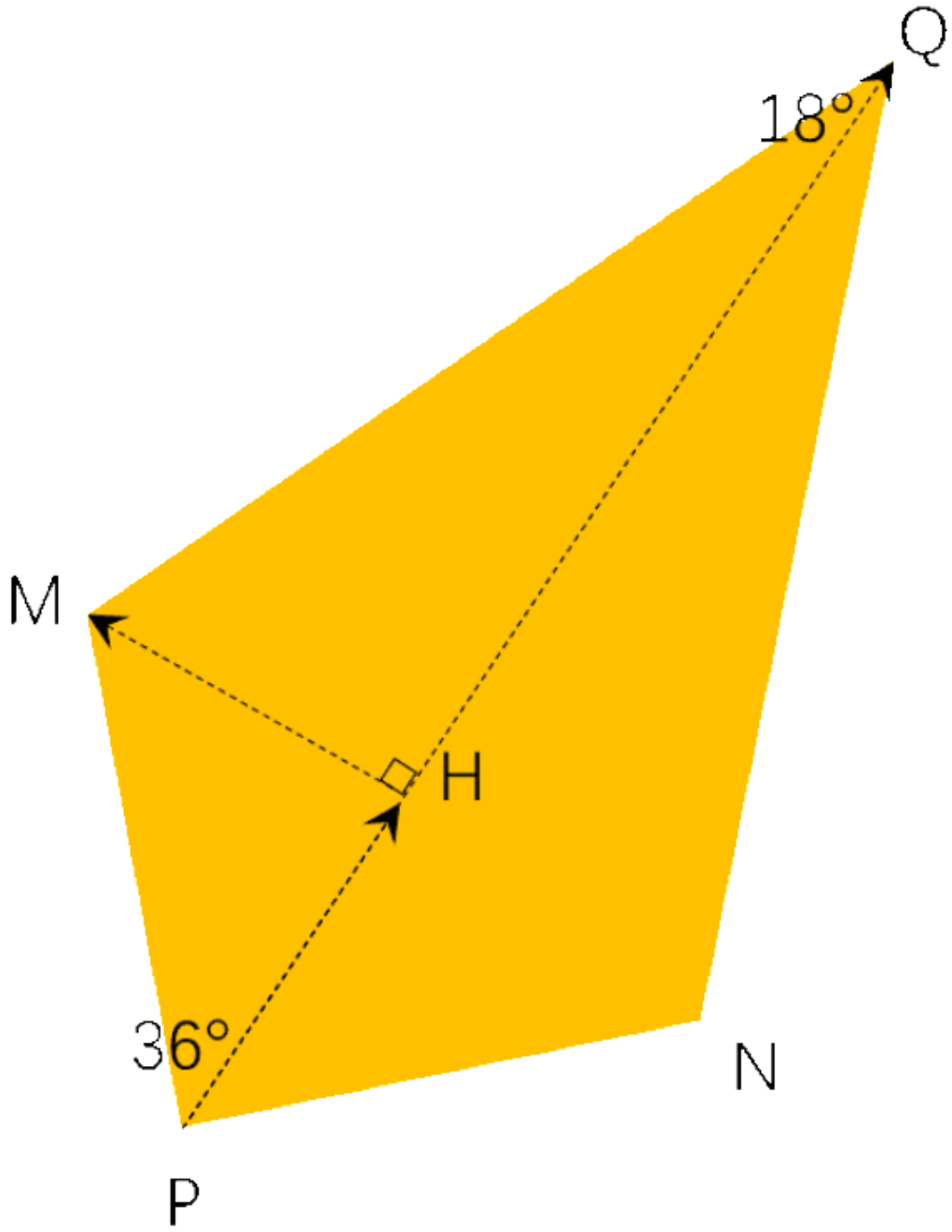
考虑到opengl只支持凸多边形，如果每个五角星都把十个顶点坐标找出来，用线段连接起来，势必太过麻烦，因此我想到了分割，**将单个五角星分割成几个凸多边形**，分割的方法很多，可以是：
①外边五个三角形里面一个五边形
②10个小三角形
③五个菱形。其中①涉及两种几何图形，不具有较强的规律性，舍弃；②中的三角形各自全等，但仍需要分别对两个基础三角形（这俩轴对称）进行旋转合成，并且三角形本身不具有特殊性，舍弃（其实用起来也不是很麻烦…）；③中五个菱形完全全等，且本身也为轴对称图形，其余四个可以通过其中一个旋转得到，因此我的思路到这就很清晰了：



画出一个菱形 -> 将相关点、向量旋转4次，每次72°合成单个五角星->根据相关点的位置画出其余四个五角星

1.画一个菱形

菱形可以使用`glBegin(GL_POLYGON)`函数，但考虑到菱形本身角度已知且为轴对称，因此没必要直接使用四个坐标，可以利用**三角函数**和**向量**计算得到：



已知向量PQ，由直角三角形MHP，和MHQ可得：

$|PH| = tan18/(tan36 + tan18)|PQ|$

$|HM| = tan36|PH|$

根据向量关系可由PQ坐标求得M、N坐标；

代码实现：

```
void glstar_part(glpoint2f P, glpoint2f Q) {//P为中心，Q为P对应顶点，MN为两侧顶点
        float k1 = tan(PI / 10) / (tan(PI / 5) + tan(PI / 10)); //tan18 /  (tan36 + tan18
        float k2 = tan(PI / 5);//tan36
    //glvector2f为定义的点/向量结构体
        glvector2f PQ = { Q.x - P.x , Q.y - P.y };
        glvector2f PH = { k1 * PQ.x , k1 * PQ.y };
        glvector2f HM = { k2 * PH.y , (-k2) * PH.x };

        glpoint2f M = { P.x + PH.x + HM.x , P.y + PH.y + HM.y };
        glpoint2f N = { P.x + PH.x - HM.x , P.y + PH.y - HM.y };

        glBegin(GL_POLYGON);//画四边形
        glVertex2f(P.x, P.y);
        glVertex2f(M.x, M.y);
        glVertex2f(Q.x, Q.y);
        glVertex2f(N.x, N.y);
        glEnd();
}
```

2.旋转

根据一条向量旋转α度后的坐标公式即可得
代码:

```
void rotate(glpoint2f P, glpoint2f *Q) {//注意此处传入指针/地址
        float rad = 2 * PI / 5;//72°

        glvector2f PQ = { Q->x - P.x , Q->y - P.y };
        glvector2f Pq = {
                cos(rad) * PQ.x + sin(rad) * PQ.y,
                cos(rad) * PQ.y + sin(rad) * (-PQ.x)
        };
        Q->x = P.x + Pq.x;
        Q->y = P.y + Pq.y;
}
```

3.合成

循环5次步骤1和2即可
代码:

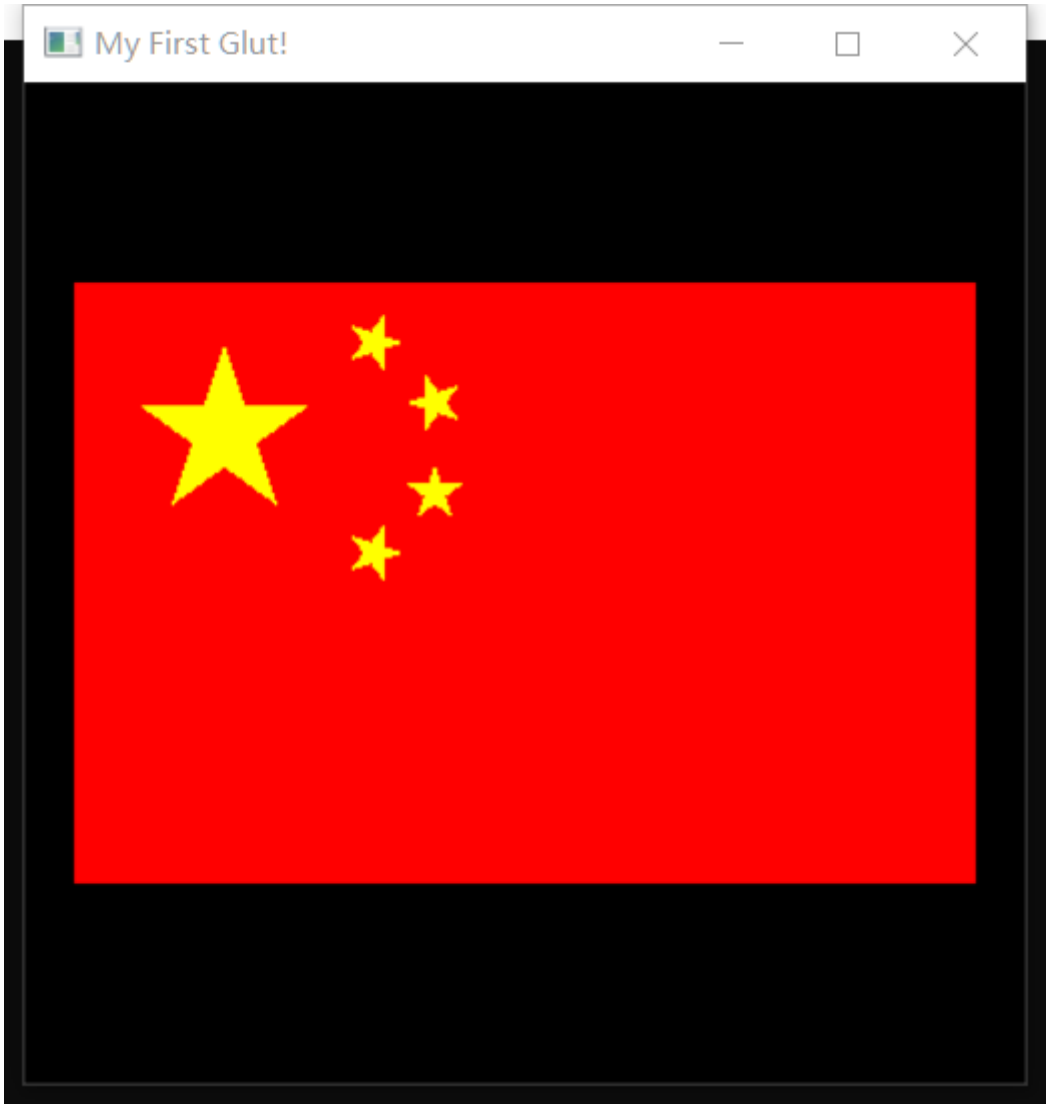
```
void glstar(GLfloat P_x, GLfloat P_y, GLfloat Q_x, GLfloat Q_y) {
        glpoint2f P = { P_x , P_y };
        glpoint2f Q = { Q_x , Q_y };
        for (int i = 0; i < 5; ++i) {
                glstar_part(P, Q);
                rotate(P, &Q);
        }
}
```

4.绘制其他小五角星

在这里我使用网上的标准五星红旗图片，直接分别带入中心坐标和对应顶点坐标。一是坐标本身并不难得到，二是我认为这个实验的关键在于**通过简洁地绘制出一个任意大小的五角星达到熟悉opengl的作用**，而不在于纠结大五角星和小五角星严格的相对位置关系。

五、结果和心得

结果:



心得:
这次实验是第一次练习在visual studio 上使用opengl，说实话，并不是很容易。

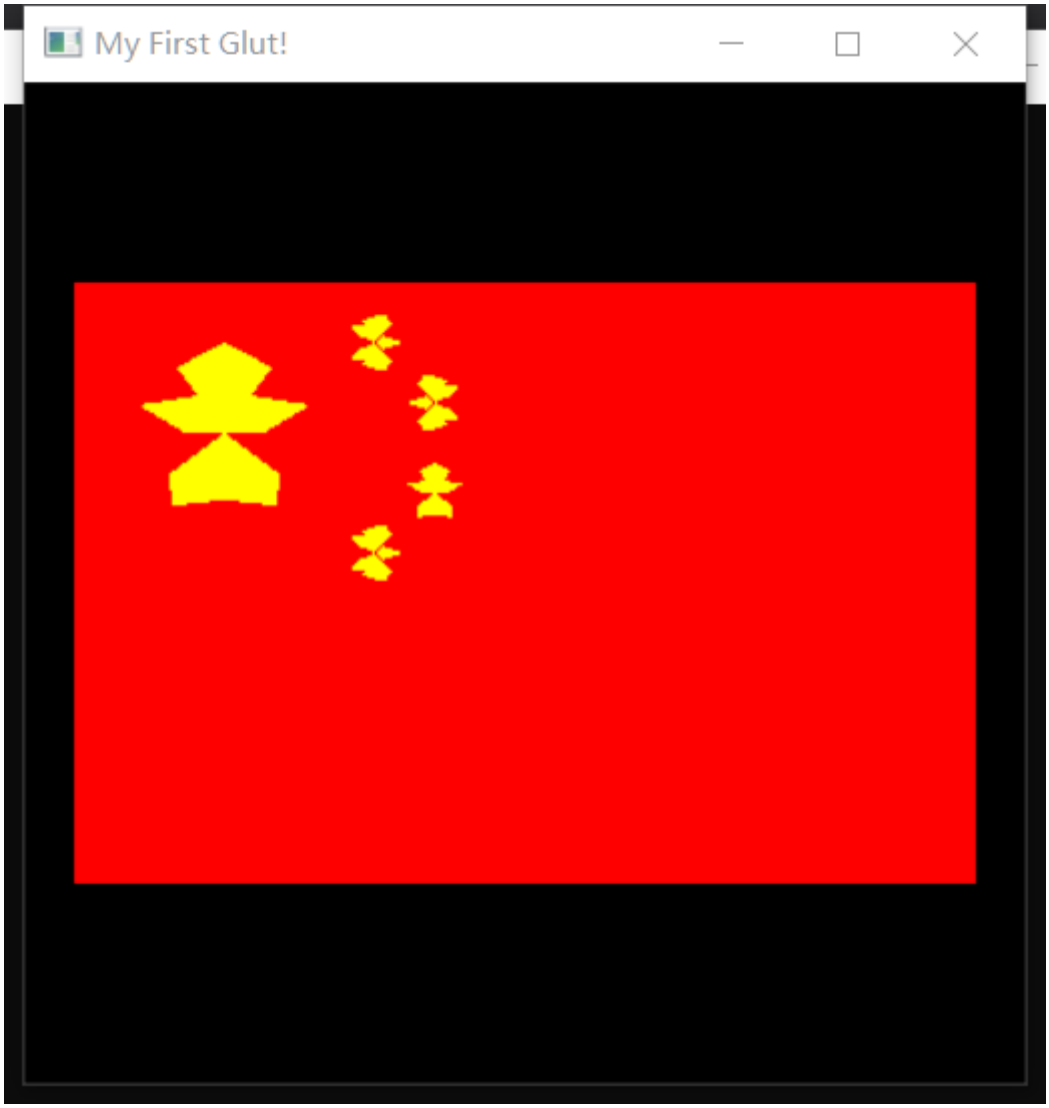
一开始是上星期搭建环境时，因为之前都是用的vscode，没有用过visual studio,所以我熟悉vs 的 workplace 和 props花了不少时间 ，另一方面因为"opengl","glut","glew","gflw"，这些东西一开始傻傻分不清楚，也下载了很多重复的文件。

然后是实验课正式着手做时，因为对opengl库的陌生和相关函数的不熟悉，过分纠结在一些很小的语法细节上，导致耽误了不少时间，也弄得自己不能静下心来思考图形内部的几何关系。

于是我选择在网上找了几个opengl的简单示例，通过阅读别人的代码和注释熟悉了opengl绘制最基本的二维图形的语法，再通过分析五角星的内部图形的几何关系，把实验思路理了出来。

但在具体代码实现时，也碰到不少问题：一个最明显的是代码里面数学公式太多，很容易就写错了，又不像普通c++那样很方便的直接打断点调试看变量，因为这里面的变量就只有坐标，而坐标如果不对我又没法肉眼就判断到底对应哪行代码不对了...

下面这个图就是:



一开始我以为是旋转公式错了，又重新推了一遍，没问题，最后才发现在第一步绘制菱形时，`glvector2f PH = { k1 * PQ.x , k1 * PQ.y };`
中的第二个k2写成k1。。。

总的来说：这次实验一方面熟悉了vs和opengl的简单操作，另一方面复习了向量和三角函数有关知识，学会了如何在代码中表示简单的几何元素