

**计算机图形学作业1**

学生姓名： 周麟

班 学 号： 111181

指导教师： 罗忠文

**中国地质大学信息工程学院**

**2020年 12 月 1 日**

**作业简介**

在本作业中，完成了对于多个Bezier，Bspline曲线控制点画出曲线的操作，进而完成了Bezier曲线与Bspline曲线的相互转化，并且能够在生成的曲线上进行增加、删除、移动点的操作。对于给定的样条曲线，能够通过旋转生成立体模型，以.obj形式输出。

**具体实现**

**Paint函数的实现**

Paint函数主要完成的是绘制控制点，并且将控制点连线的功能，每一个子类都需要调用该方法，所以只需要在父类Curve中实现即可。

先在初始化方法中保存输入的控制点数据和控制点个数。然后按照先画点、在画线的顺序进行，各点的坐标取值只需要通过类Vec3f内置函数实现；例如，vertices[i].Get(x, y, z);。画点时使用：

glBegin(GL\_POINTS); //表示画点开始

glColor3f(1.0, 0.0, 0.0); //给点设定颜色

glVertex3f(x, y, z); //画点

glEnd(); //画点结束标志

画线需要得到两个点的数据，才能两点连线形成一条线段。

glBegin(GL\_LINES); //表示画线开始

glColor3f(0.0, 0.0, 1.0); //给线设定颜色

glVertex3f(x1, y1, z1); //画第一个点

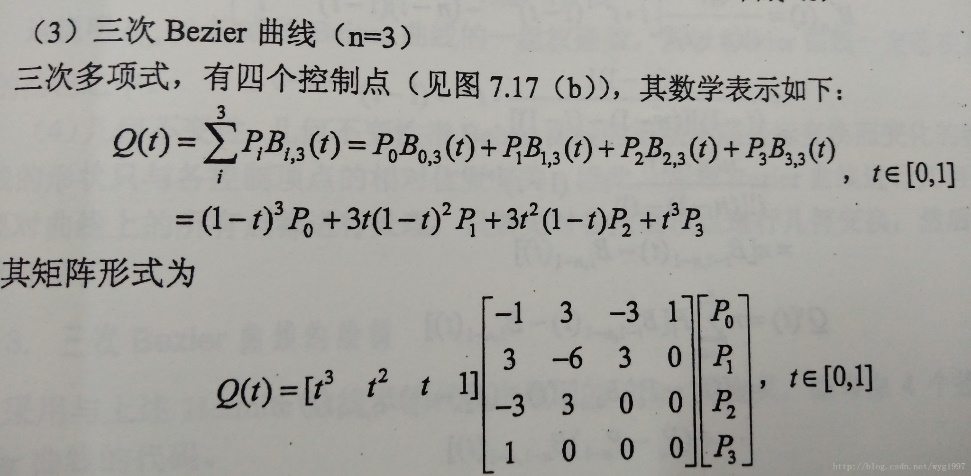
glVertex3f(x2, y2, z2); //连接到第二个点

glEnd(); //画线结束

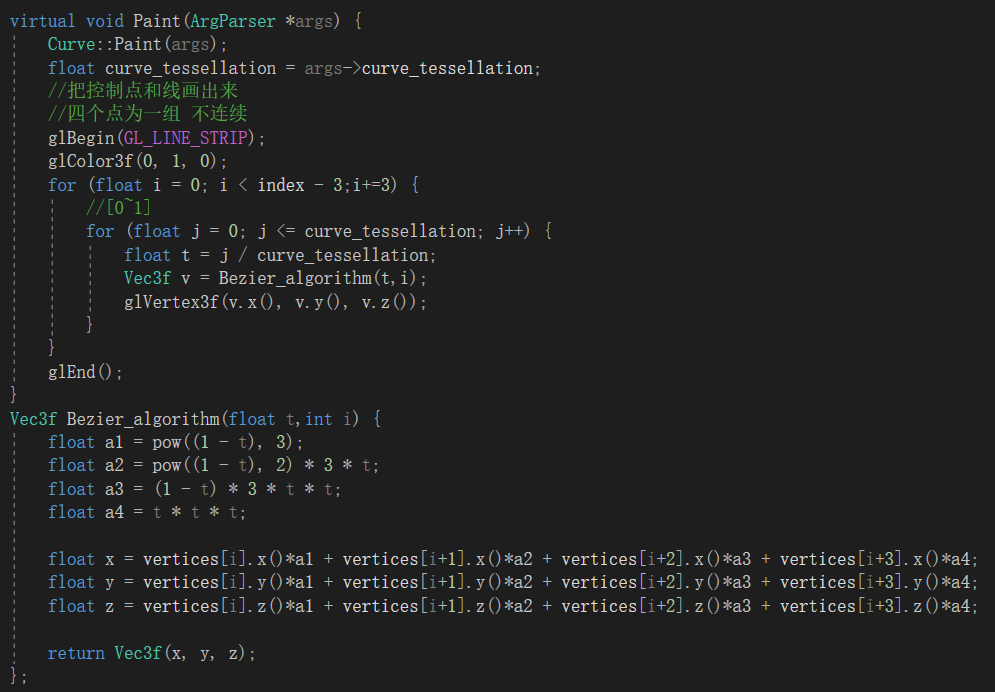
**绘制 Bezier 曲线和 BSplines 曲线**

具体绘制步骤为：首先读入曲线的细分程度，再使用for循环读入分别读入所有控制点，并且按照公式，得到每个曲线上点的数据，并且画到屏幕上。

绘制Bezier曲线的公式为

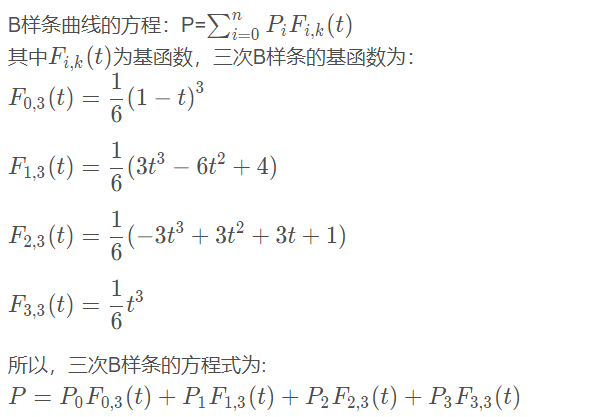


由图上式子可得，Bezier曲线上的每一个点根据式子，并且由t的取值决定（t在[0,1]中），因为读入文件中给出了细分程度curve\_tessellation，t的取值是从0开始，以1/curve\_tessellation为增长数增加。

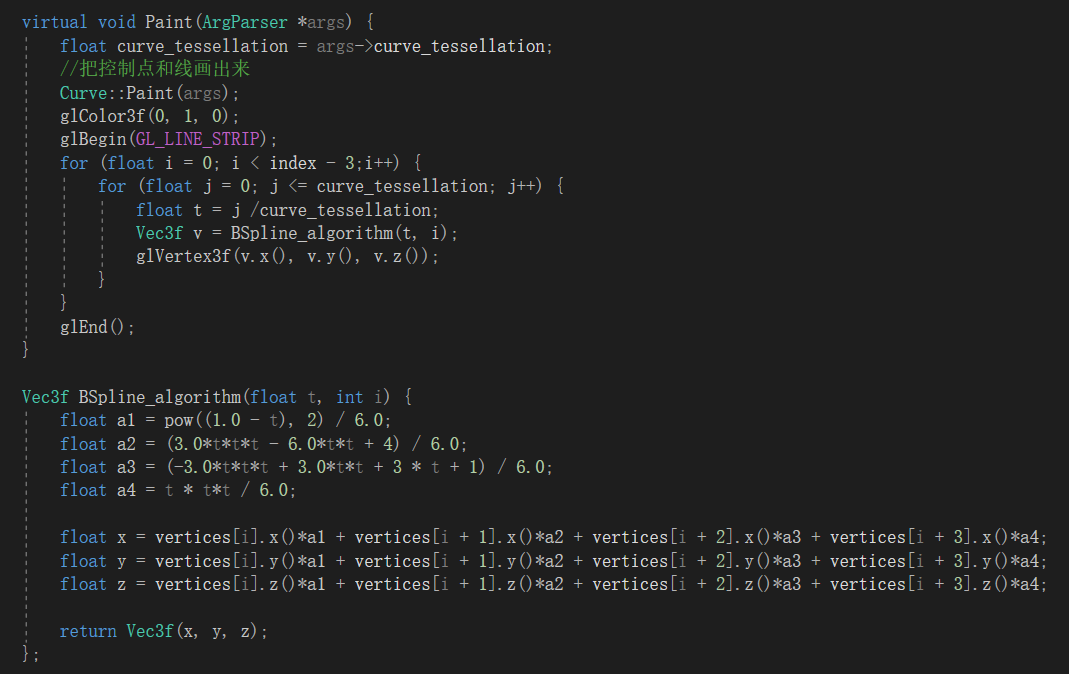


Spline曲线实现步骤相同，只不过式子不相同。

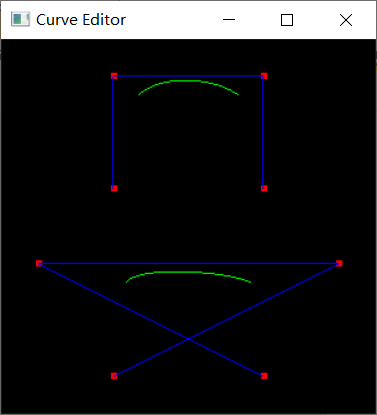
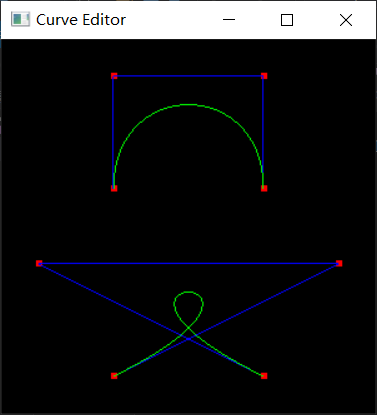
Spline曲线公式



实现代码：

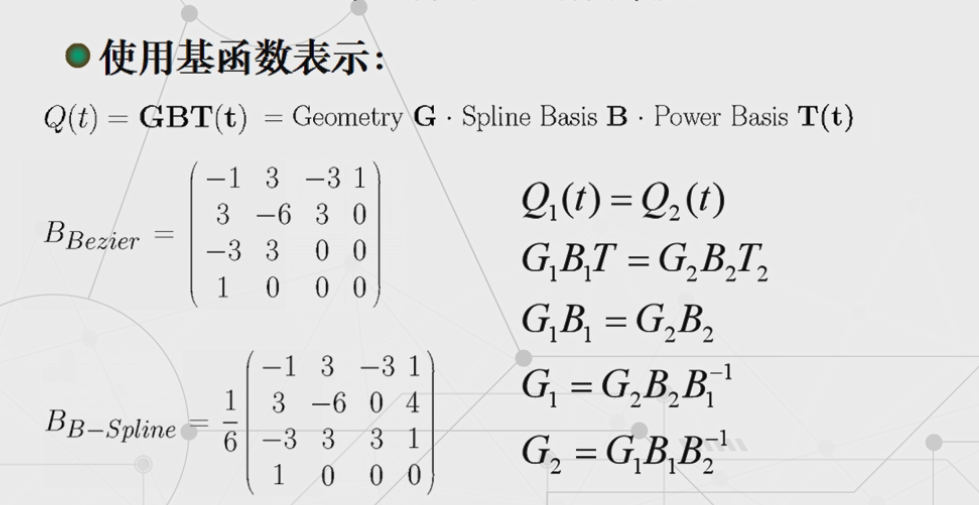


运行结果：



Bezier 曲线和 BSplines 曲线之间的转换

对于两类曲线的相互转化，有相应的式子实现

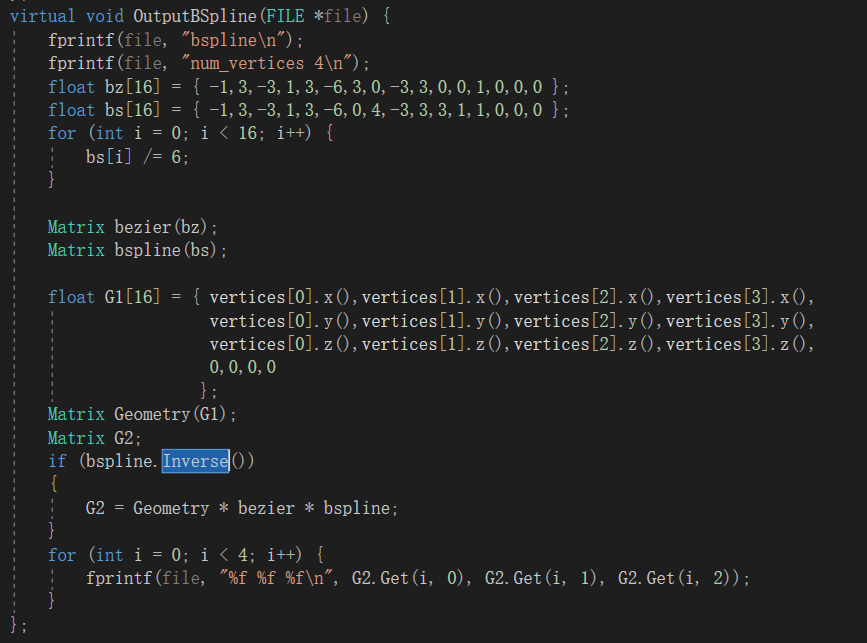


原工程中提供了很多功能进行矩阵操作，比如Matrix类提供了矩阵初始化，矩阵转置Inverse，都是曲线互相转换之间需要进行的操作。

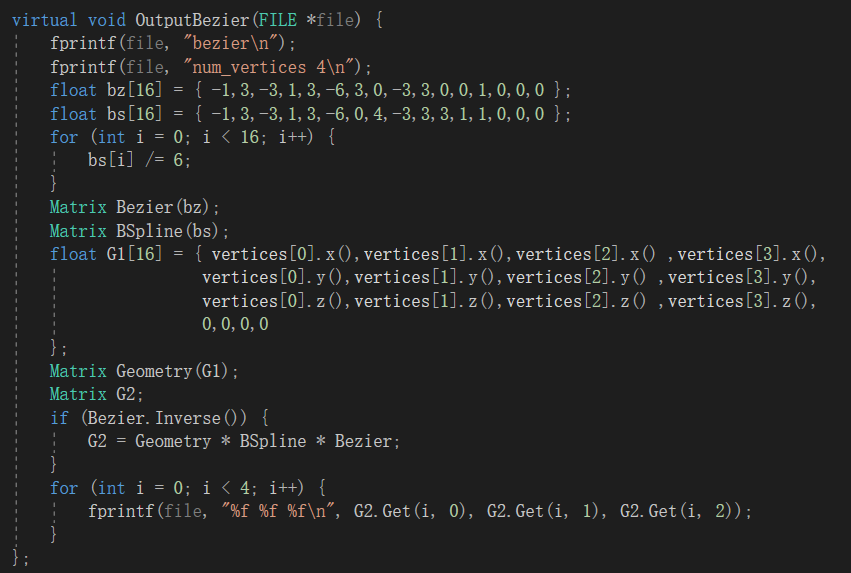
根据以上方程将数据都转化成矩阵形式，然后通过转置，就可以得到不同曲线的相同图像不同控制点的数据。

然后按照一定的格式输出至文件中即可。

Bezier转Spline

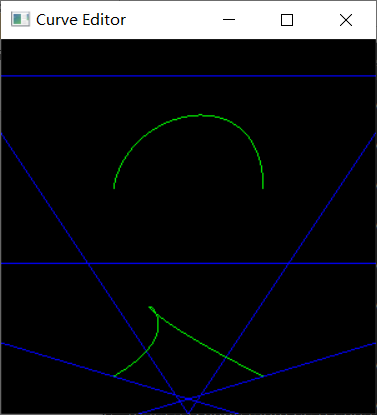
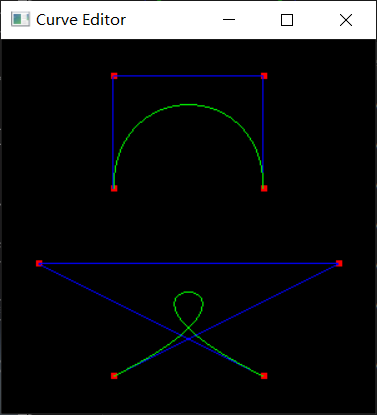


Spline转Bezier

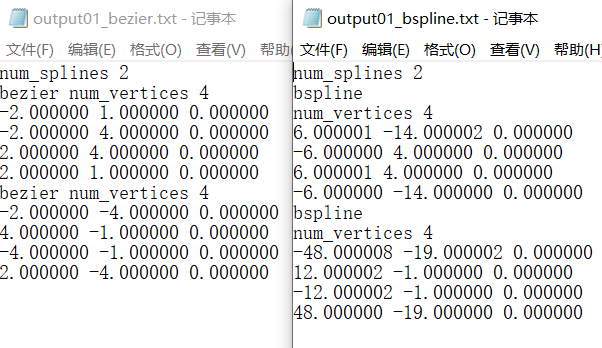


运行结果

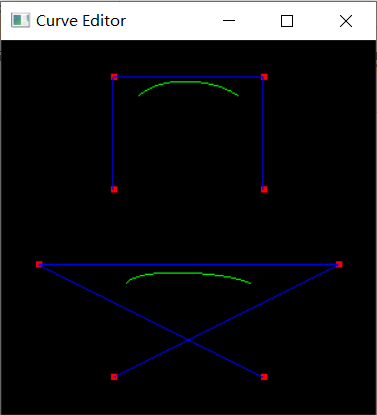
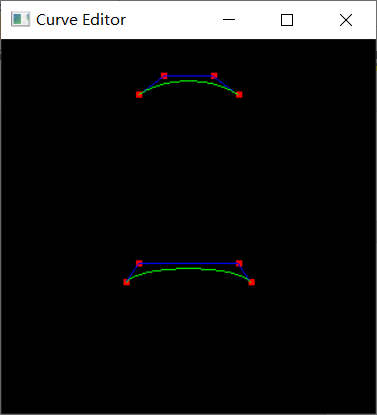
Bezier转Spline



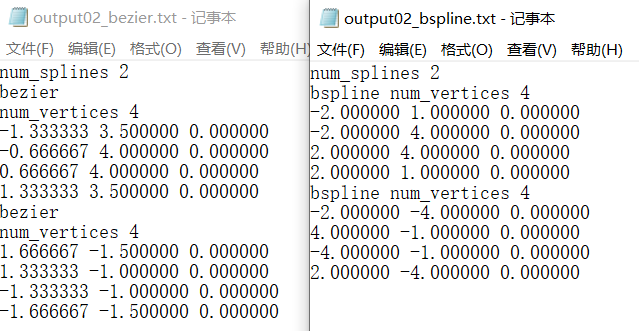
数据对比：



Spline转Bezier

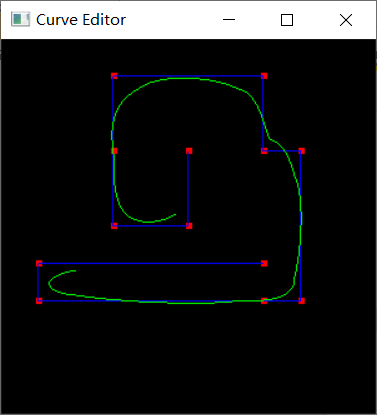
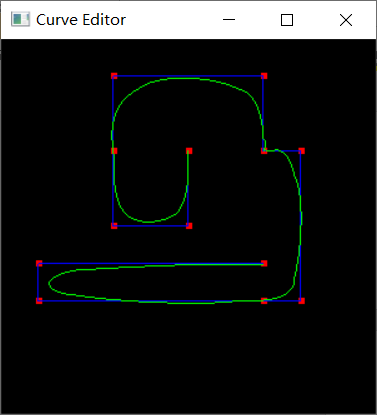
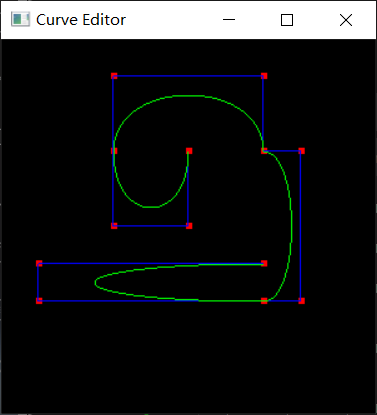


数据对比



实现多个控制点的曲线转换

只需要将保存点数据的容器大小尽可能地设置得大就可以，保存完数据直接用paint函数画出来。这里将Vec3f\*vertices数组大小设置为1000，这样一定可以保存所有输入的点数据，也可以使用vector数组，因为vector的大小可以进行伸缩。

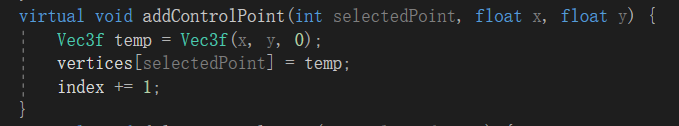


曲线编辑器

完成增加点个数、删除点个数、移动点的操作。原工程已经帮助实现了为增点判断鼠标选择坐标，为删除、移动点判断最近点的坐标的工作。

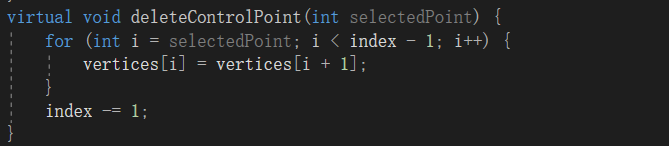
增加点个数

因为上一部分已经对数组大小尽可能地设置得大，所以只需要传入下标中对应的控制点位置中设置值即可。



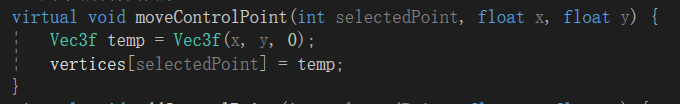
删除点个数

因为没有使用vector数组，所以不能对数组元素直接使用pop操作，所以使用元素向前移动，覆盖删除的点的数据实现。

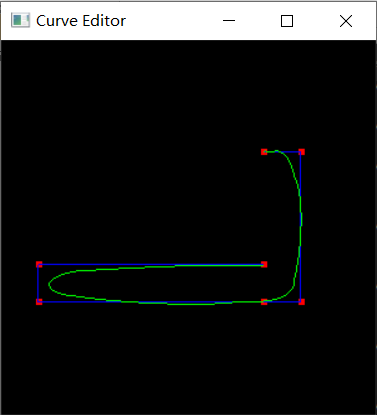
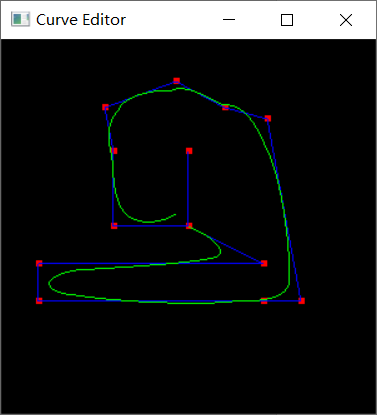
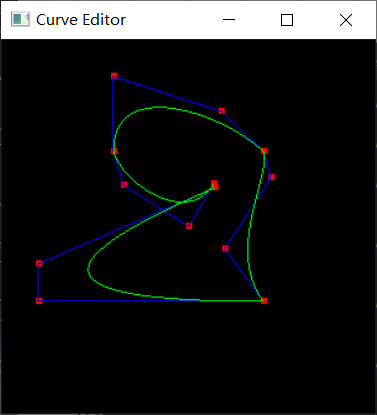


移动点

修改传入控制点下标位置的数据即可。



运行结果



实现SurfaceOfRevolution类

提取输入参数curve\_tessellation，revolution\_tessellation，给TriangleNet类初始化，为了生成三角形网。但是初始化赋值时，每一维的面数通过公式计算得出

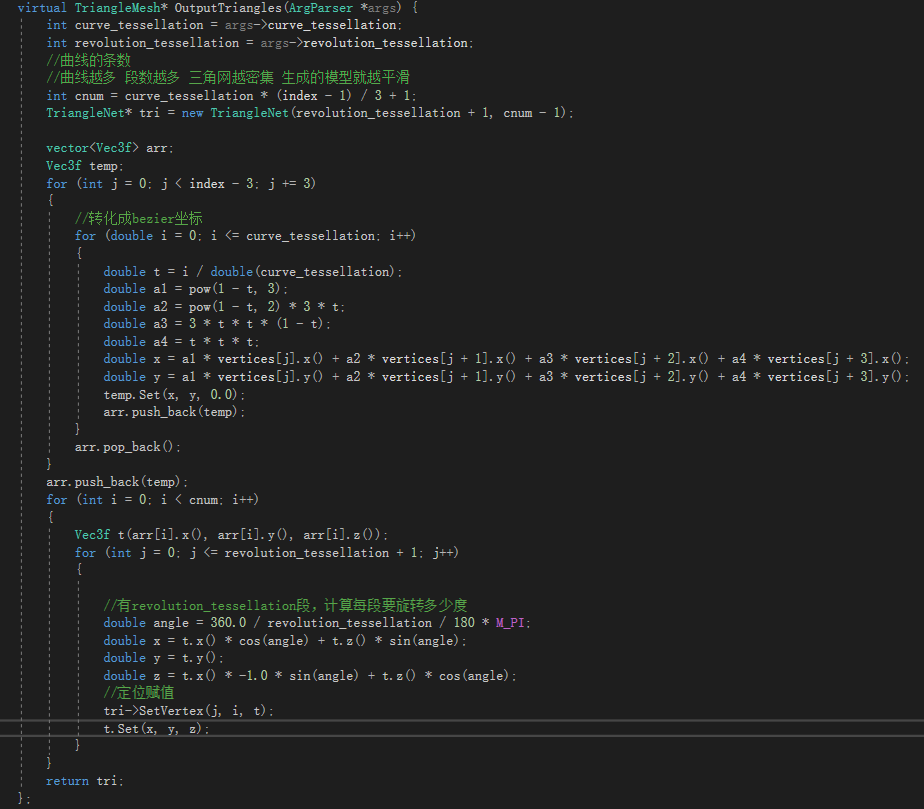
int cnum = curve\_tessellation \* (index - 1) / 3 + 1;

在初始化时要-1，+1是为了之后操作方便，并且要注意每一维上顶点数要多一个。

实现步骤

* 输入参数对TriangleNet进行初始化。
* 对每个控制点在不同旋转段数的值进行Bezier、Bspline转化。
* 对于保存的点按不同面的不同旋转段数使用欧拉旋转矩阵方程进行点的转换。
* 将转换的点使用SetTriangle()给三角形网格中每一个顶点编号。

代码：



因为配置的问题，不能运行raytracer，但是可以通过本机的3D查看器查看。

运行结果：

