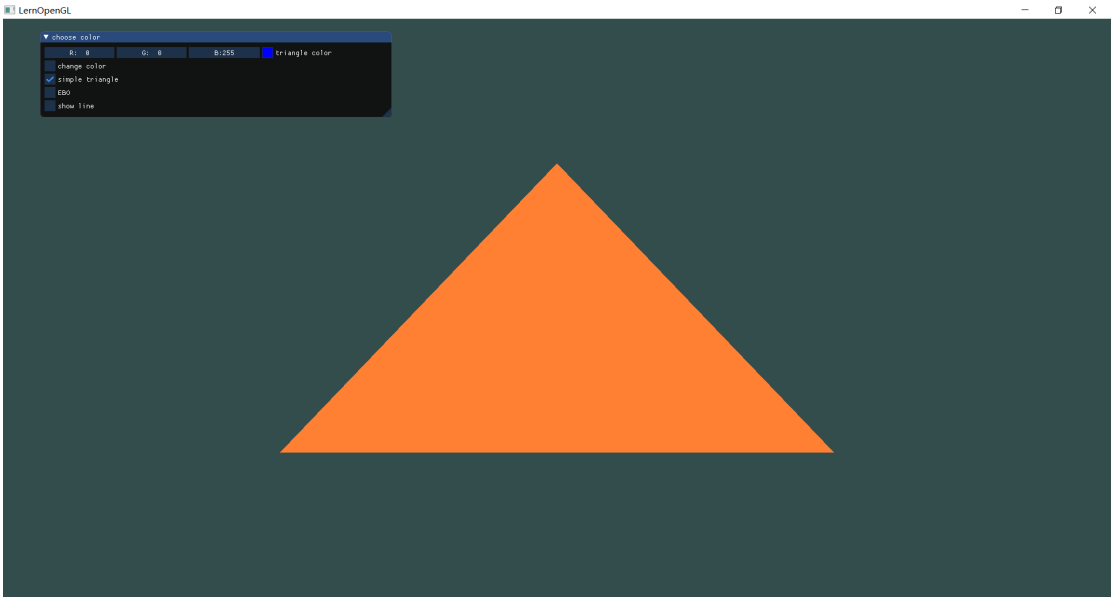
实验报告

1. 使用OpenGL(3.3及以上)+GLFW或freeglut画一个简单的三角形

   实验结果：



   实现思路：

   （1） 实例化窗口、创建窗口对象、加载GLAD、设置窗口维度

   （2） 顶点输入：将三角形的三个顶点坐标存到一个数组里面，并使用glGenBuffers函数创建一个VBO对象，将其绑定到GL_ARRAY_BUFFER目标上，调用glBufferData函数将顶点数据复制到缓冲内存中

   （3） 顶点着色器：编写顶点着色器，这个三角形颜色在片段着色器里设置了，因此顶点着色器只需要关心位置

   （4） 片段着色器：用于计算像素最后输出，这里设置了三角形的颜色，因此直接设置三角形颜色

   （5） 编译着色器：通过glCreateShader函数创建一个着色器对象，通过函数glShaderSource将着色器源码附加到着色器对象上，通过函数glCompileShader编译。检查编译是否成功：通过glGetShaderiv函数获得是否成功，通过glGetShaderInfoLog函数获得失败输出

   （6） 着色器程序：通过glCreateProgram函数创建程序对象，然后通过glAttachShader函数将前面编译好的着色器附加到程序上，使用glLinkProgram函数链接。最后通过glDeleteShader删除着色器对象

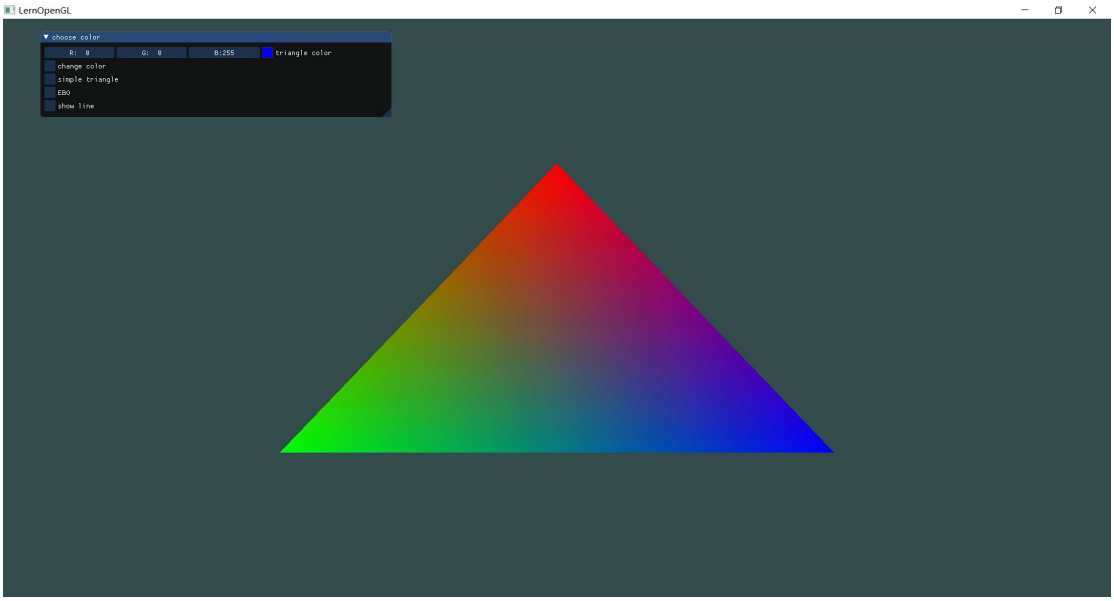
   （7） 顶点数组对象：通过glGenVertexArrays函数创建一个VAO对象，通过glBindVertexArray绑定，通过glBindBuffer将VBO存到缓冲中，通过glBufferData将数组数据复制到缓冲中，通过glVertexAttribPointer和glEnableVertexAttribArray函数设置顶点属性指针。

   （8） 绘制物体：通过glUseProgram函数激活程序对象，通过glBindVertexArray绑定VAO，再通过glDrawArrays函数绘制图形。

2. 对三角形的三个顶点分别改为红绿蓝，像下面这样。并解释为什么会出现这样的结

果。

实验结果：



实现思路：

（1） 实例化窗口、创建窗口对象、加载GLAD、设置窗口维度

（2） 顶点输入：将三角形的三个顶点坐标和颜色信息存到一个数组里面，并使用glGenBuffers函数创建一个VBO对象，将其绑定到GL_ARRAY_BUFFER目标上，调用glBufferData函数将顶点数据复制到缓冲内存中

（3） 顶点着色器：编写顶点着色器，设置位置的不变，将颜色变量属性位置值为1，并将颜色值作为输入，输出颜色

（4） 片段着色器：使用顶点着色器的输出变量作为传递进来的片段的颜色

（5） 编译着色器：通过glCreateShader函数创建一个着色器对象，通过函数glShaderSource将着色器源码附加到着色器对象上，通过函数glCompileShader编译。检查编译是否成功：通过glGetShaderiv函数获得是否成功，通过glGetShaderInfoLog函数获得失败输出

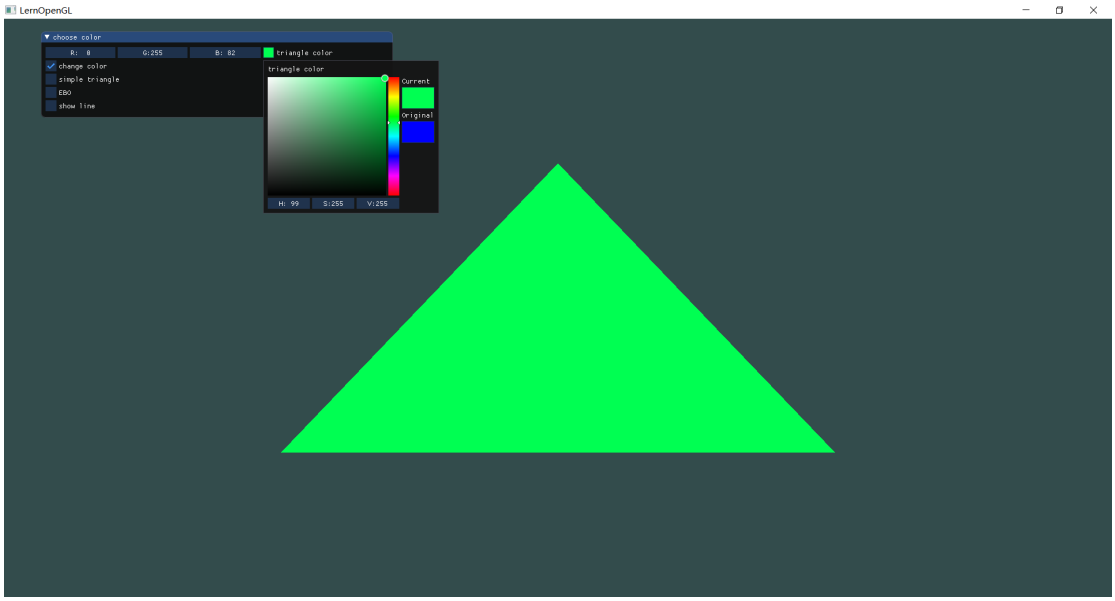
（6） 着色器程序：通过glCreateProgram函数创建程序对象，然后通过glAttachShader函数将前面编译好的着色器附加到程序上，使用glLinkProgram函数链接。最后通过glDeleteShader删除着色器对象

（7） 顶点数组对象：通过glGenVertexArrays函数创建一个VAO对象，通过glBindVertexArray绑定，通过glBindBuffer将VBO存到缓冲中，通过glBufferData将数组数据复制到缓冲中，通过glVertexAttribPointer和glEnableVertexAttribArray函数设置顶点属性指针，其中这里要设置位置属性和颜色属性。

（8） 绘制物体：通过glUseProgram函数激活程序对象，通过glBindVertexArray绑定VAO，再通过glDrawArrays函数绘制图形。

出现这种结果的原因是：片段着色器中进行的是片段插值，渲染三角形时，光栅化的阶段会造成比顶点更多的片段，而着色则通过在三角形相对位置决定片段位置，也就是插值的结果。而对于三原色插值，则会得到一定比例混合的颜色，也就是调色板的状态。

3. 给上述工作添加一个GUI，里面有一个菜单栏，使得可以选择并改变三角形的颜色。
实验结果：



实现思路：（这里基本函数与上一题类似，下面只列出不同的地方）

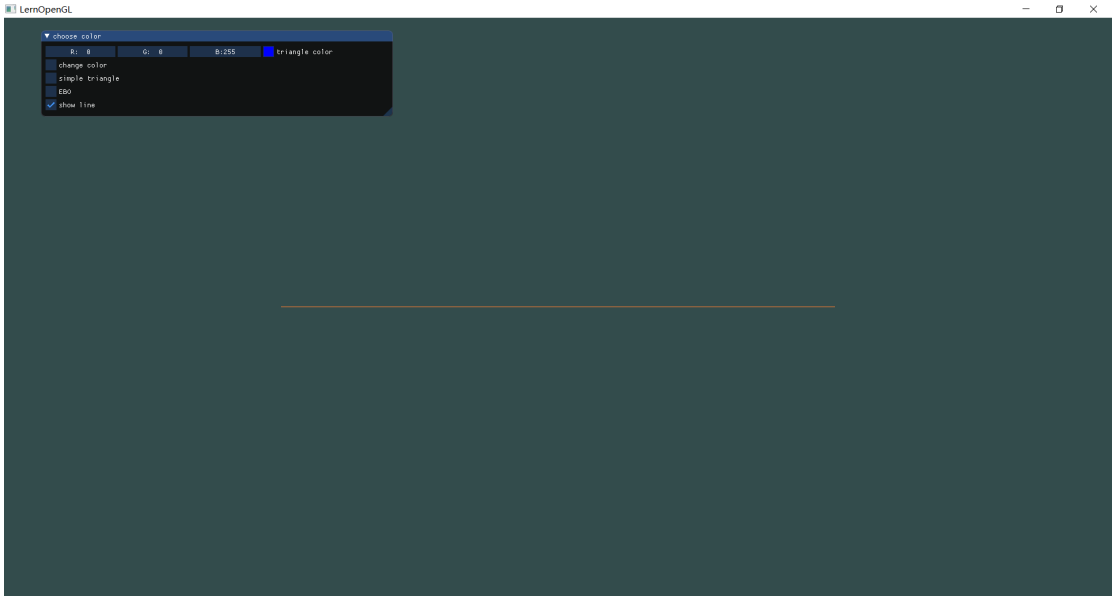
（1） 初始化ImGui

（2） 创建ImGui：调用ImGui_ImplOpenGL3_NewFrame函数、ImGui_ImplGlfw_NewFrame函数和NewFrame函数，通过Begin函数开始设置，传入GUI的标题，通过ColorEdit3函数添加颜色选择，传入参数分别是标题和选择后的结果存储地址，通过Checkbox函数添加选择，传入参数分别是标题和选择后的结果存储地址，End函数结束设置

（3） 当选择了设置颜色的选择框，则将三个顶点的颜色设置为选择的颜色

（4） 调用函数Render和ImGui_ImplOpenGL3_RenderDrawData渲染

4. 绘制其他的图元，除了三角形，还有点、线等。
实验结果：



实现思路：（这里基本函数与上一题和第一题类似，下面只列出不同的地方）

（1） 当选择了显示线的选框后，传入glBufferData函数的数据变成线的数据：只有两个顶点

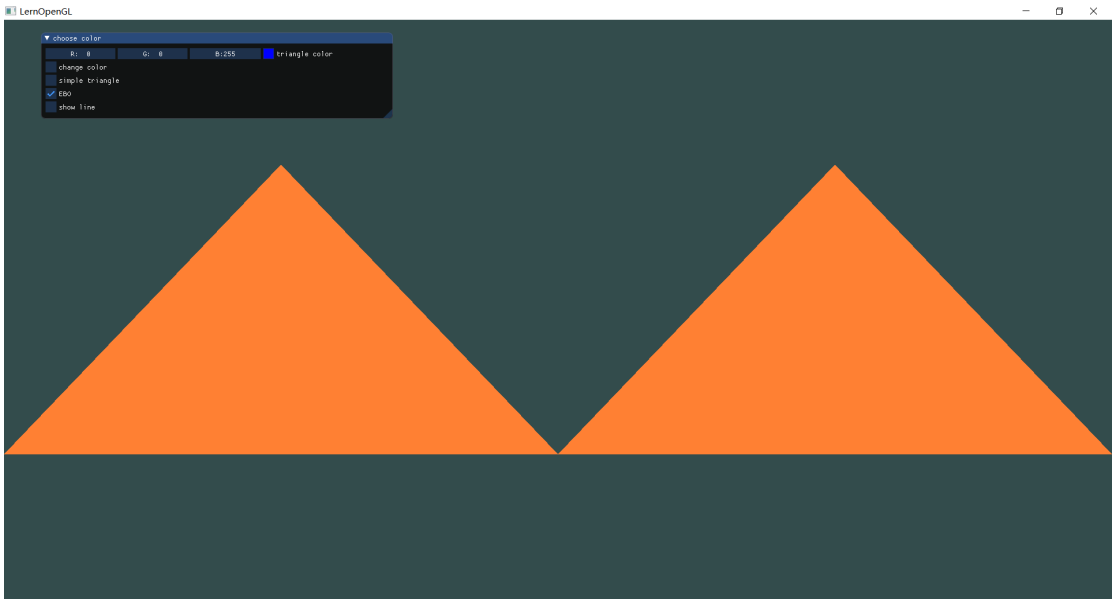
（2） 着色器代码使用的是第一题的代码

（3） 其余部分与第一题的内容相同

（4） 绘图部分，传入函数glDrawArrays第一个参数是GL_LINE_STRIP，用于画直线，第三个参数是2，由于只有两个顶点

5. 使用EBO(Element Buffer Object)绘制多个三角形。

实验结果：



实现思路：（这里基本函数与第一题类似，下面只列出不同的地方）

（1） 在选择了EBO选框后，三角形的传入数据变成两个数组，一个是五个点的位置（两个三角形重叠的点只需要用一个），另一个是两个三角形使用了哪些点的索引

（2） 在生成VBO对象并把五个点的数据传入缓冲后，初始化EBO对象，并将索引数组传入缓冲

（3） 最后绘图部分，通过glDrawElements函数绘制，传入参数分别是：GL_TRIANGLES、6（有多少个元素）、GL_UNSIGNED_INT、0