Homework 4 作业报告

16337341 朱志儒

一、 程序说明

本次作业的两个题目均使用立即渲染模式编写的，题目 2 使用了轻量级第三方库 OpenMesh。

在编写题目 1 的过程与前几个作业相似，但对每次鼠标点击操作的相应有所不同，每当有鼠标点击时，将鼠标点击的坐标存入名为 points 的数组中以供绘制图像使用。

编写初始化函数 Init()时，和之前几个作业相似，首先清除颜色缓冲区，设置点的大小，接着设置为对模型视景的操作，然后将矩阵转换为单位矩阵，最后调用二维剪裁函数设置整个坐标系。

编写 drawLine()函数，用于绘制线段。首先设置线段的颜色和大小，接着根据鼠标输入的点的坐标绘制相应的线段。

编写 drawCurve()函数，用于绘制三次 Bezier 曲线，首先设置曲线的颜色和大小，接着使用公式

$$\mathbf{B}(t) = \mathbf{P}_0(1-t)^3 + 3\mathbf{P}_1 t(1-t)^2 + 3\mathbf{P}_2 t^2(1-t) + \mathbf{P}_3 t^3 \ , \ t \in [0,1]$$

（P_0、P_1、P_2、P_3 分别代表四个点）

求出曲线中点(x，y)的坐标值，然后将这些点绘制出来即可得到 Bezier 曲线。

编写 Display()函数，用于绘制整个图像。首先清除颜色缓冲区，当存储的点的数目少于 4 时，只绘制点，当存储的点的数目等于 4 时，绘制线段和曲线。

编写 Mouse()函数，用于响应用户的鼠标操作。每当有鼠标点击操作时，先判断当前 points 数组的长度，若小于 4，则将坐标点存入 points 数组中，若等于

4，则清除 points 数组，再将坐标点存入 points 数组中。

最后编写主函数，先初始化 GLUT 库，设置 RGB 像素格式窗口，接着设置窗口的位置、大小和名称，然后调用初始化函数 Init()，接着设置绘制函数 Display()、鼠标响应函数的回调 Mouse()，最后让绘制线程循环。

编写题目 2 十分的困难，编写的过程也十分的艰辛，这个题目花了我大量的时间，不过收获颇丰。

在编写题目 2 时，首先声明多个全局变量，方便多个函数共同使用。声明鼠标当前状态 mouse_state 为 0，点击之前物体的位置(ox, oy)为(0, 0)，旋转角度相关的两个参数 xr 和 yr，与放缩矩阵相关的参数 scale，声明读取的文件名 file1、file2、file3，当前读取的文件号 current_file，声明两个显示列表 show_face_list、show_wire_list，显示状态 show_face、show_wire、show_flat_lines，用于设置显示模式。

编写初始化函数 initial_GL()，和题目 1 相似，但是需要开启深度缓冲区、启用光照功能和设置第一个光源，再申请连续索引值赋给显示列表，接着定义第一个显示列表 show_wire_list，再定义第二个显示列表 show_face_list。在定义 show_wire_list 时，先关闭光照功能，再设置线的大小和颜色，然后根据读取的多边形网格数据以 Wireframe 模式绘制，最后开启光照功能。在定义 show_face_list 时，根据读取的多边形网格数据以 Flat 模式绘制。

编写读取文件函数 read_file(string file)，首先请求顶点法线，如果存在则读取文件，如果不存在顶点法线，就通过面法线计算顶点法线，最后释放面法线。

编写 reshape(Glint w, Glint h)函数，用于设置显示窗口的属性，把用 glOrtho 创建的正交平行视景体截取图像显示到窗口。

编写显示函数 display()，首先清除之前的颜色缓存和深度缓存，将矩阵置为单位矩阵，再设置旋转矩阵、位移矩阵和缩放矩阵，最后根据显示状态 show_face、show_wire、show_flat_lines 来执行相应的显示列表以实现 Flat、Wireframe 和 Flat lines 三个显示模式。

编写函数 special(int key, int x, int y)，用于响应用户的按键操作。当用户按下 F1 时，显示 cow.obj 文件；按下 F2 时，显示 cactus.ply 文件；按下 F3 时，显示 Armadillo.off 文件。当用户按下 F4 时，显示模式在 Flat、Wireframe 和 Flat lines 中切换。

编写 mouse_move()函数和 mouse()函数，用于响应用户的鼠标操作。编写 mouse_move()函数时，根据鼠标位置的变化情况修改物体旋转的角度。编写 mouse()函数时，首先判断鼠标的操作，如果是点击操作，则修改鼠标的位置坐标。如果是滑动滚轮，则修改放缩矩阵参数 scale。

最后编写主函数 main()，与题目 1 相似，但增加 glutMotionFunc()和 glutSpecialFunc()用于响应用户的鼠标移动和按键操作。

二、 运行方法

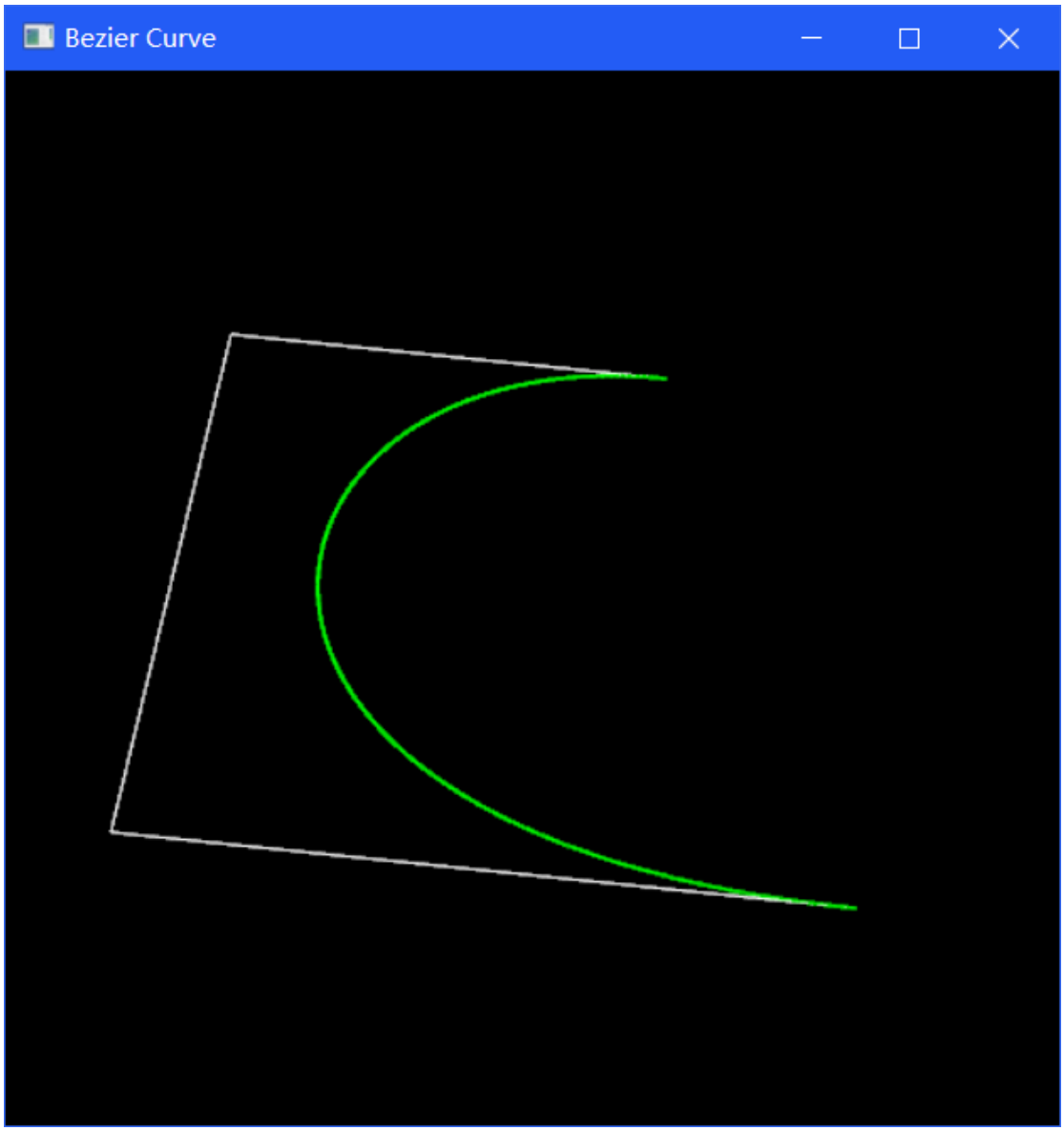
在“用 OpenGL 实现交互式三次 Bezier 曲线的构建”文件夹中，有 Bezier Curve.exe 和 freeglut.dll 两个文件。运行 Bezier Curve.exe 文件，然后在窗口中随机点击 4 下，就会在窗口中显示三条线段和一条三次 Bezier 曲线。再点击 4 下也是同样的效果。

在“用 OpenGL 实现简单的多边形网格数据读取和操作”文件夹中，有 Reading and Manipulation.exe 可执行文件、freeglut.dll 动态链接库文件和 Armadillo.off、cactus.ply、cow.obj 三个多边形网格文件。运行 Reading and Manipulation.exe 文

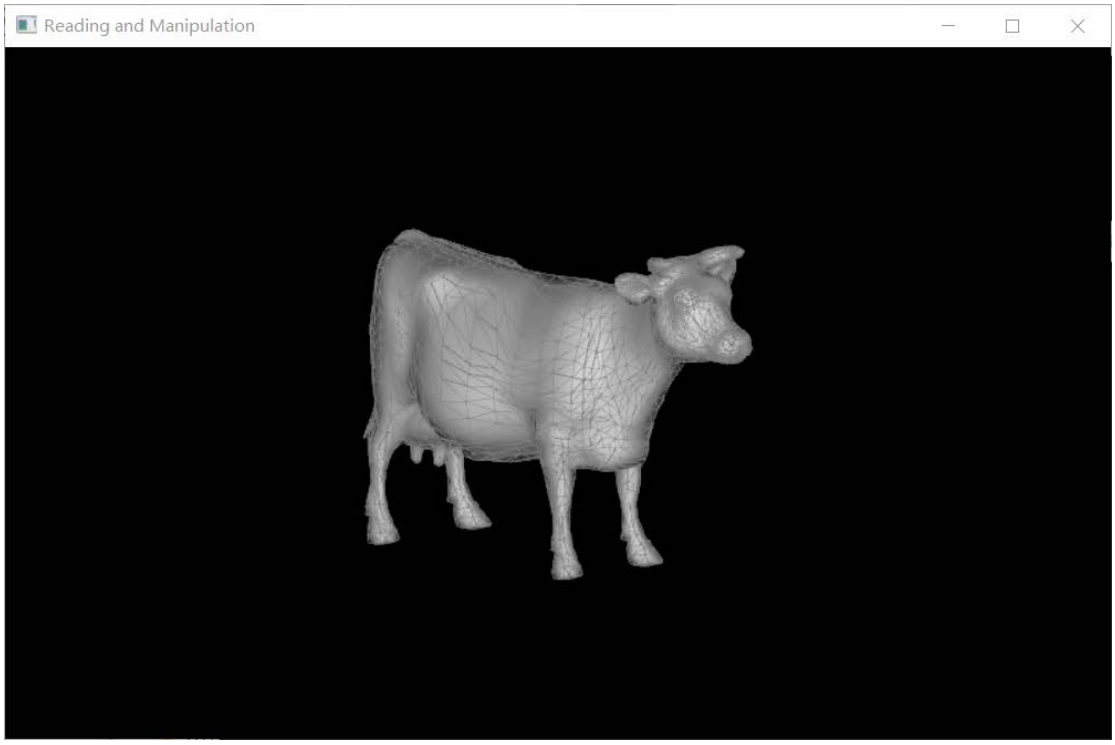
件，读取并显示 cow.obj 文件，此时可以使用鼠标拖动物体使其旋转，也可操作滚轮来放大或缩小视角，按下 F4 即可切换显示模式。再按下 F2 则读取并显示 cactus.ply 文件，按下 F3 则读取并显示 Armadillo.off 文件，在显示这两个文件时均可进行上述操作修改显示效果。

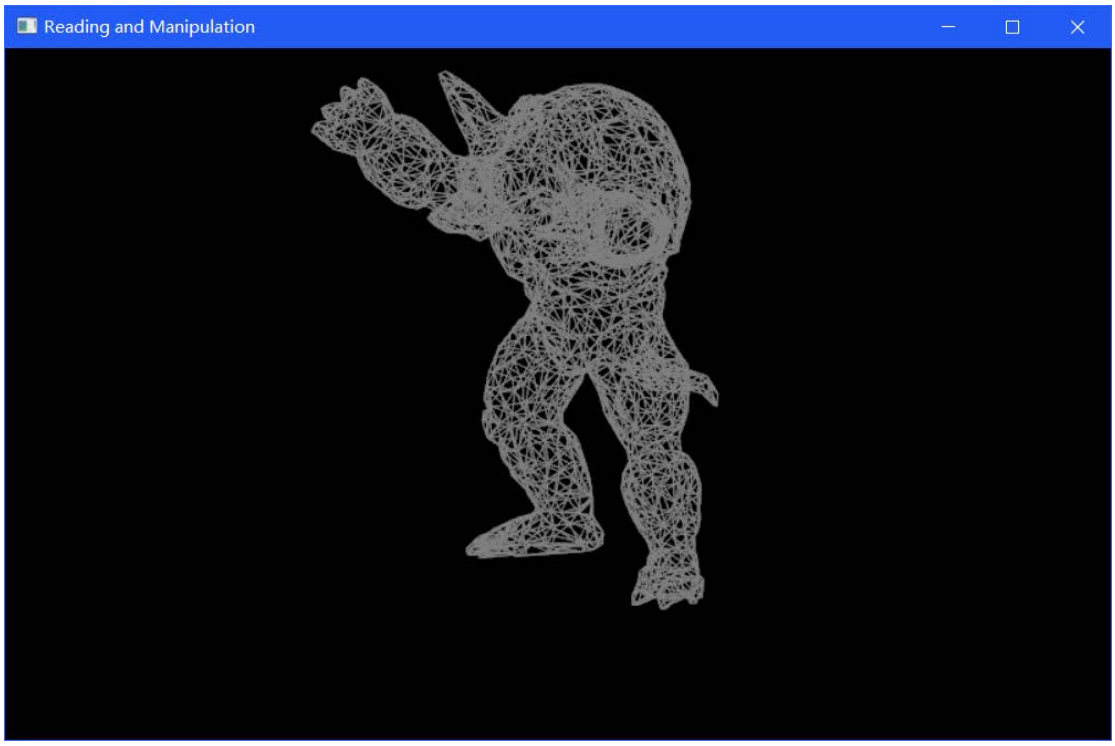
三、 程序运行结果

运行 Bezier Curve.exe 文件效果如图所示。



运行 Reading and Manipulation.exe 文件效果如图所示。




Reading and Manipulation