**电子科技大学**

**计算机科学与工程学院**

**标 准 实 验 报 告**

**（实验）课程名称 计算机图形学**

**电子科技大学教务处制表**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：宋邦睿 学 号：2023110903001 指导教师：肖逸飞**

**实验地点：主楼A2 413-2 实验时间：7-10周周六**

**一、实验项目名称：计算机图形学**

**二、实验学时： 10学时**

**三、实验原理：**

**1、贝塞尔曲线**

**a):Bezier绘制：**根据贝塞尔函数的定义绘制控制点

**用户操作：**通过glutMouseFunc(mouseButton);

glutMotionFunc(mouseMotion);glutKeyboardFunc(keyboard);三个函数监听用户输入。通过unProject函数接口从屏幕坐标系逆映射回世界坐标系。实现了移动控制点、自定义控制点位置等操作。

**b):多Bezier绘制**：通过GLWord类管理所有控制点，实现了绘制多条Bezier曲线。

**c):自动化移动控制点：**自定义AutoMove（）函数，配合OpenGL的库函数glutTimerFunc（）实现程序重复自动调用AutoMove（）随机改变选中控制点的位置，实现控制点自动移动。为使多控制点自动移动，设置了控制点储存容器储存选中控制点，同时被AutoMove（）调用

**2、分形几何**

**A）：通过迭代实现分形**

**B）：设置视口个数：通过glViewport函数建立四个视口，每个视口独立迭代**

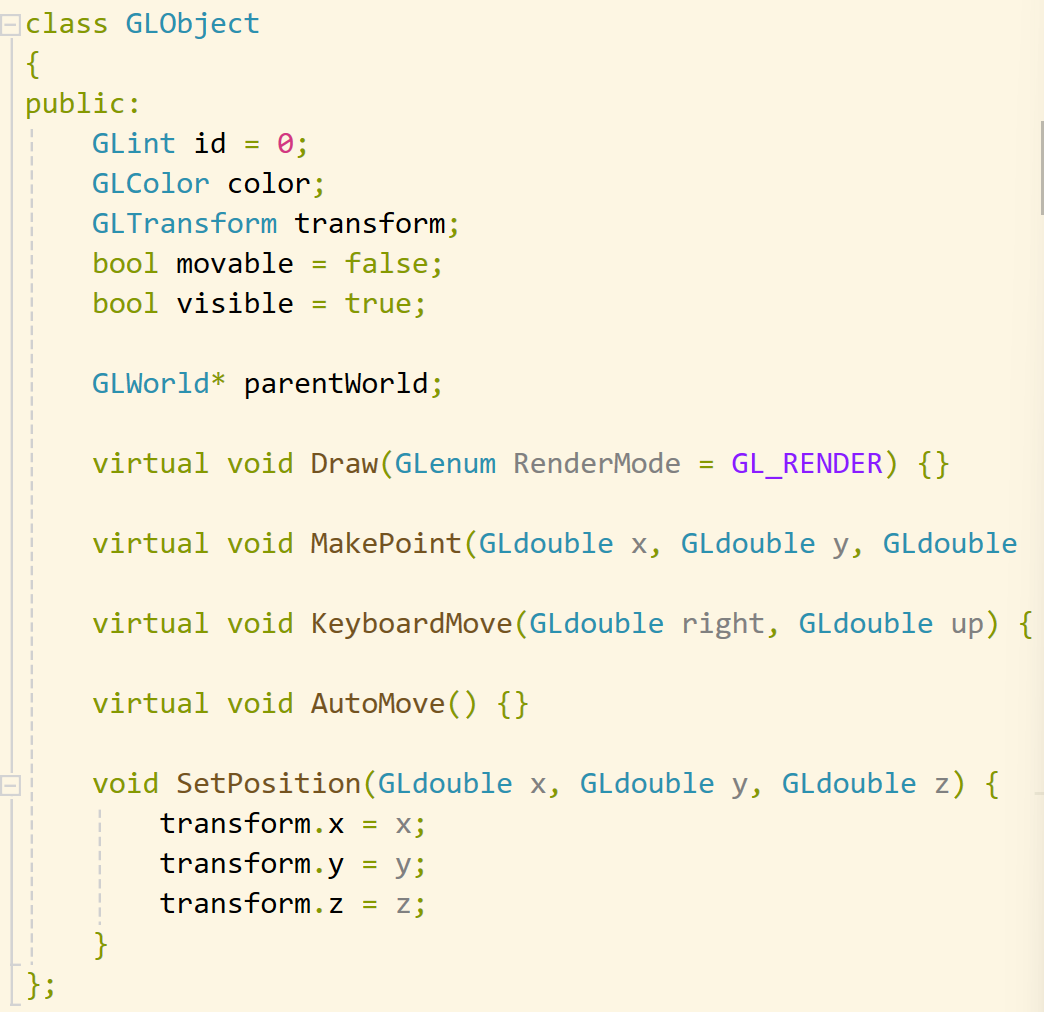
**C）：自动迭代：**glutTimerFunc（）实现自动迭代

**四、实验内容及步骤：**

1、Bezier曲线

a):设置基类GLObject和GLWorld管理所有控制点的生成、删除、绘制。绘制时调用派生类自己的绘制函数。体现多态特性。

GLObject类如下



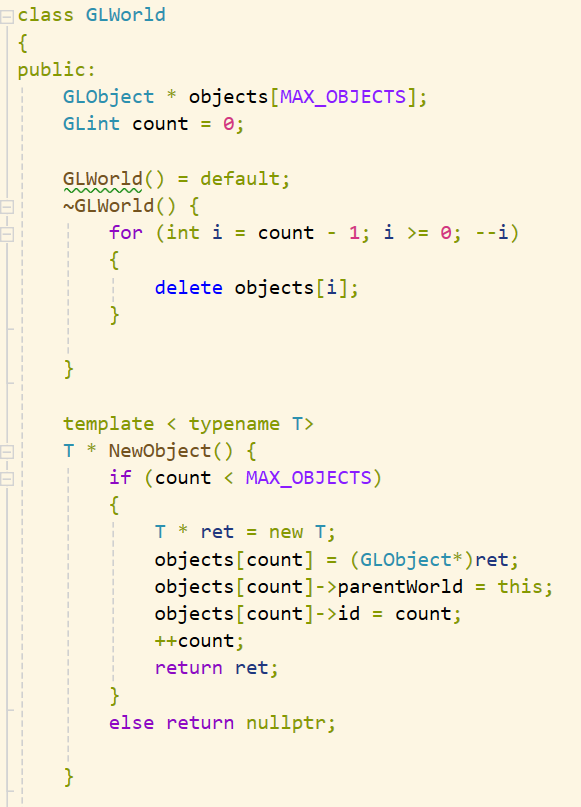
**Draw**由子类重写，用于绘制

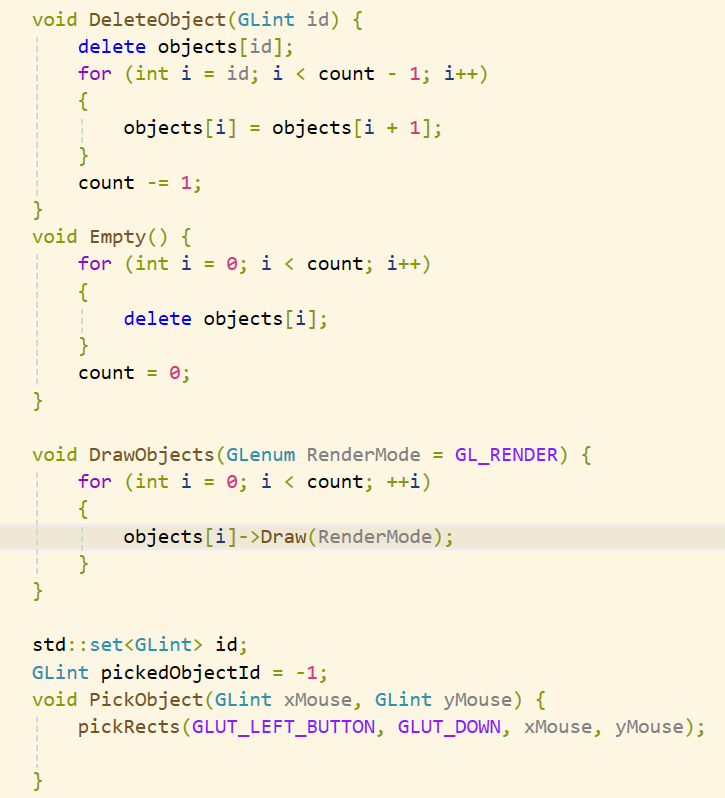
**MakePoint**由子类重写，内部调用GLWorld的NewObject方法创建对象

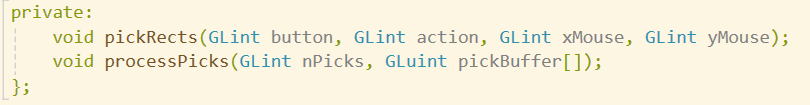
**AutoMove**由子类重写，负责程序自动完成控制点变化

**SetPosition用于设置物体位置**

**GLWorld类如下**





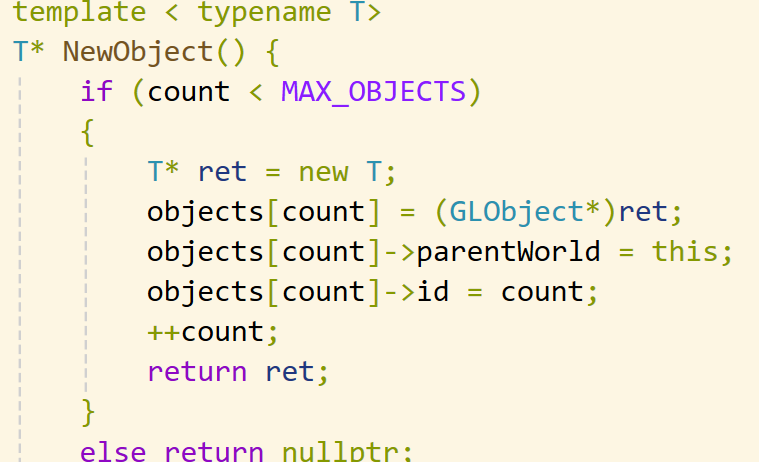


**DeleteObject和Empty**用于清空指定对象和重置所有对象。

**DrawObject**调用派生类**Draw**方法进行绘制

B）：对象的实例化

对象的实例化依靠GLWorld的NewObjec方法，根据传入对象的指针类型返回实例化对象指针，并记录在列表object[]中统一管理。



C）：对象拾取功能

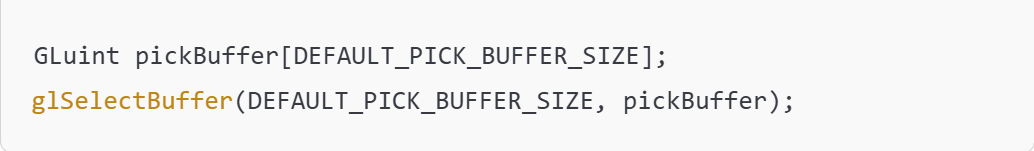
**pickRects**实现对象拾取功能

button 和 action: 指示鼠标按键和动作（例如左键按下）。

xMouse, yMouse: 鼠标点击的位置（屏幕坐标）。

定义选择缓冲区 pickBuffer，用来存储拾取信息（如对象名称和深度信息）。

glSelectBuffer 告诉 OpenGL 使用指定的缓冲区记录拾取数据。



glInitNames 清空命名栈（名字栈）。

glPushName(MAX\_OBJECTS) 向命名栈推入一个值，作为初始栈顶值。

这个栈用于标识绘制的对象，后续可以使用 glLoadName(id) 替换栈顶元素，标识具体的对象。



切换到投影矩阵，保存当前矩阵状态。

设置拾取区域：使用 gluPickMatrix 创建一个以鼠标点击为中心的小矩形区域作为拾取区域，大小由 DEFAULT\_PICK\_WINDOW\_SIZE 决定。

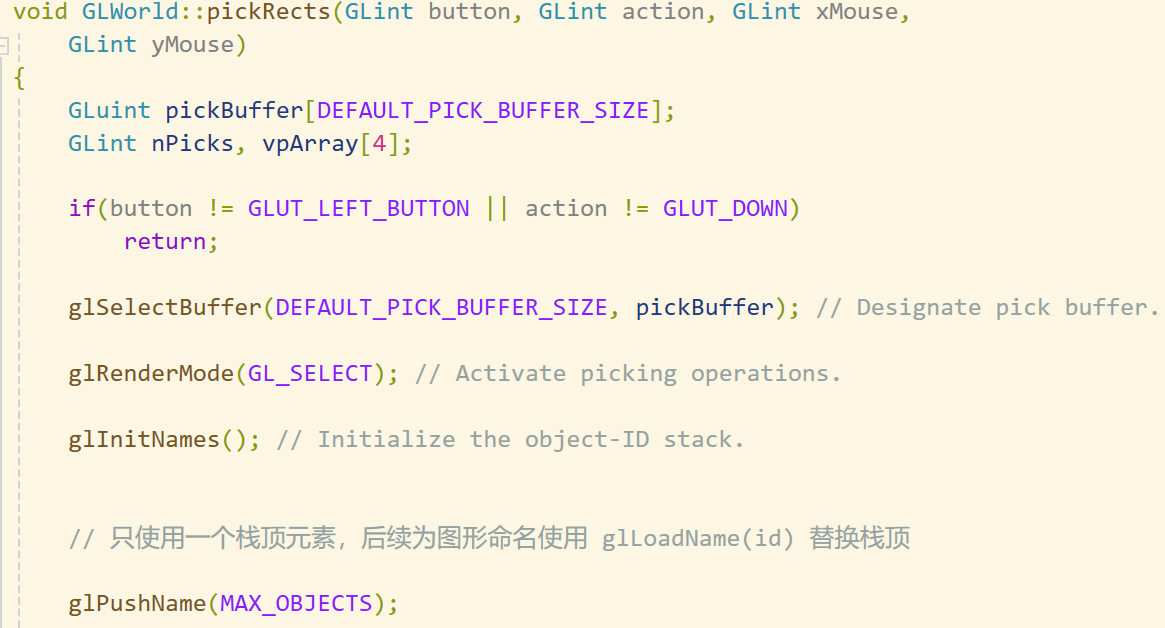
调用 gluOrtho2D 设置拾取区域的正交投影，以匹配应用的世界坐标范围。

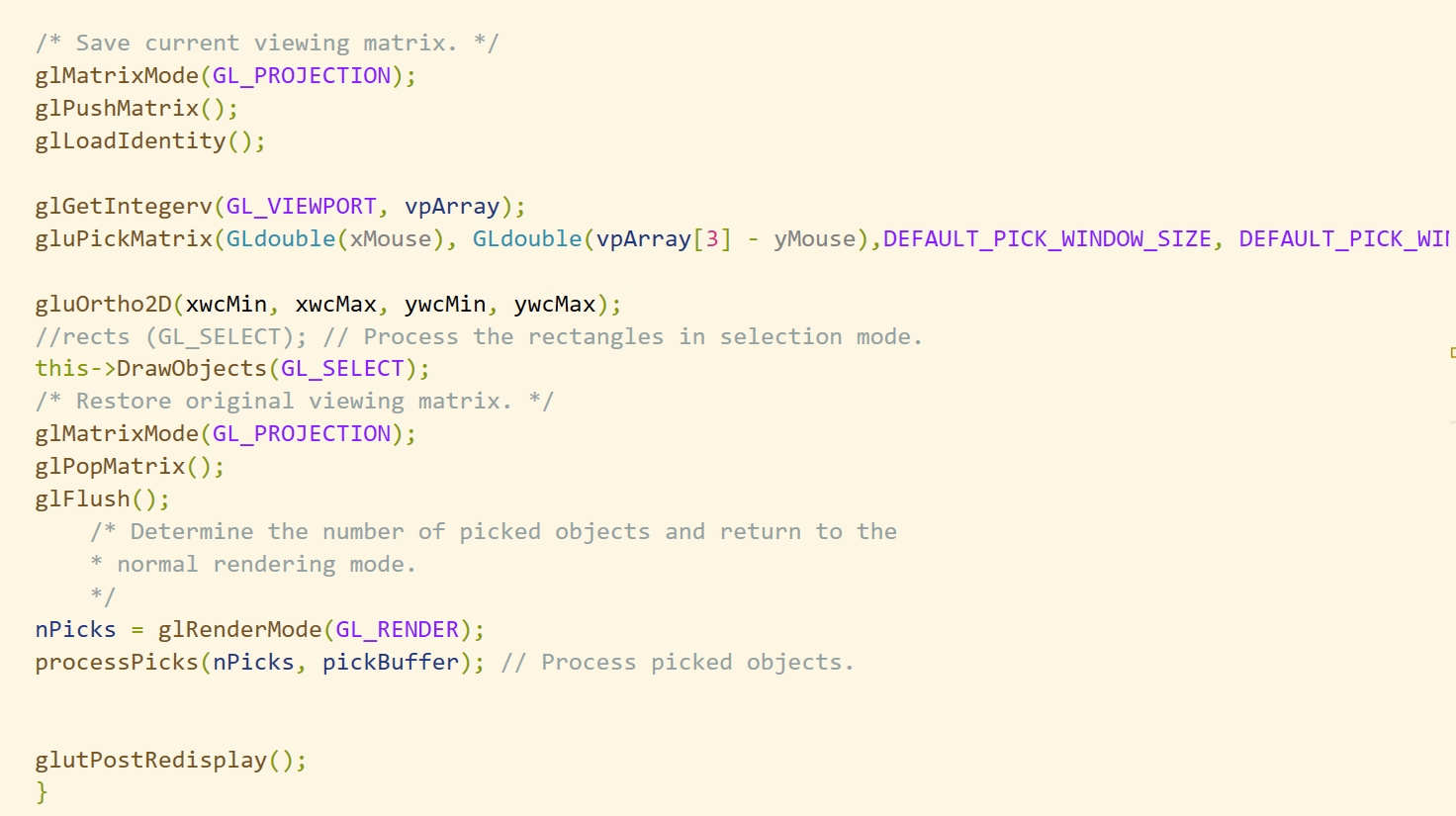
调用 glRenderMode(GL\_RENDER) 结束选择模式，返回拾取到的对象数 nPicks。

缓冲区中存储了拾取到的对象信息（如名称、深度等）。

调用 processPicks 函数对拾取到的对象进行处理。

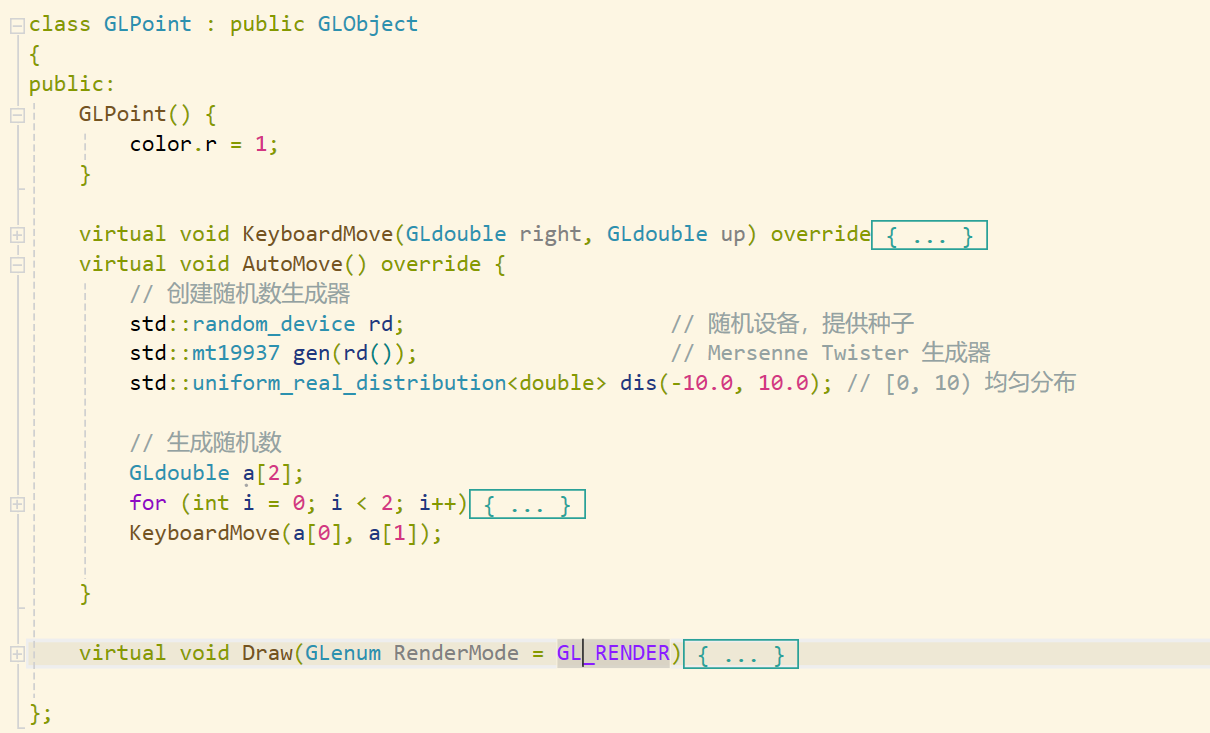
对**processPicks**的进一步解释另作分析。





D）：GLPoint类实现

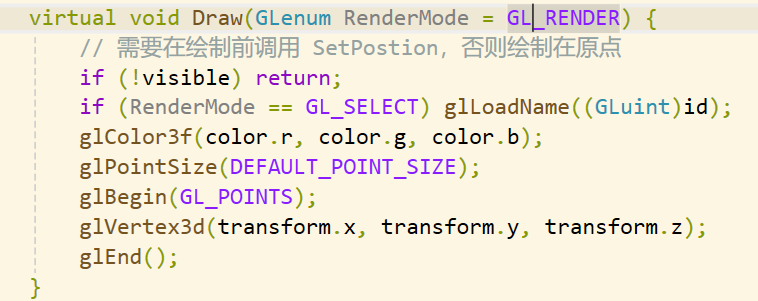
GLPoint类作为Bezier的基类，需要实现对应Draw方法、基础的位置计算、变换方法：



**KeyboardMove**用于响应用户操作，具体实现见下文

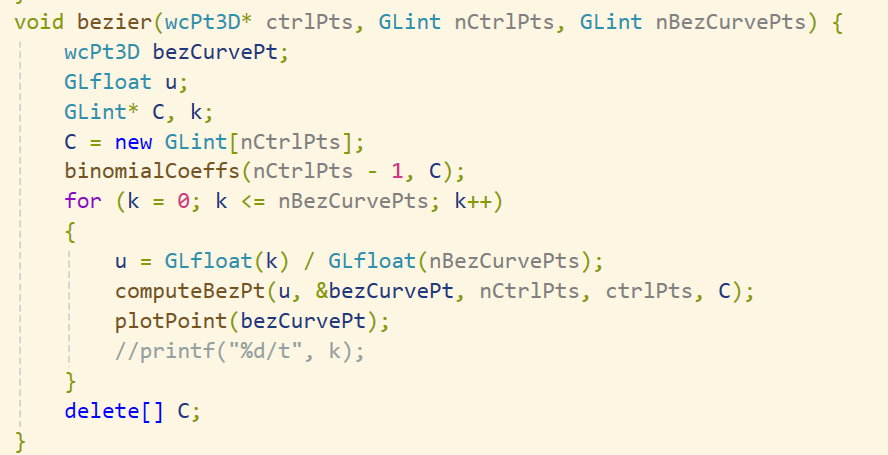
**AutoMove**用于执行程序自动控制控制点移动

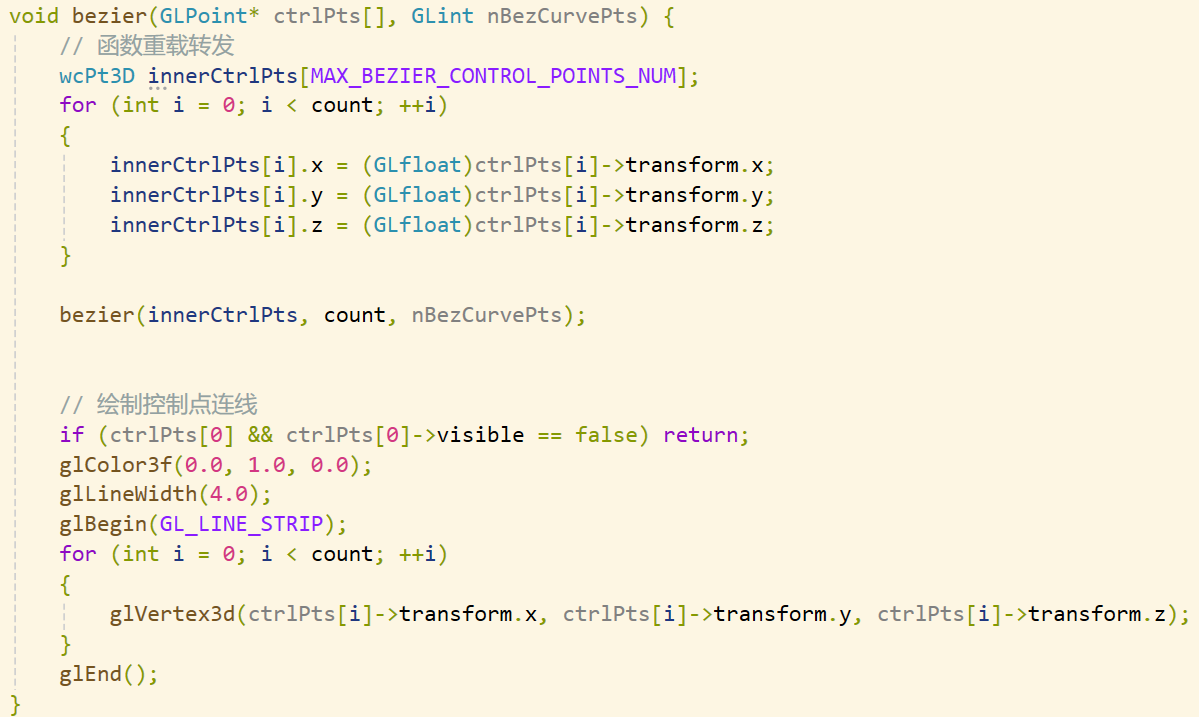
**Draw**作为绘制方法，实现为：



E）：Bezier曲线计算

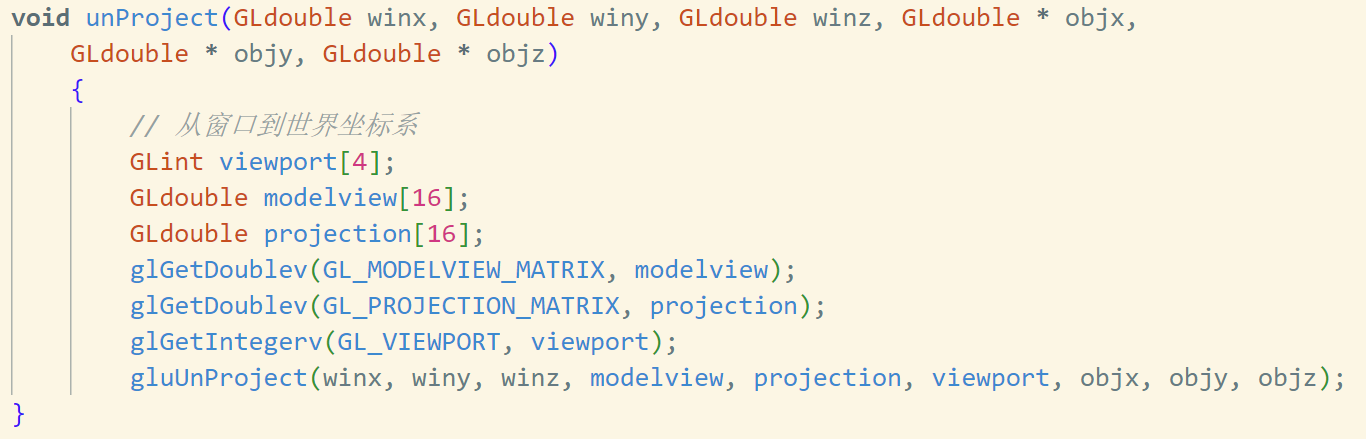
由书中给出的计算公式，实现相应计算部分。



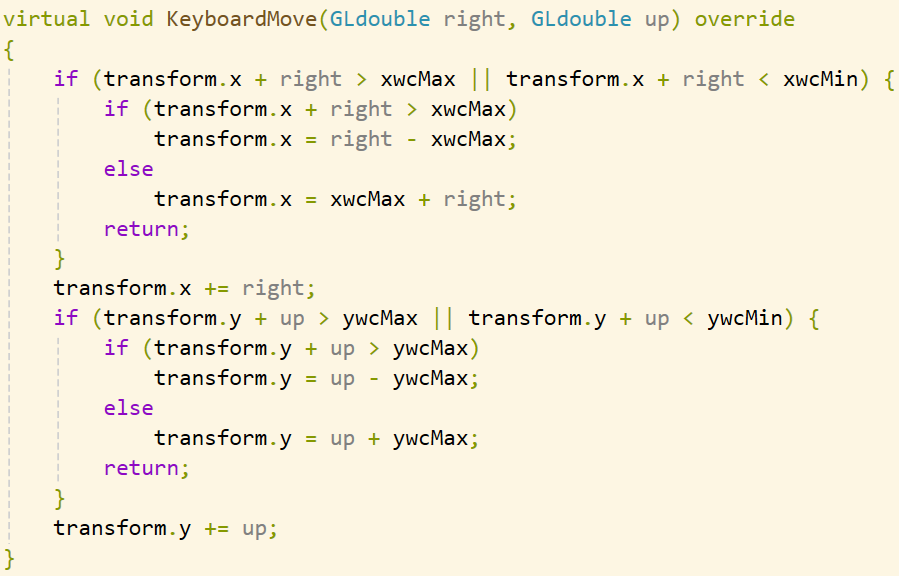


D）：响应用户操作

unProject函数响应用户鼠标点击位置逆映射计算世界坐标系位置，实现自定义控制点生成。



添加了KeyboardMove方法，便于用户用方向键控制控制点移动。同时给AutoMove函数提供接口，实现程序自动控制控制点移动。KeyboardMove函数增加了越界判断，防止控制点离开屏幕坐标系。



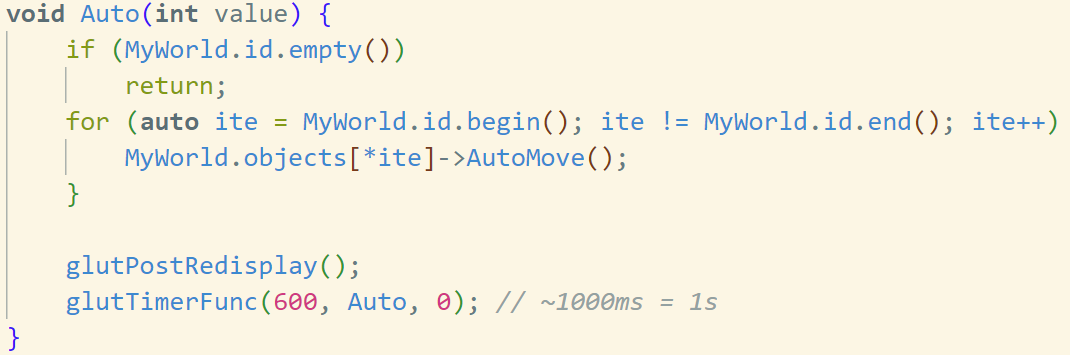
E）：程序自动控制控制点位置的实现

增加AutoMove方法配合OpenGl库函数glutTimerFunc（600, Auto, 0）每隔1s自动调用实例对象的AutoMove方法实现控制点自动移动。其中Auto函数设置在主函数中，调用GLWorld中选中的控制点，实现多控制点的同时自动化控制。

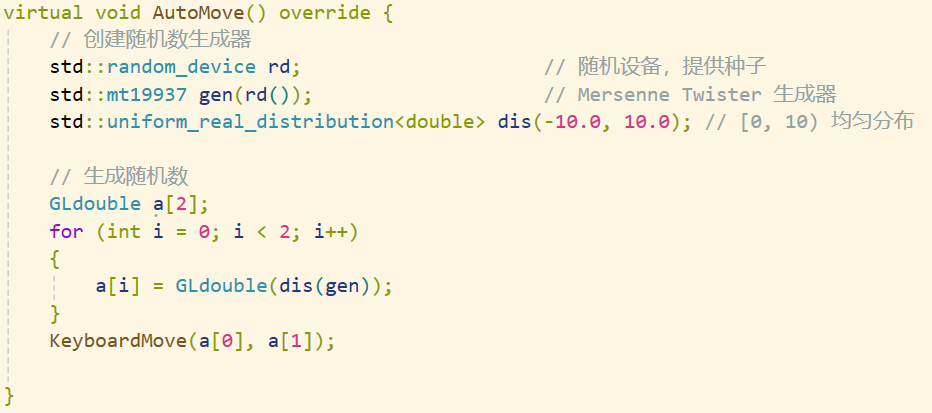
为了更好的控制多个控制点，定义集合容器set存储选中的控制点。processPicks函数中将鼠标选中的控制点添加入set容器id中，id将管理所有选中的控制点指针，便于自动控制。



Auto函数的实现：



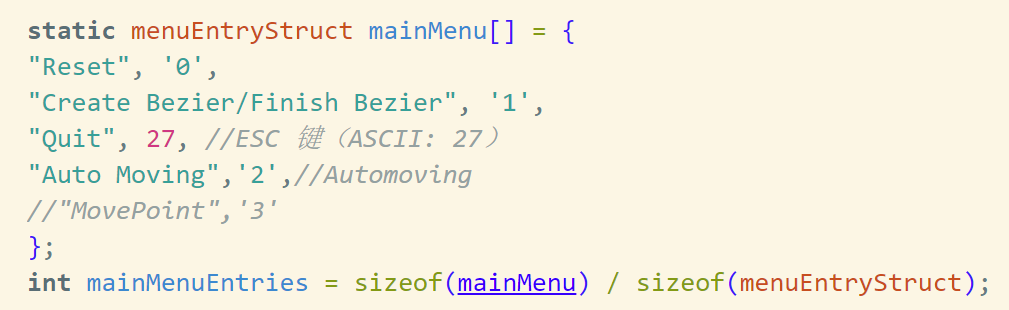
AutoMove函数的实现：



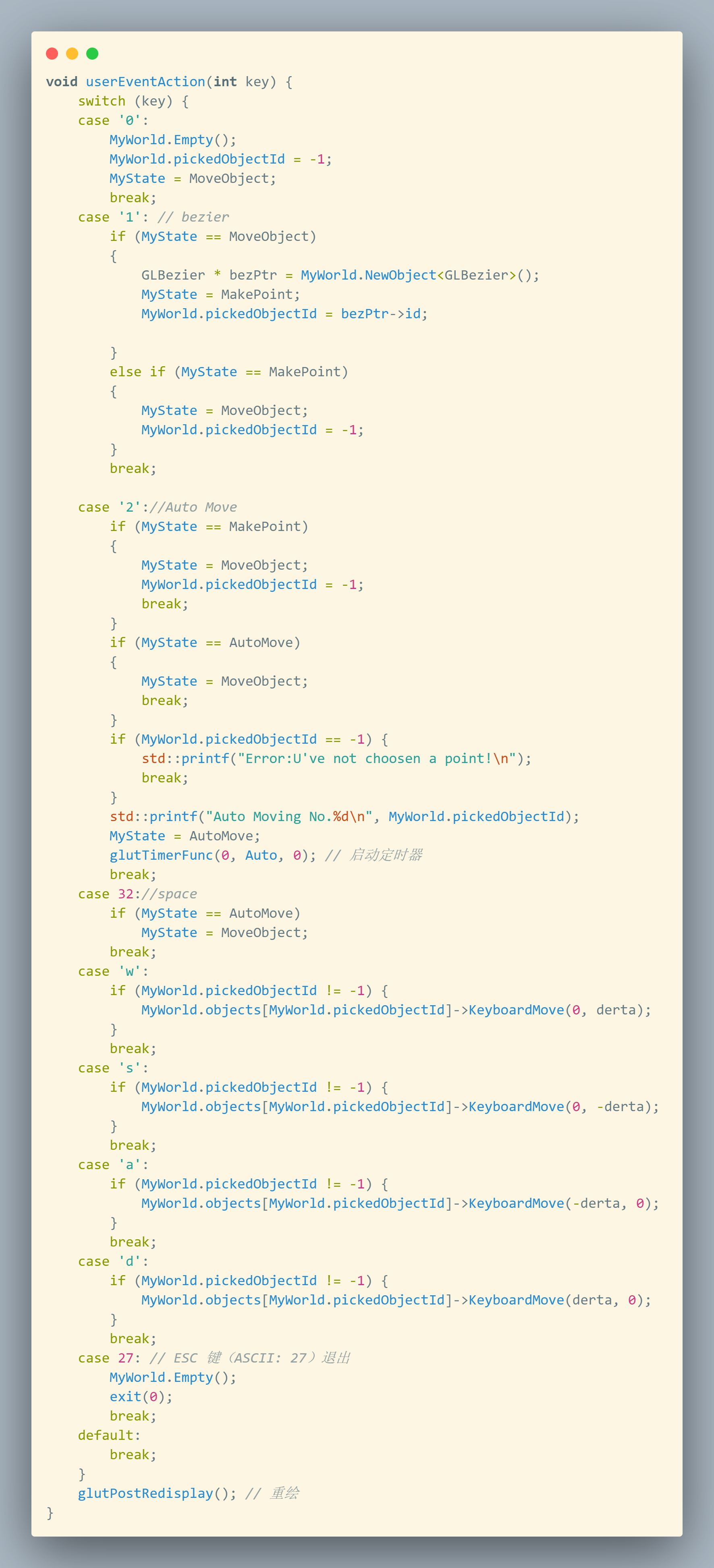
综上，实现生成贝塞尔曲线，测试如下：

F）主函数部分

**设置菜单绑定鼠标滚轮，菜单映射init部分**



功能执行部分完整代码如下：



其中wasd用于键盘移动控制点，实现逻辑为调用对象的KeyboardMove（）方法进行坐标运算

执行绘制贝塞尔曲线操作时，需要先判断当前状态是否为AutoMove、MoveObject、MakePoint，若为MoveObject，则开始绘制。若为MakePoint则结束绘制，释放MyWorld.pickedObjectId指针。若为AutoMove，需要停止执行自动控制命令。



利用OpenGL的 mouseMotion、mouseButton自定义鼠标操作，用于判断选中点、逆映射屏幕坐标生成控制点。鼠标按下时需要判断当前状态，若为MovePoint，需要判断是否选中控制点，若为MakePoint，需要调用**unProject**方法将屏幕坐标逆映射回世界坐标，生成控制点。

完整鼠标逻辑判断如下：



综上，实现Bezier相关功能，测试如下：

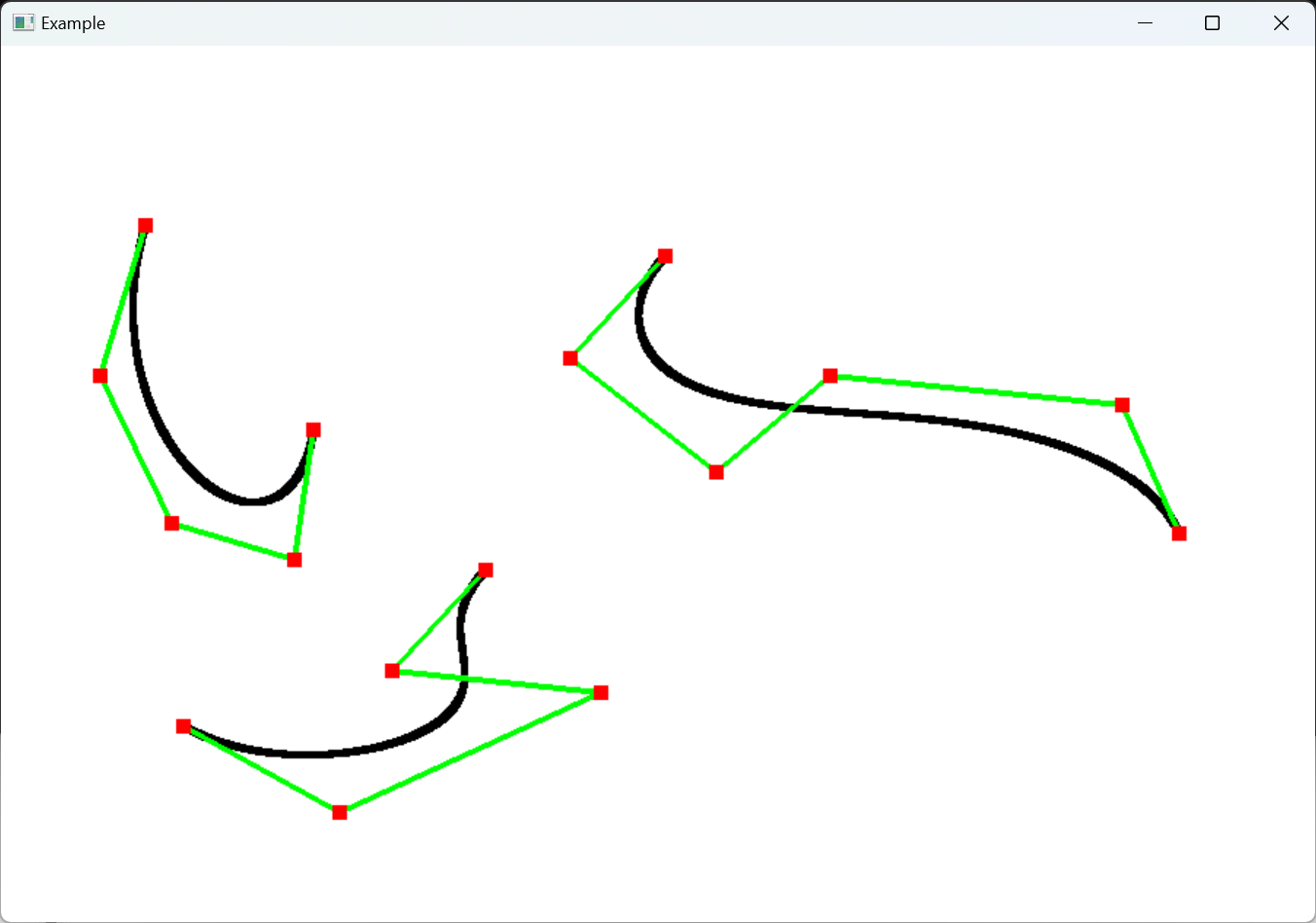


图 1‑1多Bezier曲线

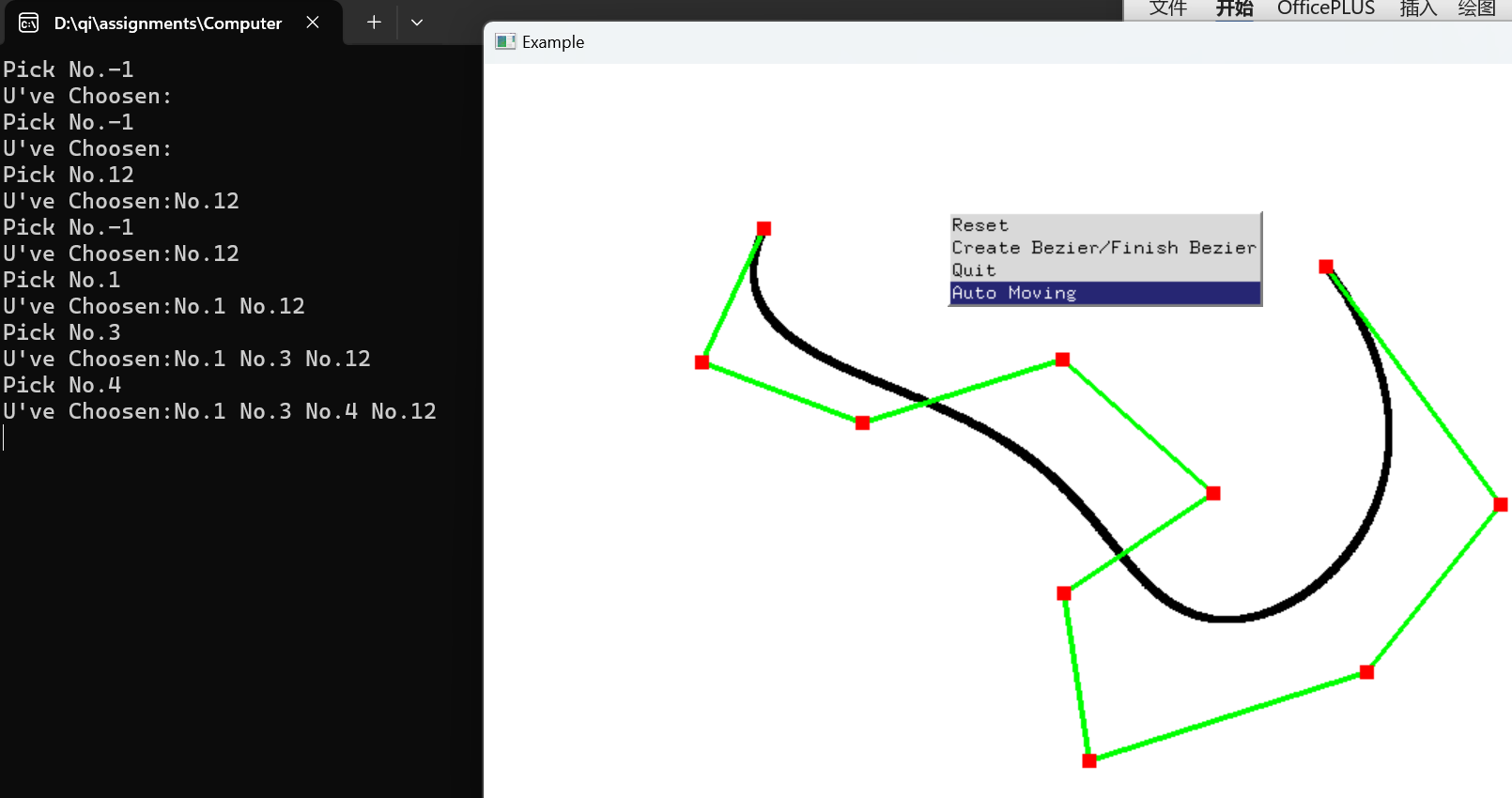


图 1-2程序自动变换多控制点

**2、分形几何**

**A）：定义视口坐标范围：设置四个视口，每个视口单独迭代**

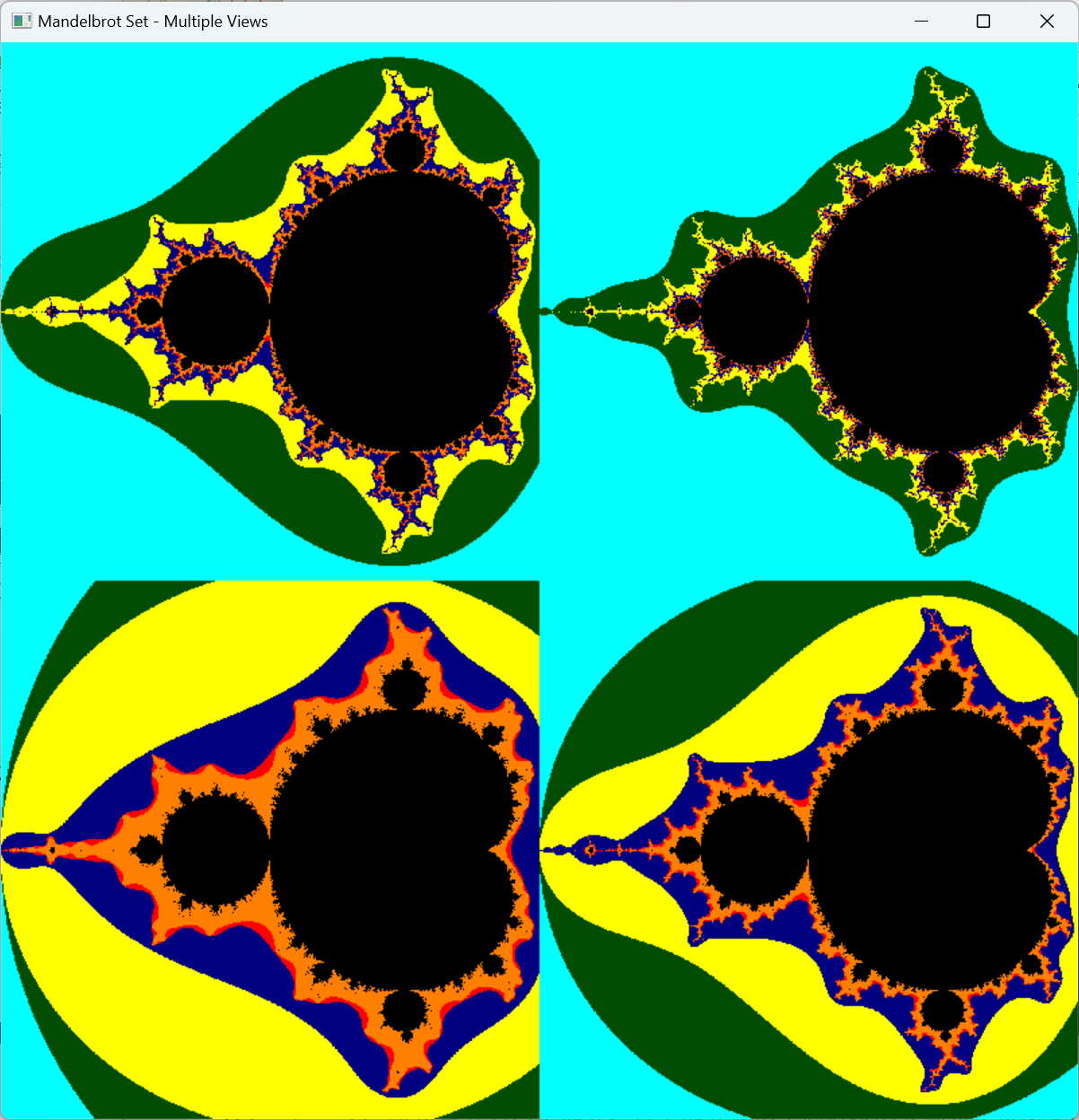


**B）：自动迭代：**

通过glutTimerFunc(100, animate, 0); // 每100毫秒调用一次函数重复调用迭代方法，当达到迭代上限则停止



**C）：测试如下：**



**五、总结及心得体会：**

**学习了bezier曲线构造、分形图形原理，体会到了opengl视角变换逻辑、世界坐标系和窗口坐标系的区别和转化**

学习了分形几何相关构造原理，学会用多视口绘制的方法

**六、对本实验过程及方法、手段的改进建议：**

希望封装用户操作函数能在分形程序中指定迭代次数

**报告评分：**

**指导教师签字：**