Universidade Federal de Santa Maria
Departamento de Computação Aplicada
Prof. Cesar Tadeu Pozzer
Disciplina: Computação Gráfica
pozzer@inf.ufsm.br
12/06/2022

Curvas

Usos

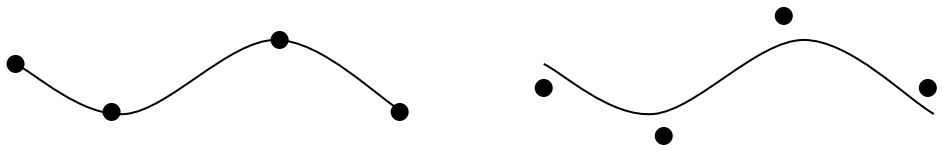
- Desenho de curvas e superfícies
- Especificação de caminhos para animação e movimentação de câmera
- CAD: Modelagem de carroceria de veículos (Bézier – Renault (1960), Casteljau – Citroen (1960)), aviões, etc.

As curvas mais conhecidas são:

- Bézier
- B-Splines
- Splines
- NURBS
- Hermite

Tipos:

- Interpolação
  - A curva passa sobre os pontos de controle
  - Usado em digitalização, refinamento de heightmaps, etc.
- Aproximação
  - A curva aproxima os pontos de controle
  - Usando na modelagem de objetos



Representação de curvas

1. Curvas são **representadas por partes (segmentos de curva)** através de polinômios de grau baixo (***piecewise polynomial*** *function: piecewise function is a function defined by multiple sub-functions, each sub-function applying to a certain interval of the main function's domain, a sub-domain*. Como exemplo, $f(x) = |x| = \begin{cases} -x, & \text{se } x < 0 \\ x, & \text{se } x \geq 0 \end{cases}$. Fonte: Wikipedia).

   - Algumas curvas **não podem ser facilmente** descritas por expressões analíticas em toda sua extensão. Assim utilizam-se conjuntos de (segmentos de) curvas, unidas pelas extremidades.

2. Utilizam-se geralmente polinômios de **grau 3**

   - Quanto maior for o grau dos polinômios, mais complexos são os cálculos, além de apresentar outros problemas como instabilidade numérica.
   - Polinômios de grau 3 são flexíveis e suprem a maioria dos requisitos de aplicações práticas
   - Representam espaço tridimensional, ou seja, **não são planares**.
   - Curvas quádráticas são funções de 3 pontos de controle e em um espaço 3D estão confinadas a um plano, e somente tem aplicação prática em aplicações 2D.

3. Cada segmento de curva é definido por um **conjunto de pontos discretos** (pontos de controle – *control points*), juntamente com um **conjunto de funções básicas** (ou funções que combinam a influência dos pontos – *blending functions*).

    - Para polinômios de grau 3, utilizam-se 4 pontos de controle (P_0, P_1, P_2 e P_3), e 4 funções bases (com representação paramétrica).
    - As funções de *blending* são usadas para gerar a curva a partir dos pontos de controle.

4. Utilizam-se **funções paramétricas**

    - As funções (geralmente cúbicas) são representadas na forma paramétrica

$$x = x(t)$$
$$y = y(t)$$
$$P(t) = (x(t), y(t))$$

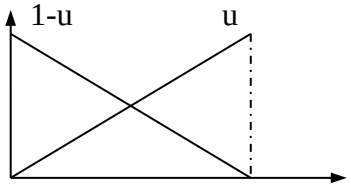
      Para 3D, basta adicionar mais uma coordenada

$$P(t) = (x(t), y(t), z(t))$$

    - Os extremos e o comprimento da curva são fixos pelo intervalo de variação do parâmetro, freqüentemente normalizado para $0 \le t \le 1$.

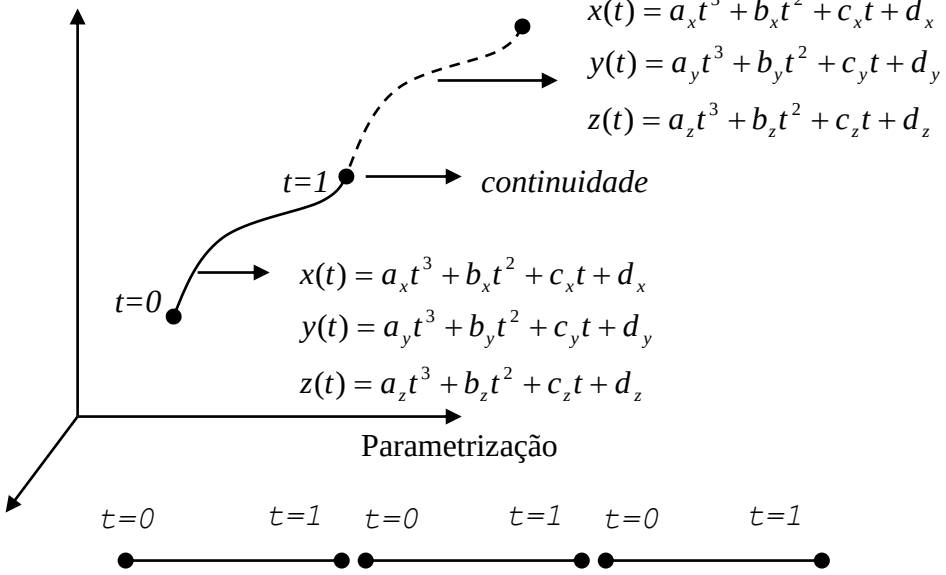
    - **Vantagens**:

        - Pontos na curva podem ser **computados sequencialmente**, ao invés de ter-se que calcular sistemas de equações não lineares para cada ponto em uma representação implícita
        - Curvas paramétricas são **facilmente transformáveis** (Transformações lineares), visto que são definidas independente do sistema de coordenadas
        - Geralmente aplicações querem **curvas/superfícies** complexas que não podem ser descritas por simples funções. Polinômios de alto grau podem representar curvas complexas mas requerem:
            - um grande número de coeficientes,
            - gastam maior tempo de processamento; e
            - podem introduzir oscilações não desejadas na curva.
        - Função de grau 1



$$\vec{P}(u) = (1-u)\vec{P}_0 + u\,\vec{P}_1$$

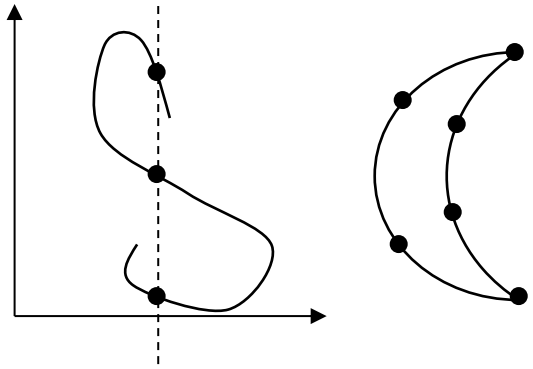
        - Função de grau 3



5. Para que a união das curvas seja feita de forma adequada, em muitos casos é necessário que a curva resultante da união tenha **curvatura contínua (restrições de continuidade)**.

Requisitos

- Controle local: Alterando-se um ponto de controle (*control point*) altera-se a curva apenas na proximidade do ponto;
- Independente do sistema de coordenadas.
    - Isso porque cada ponto da curva é determinado por um único parâmetro *t* (característica de funções paramétricas).
    - Facilita a aplicação de transformações geométricas como rotação, translação, escala.
- Valores múltiplos. Para um mesmo valor de x, pode-se ter vários valores de y.
- Continuidade variável
- Versatilidade: possibilidade de fazer qualquer tipo de curva.

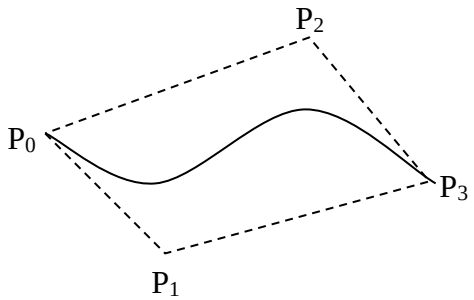


Processo interativo para geração de curvas

- Criar pontos de controle
- Visualizar a curva
- Reposicionar pontos

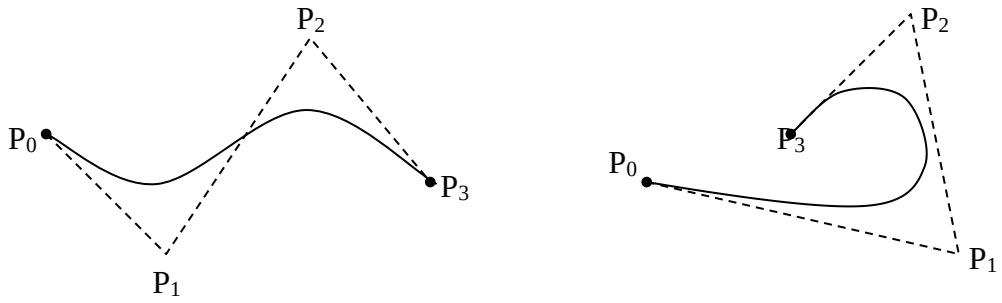
Fecho Convexo (*Convex Hull*)

- Polígono que engloba todos os pontos de controle
- Fisicamente pode ser visto como um elástico que toca os pontos de controle. Um ponto de controle está dentro desta área ou no perímetro.
- Alguns tipos de curvas (como a de Bézier e Splines) estão sempre dentro do fecho convexo.



Grafo de Controle (*Control Graph*)

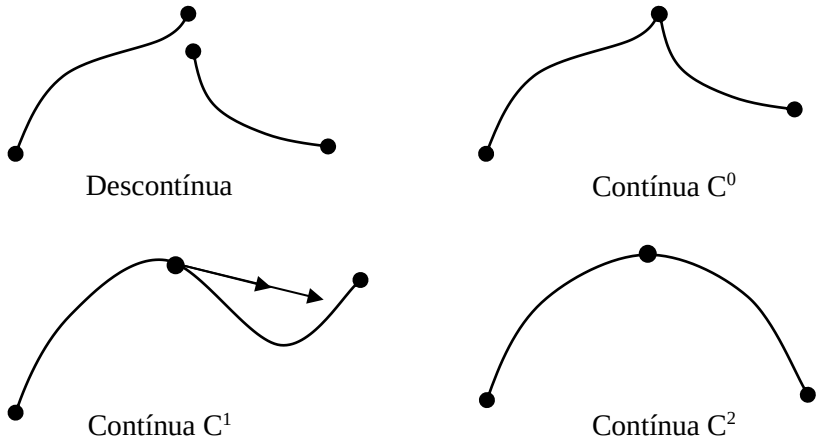
Segmentos de linhas que conectam os pontos de controle segundo a ordem que foram especificados. Auxilia o usuário na compreensão da curva gerada



Continuidade

Como as curvas são compostas pela junção de pequenas curvas, deve-se observar a **continuidade paramétrica** da junção entre cada curva.

- C^0 – duas curvas se encontram.
- C^1 – As duas curvas devem ter tangentes comuns no ponto de junção (mesma direção e comprimento). Em outras palavras, a primeira derivada paramétrica das duas curvas é igual no ponto de interseção.
- C^2 – Devem ter a **mesma curvatura**, ou seja, tanto a primeira como a segunda derivadas das duas curvas são as mesmas no ponto de interseção.



Em uma continuidade C^1, as taxas de variação dos vetores tangentes podem ser muito diferentes, de modo que a forma das duas seções adjacentes pode mudar bruscamente. Esta continuidade é suficiente para digitalização de desenhos e aplicações de design.

Em uma continuidade C^2, as taxas de variação dos vetores tangentes para as duas funções na junção são iguais. Desta forma, a tangente faz uma transição suave de uma seção para outra da curva. Esta continuidade é necessária para definição de caminhos de movimentação de câmeras sintéticas e para aplicações de CAD que requerem precisão. A geração de um *path* para câmera usando continuidade C^1 pode causar uma mudança brusca na aceleração da câmera, devido a descontinuidade de movimento, causando desconforto para quem esteja assistindo.

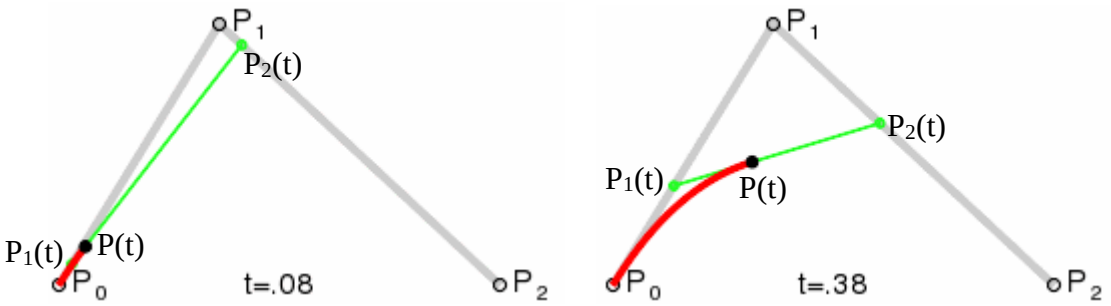
1. Curvas de Bézier

Desenvolvida por Pierre Bézier (1960). A principal **vantagem** à Hermite é que a determinação das tangentes nos pontos de início e fim é feita por pontos de controle e não vetores. Curva utilizada em diversos programas gráficos, como 3D MAX, Corel Draw, Paint Shop Pro, dentre outros.

Faz uso de um polinômio de grau *n*. Para isso deve-se ter *n+1* pontos de controle. Geralmente utiliza-se um polinômio de grau 3 e conseqüentemente 4 pontos de controle.

A curva sempre passa no primeiro e último ponto de controle. Para os demais, a curva é apenas aproximada.

A formulação de Bezier é derivada da interpolação linear paramétrica entre dois pontos (formando uma reta), como mostrada nas seguintes figuras. Para uma situação com 3 pontos de controle, deve-se fazer a interpolação paramétrica das retas paramétricas que unem cada dois pontos de controle, neste caso as retas $P_1(t)$ e $P_2(t)$. Este processo recursivo pode ser aplicado para curvas com qualquer número de pontos de controle.



http://en.wikipedia.org/wiki/B%C3%A9zier_curve

$P_1(t) = (1-t)P_0 + (t)P_1$
$P_2(t) = (1-t)P_1 + (t)P_2$

$P(t) = (1-t)P_1(t) + (t)P_2(t)$
$P(t) = (1-t)\,[(1-t)P_0 + (t)P_1] + (t)\,[(1-t)P_1 + (t)P_2]$
$P(t) = (1-2t+t^2)P_0 + (t-t^2+t-t^2)P_1 + t^2P_2$
$P(t) = (1-t)^2P_0 + 2t(1-t)P_1 + t^2P_2$

A curva paramétrica de Bézier é definida como

$$P(t) = \sum_{i=0}^{n} B_i J_{n,i}(t), \quad 0 \le t \le 1$$

onde B_i representa cada um dos *n+1* pontos de controle e $J_{i,n}(t)$ são as *blending functions* (funções de combinação). Essas funções são descritas pelos **polinômios de Bernstein** como:

$$J_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i} \quad \text{onde} \quad \binom{n}{i} = \frac{n!}{i!(n-i)!}$$

são os coeficientes binomiais. As funções $J_{n,i}$ devem satisfazer as condições

$$j_{n,i}(t) > 0, \quad 0 \le t \le 1$$

$$\sum_{i=0}^{n} J_{n,i}(t) = 1, \quad 0 \le t \le 1$$

Essa propriedade assegura que a curva gerada fica dentro do fecho convexo (*convex hull*) definido pelos pontos de controle.

Para *n=2*, temos 3 pontos de controle e as seguintes funções

$(1-t)^2$
$2t(1-t)$
t^2

De forma mais explícita, a curva em 2D definida por $P_0P_1P_2$ é dada por

$$x(t) = P_{x0}(1-t)^2 + P_{x1}\big(2t(1-t)\big) + P_{x2}(t^2)$$
$$y(t) = P_{y0}(1-t)^2 + P_{y1}\big(2t(1-t)\big) + P_{y2}(t^2)$$

ou (com sobrecarga de operadores da linguagem C++)

$$P(t) = P_0(1-t)^2 + P_1\big(2t(1-t)\big) + P_2(t^2)$$

A matriz que representa as funções é sempre simétrica em relação à diagonal principal e o canto inferior direito é sempre zero.

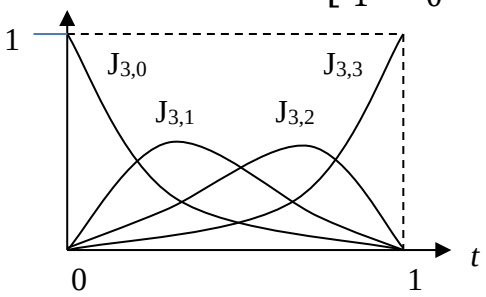
$$P(t) = [t^2 \quad t \quad 1] \cdot \begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix}$$

Para *n=3*, temos 4 pontos de controle e 4 *blending functions*

$(1-t)^3$ (valor máximo em t=0)
$3t(1-t)^2$ (valor máximo em t=1/3)
$3t^2(1-t)$ (valor máximo em t=2/3)
t^3 (valor máximo em t=1)

Matriz simétrica

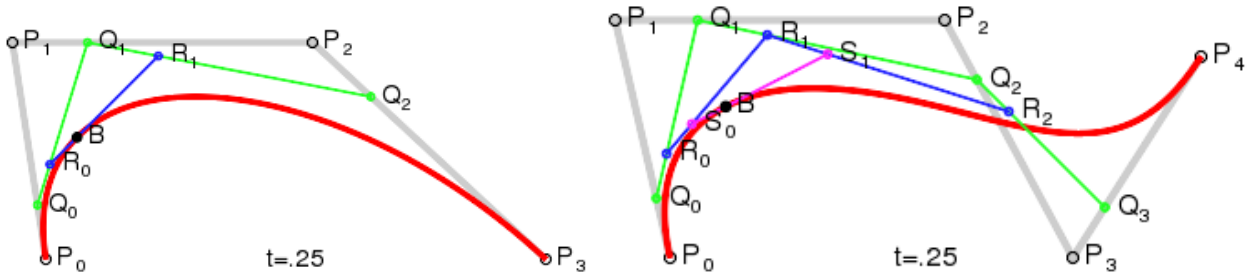
$$P(t) = [t^3 \quad t^2 \quad t \quad 1] \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$



Alterando-se um ponto de controle de curva, toda curva sofre alteração. Isso pode ser verificado observando-se que as funções base são não zero para valores de *t* entre 0 e 1.

A **somatória** das *blending functions* par qualquer valore de *t* é sempre 1.

As seguintes figuras ilustram o processo de geração das curvas para situações com 4 e 5 pontos de controle, usando o mesmo processo recursivo apresentado anteriormente.

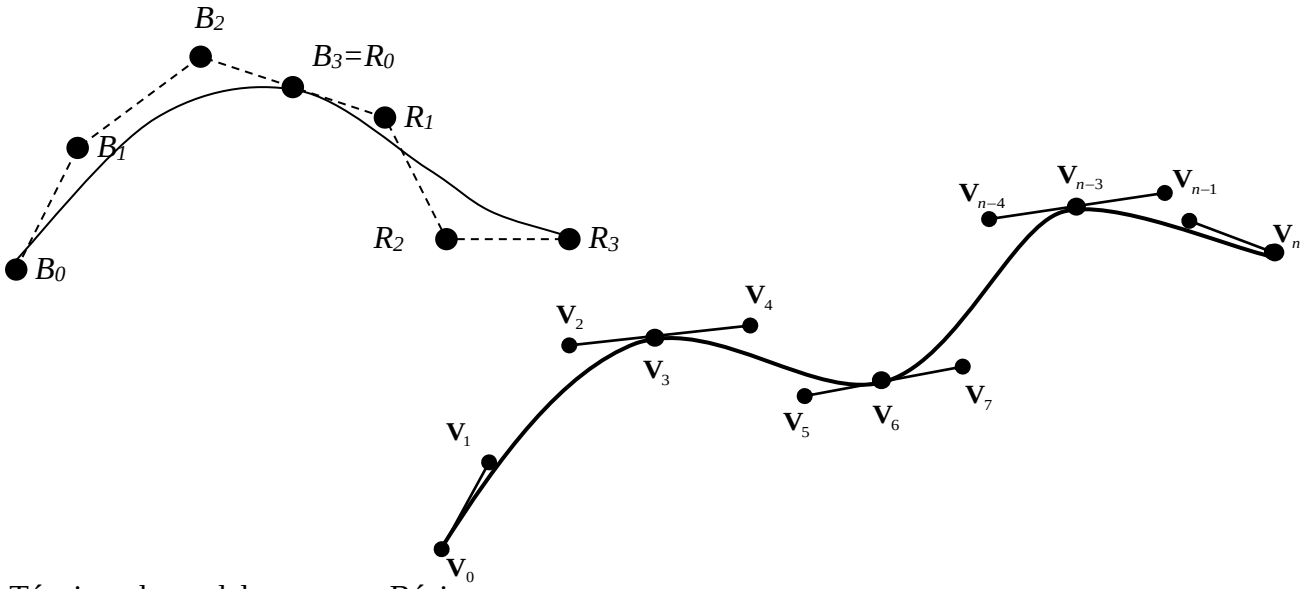


Quanto maior for o grau das funções (número de pontos de controle)

- Maior é o tempo de processamento
- Maior é o controle que se tem sobre a curva. Curvas de baixo grau, alterando-se um ponto, toda curva sofre forte influência.
- Solução para isso é quebrar a curva em vários segmentos de curva. Para assegurar continuidade (ao menos C^1) no ponto de junção, deve-se garantir que
  - Os dois segmentos de curva devem ter um ponto em comum (C^0)
  - As duas curvas devem ter a mesma inclinação no ponto de união. (Derivadas contínuas)
  - Deve-se ter 3 pontos em linha reta, sendo o ponto intermediário o ponto comum as duas retas, e os extremos pontos vizinhos ao ponto em comum.
  - Esta propriedade é válida pois a **tangente da curva** em um ponto extremo é dada pela reta que une a extremidade com o ponto de controle adjacente.

$$(B_3-B_2) = k(R_1-R_0)$$

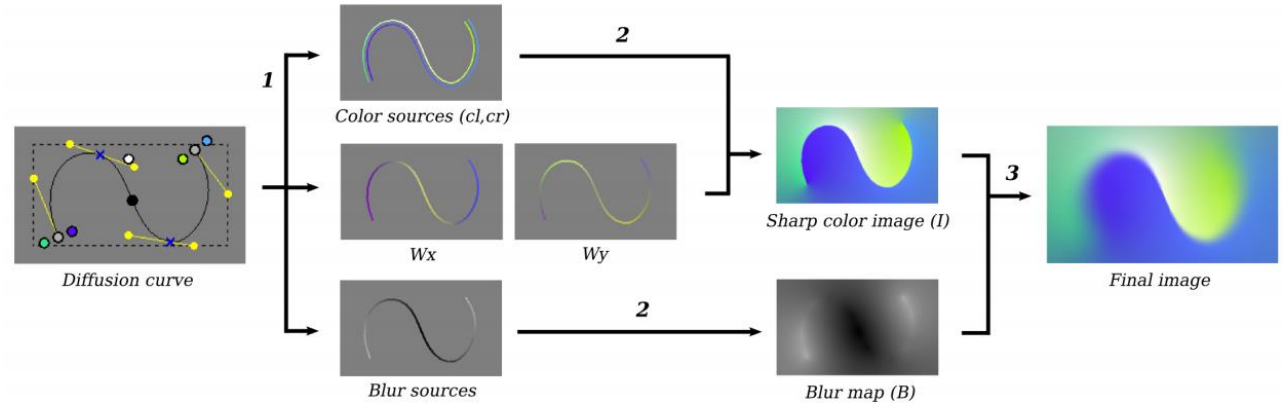
(Os vetores devem ser Linearmente Dependentes – LD)



Técnicas de modelagem com Bézier

- Para gerar um círculo, o primeiro e último pontos devem ser coincidentes. Deve-se também observar a continuidade no ponto de junção
- Pode-se aumentar a influência de um ponto de controle se ele for duplicado (mais de um ponto de controle em uma mesma coordenada).

Um exemplo de aplicação bem recente de curvas de Bézier está no trabalho de Alexandrina Orzan at al.: "*Diffusion Curves: A Vector Representation for Smooth-Shaded Images*", onde juntamente com curvas de difusão pode-se facilmente desenhar figuras com gradientes de cores.



3. B-Spline

A curva B-spline não passa por nenhum ponto de controle (aproximação dos pontos).

Vantagens em relação à Bézier:

- O grau dos polinômios pode ser definido independentemente do número de pontos de controle (com certas limitações);
- Permitem controle local na forma da curva ou superfície.

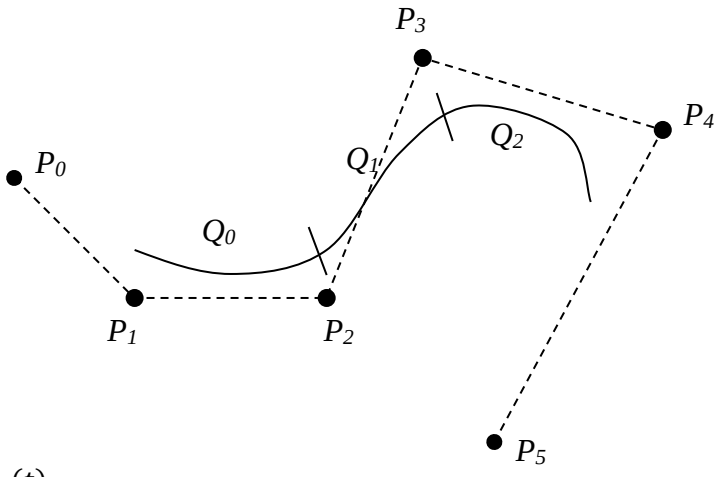
A curva gerada está confinada dentro do **fecho convexo** formado pelos pontos de controle.

Pode ser gerada para um número qualquer de pontos de controle e grau de polinômio. O grau do polinômio independe do número de pontos de controle, seguindo a seguinte regra: C^{d-2}, onde d-1 é o grau da função. Para um polinômio de grau 3, tem-se continuidade C^2. (grau = 3 = 4-1 → $C^{4-2} = C^2$)

Geralmente faz uso de polinômios de grau 3 com 1ª e 2ª derivadas contínuas (C^2), uma vez que as *blending functions*, de grau 3, são polinômios C^2.

O número de *blending functions* é igual ao número de pontos de controle para cada segmento da curva.

O segmento Q_0 é definido por $P_0P_1P_2P_3$ que são escalados por $B_0B_1B_2B_3$
O segmento Q_1 é definido por $P_1P_2P_3P_4$ que são escalados por $B_0B_1B_2B_3$
O segmento Q_2 é definido por $P_2P_3P_4P_5$ que são escalados por $B_0B_1B_2B_3$



$$Q_i(t) = \sum_{k=0}^{3} P_{i+k} B_k(t)$$

$$\sum_{i=0}^{m} B_i(t) = 1 \quad (a\ curva\ fica\ dentro\ do\ fecho\ convexo)$$

O valor em cada segmento de curva varia entre 0 e 1. Assim a B-Spline é definida como uma série de *m-2* segmentos de curvas, onde existem *m+1* pontos de controle.

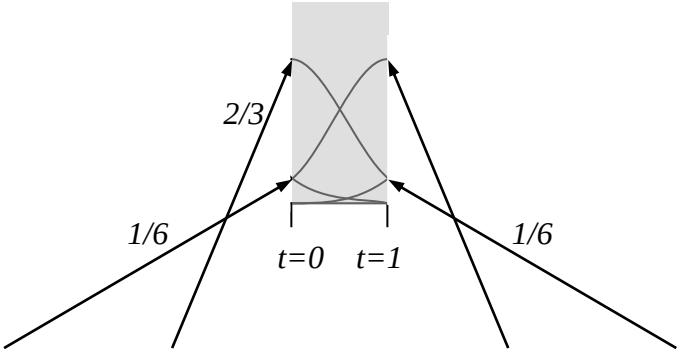
Cada segmento de curva é definido por 4 pontos de controle. Assim existem 3 vezes mais pontos de controle e 3 vezes mais funções bases que segmentos de curva.

O ponto de junção, no valor de *t*, entre segmentos de curva é chamado de ***knot value***. Quando os *knot values* estão igualmente espaçados tem-se uma ***B-Spline*** **uniforme** em relação ao parâmetro *t* (ou seja, $0 \leq t \leq 1$ em qualquer segmento de curva) Caso contrário, tem-se uma ***B-Spline*** **não uniforme**.

Blending Functions

$$B_0 = {}^1\!/_6\,(1 - t^3)$$
$$B_1 = {}^1\!/_6\,(3t^3 - 6t^2 + 4)$$
$$B_2 = {}^1\!/_6\,(-3t^3 + 3t^2 + 3t + 1)$$
$$B_3 = {}^1\!/_6\,t^3$$

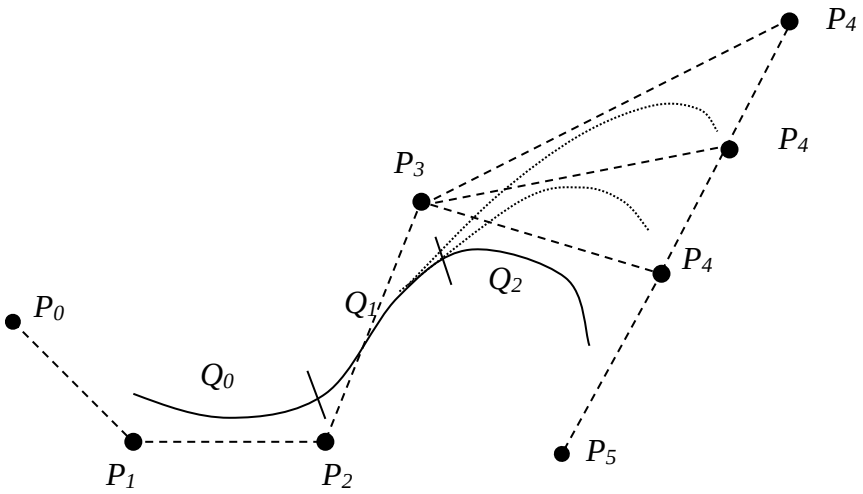
A seguinte figura ilustra a variação das blending functions ao longo de t.



For *i* = 0, ..., *n*
For *t* = 0, ..., 1

$$P(t) = \frac{(1-t)^3}{6} P_i + \frac{(3t^3 - 6t^2 + 4)}{6} P_{i+1} + \frac{(-3t^3 + 3t^2 + 3t + 1)}{6} P_{i+2} + \frac{t^3}{6} P_{i+3}$$

Propriedade: A curva apresenta controle local, pois mudando-se um ponto, a curva sofre influência apenas na proximidade deste ponto.



Duplicação de pontos de controle:

- Aumenta a influência pela repetição. Cada ponto será usado mais de uma vez em um único segmento
- Pode ser usado para interpolar pontos intermediários ou extremos da curva
- Pode-se ter:
  - 1 ponto de controle: caso convencional
  - 2 pontos: a curva passa muito próximo do ponto
  - 3 pontos: a curva toca o ponto de controle

Leitura Complementar

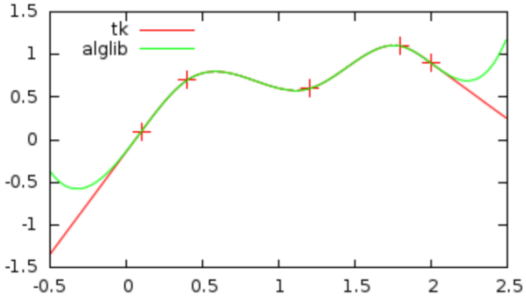
B-Splines não uniformes
β-Splines
NURBS (*Non-Uniform Rational Basis Spline)*

4. Curvas Spline

Spline é uma haste flexível para gerar curvas suaves que passam por um conjunto de pontos. O termo spline é usado para se referir a uma grande classe de funções que são usadas em aplicações que requerem interpolação ou suavização de dados [Wikipedia]. Todos os tipos de curvas apresentados nesse material são fundamentados em Splines, que é um campo de estudo muito amplo. A B-spline é um caso específico de spline.

A spline mais comum é a Spline de Interpolação Cúbica. A curva passa por todos os pontos de entrada. O cálculo da spline é bem complexo, e ao contrário de uma B-Spline, que faz uso sempre das mesmas funções bases, para uma Spline deve-se calcular as funções bases por um processo recursivo com o algoritmo de Cox-de Boor.

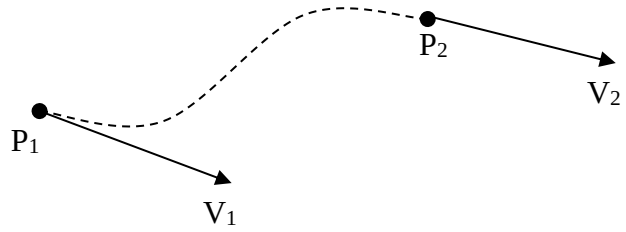
Existem várias implementações disponíveis na internet de splines. Para o caso 1D, sugere-se a biblioteca TK (https://kluge.in-chemnitz.de/opensource/spline/), que apresenta implementação completa em C++. Para casos 1D, 2D e 3D, sugere-se a biblioteca ALGLIB (https://www.alglib.net/), com implementações completas de código aberto em C++, C#, Python e Delphi.



5. Curvas de Hermite

Criadas por Charles Hermite (1822-1901)
.
São necessários 4 fatores para determinar uma curva:

- Dois pontos de controle P_1 e P_2
- Dois vetores V_1 e V_2, para descrever as **tangentes** e seus pesos na curva em P_1 e P_2, ou seja
  - V_1 indica como a curva deixa o ponto P_1
  - V_2 como encontra o ponto P_2.
  - O módulo dos vetores V_1 e V_2 funciona como um peso que muda a forma da curva. V_1 e V_2 são definidos em relação a origem.
- Não permite ajuste global pois a curva é definida pelos pontos extremos. A curva pode ser ajustada localmente visto que cada segmento de curva depende somente das restrições dos limites (*endpoint constraints*)
- A curva interpola os pontos extremos (P_1 e P_2)



O cálculo de um ponto da curva (t) é dado pelos fatores de controle P_1, P_2, V_1 e V_2 ponderados.

$P(t)$ representa um ponto de função cúbica para a seção da curva entre os pontos P_1 e P_2. Assim, as condições de contorno são:

$P(0) = P_1$ (para t=0, $P(t) = P_1$)
$P(1) = P_2$ (para t=1, $P(t) = P_2$)
$P'(0) = P_1' = V_1$ (derivada paramétrica no ponto de controle P_1) – Descreve como a curva deixa o ponto P_1.
$P'(1) = P_2' = V_2$ (derivada paramétrica no ponto de controle P_2) – Descreve como a curva chega ao ponto P_2.

As curvas são definidas por polinômios de grau 3, no seguinte formato:

$$P(t) = at^3 + bt^2 + ct + d, \quad 0 \le t \le 1$$

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$
$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$
$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$$

ou em formato matricial e substituindo os valores dos extremos 0 e 1 no parâmetro u temos

$$P(t) = [t^3 \quad t^2 \quad t \quad 1]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \begin{cases} t = 0 \rightarrow [0 \quad 0 \quad 0 \quad 1][a \quad b \quad c \quad d]^T \\ t = 1 \rightarrow [1 \quad 1 \quad 1 \quad 1][a \quad b \quad c \quad d]^T \end{cases}$$

Calculando as derivadas da função

$$P'(t) = [3t^2 \quad 2t \quad 1 \quad 0]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \begin{cases} t = 0 \rightarrow [0 \quad 0 \quad 1 \quad 0][a \quad b \quad c \quad d]^T \\ t = 1 \rightarrow [3 \quad 2 \quad 1 \quad 0][a \quad b \quad c \quad d]^T \end{cases}$$

Deve-se encontrar o valor de **a, b, c e d** para os valores P_1, P_2, V_1 e V_2 dados.

$$P_1 = [0 \quad 0 \quad 0 \quad 1]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \quad P_2 = [1 \quad 1 \quad 1 \quad 1]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \qquad V_1 = [0 \quad 0 \quad 1 \quad 0]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \quad V_2 = [3 \quad 2 \quad 1 \quad 0]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

A seguinte matriz engloba as 4 restrições (condições) P_1, P_2, V_1 e V_2.

$$\begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

Evidenciando os coeficientes das equações

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1}\begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \end{bmatrix} = M_H \cdot \begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \end{bmatrix}$$

onde M_H é a matriz de Hermite, definida pelo inverso da matriz de restrições. Assim,

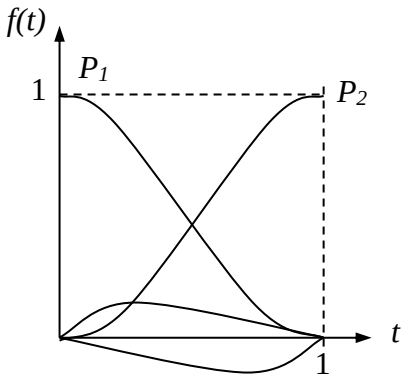
$$P(t) = [t^3 \quad t^2 \quad t \quad 1]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = P(t) = [t^3 \quad t^2 \quad t \quad 1] \cdot M_H \cdot \begin{bmatrix} P_1 \\ P_2 \\ V_1 \\ V_2 \end{bmatrix}$$

$$P(t) = P_1\underbrace{(2t^3 - 3t^2 + 1)}_{t(0)=1} + P_2\underbrace{(-2t^3 + 3t^2)}_{t(1)=1} + V_1\underbrace{(t^3 - 2t^2 + t)}_{t[0,1]=0} + V_2\underbrace{(t^3 - t^2)}_{t[0,1]=0}$$

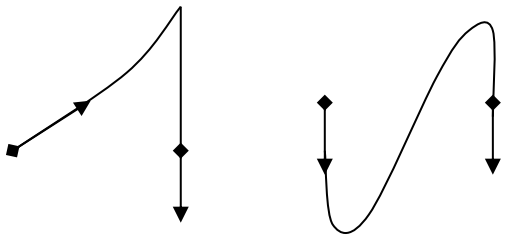
ou seja, nos pontos de controle as funções valem 1 para *t=0* e *t=1*, respectivamente

$$P(t) = P_1 H_0(u) + P_2 H_1(u) + V_1 H_2(u) + V_2 H_3(u)$$

onde $H_n(u)$ são as *blending functions*, como mostrado na seguinte figura



Exemplos de curvas



6. Exercícios

1. Faça um programa para plotar as funções de interpolação de Bezier e B-spline.
2. Faça um programa interativo para criação e edição de curvas de Bezier e B-sline com polinômios de grau 3.
3. O que acontece com a curva se o parâmetro *t* ultrapassar o valor 1 em cada tipo de curva?

Referências

[1] Azevedo, E., Conci, A. **Computação Gráfica, Teoria e Prática**. Editora Elsevier, 2003.

[2] Gattass, M. **Notas de aula**. PUC-Rio

[3] Watt, A. **3D Computer Graphics,** Addison Wesley; 3 edition (December 6, 1999)

[4] Hearn, D., Baker, M. P. **Computer Graphics, C Version** (2nd Edition), Prentice Hall, New Jersey, 1997

[5] Rogers, D., Adams, J. **Mathematical Elements for Computer Graphics**, 2nd Edition. Mc Graw Hill, 1990.