计算机图形学作业 1 报告

徐琢雄 521021910560

1.概述

绘制由多个三角形组成的六边形，实现快捷键三角形颜色渐变，快捷键实现三角形边的高亮显示，同时所有三角形顶点出现大小不一的圆点，每帧刷新亮点的大小和亮度，实现闪耀的视觉效果，以及设计视觉特效。

在 homework1 文件夹中使用 cmakelist 构建项目，构建 target homework1 得到可执行文件，即可运行。

程序执行过程中，按下快捷键 C 三角形颜色渐变开始，再次按下渐变停止，可以在渐变过程中随时停止；按下快捷键 S 三角形边高亮，显示圆点，边上光斑效果，再次按下效果消失。

2.实现方案

2.1.三角形显示

首先实现了一个坐标旋转函数，利用坐标旋转固定角度生成六边形的六个顶点，然后加上三个中间的顶点就得到了顶点数组。

我选择通过引用来绘制三角形，这样就不用重复编码顶点。先建立一个三角形索引数组，内容是所有三角形的顶点索引。

之后绑定 VAO，EBO，EBO，用 glDrawElement 函数，绘制所有三角形。

2.2.三角形颜色以及渐变

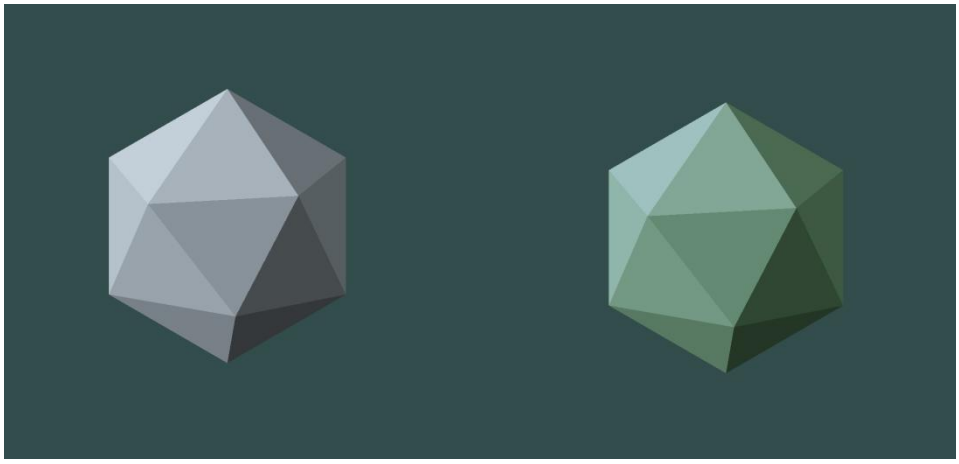
我使用了自定义的 shader，在 fragment shader 中，用 uniform 来定义 shader 着色的颜色，在每一次绘制三角形之前使用 glUniform 函数发送颜色。

在颜色的选择上，先生成了 rgb 值，然后对于每个三角形，把 rgb 颜色乘上一个 0-1 之间的系数，确保灰度渐变同时三角形都在同一色系。

渐变的实现上，先获取时间值，然后使用三角函数来改变 rgb 的数值，之后同上乘以系数达到灰度渐变。颜色变化公式如下：

$$y=A\cdot\sin(w\cdot timeValue+offfset_0)+offset_1$$

显示结果如下：



2.3. 边高亮和特效

边高亮一开始的实现方案是用线框模式重新绘制一遍整个图像，就实现了边高亮。

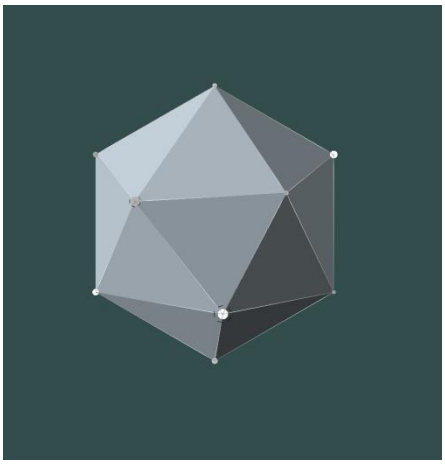
实现光斑能量传递的特效时，采用了另一种实现，就是绘制所有线段。为了便于控制和调整颜色，我用了第二个自定义的 shader，将颜色信息和坐标信息一起编码到顶点数组中，然后通过 glVertexAttribPointer 分别定义位置和颜色信息的偏移量，这样 shader 就能正确读取不同属性了。

之后我建立了所有边的索引数组，然后用时间作为偏移量，用偏移量作为权重，加权平均边的端点的坐标，就得到了边上一点的坐标，如下所示：

$$(x_p,y_p)=weight\cdot(x_1,y_1)+(1-weight)\cdot(x_2,y_2)$$

将端点和中间点设置为不同颜色，就显示了光斑效果。每一帧的绘制时，先使用第二个 shader program，然后用 time value 生成边和光斑的数据传给 shader，绘制边线。

而权重又是随时间变化的，所以光斑可以呈现周期性的能量传送效果。如下所示：



2.4. 圆点显示

因为 openGL 没有画圆的函数，故采用扇形模式画圆，我实现了一个函数用来生成一

个圆需要的顶点，输入顶点坐标和半径，使用之前的旋转函数，生成围绕顶点等距的 20 个点，均匀排列在顶点周围，连接这些点来画出圆形。再设置一个新的 buffer 存储圆点的数据，并且用 GL_DYNAMIC_DRAW，这是因为顶点大小和颜色会变化，数组要经常刷新。

在每一帧的绘制时，绑定新的 VAO，然后生成点，生成点的半径和颜色在原来的半径基础上乘一个随时间变化的系数，同样用三角函数和偏移量，实现点的大小和颜色随着时间周期性变化。这个函数的系数和点的索引有关，实现的效果是不同的点大小和颜色变化的周期都不同。最后用 GL_TRIANGLE_FAN 画扇形，连接成一个圆。