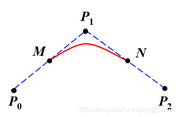
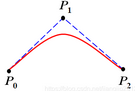
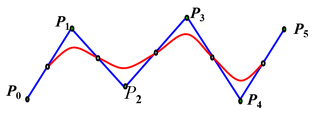
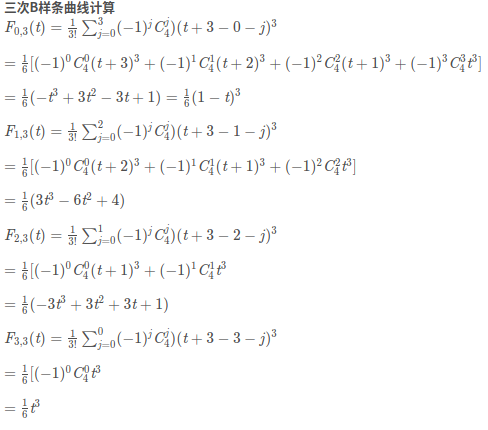
 B样条曲线从Bezier曲线演变而来，由于每条Bezier都经过端点，导致分段的Bezier曲线在端点处难以平滑过渡。二次B样条曲线克服了这个缺点，把端点移到线段中点（如下图所示），这样就能保证各段曲线在连接处能够一阶导数连续。如图左侧为贝塞尔，右侧为B样条曲线。



如图所示的折线，B样条曲线能保证一阶导数连续，而贝塞尔曲线则不能。

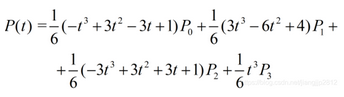


二阶和三阶B样条曲线公式如下：



2020-11-26 16-48-40 的屏幕截图

2020-11-26 16-29-42 的屏幕截图



**程序实现了二次与三次B样条曲线，封装成了BSpline类**

**BSpine.h**

#pragma once

//#include "position.h"

typedef struct tagPosition

{

double x;

double y;

tagPosition(double \_x,double \_y) { x=\_x; y=\_y;}

tagPosition() {};

bool operator==(const tagPosition & pt) { return (x==pt.x && y==pt.y);}

} CPosition;

class CBSpline

{

public:

CBSpline(void);

~CBSpline(void);

void TwoOrderBSplineSmooth(CPosition \*pt,int Num);

void TwoOrderBSplineInterpolatePt(CPosition \*&pt,int &Num,int \*InsertNum);

double F02(double t);

double F12(double t);

double F22(double t);

void ThreeOrderBSplineSmooth(CPosition \*pt,int Num);

void ThreeOrderBSplineInterpolatePt(CPosition \*&pt,int &Num,int \*InsertNum);

double F03(double t);

double F13(double t);

double F23(double t);

double F33(double t);

};

**BSpine.cpp**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* BSpline.cpp \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// 包含功能：二次B样条平滑，三次B样条平滑；二次B样条平滑后节点插值

//

// 作者： 蒋锦朋 1034378054@qq.com

// 单位： 中国地质大学（武汉） 地球物理与空间信息学院

// 日期： 2014/12/03 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include "StdAfx.h"

#include "BSpline.h"

CBSpline::CBSpline(void)

{

}

CBSpline::~CBSpline(void)

{

}

//======================================================================

// 函数功能： 二次B样条平滑，把给定的点，平滑到B样条曲线上，不增加点的数目

// 输入参数： \*pt ：给定点序列，执行完成后，会被替换成新的平滑点

// Num：点个数

// 返回值： 无返回值

// 编辑日期： 2014/12/03 //===================================================================

· void CBSpline::TwoOrderBSplineSmooth(CPosition \*pt,int Num)

· · {

· · CPosition \*temp=new CPosition[Num];

· · for(int i=0;i<Num;i++)

· · temp[i]=pt[i];

· ·

· · temp[0].x=2\*temp[0].x-temp[1].x; // 将折线两端点换成延长线上两点

· · temp[0].y=2\*temp[0].y-temp[1].y;

· ·

· · temp[Num-1].x=2\*temp[Num-1].x-temp[Num-2].x;

· · temp[Num-1].y=2\*temp[Num-1].y-temp[Num-2].y;

· ·

· · CPosition NodePt1,NodePt2,NodePt3;

· · double t;

· · for(int i=0;i<Num-2;i++)

· · {

· · NodePt1=temp[i]; NodePt2=temp[i+1]; NodePt3=temp[i+2];

· · if(i==0) // 第一段取t=0和t=0.5点

· · {

· · t=0;

· · pt[i].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[i].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · t=0.5;

· · pt[i+1].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[i+1].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · }else if(i==Num-3) // 最后一段取t=0.5和t=1点

· {

· · t=0.5;

· · pt[i+1].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[i+1].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · t=1;

· · pt[i+2].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[i+2].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · }else // 中间段取t=0.5点

· · {

· · t=0.5;

· · pt[i+1].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[i+1].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · }

· · }

· · delete []temp;

· · }

· ·

· · //================================================================

· · // 函数功能： 二次B样条拟合,在节点之间均匀插入指定个数点

· · // 输入参数： \*pt ：给定点序列，执行完成后，会被替换成新的数据点

· · // Num：节点点个数

· · // \*InsertNum: 节点之间需要插入的点个数指针

· · // 返回值： 无返回值

· · //

· · // 编辑日期： 2014/12/07

· · //=================================================================

· void CBSpline::TwoOrderBSplineInterpolatePt(CPosition \*&pt,int &Num,int \*InsertNum)

· · {

· · if(pt==NULL || InsertNum==NULL) return;

· ·

· · int InsertNumSum=0; // 计算需要插入的点总数

· · for(int i=0;i<Num-1;i++) InsertNumSum+=InsertNum[i];

· ·

· · CPosition \*temp=new CPosition[Num]; // 二次B样条不需要增加点数，需要将首尾点替换掉

· · for(int i=0;i<Num;i++)

· · temp[i]=pt[i];

· ·

· · temp[0].x=2\*temp[0].x-temp[1].x; // 将折线两端点换成延长线上两点

· · temp[0].y=2\*temp[0].y-temp[1].y;

· ·

· · temp[Num-1].x=2\*temp[Num-1].x-temp[Num-2].x;

· · temp[Num-1].y=2\*temp[Num-1].y-temp[Num-2].y;

· ·

· · delete []pt; // 点数由原来的Num个增加到Num+InsertNumSum个，删除旧的存储空间，开辟新的存储空间

· ·

· · pt=new CPosition[Num+InsertNumSum];

· ·

· · CPosition NodePt1,NodePt2,NodePt3,NodePt4; // 两节点间均匀插入点，需要相邻两段样条曲线,因此需要四个节点

· ·

· · double t;

· · int totalnum=0;

· for(int i=0;i<Num-1;i++) // 每条线段均匀插入点

· · {

· · if(i==0) // 第一段只需计算第一条样条曲线，无NodePt1

· · {

· · NodePt2=temp[i]; NodePt3=temp[i+1]; NodePt4=temp[i+2];

· ·

· · double dt=0.5/(InsertNum[i]+1);

· · for(int j=0;j<InsertNum[i]+1;j++)

· · {

· · t=0+dt\*j;

· · pt[totalnum].x=F02(t)\*NodePt2.x+F12(t)\*NodePt3.x+F22(t)\*NodePt4.x;

· · pt[totalnum].y=F02(t)\*NodePt2.y+F12(t)\*NodePt3.y+F22(t)\*NodePt4.y;

· · totalnum++;

· · }

· · }else if(i==Num-2) // 最后一段只需计算最后一条样条曲线，无NodePt4

· · {

· · NodePt1=temp[i-1]; NodePt2=temp[i]; NodePt3=temp[i+1];

· ·

· · double dt=0.5/(InsertNum[i]+1);

· · for(int j=0;j<InsertNum[i]+2;j++)

· · {

· · t=0.5+dt\*j;

· · pt[totalnum].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[totalnum].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · totalnum++;

· · }

· }else

· · {

· · NodePt1=temp[i-1],NodePt2=temp[i]; NodePt3=temp[i+1]; NodePt4=temp[i+2]; // NodePt1,2,3计算第一条曲线，NodePt2,3,4计算第二条曲线

· ·

· · int LeftInsertNum,RightInsertNum; // 计算线段间左右曲线段上分别需要插入的点数

· · double rightoffset=0; // 左边曲线段从t=0.5开始，又边曲线段从t=rightoffset开始

· · double Leftdt=0,Rightdt=0; // 左右曲线取点t步长

· · if(InsertNum[i]==0 )

· · {

· · LeftInsertNum=0;

· · RightInsertNum=0;

· · }else if(InsertNum[i]%2==1) // 插入点数为奇数，左边曲线段插入点个数比右边多一点

· · {

· · RightInsertNum=InsertNum[i]/2;

· · LeftInsertNum=RightInsertNum+1;

· · Leftdt=0.5/(LeftInsertNum);

· · Rightdt=0.5/(RightInsertNum+1);

· · rightoffset=Rightdt;

· · }else // 插入点数为偶数，左右边曲线段插入个数相同

· · {

· · RightInsertNum=InsertNum[i]/2;

· · LeftInsertNum=RightInsertNum;

· · Leftdt=0.5/(LeftInsertNum+0.5);

· · Rightdt=0.5/(RightInsertNum+0.5);

· · rightoffset=Rightdt/2;

· · }

· for(int j=0;j<LeftInsertNum+1;j++)

· · {

· · t=0.5+Leftdt\*j;

· · pt[totalnum].x=F02(t)\*NodePt1.x+F12(t)\*NodePt2.x+F22(t)\*NodePt3.x;

· · pt[totalnum].y=F02(t)\*NodePt1.y+F12(t)\*NodePt2.y+F22(t)\*NodePt3.y;

· · totalnum++;

· · }

· ·

· · for(int j=0;j<RightInsertNum;j++)

· · {

· · t=rightoffset+Rightdt\*j;

· · pt[totalnum].x=F02(t)\*NodePt2.x+F12(t)\*NodePt3.x+F22(t)\*NodePt4.x;

· · pt[totalnum].y=F02(t)\*NodePt2.y+F12(t)\*NodePt3.y+F22(t)\*NodePt4.y;

· · totalnum++;

· · }

· · }

· · }

· · delete []temp;

· · Num=Num+InsertNumSum;

· ·

· · }

· //================================================================

· · // 函数功能： 二次样条基函数

· · //

· · // 编辑日期： 2014/12/03

· · //================================================================

· · double CBSpline::F02(double t)

· · {

· · return 0.5\*(t-1)\*(t-1);

· · }

· · double CBSpline::F12(double t)

· · {

· · return 0.5\*(-2\*t\*t+2\*t+1);

· · }

· · double CBSpline::F22(double t)

· · {

· · return 0.5\*t\*t;

· · }

· · //========================================================================

· · // 函数功能： 三次B样条平滑，把给定的点，平滑到B样条曲线上，不增加点的数目

· · // 输入参数： \*pt ：给定点序列，执行完成后，会被替换成新的平滑点

· · // Num：点个数

· · // 返回值： 无返回值

· · //

· · // 编辑日期： 2014/12/03

· · //========================================================================

· void CBSpline::ThreeOrderBSplineSmooth(CPosition \*pt,int Num)

· · {

· · CPosition \*temp=new CPosition[Num+2];

· · for(int i=0;i<Num;i++)

· · temp[i+1]=pt[i];

· ·

· · temp[0].x=2\*temp[1].x-temp[2].x; // 将折线延长线上两点加入作为首点和尾点

· · temp[0].y=2\*temp[1].y-temp[2].y;

· ·

· · temp[Num+1].x=2\*temp[Num].x-temp[Num-1].x;

· · temp[Num+1].y=2\*temp[Num].y-temp[Num-1].y;

· ·

· · CPosition NodePt1,NodePt2,NodePt3,NodePt4;

· · double t;

· · for(int i=0;i<Num-1;i++)

· · {

· · NodePt1=temp[i]; NodePt2=temp[i+1]; NodePt3=temp[i+2]; NodePt4=temp[i+3];

· ·

· · if(i==Num-4) // 最后一段取t=0.5和t=1点

· · {

· · t=0;

· · pt[i].x=F03(t)\*NodePt1.x+F13(t)\*NodePt2.x+F23(t)\*NodePt3.x+F33(t)\*NodePt4.x;

· · pt[i].y=F03(t)\*NodePt1.y+F13(t)\*NodePt2.y+F23(t)\*NodePt3.y+F33(t)\*NodePt4.y;

· · t=1;

· · pt[i+1].x=F03(t)\*NodePt1.x+F13(t)\*NodePt2.x+F23(t)\*NodePt3.x+F33(t)\*NodePt4.x;

· · pt[i+1].y=F03(t)\*NodePt1.y+F13(t)\*NodePt2.y+F23(t)\*NodePt3.y+F33(t)\*NodePt4.y;

· · }else // 中间段取t=0.5点

· {

· · t=0;

· · pt[i].x=F03(t)\*NodePt1.x+F13(t)\*NodePt2.x+F23(t)\*NodePt3.x+F33(t)\*NodePt4.x;

· · pt[i].y=F03(t)\*NodePt1.y+F13(t)\*NodePt2.y+F23(t)\*NodePt3.y+F33(t)\*NodePt4.y;

· · }

· · }

· · delete []temp;

· · }

· ·

· · //================================================================

· · // 函数功能： 三次B样条拟合,在节点之间均匀插入指定个数点

· · // 输入参数： \*pt ：给定点序列，执行完成后，会被替换成新的数据点

· · // Num：节点点个数

· · // \*InsertNum: 节点之间需要插入的点个数指针

· · // 返回值： 无返回值

· · //

· · // 编辑日期： 2014/12/07

· · //=================================================================

· · void CBSpline::ThreeOrderBSplineInterpolatePt(CPosition \*&pt,int &Num,int \*InsertNum)

· · {

· · if(pt==NULL || InsertNum==NULL) return;

· ·

· · int InsertNumSum=0; // 计算需要插入的点总数

· · for(int i=0;i<Num-1;i++) InsertNumSum+=InsertNum[i];

· ·

· · CPosition \*temp=new CPosition[Num+2];

· · for(int i=0;i<Num;i++)

· · temp[i+1]=pt[i];

· temp[0].x=2\*temp[1].x-temp[2].x; // 将折线延长线上两点加入作为首点和尾点

· · temp[0].y=2\*temp[1].y-temp[2].y;

· ·

· · temp[Num+1].x=2\*temp[Num].x-temp[Num-1].x;

· · temp[Num+1].y=2\*temp[Num].y-temp[Num-1].y;

· ·

· · CPosition NodePt1,NodePt2,NodePt3,NodePt4;

· · double t;

· ·

· · delete []pt; // 点数由原来的Num个增加到Num+InsertNumSum个，删除旧的存储空间，开辟新的存储空间

· ·

· · pt=new CPosition[Num+InsertNumSum];

· ·

· · int totalnum=0;

· · for(int i=0;i<Num-1;i++) // 每条线段均匀插入点

· · {

· · NodePt1=temp[i]; NodePt2=temp[i+1]; NodePt3=temp[i+2]; NodePt4=temp[i+3];

· · double dt=1.0/(InsertNum[i]+1);

· ·

· · for(int j=0;j<InsertNum[i]+1;j++)

· · {

· · t=dt\*j;

· · pt[totalnum].x=F03(t)\*NodePt1.x+F13(t)\*NodePt2.x+F23(t)\*NodePt3.x+F33(t)\*NodePt4.x;

· · pt[totalnum].y=F03(t)\*NodePt1.y+F13(t)\*NodePt2.y+F23(t)\*NodePt3.y+F33(t)\*NodePt4.y;

· · totalnum++;

· · }

· if(i==Num-2){ // 最后一个尾点

· · t=1;

· · pt[totalnum].x=F03(t)\*NodePt1.x+F13(t)\*NodePt2.x+F23(t)\*NodePt3.x+F33(t)\*NodePt4.x;

· · pt[totalnum].y=F03(t)\*NodePt1.y+F13(t)\*NodePt2.y+F23(t)\*NodePt3.y+F33(t)\*NodePt4.y;

· · totalnum++;

· · }

· · }

· ·

· · delete []temp;

· · Num=Num+InsertNumSum;

· ·

· · }

· ·

· · //================================================================

· · // 函数功能： 三次样条基函数

· · //

· · // 编辑日期： 2014/12/03

· · //================================================================

· · double CBSpline::F03(double t)

· · {

· · return 1.0/6\*(-t\*t\*t+3\*t\*t-3\*t+1);

· · }

· · double CBSpline::F13(double t)

· · {

· return 1.0/6\*(3\*t\*t\*t-6\*t\*t+4);

· · }

· · double CBSpline::F23(double t)

· · {

· · return 1.0/6\*(-3\*t\*t\*t+3\*t\*t+3\*t+1);

· · }

· · double CBSpline::F33(double t)

· · {

· · return 1.0/6\*t\*t\*t;

· · }

**程序调用：**

· #include "stdafx.h"

· · #include "math.h"

· · #include "BSpline.h"

· ·

· · int \_tmain(int argc, \_TCHAR\* argv[])

· · {

· · int num=8;

· · double x[8]={9.59,60.81,105.57,161.59,120.5,100.1,50.0,10.0};

· · double y[8]={61.97,107.13,56.56,105.27,120.5,150.0,110.0,180.0};

· ·

· · CPosition \*testpt=new CPosition[num];

· · for(int i=0;i<num;i++) testpt[i]=CPosition(x[i],y[i]);

· ·

· · int \*Intnum=new int[num-1];

· · for(int i=0;i<num-1;i++){

· · Intnum[i]=10; // 每一个样条曲线内插入10个点

· · }

· ·

· · int num2=num;

· · CBSpline bspline;

· · bspline.TwoOrderBSplineInterpolatePt(testpt,num2,Intnum); // 二次B样条曲线

· · //bspline.ThreeOrderBSplineInterpolatePt(testpt,num2,Intnum); // 三次B样条曲线

· ·

· · //bspline.TwoOrderBSplineSmooth(testpt,num2); // 二次B样条平滑

· · //bspline.ThreeOrderBSplineSmooth(testpt,num2); // 三次B样条平滑

· · delete Intnum;

· ·

· ·

· · FILE \*fp\_m\_x = fopen("Bspline\_test\_x.txt", "wt");

· · FILE \*fp\_m\_y = fopen("Bspline\_test\_y.txt", "wt");

· · for (int i = 0; i < num2; i++){

· · fprintf(fp\_m\_x, "%lf\n", testpt[i].x);

· · fprintf(fp\_m\_y, "%lf\n", testpt[i].y);

· · }

· · fclose(fp\_m\_x);

· · fclose(fp\_m\_y);

· ·

· · return 0;

· · }

**附matlab绘图代码：**

· clear all;

· · clc;

· ·

· · load Bspline\_test\_x2.txt; % 导入二次B样条计算的结果

· · load Bspline\_test\_y2.txt;

· · load Bspline\_test\_x3.txt; % 导入三次B样条计算的结果

· · load Bspline\_test\_y3.txt;

· ·

· · load Bspline\_test\_smooth\_x2.txt

· · load Bspline\_test\_smooth\_y2.txt

· · load Bspline\_test\_smooth\_x3.txt

· · load Bspline\_test\_smooth\_y3.txt

· ·

· · control\_point\_x=[9.59,60.81,105.57,161.59,120.5,100.1,50.0,10.0];

· · control\_point\_y=[61.97,107.13,56.56,105.27,120.5,150.0,110.0,180.0];

· ·

· · figure;

· · h1=plot(control\_point\_x,control\_point\_y,'--ob'); hold on;

· · h2=plot(Bspline\_test\_x2,Bspline\_test\_y2,'--k','LineWidth',2);

· · h3=plot(Bspline\_test\_x3,Bspline\_test\_y3,':r','LineWidth',2);

· · h4=plot(Bspline\_test\_smooth\_x2,Bspline\_test\_smooth\_y2,'og','LineWidth',1);

· · h5=plot(Bspline\_test\_smooth\_x3,Bspline\_test\_smooth\_y3,'^b','LineWidth',1);

· ·

· · legend([h1 h2 h3 h4 h5],'控制点','二次B样条曲线','三次B样条曲线','二次B样条平滑','三次B样条平滑');

· · axis([0 200 40 200]);

· · set(gca,'FontSize',15); set(gca,'FontWeight','bold'); set(gca,'LineWidth',2);

· · set(get(gca,'XLabel'),'Fontsize',15); set(get(gca,'YLabel'),'FontSize',15);