浙江大学

本科实验报告

课程名称：计算机图形学

姓　　名：王晨宇

学　　院：计算机科学与技术学院

系：

专　　业：计算机科学与技术

学　　号：3180104919

指导教师：唐敏

目录

一、实验目的和要求 **1**

二、实验环境 **1**

三、实验内容和原理 **1**

3.1 五星红旗的坐标位置 1

四、操作方法和实验步骤 **2**

4.1 实验环境的配置 2

4.2 坐标和标准设备坐标的转换 2

4.3 五角星的绘制 2

代码实现 2

4.4 国旗的绘制 3

五、实验结果 **4**

六、讨论和心得 **6**

浙江大学实验报告

课程名称：	计算机图形学	实验类型：	综合		
实验项目名称：	GLFW 程序设计				
学生姓名：	王晨宇	专业：	计算机科学与技术	学号：	3180104919
同组学生姓名：	None	指导老师：	唐敏		
实验地点：	紫金港机房	实验时间：	2021-03-26		

一、 实验目的和要求

学会配置 GLFW、GLAD 和 GLM 开发库并使用 Visual Studio C++ 开发 OpenGL 程序。

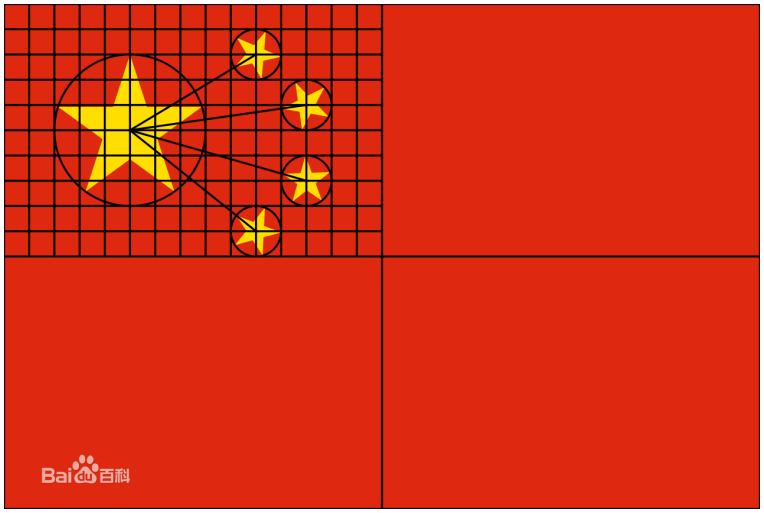
二、 实验环境

- Visual Studio C++ 2019
- GLFW、GLAD、GLM

三、 实验内容和原理

3.1 五星红旗的坐标位置

参考了标准的国旗画法：



四、 操作方法和实验步骤

4.1 实验环境的配置

参考助教学长的配置过程就不再赘述

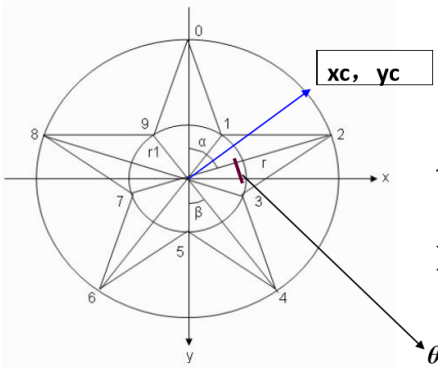
4.2 坐标和标准设备坐标的转换

为了方便坐标的确定，使用了一个从相对坐标 $(x, y) \in (w, h)$ 转换到标准设备坐标的函数：

```
glm::vec2 pos2ndc(int x, int y, int w, int h) {
  return glm::vec2(-0.5f + (float)x / w, 0.5f - (float)y / h);
}
```

4.3 五角星的绘制

把五角星看成中点出发的十个三角形



外接圆和内接圆半径比通过余弦定理计算得到为：

$$d = \frac{\cos 72^\circ}{\cos 36^\circ}$$

因为最终的坐标都要映射到标准设备坐标，所以需要考虑长宽比，我们在用向量得到坐标时，横向的方向需要乘上伸缩的比例 $aspect = \frac{width}{height}$ 为了方便修改五角星的坐标位置，我在构造函数中只是简单地绑定了 vao 和 vbo，同时给坐标一个初始位置。通过 set 函数来设定具体位置。

代码实现

```
Star::Star() {
  for (int i = 0; i < 30; i++) _vertices[i] = glm::vec2(0, 0);
```

```
3     glGenBuffers(1, &_vbo);
4     glGenVertexArrays(1, &_vao);
5   }
6   void Star::check() {
7     printf("%f %f\n", _position[0], _position[1]);
8   }
9   // 每次调整坐标
10  void Star::set(const glm::vec2& position, float rotation, float radius, float
        aspect) {
11    _position = position, _rotation = rotation, _radius = radius;
12    float angle = rotation;
13    float k = 0.382;
14    printf("%f\n", aspect);
15    // 画10个三角形
16    for (int i = 0; i < 5; ++i) {
17      _vertices[6 * i] = _position;
18      _vertices[6 * i + 1] = _position + glm::vec2(cos(angle) * radius / aspect,
            sin(angle) * radius);
19      angle = angle + theta;
20      _vertices[6 * i + 2] = _position + glm::vec2(cos(angle) * radius * k / aspect
            , sin(angle) * radius * k);
21      _vertices[6 * i + 3] = _position;
22      _vertices[6 * i + 4] = _position + glm::vec2(cos(angle) * radius * k / aspect
            , sin(angle) * radius * k);
23      angle = angle + theta;
24      _vertices[6 * i + 5] = _position + glm::vec2(cos(angle) * radius / aspect,
            sin(angle) * radius);
25    }
26    // 调整坐标后需要重新绑定vao，把坐标数组复制到缓冲中
27    glBindVertexArray(_vao);
28    glBindBuffer(GL_ARRAY_BUFFER, _vbo);
29    glBufferData(GL_ARRAY_BUFFER, sizeof(_vertices), _vertices, GL_DYNAMIC_DRAW);
30
31    glVertexAttribPointer(0, 2, GL_FLOAT, GL_FALSE, sizeof(glm::vec2), (void*)0);
32    glEnableVertexAttribArray(0);
33    glBindVertexArray(0);
34  }
```

4.4 国旗的绘制

因为标准的国旗长宽比是 3 : 2，所以需要修改初始的长宽数据。

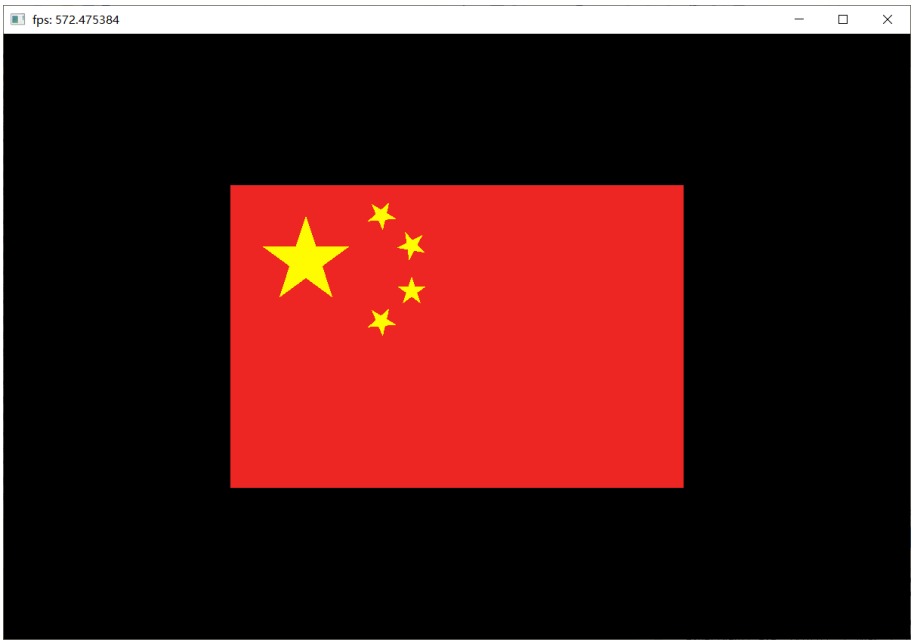
```
1   /* window info */
2     GLFWwindow* _window = nullptr;
3     std::string _windowTitle;
4     int _windowWidth = 1200;
5     int _windowHeight = 800;
```

为了保证五角星的比例不变，同时尽可能少地修改五角星的坐标。我把改变窗口大小的回调函数在 RenderFlag 类中重写。相应的要把 application 构造函数做的事情放到 RenderFlag 的构造函数里去。当然第一次初始化也要设置一个初始的坐标，这里就不再赘述。

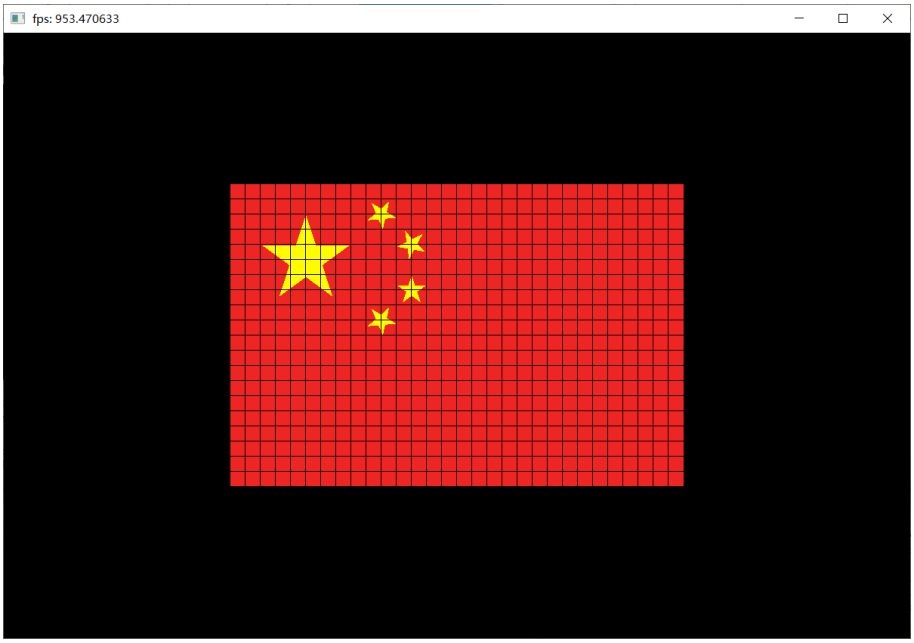
```
1  void RenderFlag::framebufferResizeCallback(GLFWwindow* window, int w, int h) {
2    RenderFlag* app = reinterpret_cast<RenderFlag*>(glfwGetWindowUserPointer(window
         ));
3    app->_windowWidth = w;
4    app->_windowHeight = h;
5    printf("%d %d\n", w, h);
6    float aspect = (float)w / h;
7    _stars[4]->set(pos2ndc(1, 1, 6, 4), pi / 2, 0.15, aspect);
8    _stars[3]->set(pos2ndc(10, 2, 30, 20), pi / 2 - atan2(3.0, 5.0), 0.05, aspect);
9    _stars[2]->set(pos2ndc(12, 4, 30, 20), pi + atan2(1.0, 7.0), 0.05, aspect);
10   _stars[1]->set(pos2ndc(12, 7, 30, 20), pi / 2, 0.05, aspect);
11   _stars[0]->set(pos2ndc(10, 9, 30, 20), pi / 2 - atan2(3.0, 5.0), 0.05, aspect);
12   glViewport(0, 0, w, h);
13 }
```

五、　实验结果

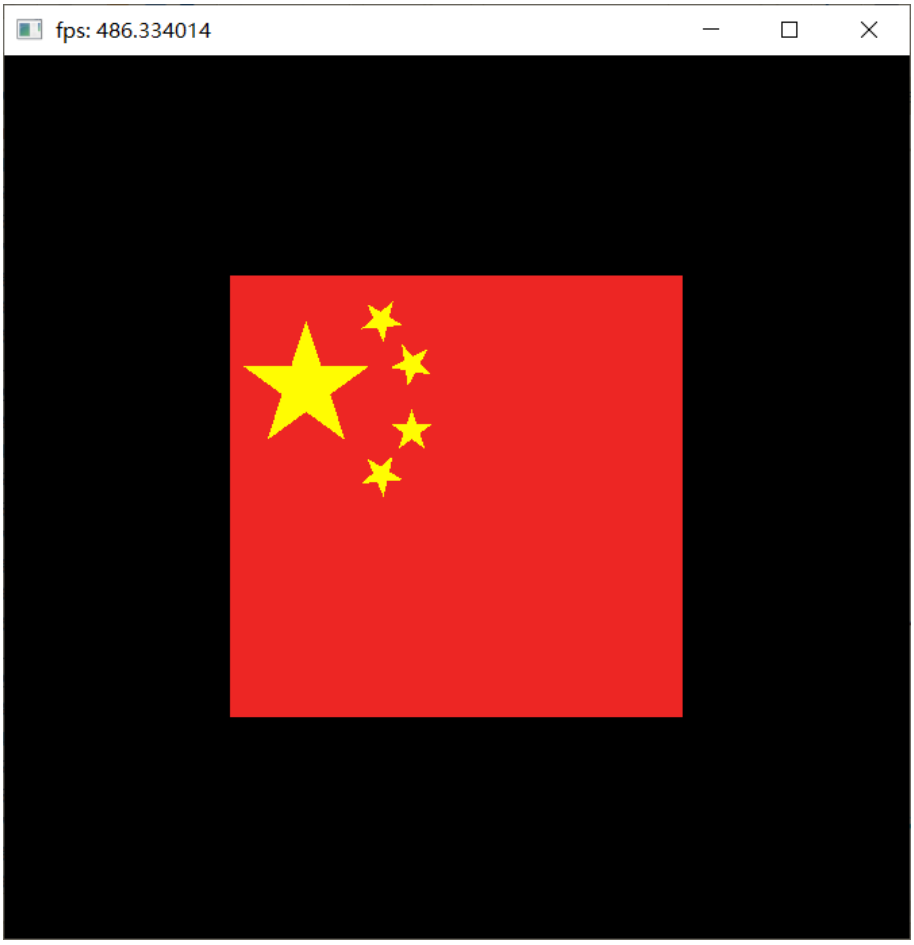
初始的样子：



加辅助线的版本（显得非常标准）



拉伸之后的效果：



六、 讨论和心得

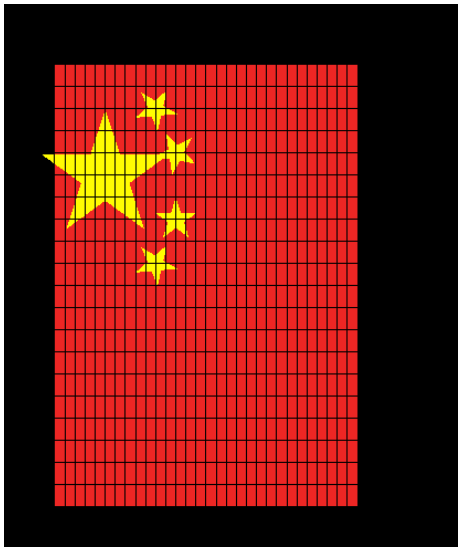
在这次实验中我使用助教学长提供的框架完成了五星红旗的绘制。在学习中我对 glfw 绘制图形流程以及一些特殊的对象（VAO、VBO）有了更进一步的了解。

在尝试解决拉伸窗口五角星比例保持不变的过程中，我也尝试了许多方法。一开始把五角星的坐标设置写在了 render 里，这样在每一次渲染前都会根据当前窗口的大小修改每个五角星的位置坐标。效果也还行，但是存在的问题是，大多数我们观察这个五角星的时候窗口并不会变化，每次渲染都重新计算、绑定同样的位置函数导致大量的资源浪费。所以最后改成了在改变窗口大小的回调函数中实现坐标的变换。

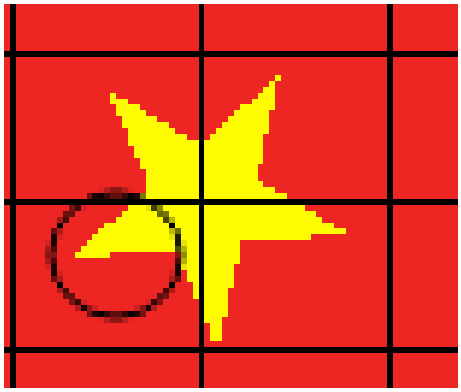
对 VAO 和 VBO 的理解可能还比较粗浅，停留在“怎么修改大概是可以跑的”的阶段，要实现更高效的代码还需要通过继续学习和实现来完全掌握它。

当然最后的结果还是存在一些不尽人意的地方，为了让 star 能在回调函数里改变大小，把 application 做的初始化工作移到了 RenderFrame 类中，而且把 stars 设置成了静态变量，感觉有点丑陋。不知道有没有更好的解决办法。

因为以纵向的单位长度为标准，所以当宽比较小时可能会出现五角星超过国旗范围的情况，~~但是这样的 corner case 处理起来没什么意思~~，就先没有去修改它：



另外可能是大量使用三角函数导致的精度问题，绘制出的小五角星边缘比较明显的不规则形状：



为了让显示效果更好，应该还有一些别的优化方法可以深挖。