Министерство образования и науки РФ

ФГБОУ ВПО «Тамбовский государственный технический университет»

Кафедра « »

Лабораторная работа №4

по дисциплине «Геометрическое моделирование»

Построение сплайна второй и третьей степени

Выполнил: студент гр. −31 Д. Ю.

Проверил: С. А.

Тамбов,

***Цели и задачи****.*

Выполнить работу по написанию алгоритма, реализующего построение квадратичных и кубических сплайнов, выявить недостатки подхода.

***Решение задачи****.*

Квадратичный сплайн строится по формуле 1:

1

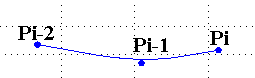
, где t изменяется от 0 до 1, P – точка (Рисунок 1)

Рисунок Индексы точек

Нюанс формулы состоит в следующих моментах:

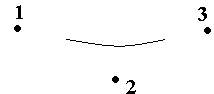
1. Алгоритм основан на обработке точек, группой по три.
2. Алгоритм из трех точек построит (Рисунок 2) одну кривую, у которой начало и конец будут серединой расстояния от точки начала до точки в середине (так же и конец кривой).

Рисунок Нюанс #2

Кубический сплайн является аналогом предыдущего сплайна с той лишь разницей, что формула 1 примет вид:

2

, формула приобрела еще одну точку, но не потеряла нюансов, описанных выше.

*В обоих подходах используется не стандартное индексирование точек, то есть самая первая точка на холсте станет не Pi, а Pi-3 (или Pi-2).*

Для построения больших кривых, точки берут со смещением на единицу (Рисунок 3)

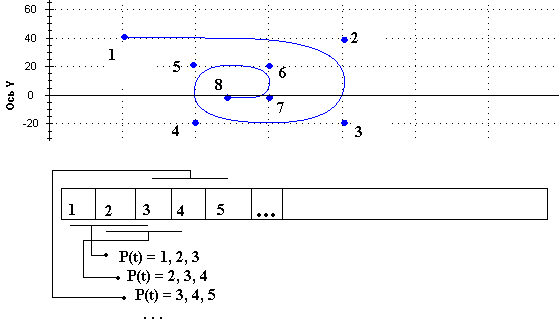
Код программы:

Рисунок Пример кривой из 8 точек и разбор по группам в теле

**internal** void btnBuild**()**

**{**

//проверить количество точек >= 3

**if** **(**\_lpointArr**.**Count **<** 3**)** **return;**

\_pointArr **=** **new** PointD**[**\_lpointArr**.**Count**];**

\_lpointArr**.**CopyTo**(**\_pointArr**);**

INum **=** \_pointArr**.**Length**;**

\_list**.**Clear**();**

#region построить график

bool flagRepitStart **=** **true;**

var tmpRes **=** **new** PointD**();**

**if** **(**\_switchFlag**)**//On

**{**//Note: Для партии по 3 точки:

//Дублировать 1 и n точку 2 раза

**for** **(**int i **=** 0**;** i **+** 2 **<** \_lpointArr**.**Count**;** i**++)**

**{**//Note: Конец - это не конец, это начало:)

PointD Pi2 **=** \_pointArr**[**i**];**

PointD Pi1 **=** \_pointArr**[**i **+** 1**];**

PointD Pi0 **=** \_pointArr**[**i **+** 2**];**

**for** **(**var t **=** .01**;** t **<=** 1.0**;** t **+=** .01**)**

**{**

**if** **(**t **+** .01 **>** 1.0 **&&** i **+** 2 **==** \_lpointArr**.**Count **-** 1**)**//Конец = 2 точки

tmpRes **=** **new** PointD**(**

.5 **\*** t **\*** t **\*** Pi0**.**X **+** **(**.75 **-** Math**.**Pow**(**t **-** .5**,** 2.0**))** **\*** Pi0**.**X **+**

.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 2.0**)** **\*** Pi2**.**X**,**

.5 **\*** t **\*** t **\*** Pi0**.**Y **+** **(**.75 **-** Math**.**Pow**(**t **-** .5**,** 2.0**))** **\*** Pi0**.**Y **+**

.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 2.0**)** **\*** Pi2**.**Y**);**

**else**

tmpRes **=** flagRepitStart

**?** **new** PointD**(**//Начало = 2 точки

.5 **\*** t **\*** t **\*** Pi0**.**X **+** **(**.75 **-** Math**.**Pow**(**t **-** .5**,** 2.0**))** **\*** Pi2**.**X **+**

.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 2.0**)** **\*** Pi2**.**X**,**

.5 **\*** t **\*** t **\*** Pi0**.**Y **+** **(**.75 **-** Math**.**Pow**(**t **-** .5**,** 2.0**))** **\*** Pi2**.**Y **+**

.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 2.0**)** **\*** Pi2**.**Y**)**

**:** **new** PointD**(**//Серединка = 3 точки

.5 **\*** t **\*** t **\*** Pi0**.**X **+** **(**.75 **-** Math**.**Pow**(**t **-** .5**,** 2.0**))** **\*** Pi1**.**X **+**

.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 2.0**)** **\*** Pi2**.**X**,**

.5 **\*** t **\*** t **\*** Pi0**.**Y **+** **(**.75 **-** Math**.**Pow**(**t **-** .5**,** 2.0**))** **\*** Pi1**.**Y **+**

.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 2.0**)** **\*** Pi2**.**Y**);**

flagRepitStart **=** **false;**

// добавим в список точку P(t)

\_list**.**Add**(**tmpRes**.**X**,** tmpRes**.**Y**);**

**}**

**}**

**}**

**else**//Off

**{**//Note: Для партии по 4 точки

//Дублировать 1 и n точку 3 раза

**for** **(**int i **=** 0**;** i **+** 3 **<** \_lpointArr**.**Count**;** i**++)**

**{**//Note: Конец - это не конец, это начало:)

PointD Pi3 **=** \_pointArr**[**i**];**

PointD Pi2 **=** \_pointArr**[**i **+** 1**];**

PointD Pi1 **=** \_pointArr**[**i **+** 2**];**

PointD Pi0 **=** \_pointArr**[**i **+** 3**];**

**if** **(**i **==** 0**)**

**{**//Псевдо равные ячейки для начала массива

Pi2 **=** \_pointArr**[**i**];**

Pi1 **=** \_pointArr**[**i **+** 1**];**

Pi0 **=** \_pointArr**[**i **+** 2**];**

**}**

**if** **(**i **+** 3 **==** \_lpointArr**.**Count **-** 1 **&&** **!**flagRepitStart**)**

**{**//Псевдо равные ячейки для конца массива

//Pi3 = \_pointArr[i + 0];

//Pi2 = \_pointArr[i + 1];

//Pi1 = \_pointArr[i + 2];

//Pi0 = \_pointArr[i + 2];

**}**

**for** **(**double t **=** .01**,** itr **=** 0.0**;** t **<=** 1.0**;** t **+=** .01**,** itr **+=** 1.0**)**

**{**

**if** **(**t **+** .01 **>** 1.0 **&&** i **+** 3 **==** \_lpointArr**.**Count **-** 1**)**//Конец = 2 точки

tmpRes **=** **new** PointD**(**

1.0 **/** 6.0 **\*** t **\*** t **\*** t **\*** Pi0**.**X **+**

**(**2.0 **/** 3.0 **-** 0.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 3.0**)** **-** Math**.**Pow**(**t **-** 1.0**,** 2.0**))** **\*** Pi0**.**X **+**

**(**2.0 **/** 3.0 **+** 0.5 **\*** t **\*** t **\*** t **-** t **\*** t**)** **\*** Pi0**.**X **+** 1.0 **/** 6.0 **\*** Math**.**Pow**(**1.0 **-** t**,** 3.0**)** **\*** Pi3**.**X**,**

1.0 **/** 6.0 **\*** t **\*** t **\*** t **\*** Pi0**.**Y **+**

**(**2.0 **/** 3.0 **-** 0.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 3.0**)** **-** Math**.**Pow**(**t **-** 1.0**,** 2.0**))** **\*** Pi0**.**Y **+**

**(**2.0 **/** 3.0 **+** 0.5 **\*** t **\*** t **\*** t **-** t **\*** t**)** **\*** Pi0**.**Y **+** 1.0 **/** 6.0 **\*** Math**.**Pow**(**1.0 **-** t**,** 3.0**)** **\*** Pi3**.**Y**);**

**else**

tmpRes **=** flagRepitStart

**?** **new** PointD**(**

1.0 **/** 6.0 **\*** t **\*** t **\*** t **\*** Pi0**.**X **+**

**(**2.0 **/** 3.0 **-** 0.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 3.0**)** **-** Math**.**Pow**(**t **-** 1.0**,** 2.0**))** **\*** Pi3**.**X **+**

**(**2.0 **/** 3.0 **+** 0.5 **\*** t **\*** t **\*** t **-** t **\*** t**)** **\*** Pi3**.**X **+** 1.0 **/** 6.0 **\*** Math**.**Pow**(**1.0 **-** t**,** 3.0**)** **\*** Pi3**.**X**,**

1.0 **/** 6.0 **\*** t **\*** t **\*** t **\*** Pi0**.**Y **+**

**(**2.0 **/** 3.0 **-** 0.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 3.0**)** **-** Math**.**Pow**(**t **-** 1.0**,** 2.0**))** **\*** Pi3**.**Y **+**

**(**2.0 **/** 3.0 **+** 0.5 **\*** t **\*** t **\*** t **-** t **\*** t**)** **\*** Pi3**.**Y **+** 1.0 **/** 6.0 **\*** Math**.**Pow**(**1.0 **-** t**,** 3.0**)** **\*** Pi3**.**Y**)**

**:** **new** PointD**(**

1.0 **/** 6.0 **\*** t **\*** t **\*** t **\*** Pi0**.**X **+**

**(**2.0 **/** 3.0 **-** 0.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 3.0**)** **-** Math**.**Pow**(**t **-** 1.0**,** 2.0**))** **\*** Pi1**.**X **+**

**(**2.0 **/** 3.0 **+** 0.5 **\*** t **\*** t **\*** t **-** t **\*** t**)** **\*** Pi2**.**X **+** 1.0 **/** 6.0 **\*** Math**.**Pow**(**1.0 **-** t**,** 3.0**)** **\*** Pi3**.**X**,**

1.0 **/** 6.0 **\*** t **\*** t **\*** t **\*** Pi0**.**Y **+**

**(**2.0 **/** 3.0 **-** 0.5 **\*** Math**.**Pow**(**t **-** 1.0**,** 3.0**)** **-** Math**.**Pow**(**t **-** 1.0**,** 2.0**))** **\*** Pi1**.**Y **+**

**(**2.0 **/** 3.0 **+** 0.5 **\*** t **\*** t **\*** t **-** t **\*** t**)** **\*** Pi2**.**Y **+** 1.0 **/** 6.0 **\*** Math**.**Pow**(**1.0 **-** t**,** 3.0**)** **\*** Pi3**.**Y**);**

flagRepitStart **=** **false;**

// добавим в список точку P(t)

\_list**.**Add**(**tmpRes**.**X**,** tmpRes**.**Y**);**

**}**

**}**

**}**

#endregion

DrawGraph**();**

**}**

Для того, чтобы кривая выходила из первой точки, необходимо начальную точку дублировать один раз (либо два), так же поступают с концом. В коде это выражено формулой с повторенной точкой старта (либо конца).

К сожалению подход, который работает в формуле 1 – не работает во втором случае с 4 точками. Из за этого при построении кривых заметен эффект ломки отдельных участков.

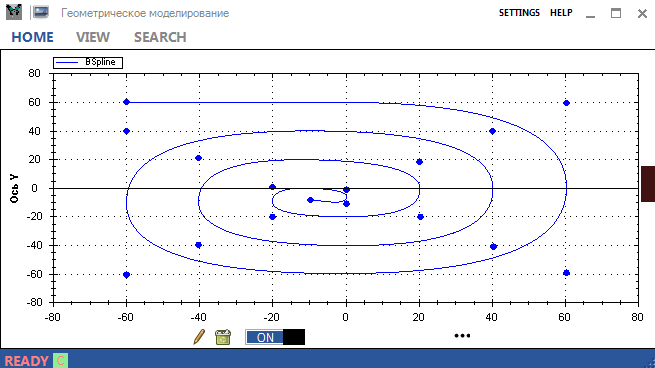
Пример построения по формуле 1 на рисунке 4.

Рисунок Квадратичный сплайн

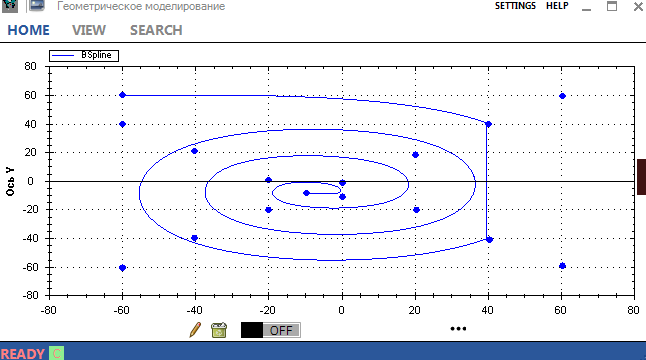
Пример по формуле 2 (Рисунок 5):

Рисунок Кубический сплайн

***Вывод.***

Сплайн, выполненный по формуле 1 идеален. То же мнение можно (наверное) сказать о кубических сплайнах и формуле 2, однако искусственно повторить подмену начальных и конечных значений мне не удалось. Остальные варианты дублирования мною не рассматривались в виду их банальности исполнения.