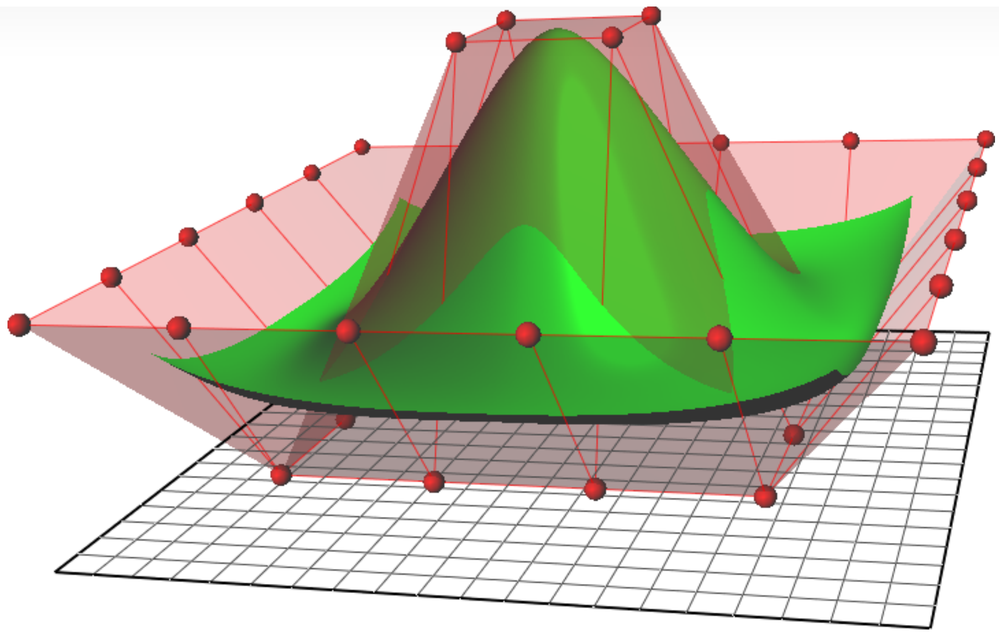


曲线杂谈（三）：青出于蓝的B-Spline和NURBS

EC果酱技术团队
来自EverCraft的大果宫酱技术团队

作者：螺蛳道长

摘要

书接上回，正所谓人无完人，金无赤足，贝塞尔曲线纵然有千般优点，也不免在有些场景下心有余而力不足。

正如百年前“物理学大厦上飘过的两朵乌云“催生了几乎整个现代物理学的内容，贝塞尔曲线也是图形学发展史上用来引玉效果极好的一块砖儿～

毕竟，人类向真理之巅攀登的过程就是边努力填坑再不断挖坑的过程。B-Spline以及各种衍生的数学工具就是极好的一个例子。

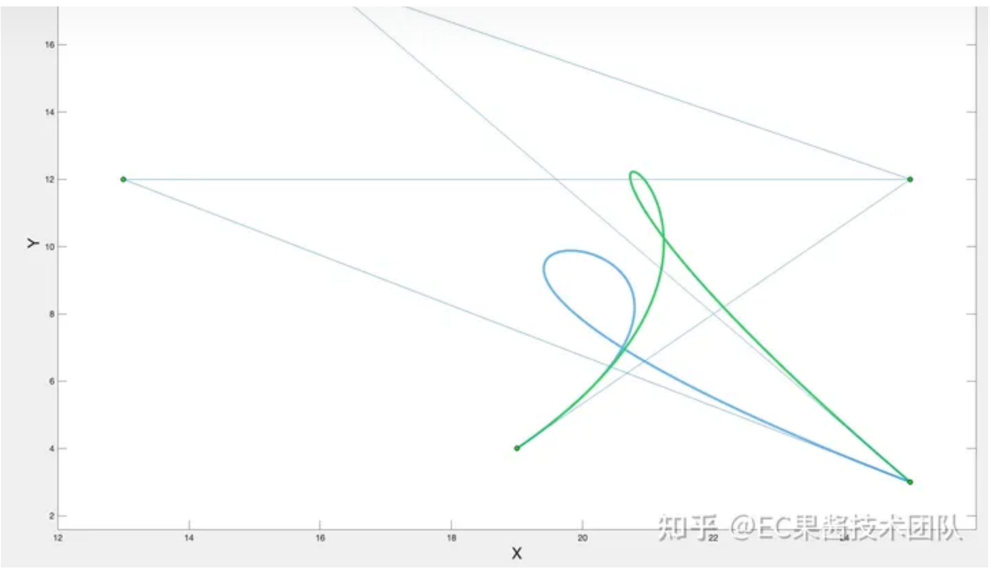
站在巨人的肩膀上的感觉大概像这样～

Bezier曲线的缺点

在正式开始介绍B-Spline之前，咱们先来聊聊贝塞尔曲线的缺陷。

- 任何一控制点的变化会引起曲线上所有点的改变，牵一发而动全身。所以，贝塞尔曲线是无法针对局部进行编辑的。

蓝色曲线，控制点为[25,3）（13,12）（25,12）（19,4）。我们只需要轻轻地将第二个控制点改为（16，18），得到绿色的贝塞尔曲线，可以清晰地看到两者轨迹的迥异。



- 样条阶数和控制点个数成正比，随着控制点的增加，计算量会快速增加。所以随着Bezier曲线的阶数增加，对于**曲线形状的控制将变得困难**。主流的图形套件所提供的贝塞尔曲线接口很少超过3阶。
- 确定了多边形的顶点数（m个），也就决定了所定义的Bezier曲线的阶次（m-1次），控制的**自由度较少。**

这些缺点决定了，如果我们需要用贝塞尔曲线作为唯一工具去绘制一个较为复杂的图形，为了保证可编辑性和渲染效率，只能将整个结构拆分成很多小段用低阶数(主要是2阶）的贝塞尔曲线进行拟合，再进行拼接。

不能说不优雅，简直是超级超级巨大巨大的不优雅～

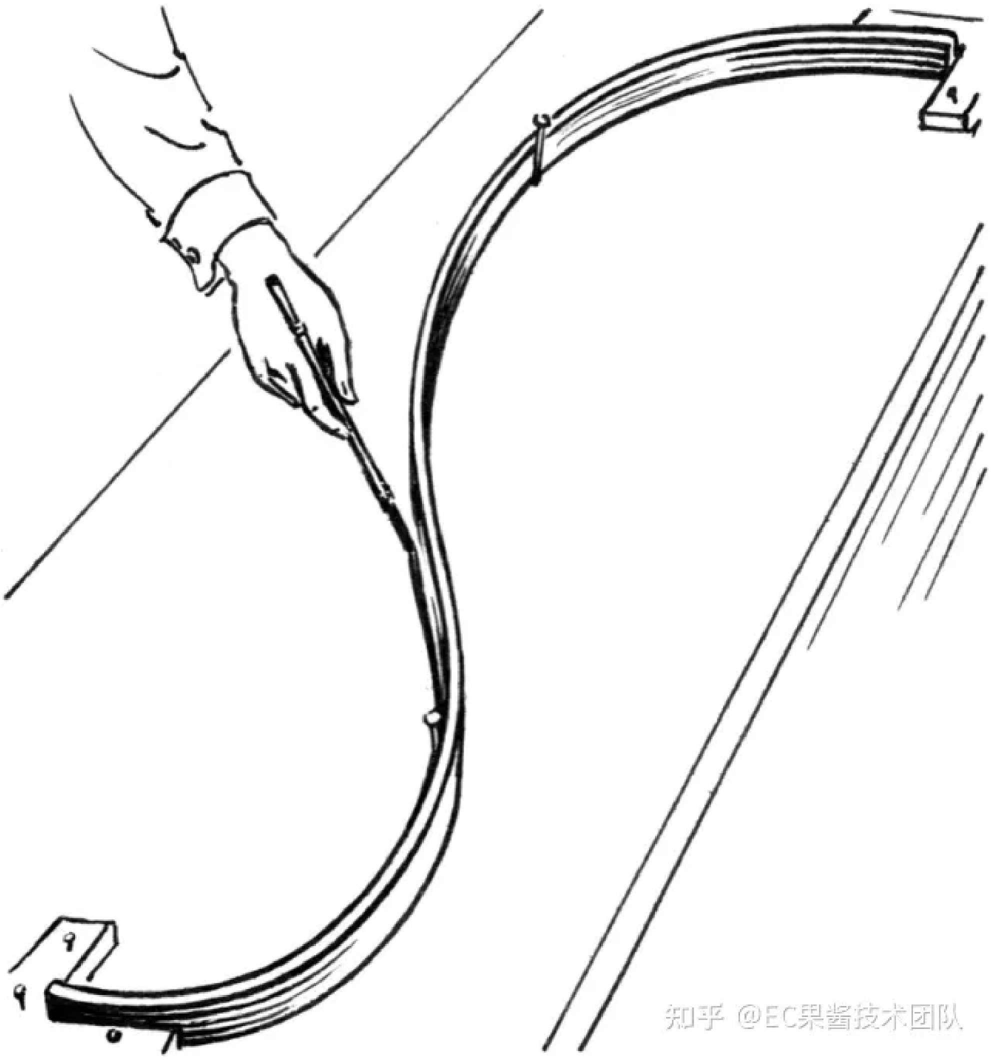
什么叫样条?

在计算机处理之前，设计一般是通过在纸上的手绘完成，该过程需要借助许多绘画工具。

样条（Spline）是早期的工人和技术员（特别是造船业），用来画光滑形状的工具，它是一根柔软但有弹性的长条物，有些像尺子。

使用方法很简单，将两端和几个点用钉子固定之后，便可以产生顺滑的曲线。

但是仍然存在许多不规则形状的曲线，例如船艏，是无法通过这些工具绘制的。虽然可以通过在手绘板上徒手完成，但由于尺寸原因这种手稿无法在实际生产中得到应用。



B样条相比贝塞尔曲升级了什么?

B样条（B-Spline）是对贝塞尔曲线的一般化，它的诞生时为了解决Bezier在应用过程中暴露出的一系列缺陷。

为了实现对曲线更好的控制力，B-Spline对控制点和阶数做了解耦，两者再无直接关系，并且引入了节点向量（Knots）。相比于贝塞尔曲线只有控制点一个自由度，B-spline拥有的自由度多达三个。

什么是节点向量（Knots）

B-Spline需要定义曲线的节点向量U，它可以对应到Bezier曲线的参数u。

只是Bezier曲线的参数u是 $\in [0,1]$ 的实数。而B-Spline对它进行了扩展，把 $[U_{min}, U_{max}]$ 区间划分成多个非减的实数段。

> $U = \{u_0, u_1, ..u_m\}$

虽然 B-spline 比 Bezier 曲线会玩儿，但节点向量 U 的取值也要遵循基本法。最重要的就是其元素个数 (m+1) 和曲线阶数 p 、控制点个数n满足：

> $m = n + p + 1$

- 重复性（multiplicity）

如果U向量中存在k个相等的元素 u_x ，则 B-spline 在 x 处具有 k 的重复度。（敲黑板！！！）这个性质在后面讲clamped的时候会用到～

- 均匀性(uniformity)

如果U每段的距离（span）是相等，那么这个B-Spline就被称为均匀B样条：

> $u_1 - u_0 = u_2 - u_1 = ... = u_n - u_{n-1}$

均匀性也可以只在局部遵守，比如 $U = \{0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1\}$,第 4 到第 8 个节点间保持 0.2 的差值，而首尾则有着 3 重复度。

这样的曲线也被称为准均匀（quasi-uniform）B样条。（敲黑板x2！！），这个性质在后面也会用到～

B-Spline公式推导

废话不多，先码公式～

和贝塞尔曲线一样，B-Spline的基函数也可以用递归的方法展开以降低阶数：

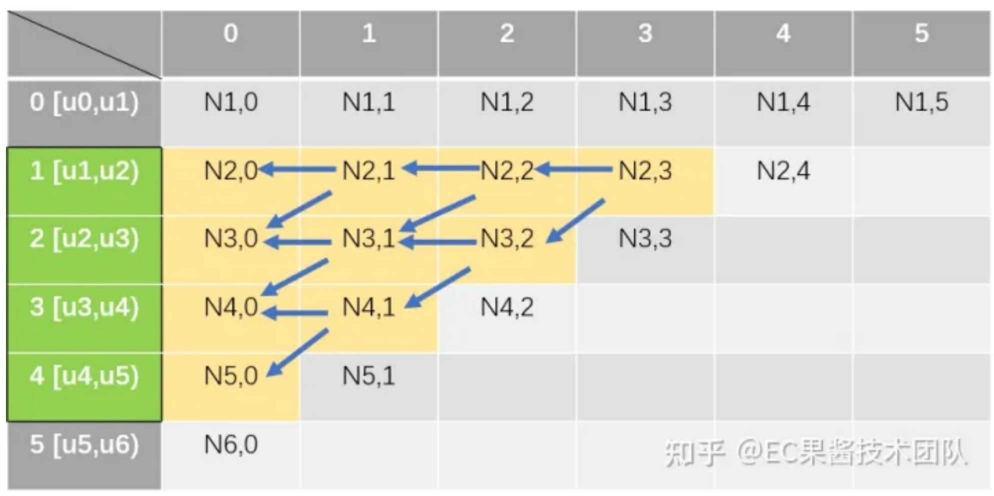
$$N_{i,p}(u) = f_{i,p}(u)N_{i,p-1}(u) + g_{i+1,p}(u)N_{i+1,p-1}$$
$$f_{i,p}(u) = \frac{u-u_i}{u_{i+p}-u_i}$$
$$g_{i,p}(u) = 1 - f_{i,p}(u) = \frac{u_{i+p}-u}{u_{i+p}-u_i}$$

对于 $N_{i,0}$ 需要特殊定义一下：

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u \in [u_i, u_{i+1}) \\ 0 & \text{if } u \notin [u_i, u_{i+1}) \end{cases}$$

从递推公式中，我们可以总结出，对于 $N_{i,n}$，$[N_{i,n}, N_{i+n,0}]$ 必然存在非零元素，也就是说小于 i 和大于 $i+n$ 的节点都不需要纳入基函数的计算了，这样就确定了内层循环的范围：

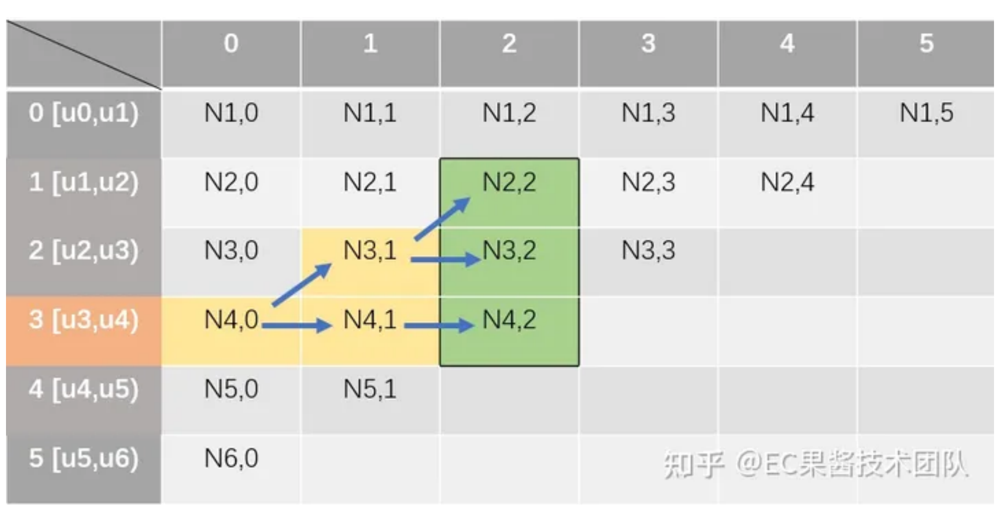
$$N_{i,p}(u) = \begin{cases} \geq 0 & \text{if } u \in [u_i, u_{i+p+1}) \\ = 0 & \text{if } u \notin [u_i, u_{i+p+1}) \end{cases}$$



	0	1	2	3	4	5
0 [u0,u1)	N1,0	N1,1	N1,2	N1,3	N1,4	N1,5
1 [u1,u2)	N2,0	N2,1	N2,2	N2,3	N2,4	
2 [u2,u3)	N3,0	N3,1	N3,2	N3,3		
3 [u3,u4)	N4,0	N4,1	N4,2			
4 [u4,u5)	N5,0	N5,1				
5 [u5,u6)	N6,0					

那么反过来，若u在节点向量的位置和曲线的阶数已经确定，我们不难得出相应阶数下（第p列）可能非0的基函数。

$$N_{x,p} \begin{cases} \geq 0 & \text{if } u \in [u_i, u_{i+1}) \quad and \quad x \in [i-p, .., i] \\ = 0 & otherwise \end{cases}$$



	0	1	2	3	4	5
0 [u0,u1)	N1,0	N1,1	N1,2	N1,3	N1,4	N1,5
1 [u1,u2)	N2,0	N2,1	N2,2	N2,3	N2,4	
2 [u2,u3)	N3,0	N3,1	N3,2	N3,3		
3 [u3,u4)	N4,0	N4,1	N4,2			
4 [u4,u5)	N5,0	N5,1				
5 [u5,u6)	N6,0					

这样对于任意的 u，我们可以通过它在节点向量空间中所处的区间和阶数，确定外层循环的范围 $[i-p, i]$：

$$C(u) = \sum_{i=\{x|N_{x,p}(u)\geq 0\}} N_{i,p}(u)P_i$$

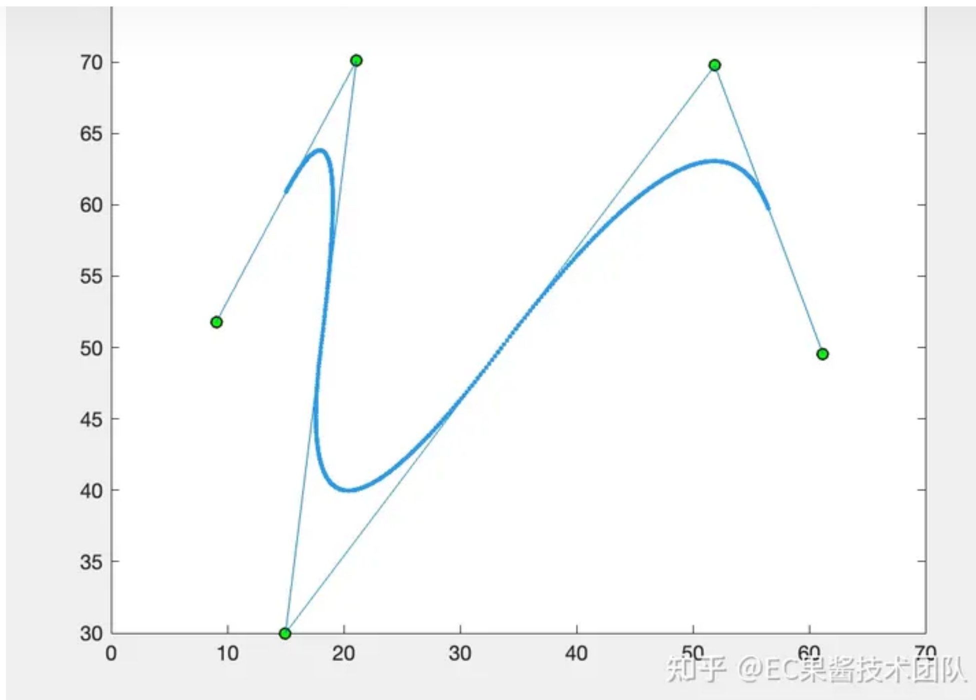
clamped

和贝塞尔曲线不同的是，B-Spline不会天然经过首尾两个控制点。

这就给拼接带来了一些麻烦。

譬如像这样～

目录 收起



但是没关系，不会天然经过，我们就创造机会让他们经过，这样的场景又被成为**clamped**。

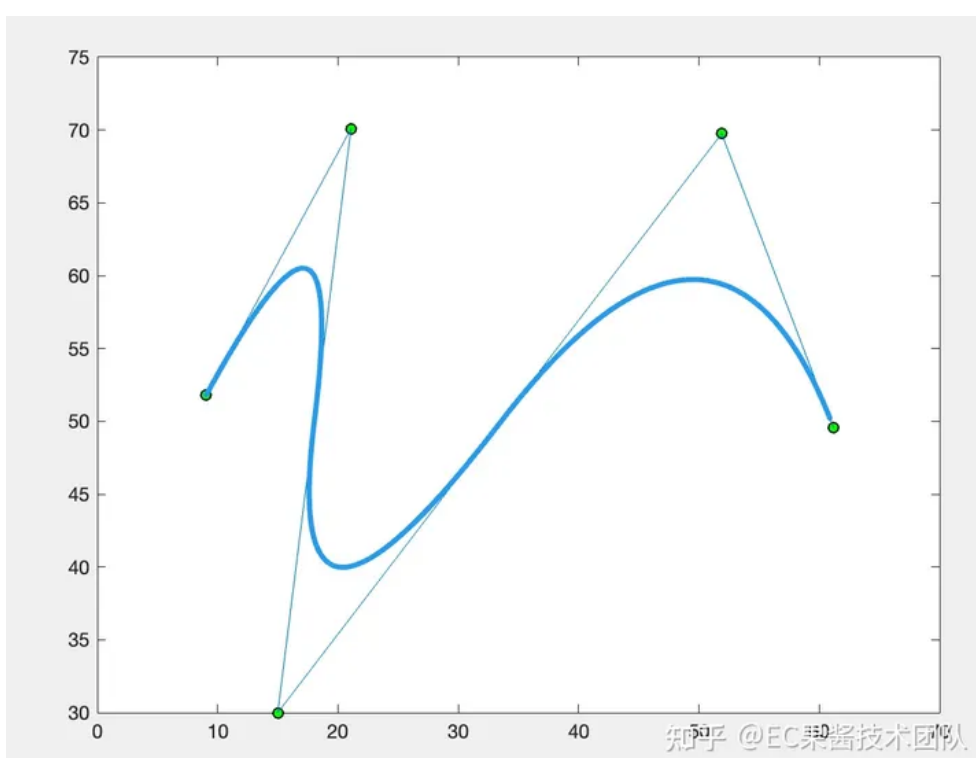
与之相对，即曲线不经过首末控制点的场景，又被称为**periodic**。

为了达成clamped的效果，在需要对节点向量进行设计。对于阶数为p的样条，首个节点和末个节点均应具有p+1的重复度（multiplicity）

> $U = \{a\ ,a\ ,\dots,a,u_{p+1},\dots,u_{m-p-1},b,b,..b\}$

U向量的前 $p+1$ 个元素都是实数 a ，后 $p+1$ 个元素都是 b 。如非特别强调，则 $a=0$ ，$b=1$ 。

So easy~



NURBS

这名字看上去非常狂拽酷炫有木有~但是把每个字拆开来，

- NU-nouniform，即非均匀，指有效节点向量区间距离非减不相等的。
- R-rational，有理，指权值（weight）各不相等。 权值代表各控制点对某一个拟合点的影响力，这里也可以发现普通的样条来说，各控制点对于同一个拟合点的影响力是相等的。
- BS-BSpline，B样条。

全称合起来就是控制点影响力和节点向量都参差不齐的B样条。是不是觉得也就那么回事儿？

NURBS的发展始于1950年代，当时的工程师们正为缺少一种数学工具能在设计中精确地定义自由曲面而苦恼，为此他们在很长的时间里只能依赖于实体模型。

受大西洋彼岸的贝塞尔等人启发，现任职于solid Edge的Dr.Ken.Versprille于1975年在其博士学位论文 **Computer-Aided Design Applications of the Rational B-Spline Approximation form**中首次提出NURBS的完整推导。

老爷子本尊，膜拜~

1991年，国际标准化组织（ISO）颁布的工业产品数据交换标准STEP中，把NURBS作为定义工业产品几何形状的唯一数学方法。

1992年，国际标准化组织又将NURBS纳入到规定独立于设备的交互图形编程接口的国际标准PHIGS（程序员层次交互图形系统）中，作为PHIGS Plus的扩充部分。

Bezier、有理Bezier、均匀B样条和非均匀B样条都被统一到NURBS中。

NURBS的公式也非常简单，就是在B-spline的基础上加入了一个权重w而已。

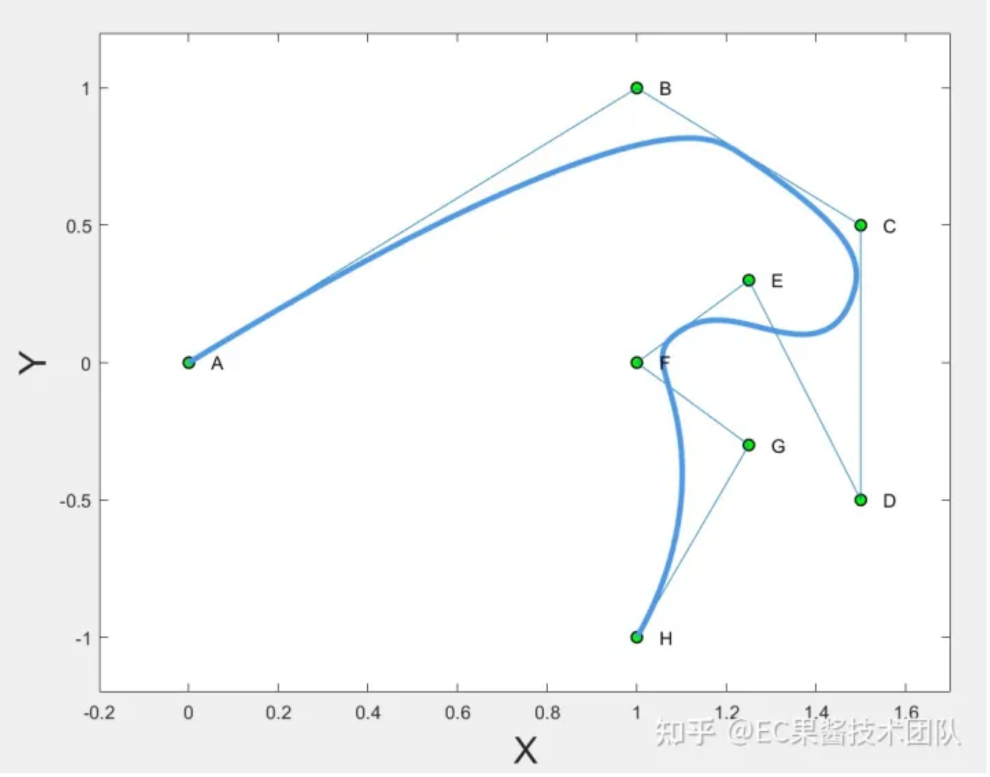
> $C(u) = \sum_{i=1}^{k}(\frac{N_{i,p}(u)w_i}{\sum_{i=1}^{k} N_{i,p}(u)w_i})P_i$
>
> $If\ w_1 = w_2 = w_3 = .. = w_i \neq 0$

扩展成曲面形式，

$$C(u,v)=\sum_{i=1}^{k}\sum_{j=1}^{l}N_{i,j}(u,v)P_{i,j}$$

其中

$$R_{i,n}=\frac{N_{i,n}(u)N_{j,m}(v)w_{i,j}}{\sum_{p=1}^{k}\sum_{q=1}^{l}N_{p,n}(u)N_{q,m}(v)w_{p,q}}$$



Source Code

最后照例是大家最爱的“show me your code”环节，今天暂时只提供曲线部分的代码。至于曲面的部分，挖个坑，什么时候公布就看什么时候点赞过百咯～

所以记得**点赞+关注**！！！

- 主函数

```
function [C]=BSpline(varargin)
    narginchk(1,7);
    ctrls=varargin{1};
    k=varargin{2};
    flag=varargin{3};
    drawable=varargin{4};

    weights=ones(size(ctrls,1),1)*(1/size(ctrls,1));
    n=size(ctrls,1)-1;
    switch flag
        case 1
            NodeVector = linspace(0, 1, n+k+2); % 均匀B样条的节点矢量
        case 2
            NodeVector = U_quasi_uniform(n, k); % 准均匀B样条的节点矢量
        case 3
            NodeVector = U_piecewise_Bezier(n, k); % 分段Bezier曲线的节点矢量
        case 4
            NodeVector=varargin{5}; %非均匀有理B样条 NURBS
            weights=varargin{6};
        otherwise
            fprintf('error!\n');
    end

    C=BSpline_gen(ctrls,n,k,NodeVector,0.001,weights);

  %绘制结果
    if drawable
        DisSpline(ctrls, n,C);
    end

end
```

- B-spline Generator

```
function [C]=BSpline_gen(vertices,n,k,NodeVector,accuracy,weights)

      %n=size(veritces,1);
```

```
            offset=0;
        end

        count=1;
        C=zeros(2,int8((1-2*offset)/accuracy));
        for u =0+offset : accuracy :1- offset-accuracy
            Dom=0;
             for i = 0 : 1 : n
                 weights(i+1,1)
                Nik_w(1, i+1) = BaseFunction(i, k , u, NodeVector)*weights(i+1,1);
                Dom=Dom+Nik_w(1,1+i);
             end
             C(:,count)=  (Nik_w/Dom)*vertices;
             count=count+1;
        end

end
```

- 基函数计算函数

```
% 计算基函数Ni,k(u),NodeVector为节点向量
function Nik_u = BaseFunction(i, k , u, NodeVector)

if k == 0       % 0次B样条
    if (u >= NodeVector(i+1)) && (u < NodeVector(i+2))
        Nik_u = 1.0;
    else
        Nik_u = 0.0;
    end
else
    % 支撑区间的长度
    Length1 = NodeVector(i+k+1) - NodeVector(i+1);
    Length2 = NodeVector(i+k+2) - NodeVector(i+2);

     % 规定0/0 = 0
    if Length1 == 0.0
        Length1 = 1.0;
    end
    if Length2 == 0.0
        Length2 = 1.0;
    end

    %递归计算
    Nik_u = (u - NodeVector(i+1)) / Length1 * BaseFunction(i, k-1, u, NodeVector) ...
        + (NodeVector(i+k+2) - u) / Length2 * BaseFunction(i+1, k-1, u, NodeVector);
end
```

- 分段Bezier-节点向量生成函数

```
function NodeVector = U_piecewise_Bezier(n, k)
% 分段Bezier曲线的节点向量计算，共n+1个控制顶点，k次B样条
% 分段Bezier端节点重复度为k+1，内间节点重复度为k,且满足n/k为正整数

if ~mod(n, k) && (~mod(k, 1) && k>=1)   % 满足n是k的整数倍且k为正整数
    NodeVector = zeros(1, n+k+2);   % 节点矢量长度为n+k+2（控制点+阶数+1）
    NodeVector(1, n+2 : n+k+2) = ones(1, k+1);  % 右端节点置1

    piecewise = n / k;      % 设定内节点的值
    Flg = 0;
    if piecewise > 1
        for i = 2 : piecewise
            for j = 1 : k
                NodeVector(1, k+1 + Flg*k+j) = (i-1)/piecewise;
            end
            Flg = Flg + 1;
        end
    end

else
    fprintf('error!\n');
end
```

- 准均匀-节点向量生成函数

```
function NodeVector = U_quasi_uniform(n, k)
% 准均匀B样条的节点向量计算，共n+1个控制顶点，k次B样条
NodeVector = zeros(1, n+k+2);
piecewise = n - k + 1;       % 有效曲线的段数 起始的k+1段都是0，结尾k+1段都是1（重复度k+1）
if piecewise == 1       % 只有一段曲线时，n = k
    for i = n+2 : n+k+2
        NodeVector(1, i) = 1;
    end
else
    flag = 1;       % 不止一段曲线时
    while flag ~= piecewise
```

```
end
end
```

Demo1-准均匀B样条

```
clear all;
%控制点
P = [0 0; 1 1; 1.5 0.5; 1.5 -0.5; 1.25 0.3; 1 0; 1.25 -0.3; 1 -1];

%阶数
k=3;

%type
type=2;%准均匀B样条

%drawble
drawable=true;

%绘制spline曲线
BSpline(P,k,type,drawable);
hold on

%给控制点按顺序打上标记
hold on
for i=1:size(P,1)
    text(P(i,1)+0.05,P(i,2),char(65+i-1));
end

%设置坐标轴标签和范围
figure(1)
xlabel('X','FontSize',20);
ylabel('Y','FontSize',20);
xlim([min(P(:,1))-0.2 max(P(:,1))+0.2]);
ylim([min(P(:,2))-0.2 max(P(:,2))+0.2]);
```

Demo2-NURBS

```
clear all;
%控制点
P = [0 0; 1 1; 1.5 0.5; 1.5 -0.5; 1.25 0.3; 1 0; 1.25 -0.3; 1 -1];

%阶数
k=3;

%knots
NodeVector=U_disuniform(size(P,1)-1,k);

%type
type=4;%非均匀有理B样条 NURBS

%weights
weights=rand(size(P,1),1);
weights=weights/sum(weights);

%drawble
drawable=true;

%绘制spline曲线
BSpline(P,k,type,drawable,NodeVector,weights);
hold on

%给控制点按顺序打上标记
hold on
for i=1:size(P,1)
    text(P(i,1)+0.05,P(i,2),char(65+i-1));
end

%设置坐标轴标签和范围
figure(1)
xlabel('X','FontSize',20);
ylabel('Y','FontSize',20);
xlim([min(P(:,1))-0.2 max(P(:,1))+0.2]);
ylim([min(P(:,2))-0.2 max(P(:,2))+0.2]);
```

除了weight，knot向量也使用了随机数进行构建以满足非均匀的特性

```
function NodeVector = U_disuniform(n,k)
% 随机非均匀B样条的节点向量，共n+1个控制顶点，k次B样条
NodeVector = zeros(1, n+k+2);
piecewise = n - k + 1;       % 有效曲线的段数 起始的k+1段都是0，结尾k+1段都是1（重复度k+1）
steps=rand(piecewise,1);
steps=steps/sum(steps);
if piecewise == 1       % 只有一段曲线时，n = k
    for i = n+2 : n+k+2
        NodeVector(1, i) = 1;
```

```
            NodeVector(1, k+1+flag) = NodeVector(1, k + flag) + steps(flag);
            flag = flag + 1;
        end
        NodeVector(1, n+2 : n+k+2) = 1;
    end
end
```

Reference

1. *Les Piegl,Way Tiller-The NURBS Book 2nd*
2. Just how did NURBS come to be? - by Dr. Ken Versprille | Solid Edge
3. wiki-非均匀有理B样条

EC果酱技术团队：曲线杂谈（二）：Bezier曲线的特殊性质
34 赞同 · 3 评论 文章

EC果酱技术团队：曲线杂谈（一）：大话Bezier曲线
36 赞同 · 2 评论 文章

编辑于 2022-04-20 16:11

曲线图 样条 nurbs

写下你的评论...

15 条评论 默认 最新

Fever
BSpline函数定义脚本中好像出错了😓第28行出现了一个DisSpline函数，但是并没有被定义啊
2022-11-21 回复 1

王不留行
那个缺失的函数DisSpline，是用来画图的，可以简单的写成这种。

function DisSpline(ctrls, n,C)

plot(C(1,:),C(2,:),LineWidth=1.5)
hold on
plot(ctrls(:,1),ctrls(:,2),'-*')

end
04-21 回复 1

行者
写得蛮好的👍
2022-06-28 回复 1

EC果酱技术团队 作者
嘿嘿，以后咱们会再接再厉的

2022-07-03 回复 喜欢

鳄鱼爱爬树
“B-Spline需要定义曲线的节点向量U，它可以对应到Bezier曲线的参数u。”
看了之前的两篇文章，没找到 Bezier 曲线的参数 u 啊，不知道这个参数 u 指的是什么？
04-30 回复 喜欢

meridjan
大概是t
07-06 回复 1

marcuswu
开头提到的贝塞尔曲线参数u是什么？我在前几篇文章里没找着
03-06 回复 喜欢

多大第一骚
呜呜呜终于搞懂了权重是干啥的了，老师的课件里莫名其妙就冒出来这个玩意儿。话说大佬知不知道什么是通过最小化代数距离和几何距离绘制B样条是什么意思呀，大作业里这俩东西根本搞不懂。
2022-12-15 回复 喜欢

wrcn