# 作业三

## 一．作业内容

Phong shading 与 VBO 绘制三维物体：

(1) 通过 fragment shader 实现 Phong shading：
   ① 在 vertex shader 中输出法向量；
   ② GLSL 会自动插值并输入 fragment shader；
   ③ 在 fragment shader 中通过 Phonq shading 计算三类反射。

(2) 使用 VBO 存储顶点与连接关系：
   ① 可通过细分物体(如小球)产生足够多的三角面片。

(3) 使用多个细分迭代次数讨论以下内容：
   ① 对比 Phong shading 与 OpenGL 自带的 smoothing shading 的区别；
   ② 使用 VBO 进行绘制及通过 glVertex 进行绘制的区别；
   ③ 讨论 VBO 中是否使用 index array 的效率区别。
   ④ 对比、讨论 HW3 和 HW2 的渲染结果、效率的差别。

## 二．思路参考

1. Vertex shader 用于将齐次坐标顶点传输到屏幕坐标顶点，同时产生定点相关的属性，用法示例如下：

```
out vec3 normal;
out vec3 vertex_to_light_vector;

void main()
{
   // Transforming The Vertex
   gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;

   // Transforming The Normal To ModelView-Space
   normal = gl_NormalMatrix * gl_Normal;

   // Transforming The Vertex Position To ModelView-Space
   vec4 vertex_in_modelview_space = gl_ModelViewMatrx * gl_Vertex;

   // Calculating The Vector From The Vertex Position To The Light Position
   vertex_to_light_vector = vec3(gl_LightSource[0].position – vertex_in_modelview_space);
}
```

2. Fragment shader 用于计算片元颜色，计算中可能使用顶点属性插值后得到的片元属性：

```
in vec3 normal;
in vec3 vertex_to_light_vector;
uniform vec4 ambient_color;

void main()
{
   // Calculating The Final Color
   gl_FragColor = ambient_color;
}
```

3. 三类反射实现：首先实现环境光，通常是恒定值；接着是漫反射，取决于光源方向和法线方向的相对角度；镜面高光是由视角和光反射角度决定的；最后将全部组合起来得到最终颜色。

4. VBO 存储顶点与连接关系：

(1) 创建初始形状。

(2) 将每个三角面片分割成更小的三角形。这可以通过在每条边的中点添加新的顶点，并将原始的大三角形分割成更小的三角形来实现。这个过程可以重复多次以获得更高的细节级别。

(3) 将顶点数据（如位置、法线等）存储到 VBO 中。在 OpenGL 中，可先创建缓冲区，再绑定 VBO，最后用 glBufferData 或类似函数上传数据。

5. 使用 VBO 将需要绘制内容放在创建的 buffer 中的示例：

```
GLuint vbold; // ID of VBO
GLfloat* vertices = new GLfloat[vCount*3]; // create vertex array

// generate a new VBO and get the associated ID
glGenBuffers(1, &vbold);
// bind VBO in order to use
glBindBuffer(GL_ARRAY_BUFFER, vbold);
// upload data to VBO
glBufferData(GL_ARRAY_BUFFER, dataSize, vertices, GL_STATIC_DRAW);
// it is safe to delete after copying data to VBO
delete [] vertices;
...
// delete VBO when program terminated
glDeleteBuffers(1, &vbold);
```

6. 对比：可从概念、实现方式、特点、视觉效果、性能开销、适用场景等方面对比讨论。

# 三．模板

与 HW2 模板相同，可在已完成的 HW2 基础上完成 HW3，也可在 HW2 模板上新建项目完成 HW3。

1： CGTemplate.pro: Qt 项目文件，可以使用 Qt 项目文件生成 vs 项目文件（windows）、makefile（linux）、或 xcode 项目文件（macos）等。

2：main.cpp: 主函数，不需要修改。

3：myglwidget.cpp:在此处完成你的代码实现，请注意编程规范，每个函数都需要写清楚注释，包括函数的作用和大致的步骤，实验报告内需要说明具体的实现思路并且需要有实验结果图。

4：myglwidget.h: 对应 cpp 的头文件，注意更新函数声明。

5：utils.h: 工具头文件，不用修改 (如有修改，在报告中说明一下) 。

6：utils.cpp: 工具 cpp 文件，不用修改 (如有修改，在报告中说明一下) 。

7：glm 文件夹：向量库，用于方便的向量定义、计算。

# 四．作业提交方式

1. 提交作业内容：

(1) 项目完整文件：
- 放在一个文件夹中；
- 若有引用代码，要求分别明确标出引用部分和自己实现部分的代码和功能，在代码注释中添加说明；

(2) 演示视频 ：
- 要求展示完整的功能；
- 禁止用程序功能外的内容拉长视频时长；
- 禁止重复多次展示同一功能；

(3) 实验报告：
- 要求写明功能实现思路和过程；
- 若有必要，可适量、简洁的粘贴代码并讨论说明，与讨论对比无关的代码请不要放在报告中；
- 要求提交的实验报告格式为 PDF。

2. 文件组织方式: 将上述提交作业内容中的三个文件压缩成一个压缩包，使用 ZIP 格式。

3. 文件命名方式：

(1) 上述提交作业内容中的三个文件分别依次命名为：
- 项目完整文件的文件夹：CGHW3；
- 演示视频：video-HW3.{任意格式}；
- 实验报告：report-HW3.pdf。

(2) 上述文件组织方式中的 ZIP 压缩包的命名方式：学号-姓名-作业 n，示例：22111111-李四-HW3。

4. 提交作业网址：

请进入 https://www.scholat.com/course/cg23au 后，扫码加入课程并根据站内指引上传作业。

# 四．注意事项

1. 允许讨论代码，但严禁任何形式的抄袭。
2. 未提交迟交申请，且迟交作业的同学，本次作业不得分。
3. 如遇到特殊情况导致未能按时提交作业的，可在作业截止日期前向 TA 提出 slip days 申请，并在 slip days 结束前补交作业。
4. 如遇到特殊情况导致未能按时提交作业且已用完 slip days 的，或者有不可抗力导致需要延迟超过 slip days 额度的，可提交申请说明理由，特事特办。