# 实验报告

姓名：张立成 学号：10165102104 班级：16级计科

实验题目：编程实现低次Bezier曲线 实验时间：2019.11.12

### 问题描述

鼠标输入控制点，通过控制点绘制低次Bezier曲线，本实验支持三次/四次Bezier曲线

## 求解思路

利用Bezier曲线具有端点插值的性质，即曲线在两端点与控制多边形端点重合且端边相切推导出Bezier曲线上各点与控制点的关系，即，其中，n为Bezier的次数，相应的控制点个数为n+1，为了更加贴合地模拟曲线，遍历t在[0, 1]范围内，以步长0.001递增，然后计算得到曲线上点坐标，绘制后即可得到Bezier曲线

### 程序代码

绘制Bezier曲线代码：

**def** draw\_bezier**(**self**,** color**):**

**if** len**(**self**.**control\_vertexes**)** **==** 4**:**

x0**,** y0 **=** self**.**control\_vertexes**[**0**]**

x1**,** y1 **=** self**.**control\_vertexes**[**1**]**

x2**,** y2 **=** self**.**control\_vertexes**[**2**]**

x3**,** y3 **=** self**.**control\_vertexes**[**3**]**

**for** t **in** range**(**1000**):**

t **=** t **/** 1000

x **=** **(**1**-**t**)\*\***3 **\*** x0 **+** 3**\***t**\*(**1**-**t**)\*\***2 **\*** x1 **+** 3**\***t**\***t**\*(**1**-**t**)** **\*** x2 **+** t**\*\***3 **\*** x3

y **=** **(**1**-**t**)\*\***3 **\*** y0 **+** 3**\***t**\*(**1**-**t**)\*\***2 **\*** y1 **+** 3**\***t**\***t**\*(**1**-**t**)** **\*** y2 **+** t**\*\***3 **\*** y3

self**.**paintPoint**(**x**,** y**,** color**)**

**elif** len**(**self**.**control\_vertexes**)** **==** 5**:**

x0**,** y0 **=** self**.**control\_vertexes**[**0**]**

x1**,** y1 **=** self**.**control\_vertexes**[**1**]**

x2**,** y2 **=** self**.**control\_vertexes**[**2**]**

x3**,** y3 **=** self**.**control\_vertexes**[**3**]**

x4**,** y4 **=** self**.**control\_vertexes**[**4**]**

**for** t **in** range**(**1000**):**

t **=** t **/** 1000

x **=** **(**1**-**t**)\*\***4 **\*** x0 **+** 4**\***t**\*(**1**-**t**)\*\***3 **\*** x1 **+** 6**\***t**\***t**\*(**1**-**t**)\*\***2 **\*** x2 **+** 4**\***t**\***t**\***t**\*(**1**-**t**)** **\*** x3 **+** t**\*\***4 **\*** x4

y **=** **(**1**-**t**)\*\***4 **\*** y0 **+** 4**\***t**\*(**1**-**t**)\*\***3 **\*** y1 **+** 6**\***t**\***t**\*(**1**-**t**)\*\***2 **\*** y2 **+** 4**\***t**\***t**\***t**\*(**1**-**t**)** **\*** y3 **+** t**\*\***4 **\*** y4

self**.**paintPoint**(**x**,** y**,** color**)**

**else:**

**return**

全部代码：

**import** sys

**from** PyQt5 **import** QtCore

**from** PyQt5**.**QtWidgets **import** **(**QApplication**,** QWidget**)**

**from** PyQt5**.**QtGui **import** **(**QPainter**,** QPen**,** QPixmap**)**

**from** PyQt5**.**QtCore **import** Qt

**class** **BezierCurve(**QWidget**):**

**def** \_\_init\_\_**(**self**):**

super**(**BezierCurve**,** self**).**\_\_init\_\_**()**

self**.**resize**(**800**,** 600**)**

self**.**board **=** QPixmap**(**800**,** 600**)**

self**.**board**.**fill**(**Qt**.**white**)**

# setMouseTracking设置为False，否则不按下鼠标时也会跟踪鼠标事件

self**.**setMouseTracking**(False)**

self**.**painter **=** QPainter**()**

self**.**control\_vertexes **=** **[]**

**def** paintEvent**(**self**,** paintEvent**):**

self**.**painter**.**begin**(**self**)**

self**.**painter**.**drawPixmap**(**0**,** 0**,** self**.**board**)**

self**.**painter**.**end**()**

**def** paintPoint**(**self**,** x**,** y**,** color**):**

self**.**painter**.**begin**(**self**.**board**)**

self**.**painter**.**setPen**(**QPen**(**color**,** 6**))**

self**.**painter**.**drawPoint**(**x**,** y**)**

self**.**painter**.**end**()**

self**.**update**()**

**def** join\_vertexes**(**self**,** vertexes**,** color**):**

**if** len**(**vertexes**)** **<** 3**:**

**return**

**else:**

# 依次连接各点

**for** i **in** range**(**len**(**vertexes**)** **-** 1**):**

self**.**bresenham\_drawline**(**vertexes**[**i**][**0**],** vertexes**[**i**][**1**],** vertexes**[**i **+** 1**][**0**],**

vertexes**[**i **+** 1**][**1**],** color**)**

**def** draw\_bezier**(**self**,** color**):**

**if** len**(**self**.**control\_vertexes**)** **==** 4**:**

x0**,** y0 **=** self**.**control\_vertexes**[**0**]**

x1**,** y1 **=** self**.**control\_vertexes**[**1**]**

x2**,** y2 **=** self**.**control\_vertexes**[**2**]**

x3**,** y3 **=** self**.**control\_vertexes**[**3**]**

**for** t **in** range**(**1000**):**

t **=** t **/** 1000

x **=** **(**1**-**t**)\*\***3 **\*** x0 **+** 3**\***t**\*(**1**-**t**)\*\***2 **\*** x1 **+** 3**\***t**\***t**\*(**1**-**t**)** **\*** x2 **+** t**\*\***3 **\*** x3

y **=** **(**1**-**t**)\*\***3 **\*** y0 **+** 3**\***t**\*(**1**-**t**)\*\***2 **\*** y1 **+** 3**\***t**\***t**\*(**1**-**t**)** **\*** y2 **+** t**\*\***3 **\*** y3

self**.**paintPoint**(**x**,** y**,** color**)**

**elif** len**(**self**.**control\_vertexes**)** **==** 5**:**

x0**,** y0 **=** self**.**control\_vertexes**[**0**]**

x1**,** y1 **=** self**.**control\_vertexes**[**1**]**

x2**,** y2 **=** self**.**control\_vertexes**[**2**]**

x3**,** y3 **=** self**.**control\_vertexes**[**3**]**

x4**,** y4 **=** self**.**control\_vertexes**[**4**]**

**for** t **in** range**(**1000**):**

t **=** t **/** 1000

x **=** **(**1**-**t**)\*\***4 **\*** x0 **+** 4**\***t**\*(**1**-**t**)\*\***3 **\*** x1 **+** 6**\***t**\***t**\*(**1**-**t**)\*\***2 **\*** x2 **+** 4**\***t**\***t**\***t**\*(**1**-**t**)** **\*** x3 **+** t**\*\***4 **\*** x4

y **=** **(**1**-**t**)\*\***4 **\*** y0 **+** 4**\***t**\*(**1**-**t**)\*\***3 **\*** y1 **+** 6**\***t**\***t**\*(**1**-**t**)\*\***2 **\*** y2 **+** 4**\***t**\***t**\***t**\*(**1**-**t**)** **\*** y3 **+** t**\*\***4 **\*** y4

self**.**paintPoint**(**x**,** y**,** color**)**

**else:**

**return**

**def** mousePressEvent**(**self**,** event**):**

x **=** event**.**pos**().**x**()**

y **=** event**.**pos**().**y**()**

**if** event**.**buttons**()** **==** QtCore**.**Qt**.**LeftButton**:** # 左键按下

self**.**control\_vertexes**.**append**((**x**,** y**))**

self**.**paintPoint**(**x**,** y**,** Qt**.**black**)**

**elif** event**.**buttons**()** **==** QtCore**.**Qt**.**RightButton**:**

self**.**join\_vertexes**(**self**.**control\_vertexes**,** Qt**.**black**)**

self**.**draw\_bezier**(**Qt**.**blue**)**

# 画板清零，回到初始状态

**def** mouseDoubleClickEvent**(**self**,** event**):**

self**.**board**.**fill**(**Qt**.**white**)**

self**.**update**()**

# 回到初始状态

self**.**control\_vertexes**.**clear**()**

**def** bresenham\_drawline**(**self**,** x0**,** y0**,** x1**,** y1**,** color**):**

dx **=** abs**(**x1 **-** x0**)**

dy **=** abs**(**y1 **-** y0**)**

x **=** x0

y **=** y0

# 通过两点的位置关系和斜率区分不同情况

**if** x1 **>** x0**:**

step\_x **=** 1

**else:**

step\_x **=** **-**1

**if** y1 **>** y0**:**

step\_y **=** 1

**else:**

step\_y **=** **-**1

**if** dy **>** dx**:**

# 斜率绝对值大于1，交换dx, dy再代入公式，并记录交换的标志位

tmp **=** dx

dx **=** dy

dy **=** tmp

flag\_change **=** **True**

**else:**

flag\_change **=** **False**

delta **=** 2 **\*** dy **-** dx

**for** i **in** range**(**1**,** dx **+** 1**):**

# 调用封装好的画点函数画点

self**.**paintPoint**(**x**,** y**,** color**)**

**if** delta **>=** 0**:**

**if** flag\_change**:**

x **+=** step\_x

**else:**

y **+=** step\_y

delta **-=** 2 **\*** dx

**if** flag\_change**:**

y **+=** step\_y

**else:**

x **+=** step\_x

delta **+=** 2 **\*** dy

**if** \_\_name\_\_ **==** "\_\_main\_\_"**:**

app **=** QApplication**(**sys**.**argv**)**

curve\_pyqt **=** BezierCurve**()**

curve\_pyqt**.**show**()**

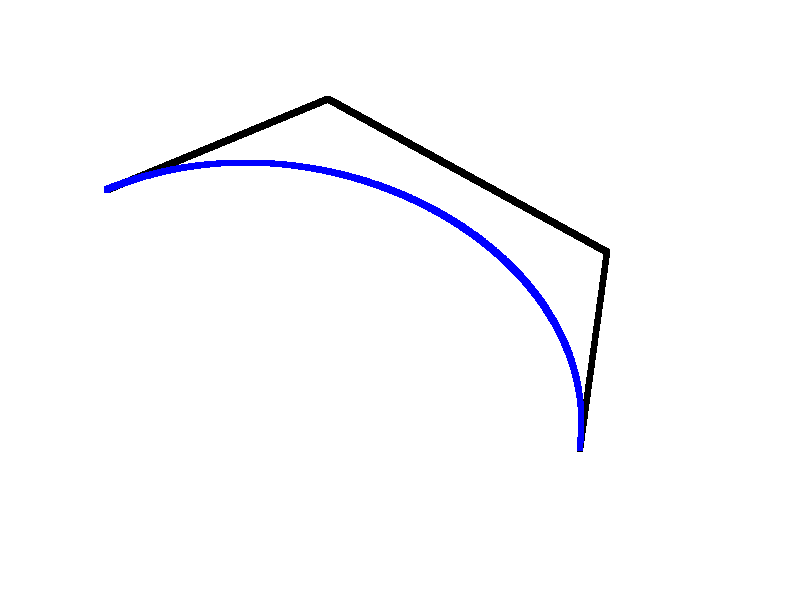
app**.**exec\_**()**

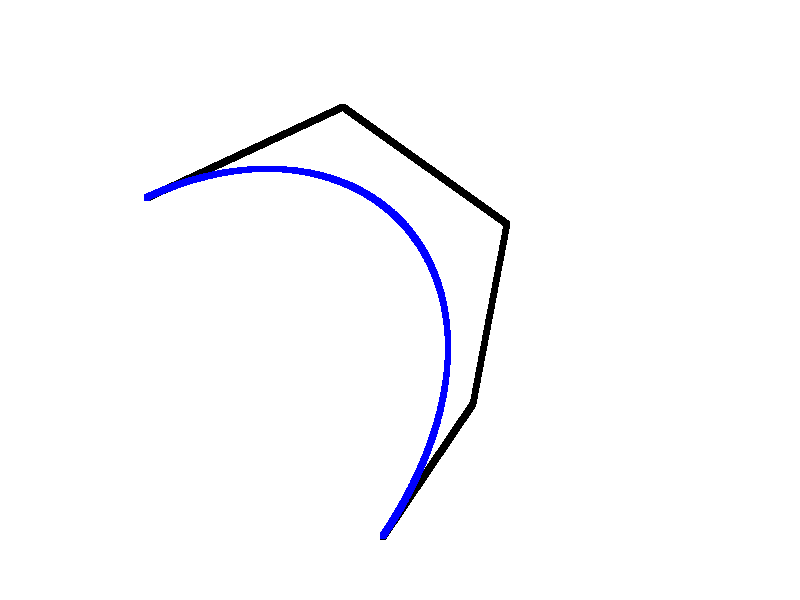
### 实验结果

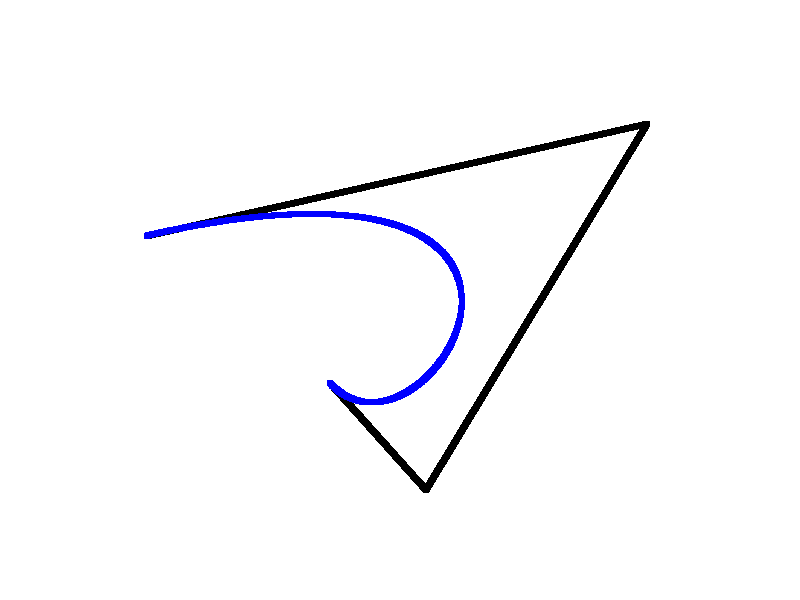
首先鼠标左键绘制控制点：

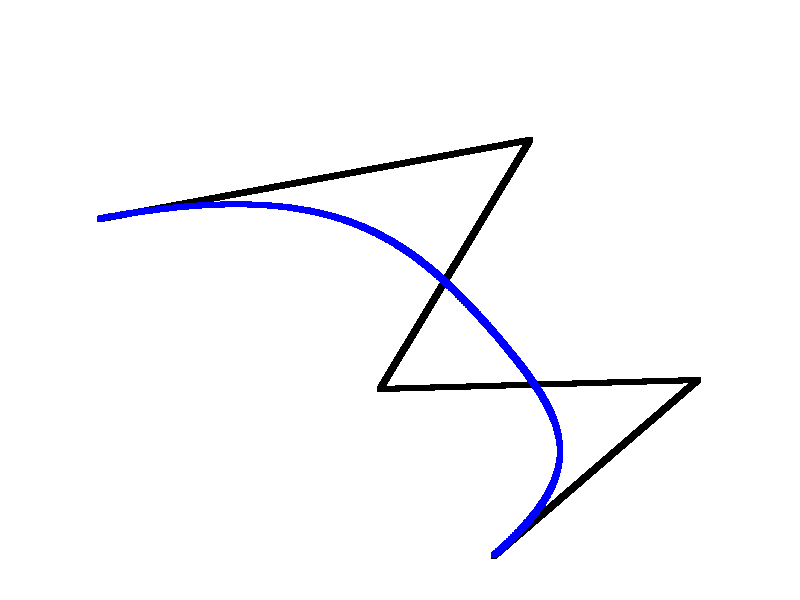


点击右键绘制曲线：









### 实验心得

此次实验难点主要在于利用Bezier曲线的性质得出曲线上各点与控制点之间的关系，推出公式之后代码实现并不难，体会到解决很多问题的时候，思路和推导其实是最重要也往往是比较难的，而实现相对来说是容易的。本实验只实现了绘制3次和4次Bezier曲线，其实是可以进行推广到n次曲线的，只需要将公式整合推广到n即可。