MAC420/5744: Introdução à Computação Gráfica

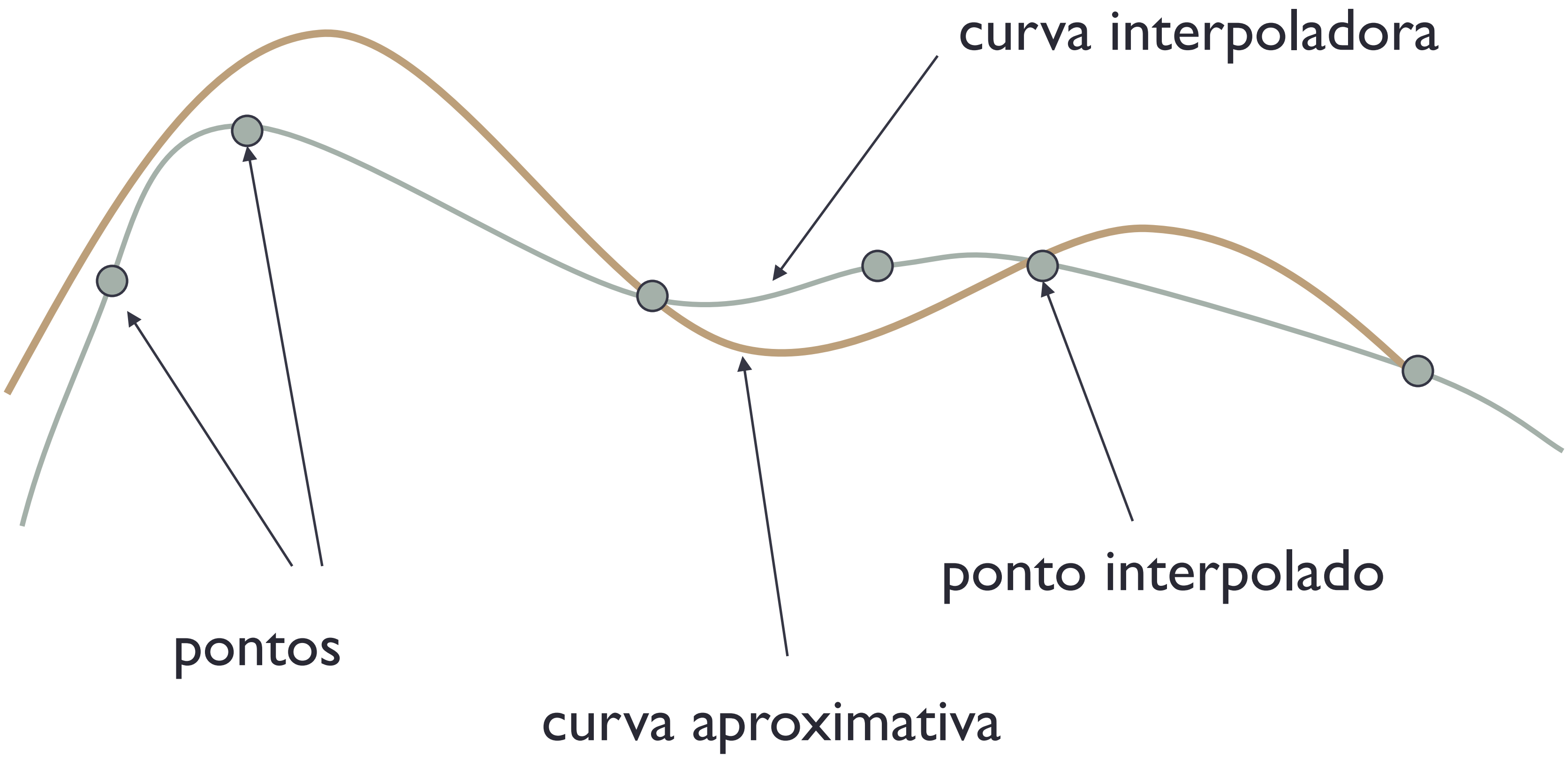
Marcel P. Jackowski
mjack@ime.usp.br

Aula #16: Curvas e superfícies

Fugindo do mundo planar

- Até este momento trabalhamos com entidades planares
  - Retas e triângulos
  - Aceleração por hardware gráfico
  - Matematicamente simples
- O mundo no entanto não é composto por entidades planares
  - Curvas e superfícies curvas
  - Usamos tais entidades somente ao nível da aplicação
  - Renderizá-las através de primitivas planares

Modelagem de curvas



Representações

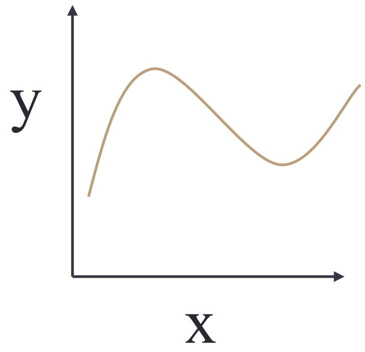
- Há muitas maneiras de representar curvas e superfícies
- Normalmente, queremos que tal representação seja:
  - Estável numericamente
  - Suave, sem discontinuidades e ondulações
  - Fácil de calcular
- Requisitos adicionais
  - Precisamos que a curva passe pelos pontos de controle ?
  - Precisaremos de suas derivadas?

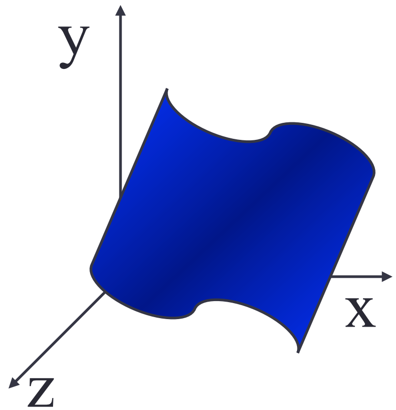
Representação explícita

- Talvez seja forma mais familiar

  $$y=f(x)$$

- Não é possível representar todas as curvas
  - Problemas com linhas verticais
  - Círculos
    - $y = \mathrm{sqrt}(r^2 - x^2)$
- Extensão para 3D
  - $y=f(x)$, $z=g(x)$
  - A forma $z = f(x,y)$ define uma superfície





Representação implícita

- Funções bidimensionais

$$g(x,y)=0$$

- São mais robustas
  - Retas: $ax+by+c=0$
  - Círculos: $x^2+y^2-r^2=0$
- A forma 3D $g(x,y,z)=0$ define uma superfície
  - A interseção de duas superfícies definem uma curva

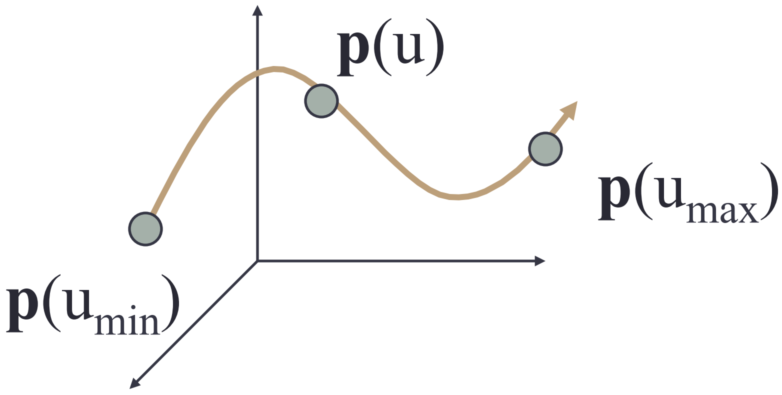
Curvas paramétricas

- Uma equação para cada variável espacial

$$p_x = x(u)$$
$$p_y = y(u)$$
$$p_z = z(u)$$

$$\mathbf{p}(u)=[x(u), y(u), z(u)]^T$$

- Para $u_{min} <= u <= u_{max}$ traçamos uma curva em 2 ou 3 dimensões



Forma implícita ou paramétrica ?

- Cada uma destas formulações possuem vantagens e desvantagens
- Porém devemos nos concentrar nas suas aplicações:
  - Amostragem de pontos: dado um objeto S, determinar um conjunto de pontos $p_1, p_2, \ldots, p_n$ tais que p_i pertence a S.
  - Classificação ponto-conjunto: dado um ponto p e um objeto S, determinar se p pertence a S.

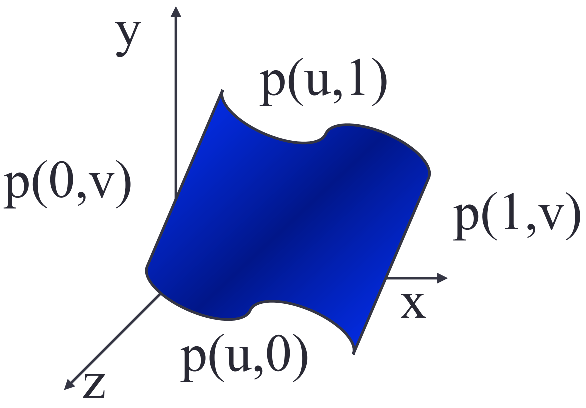
Funções paramétricas

- Normalmente conseguimos selecionar funções "razoáveis"
  - Não são únicas para uma certa curva
  - Podem aproximar ou interpolar pontos de controle
  - Desejamos funções de fácil avaliação
  - Desejamos funções de fácil diferenciação
    - Cálculo de normais
    - Conexões (segmentos)
    - Desejamos funções suaves

Superfícies paramétricas

- Superfícies requerem 2 parâmetros:

$$p_x = x(u,v)$$
$$p_y = y(u,v)$$
$$p_z = z(u,v)$$
$$\mathbf{p}(u,v) = [x(u,v),\ y(u,v),\ z(u,v)]^T$$



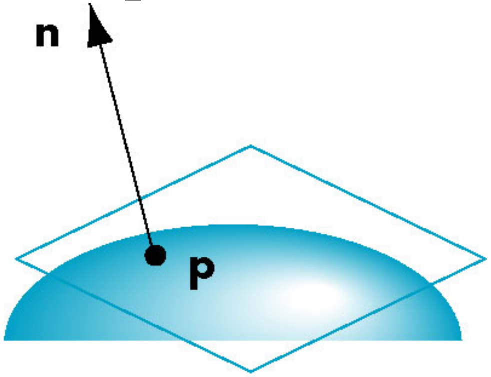
- Desejamos as mesmas propriedades das curvas paramétricas:
  - Suavidade
  - Diferenciabilidade
  - Facilidade de avaliação

Normais

Podemos diferenciar em relação aos parâmetros u e v para obter o vetor normal em qualquer ponto **p**

$$\frac{\partial \mathbf{p}(u,v)}{\partial u} = \begin{bmatrix} \partial \mathrm{x}(u,v)/\partial u \\ \partial \mathrm{y}(u,v)/\partial u \\ \partial \mathrm{z}(u,v)/\partial u \end{bmatrix} \qquad \frac{\partial \mathbf{p}(u,v)}{\partial v} = \begin{bmatrix} \partial \mathrm{x}(u,v)/\partial v \\ \partial \mathrm{y}(u,v)/\partial v \\ \partial \mathrm{z}(u,v)/\partial v \end{bmatrix}$$

$$\mathbf{n} = \frac{\partial \mathbf{p}(u,v)}{\partial u} \times \frac{\partial \mathbf{p}(u,v)}{\partial v}$$

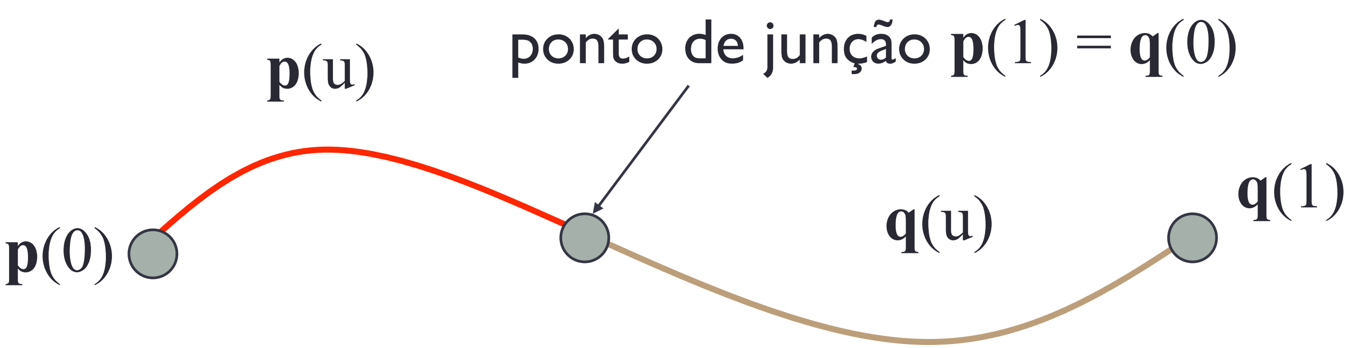


Segmentos de curva

- Podemos normalizar u, de forma que uma curva seja escrita como:

$$\mathbf{p}(u)=[x(u), y(u), z(u)]^T, \quad 0 <= u <= 1$$

- É comum desenhamos uma única curva com suporte global
- Em CG e CAD, é mais viável desenhar pequenos segmentos de curva que são interconectados



Curvas polinomiais paramétricas

$$x(u) = \sum_{i=0}^{N} c_{xi} u^i \qquad y(u) = \sum_{j=0}^{M} c_{yj} u^j \qquad z(u) = \sum_{k=0}^{L} c_{zk} u^k$$

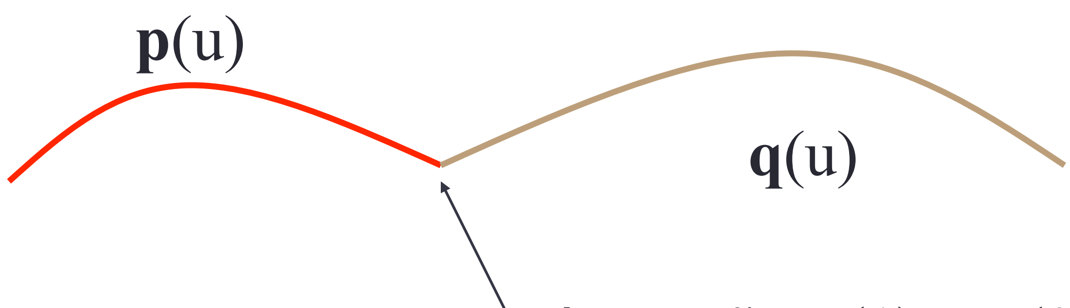
- Se N=M=L, precisamos determinar 3(N+1) coeficientes
- Cada uma das curvas para x, y e z são independentes e podem ser definidas de maneira idêntica

Usaremos a forma ao lado, onde p pode ser x, y, z

$$\mathrm{p}(u) = \sum_{k=0}^{L} c_k u^k$$

Por quê polinômios ?

- Fáceis de calcular
- Funções contínuas e diferenciáveis
- Porém devemos nos preocupar com a continuidade nos pontos entre segmentos



ponto de junção $\mathbf{p}(1) = \mathbf{q}(0)$
mas $\mathbf{p}'(1) \neq \mathbf{q}'(0)$

Curvas polinomiais cúbicas

- A atribuição de N=M=L=3, resulta na facilidade de avaliação e flexibilidade no design

$$\mathrm{p}(u) = \sum_{k=0}^{3} c_k u^k$$

- Quatro coeficientes são necessários para definir x, y e z
- Achar quatro condições independentes que resultarão em 4 equações com 4 variáveis para cada x, y e z
- Tais condições são uma mistura de requisitos de continuidade nos pontos de junção e condições de representação dos dados

Superfícies polinomiais cúbicas

$$\mathrm{p}(u, v) = [\mathrm{x}(u, v), \mathrm{y}(u, v), \mathrm{z}(u, v)]^{\mathrm{T}}$$

onde

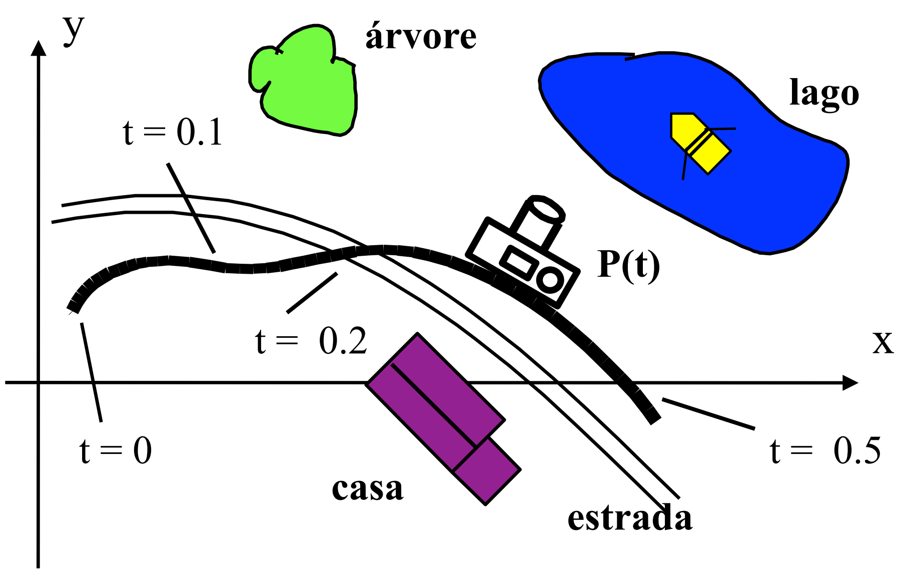
$$\mathrm{p}(u,v) = \sum_{i=0}^{3} \sum_{j=0}^{3} c_{ij} u^i v^j$$

e p representa x, y ou z

Precisamos de 48 coeficientes (3 conjuntos independentes de 16 coeficientes) para determinar um retalho ("patch") da superfície

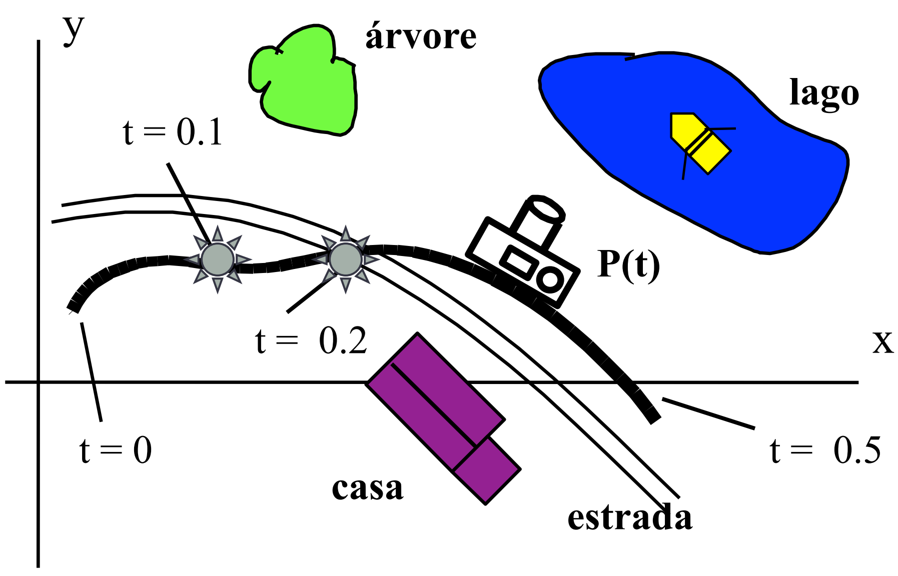
Motivação: animação da câmera

- A trajetória da câmera em uma cena deve ser especificada a cada instante de tempo
- A câmera estará localizada na posição *P*(*t*) no tempo *t*.



Animação (ii)

- Escolhemos uma função *P*(*t*) de forma que a câmera se mova conforme a trajetória desejada
- Esta câmera, por exemplo, pode tirar fotos da cena em tempos $t = 0.1$, $t = 0.2$, etc.
- A direção de visualização deve também ser especificada a cada instante.



Animação (iii)

- A câmera deve se deslocar na trajetória $P(t)$ de forma suave, sem movimentos bruscos
  - Isto impõe uma restrição na velocidade $P'(t)=v(t)$.
- Outros objetos também poderão se mover na cena: o carro, o barco, pessoas saindo da casa, etc.
  - O movimento destes objetos pode ser descrito através de funções paramétricas $F(t)$, $G(t)$, etc, apropriadas.

Suavidade de movimento

- A **velocidade** v(*t*) é um vetor que descreve a velocidade e direção de um objeto se movendo ao longo da trajetória *P*(*t*)
- É caracterizada pela primeira derivada da trajetória *P*(*t*):

$$\mathbf{v} = \frac{dP(t)}{dt} = \left(\frac{dx}{dt}, \frac{dy}{dt} \right)$$

Suavidade de movimento (ii)

- A direção **normal** à curva pode também ser determinada em cada ponto
  - Ela é definida como a direção perpedicular à reta tangencial em cada ponto
- Se a reta tangencial possui direção $\mathbf{v}(t_0)$ no tempo t_0, a direção normal em t_0 será dada pelo vetor:
  - $\mathbf{n}(t_0) = \mathbf{v}^{\perp}(t_0) = (-dy/dt, dx/dt)\,|_{t\,=\,t0}$

Formato vetorial

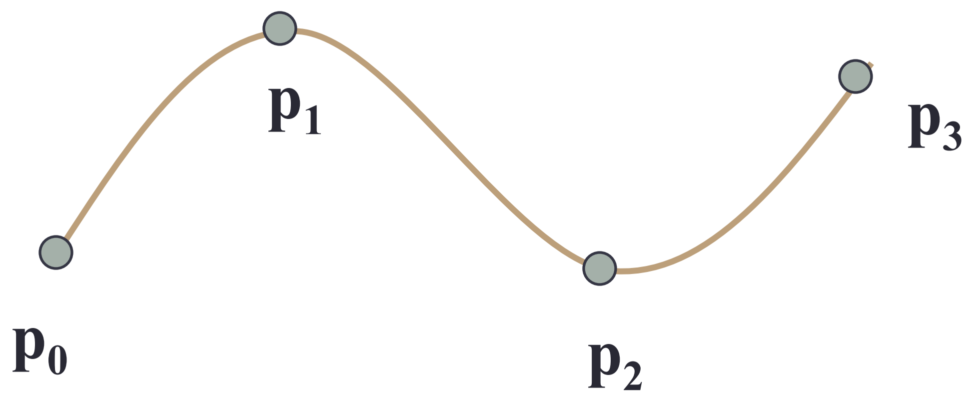
$$P(u) = \sum_{k=0}^{3} c_k u^k$$

Se fizermos:

$$\mathbf{c} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} 1 \\ u \\ u^2 \\ u^3 \end{bmatrix}$$

Então: $P(u) = \mathbf{u}^{\mathrm{T}}\mathbf{c} = \mathbf{c}^{\mathrm{T}}\mathbf{u}$

Primeiro passo: curva interpoladora



Dados quatro pontos (de controle) p_0 , p_1 , p_2 , p_3 determine a cúbica $p(u)$ que passe por eles

Devemos achar c_0, c_1, c_2, c_3

Equações de interpolação

Aplica-se as condições de interpolação em u=0, 1/3, 2/3, 1

$$p_0=p(0)=c_0$$
$$p_1=p(1/3)=c_0+(1/3)c_1+(1/3)^2c_2+(1/3)^3c_2$$
$$p_2=p(2/3)=c_0+(2/3)c_1+(2/3)^2c_2+(2/3)^3c_2$$
$$p_3=p(1)=c_0+c_1+c_2+c_2$$

escrita matricialmente abaixo, com $p = [p_0\ p_1\ p_2\ p_3]^T$

$$\mathbf{p=Ac} \qquad \mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right)^2 & \left(\frac{1}{3}\right)^3 \\ 1 & \left(\frac{2}{3}\right) & \left(\frac{2}{3}\right)^2 & \left(\frac{2}{3}\right)^3 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Matriz de interpolação

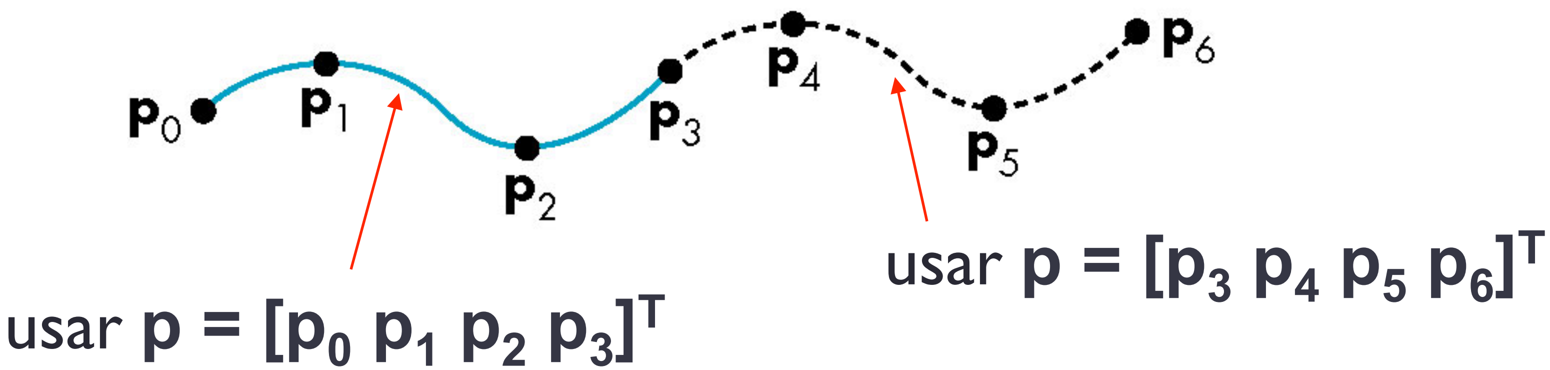
Resolvendo o sistema de equações, determinamos a matriz de *interpolação*

$$\mathbf{M}_I = \mathbf{A}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -5.5 & 9 & -4.5 & 1 \\ 9 & -22.5 & 18 & -4.5 \\ -4.5 & 13.5 & -13.5 & 4.5 \end{bmatrix}$$

$$\mathbf{c} = \mathbf{M}_I \mathbf{p}$$

Note que $\mathbf{M}_I$ não depende dos pontos de entrada e pode ser utilizada para cada segmento em x, y, e z

Interpolando múltiplos segmentos



Teremos continuidade nas junções mas não necessariamente suas derivadas.

Funções de mistura ("blending")

Podemos reescrever a equação de p(u)

$$P(u)=u^Tc=u^TM_Ip = b(u)^Tp$$

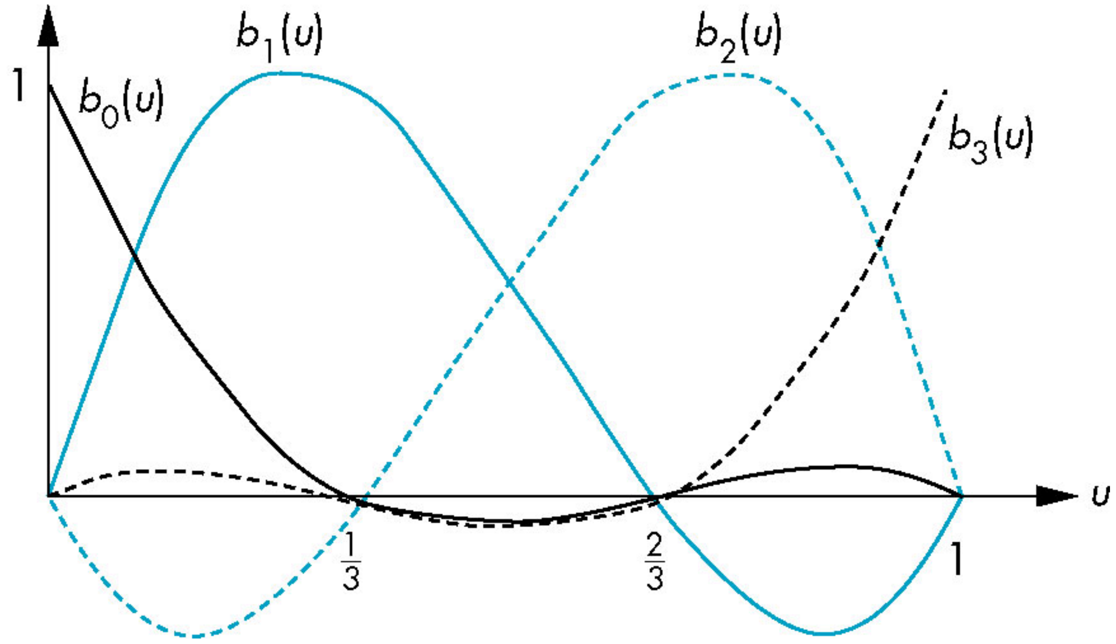
onde $b(u) = [b_0(u)\ b_1(u)\ b_2(u)\ b_3(u)]^T$ é um vetor de *polinômios de mistura* onde

$$P(u) = b_0(u)p_0 + b_1(u)p_1 + b_2(u)p_2 + b_3(u)p_3$$

$$b_0(u) = -4.5\ (u-1/3)(u-2/3)(u-1)$$
$$b_1(u) = 13.5u\ (u-2/3)(u-1)$$
$$b_2(u) = -13.5u\ (u-1/3)(u-1)$$
$$b_3(u) = 4.5u\ (u-1/3)(u-2/3)$$

Funções de mistura

- Estas funções não são suaves
  - Todos os zeros das funções estão entre 0 e 1.
  - Então a interpolação também não será.

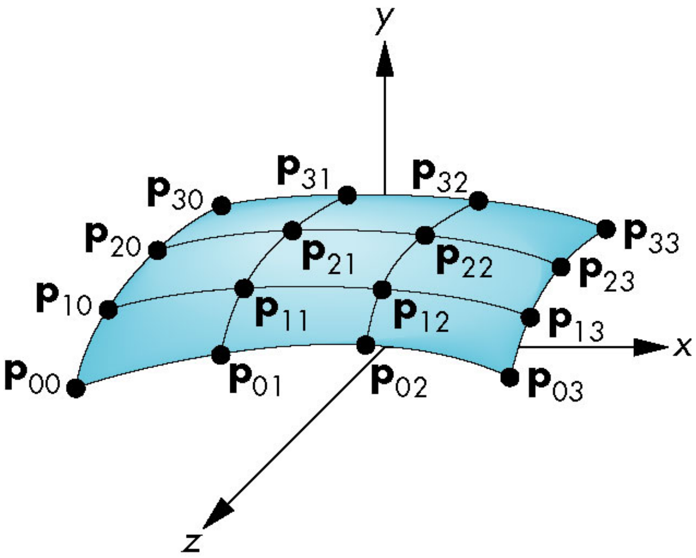


Retalho interpolador

$$P(u,v) = \sum_{i=o}^{3} \sum_{j=0}^{3} c_{ij} u^i v^j$$

Precisamos de 16 condições para determinar os 16 coeficientes

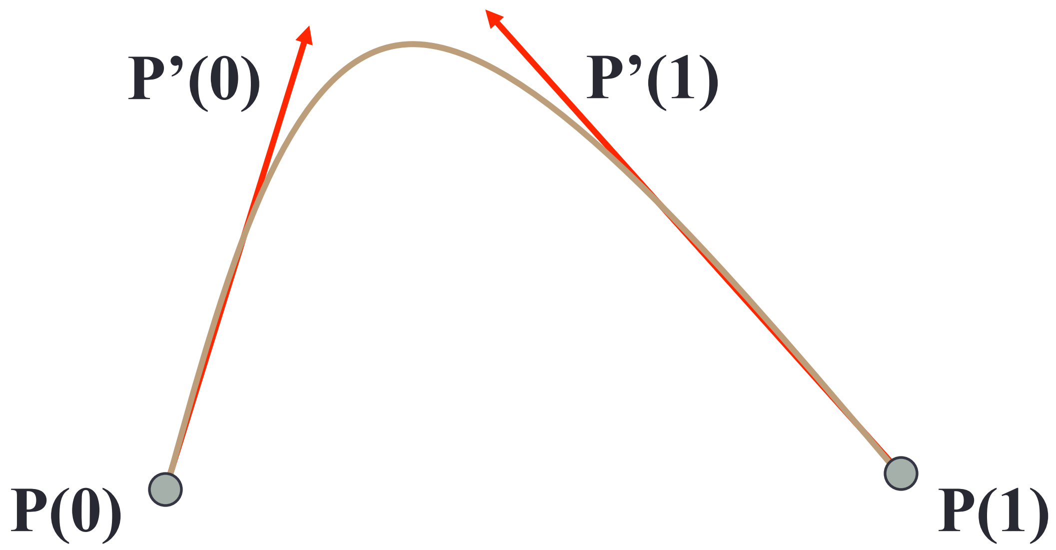
Escolheremos **u,v = 0, 1/3, 2/3, 1**



Outtros tipos de curvas e superfícies

- Como transcender as limitações da forma interpoladora
  - Ausência de suavidade
  - Discontinuidade das derivadas nos pontos de junção
- Temos 4 condições (cúbicas) que podemos aplicar em cada segmento
  - Podemos utilizá-las para outros objetivos além da interpolação
  - Precisamos somente passar bem perto dos pontos

Curvas de Hermite



Utiliza duas condições interpoladoras e
duas condições sobre as derivadas por segmento

Conserva a continuidade e primeira derivada entre segmentos

Equações

As condições de interpolação permanecem as as mesmas nos dois extremos da curva

$$\mathbf{P(0) = p_0 = c_0}$$
$$\mathbf{P(1) = p_3 = c_0+c_1+c_2+c_3}$$

Diferenciando $\mathbf{P}$ encontramos $\mathbf{P'(u) = c_1+2uc_2+3u^2c_3}$

Avaliando nos pontos extremos:

$$\mathbf{P'(0) = p'_0 = c_1}$$
$$\mathbf{P'(1) = p'_3 = c_1+2c_2+3c_3}$$

Forma matricial

$$\mathbf{q} = \begin{bmatrix} p_0 \\ p_3 \\ p'_0 \\ p'_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix} \mathbf{c}$$

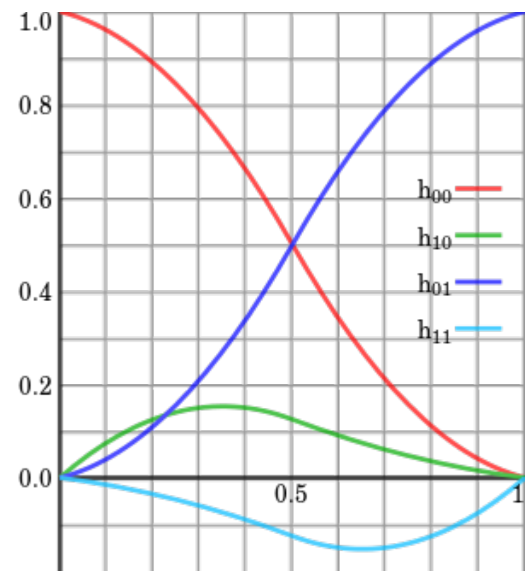
resolvendo o sistema de equações, encontramos $\mathbf{c}=\mathbf{M}_H\mathbf{q}$ onde $\mathbf{M}_H$ é a matriz de Hermite

$$\mathbf{M}_H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}$$

Polinômios de mistura

$$\mathbf{P(u) = b(u)^T q}$$

$$\mathbf{b}(u) = \begin{bmatrix} 2u^3 - 3u^2 + 1 \\ -2u^3 + 3u^2 \\ u^3 - 2u^2 + u \\ u^3 - u^2 \end{bmatrix}$$



Embora estas funções sejam suaves, a forma de Hermite não é amplamente utilizada em CG ou CAD: normalmente temos pontos de controle e não derivadas.

Todavia, a forma de Hermite é a base das curvas Bézier.

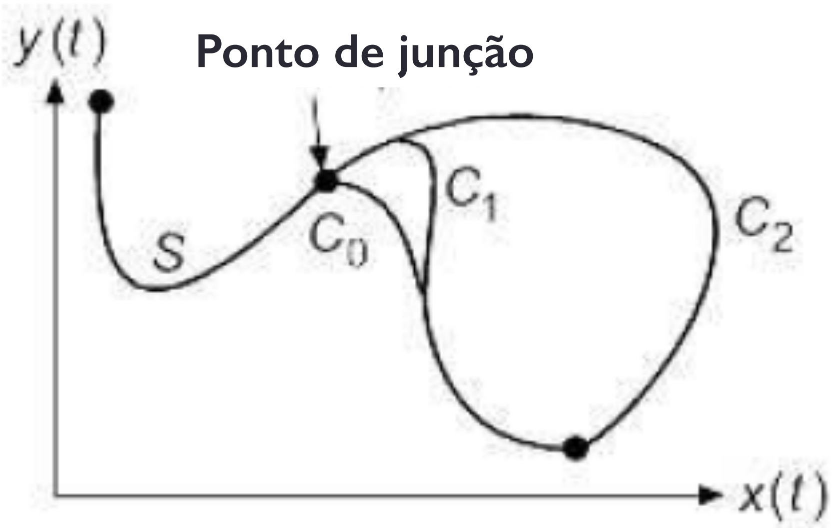
Continuidade paramétrica (C^k)

- Dizemos que uma curva $P(t)$ tem *continuidade paramétrica* de ordem k no intervalo de t E $[a, b]$ se todas as derivadas da curva até grau k, existem e são contínuas no intervalo $[a, b]$.
- Dizemos, então que, P(t) tem suavidade k no intervalo t E $[a, b]$.
- Para evitar movimentos abruptos em animações, utilizaremos curvas com suavidade 1.

Continuidade geométrica (G^k)

- A continuidade G^0 é igual a C^0: $P(t)$ é contínuo em t no intervalo $[a, b]$.
- Continuidade G^1 em $[a, b]$ implica que $P'(t\text{-dt}) = k\, P'(t\text{+dt})$ para alguma constante k e para todo c no intervalo [a,b].
- Continuidade G^2 em [a, b] implica que $P'(t\text{-dt}) = k\, P'(t\text{+dt})$ e $P''(t+dt) = m\, P''(t-dt)$ para as constantes k e m e para todo c no intervalo $[a, b]$.

Continuidade: Exemplo 1



No ponto de junção da curva ***S*** com as curvas ***C₀***, ***C₁*** e ***C₂*** temos:

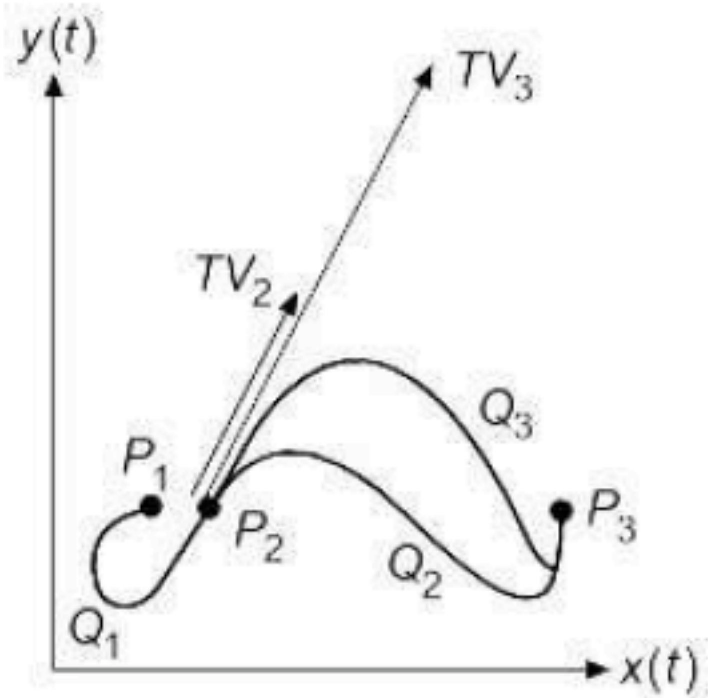
Continuidade G^0 entre ***S*** e $\boldsymbol{C_0}$

Continuidade C^1 entre ***S*** e $\boldsymbol{C_1}$

Continuidade C^2 entre ***S*** e $\boldsymbol{C_2}$

Continuidade: Exemplo 2

A continuidade paramétrica é mais restritiva que a continuidade geométrica:



Por exemplo: C^1 implica G^1

No ponto de junção P_2 temos:

Q_2 e Q_3 são G^1 com Q_1

Só Q_2 é C^1 com Q_1 (TV_1=TV_2)

Curvas polinomiais de 3º. grau

- Funções polinomiais maiores que grau 3 não são de fácil conversão para a forma paramétrica
- Polinômios cúbicos, no entanto, são úteis no desenho de curvas e superfícies
  - Utilizaremos uma coleção de pontos de controle e um algoritmo para gerar pontos na curva.
- O designer poderá editar a posição dos pontos de controle e visualizar a nova curva.
- Esta é uma abordagem visual, deixando o usuário acompanhar interativamente o progresso do design da curva.

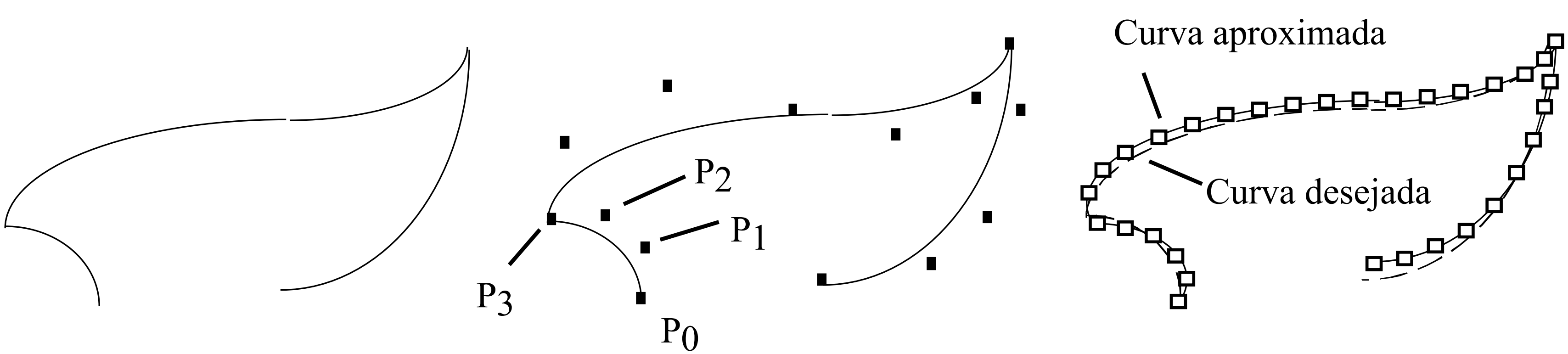
Design interativo de curvas

- Para desenhar, o operador move o ponteiro ao longo de uma curva ideal, clicando em um conjunto de pontos de controle $p_0, p_1,\ldots$ próximas à curva ideal.

a) Curva "desejada"

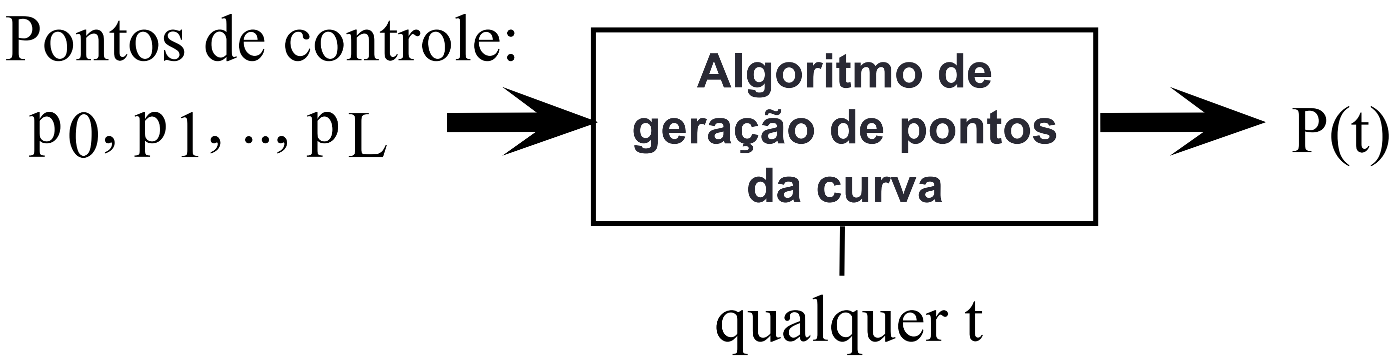
b) Operador posiciona pontos de controle

c) O algoritmo gera pontos sob uma curva próxima



Design interativo de curvas

- O papel do algoritmo é produzir um ponto $P(t)$ para qualquer valor de t. Os dados de entrada são o conjunto de pontos de controle, que determinam a forma da curva $P(t)$.



Design interativo de curvas

- Normalmente implementado como uma função

```
Point2D desenhaCurva(double t, Point2D *pts_controle);
```

- Que retorna um ponto para qualquer valor de *t* dentro de um certo intervalo.
- Para desenhar a curva, o usuário escolhe uma sequência de valores de *t*, e chama a função desenhaCurva para cada um deles;
- Finalmente, conecta-se os pontos gerados através de segmentos de reta (polilinha).

Curvas de Bézier

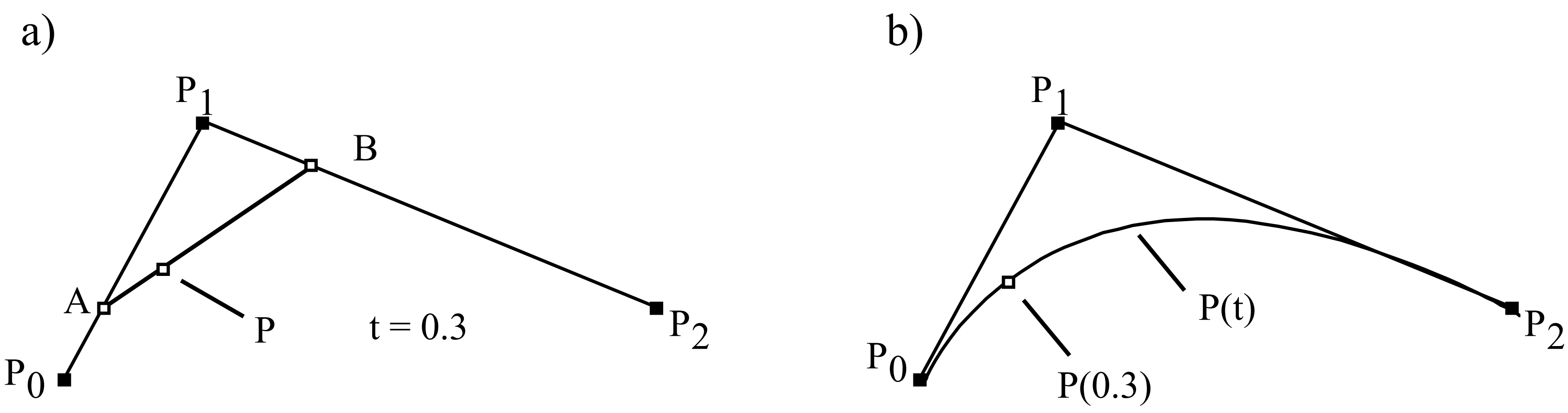
- As curvas de Bézier (curvas aproximativas) foram inventadas para auxiliar no design de automóveis
  - O algoritmo "de Casteljau"
- O algoritmo de De Casteljau baseia-se em uma seqüência de passos de transformação geométrica de fácil implementação
- Através desta transformação, é possível deduzir uma série de propriedades das que curvas que ela gera

Curvas Bézier (ii)

- Transformação de 3 pontos para obtenção de uma parábola:
  - Escolha três pontos: P_0, P_1, and P_2
  - Escolha um valor de t entre 0 and 1, ex. $t = 0.3$
  - Localize o ponto A que está a uma fração t ao longo da linha de P_0 to P_1. Analogamente, localize B a mesma fração t entre os pontos P_1 e P_2.
  - Os novos pontos serão:
    - $A(t) = (1\text{-}t)P_0 + tP_1$
    - $B(t) = (1\text{-}t)P_1 + tP_2$

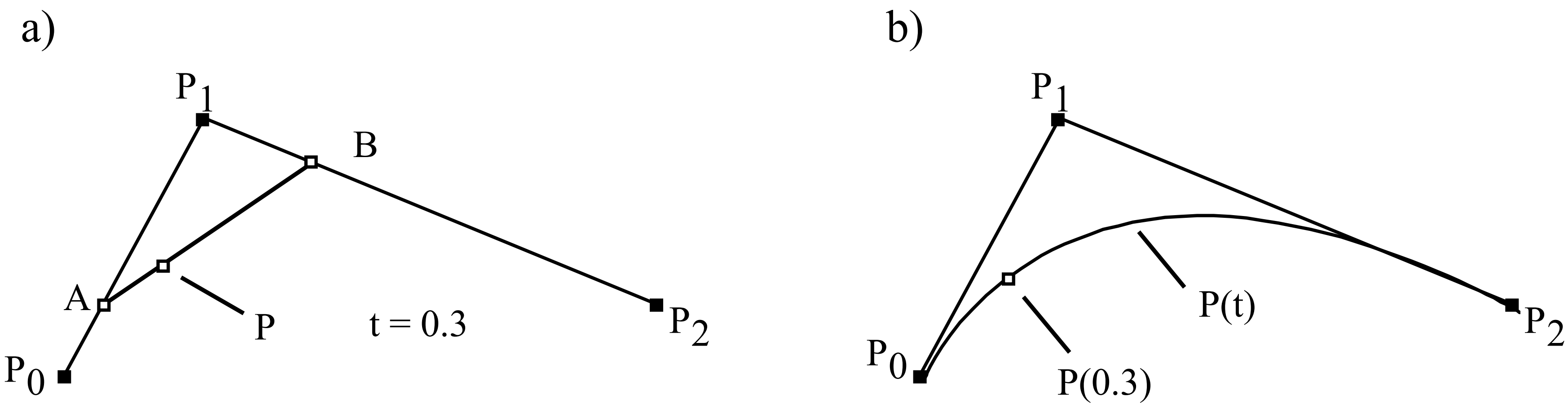
Curvas Bézier (iii)

- Agora repita a interpolação linear nestes pontos (usando o mesmo valor de *t*)
- Ache o ponto, *P*(*t*), que está na fração *t* do caminho entre A e B: P(t) = (1-t)A + tB.



Curvas Bézier (iv)

- Se este processo for repetido para *todo t* entre 0 e 1, a curva *P*(*t*) será gerada.
- A forma paramétrica resultante para tal curva será $P(t) = (1\text{-}t)^2P_0 + 2t(1\text{-}t)P_1 + t^2P_2$



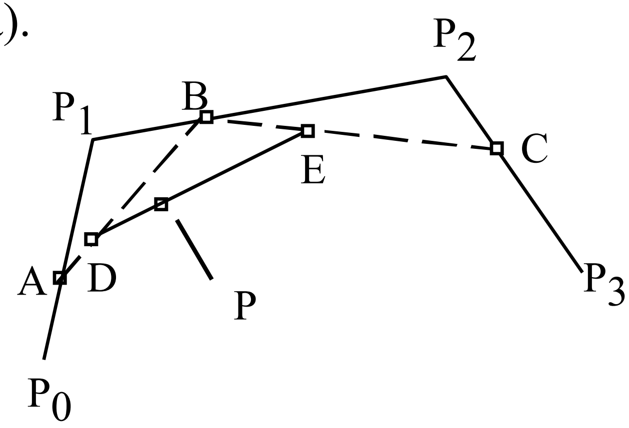
Curvas Bézier (v)

- A forma paramétrica para $P(t)$ é quadrática em t, então concluimos que tal curva é uma parábola
- Ela continuará sendo uma parábola mesmo quando t variar entre $-\infty$ to ∞.
- Ela passará em P_0 quando $t = 0$ e em P_2 quando $t = 1$
- Assim obtemos um processo bem-definido que gera uma curva parabólica suave baseada em três pontos de controle.

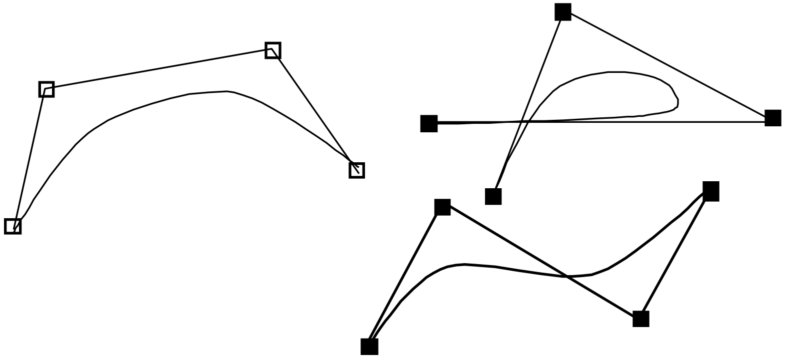
Curvas Bézier cúbicas

- Para um dado valor de *t*, o ponto *A* é posicionado a uma fração *t* entre P_0 e P_1, e similarmente para *B* e *C*.
- Então *D* é colocado a uma fração *t* do caminho entre *A* e *B*, e similarmente para o ponto *E*.
- Finalmente, o ponto desejado *P* está localizado a uma fração *t* do caminho entre *D* e *E*.

a).



b).

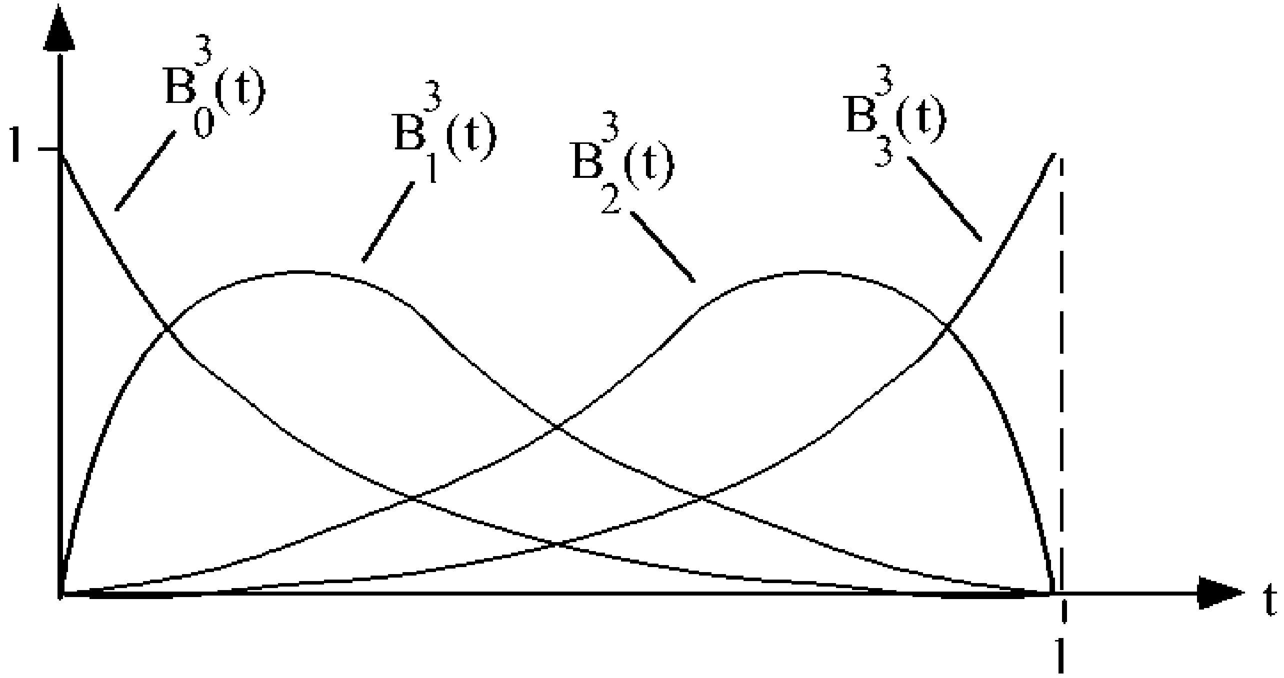


Curvas Bézier cúbica

- A curva Bézier baseada em quatro pontos de controle possui a forma paramétrica
  - $P(t) = P_0(1\text{-}t)^3 + P_1 3(1\text{-}t)^2 t + P_2 3(1\text{-}t)t^2 + P_3 t^3$.
- Cada ponto de controle P_i é pesado por um polinômio cúbico, e os termos são somados.
- Estes termos são chamados de *polinômios de Bernstein*:

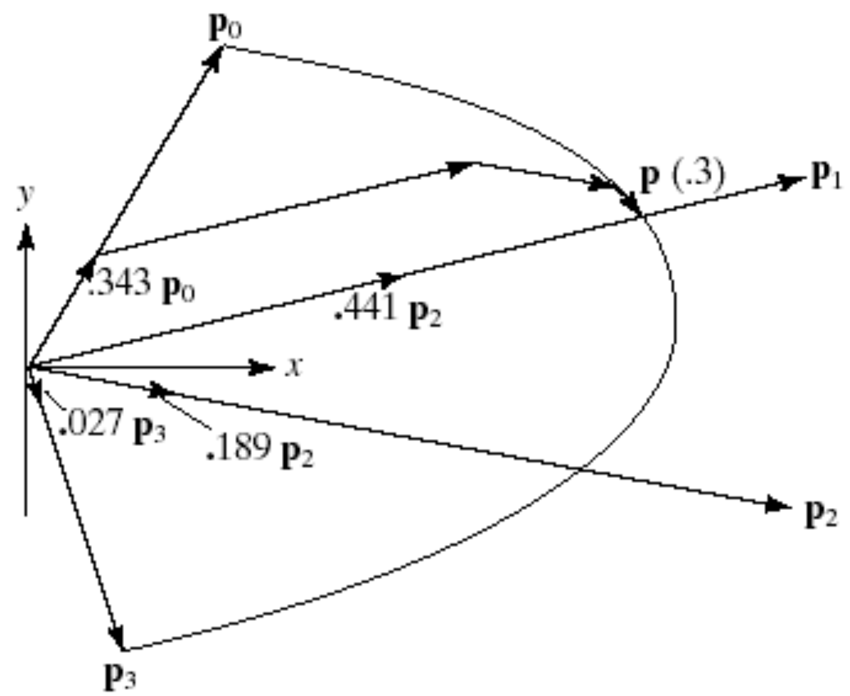
$$B_0^3 = (1-t)^3 \quad B_1^3 = 3(1-t)^2 t \quad B_2^3 = 3(1-t)t^2 \quad B_3^3 = t^3$$

Polinômios de Bernstein



Ponderação com Bernstein

- Considere pontos como vetores na origem (ex., P_0 e t = 0.3)
- Então $\mathbf{p}(0.3) = 0.343\ \mathbf{p_0} + 0.441\ \mathbf{p_1} + 0.189\ \mathbf{p_2} + 0.027\ \mathbf{p_3}$
- Nesta figura os quatro vetores são modulados e os resultados são adicionados para formar o vetor $\mathbf{p}(0.3)$.



Generalização das curvas de Bézier

- A curva resultante será:

$$B_k^L(t) = \begin{pmatrix} L \\ k \end{pmatrix} (1-t)^{L-k} t^k \qquad P(t) = \sum_{k=0}^{L} \mathrm{P}_k B_k^L(t)$$

- onde o coeficiente binomial é:

$$\begin{pmatrix} L \\ k \end{pmatrix} = \frac{L!}{k!(L-k)!}$$

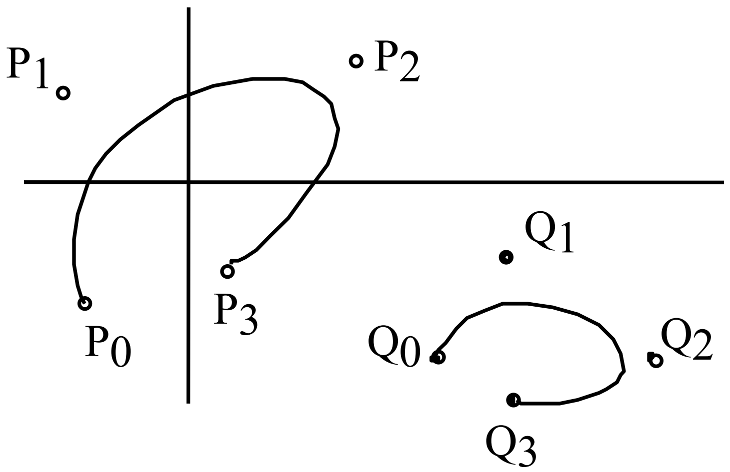
Propriedades das curvas Bézier

- As propriedades das curvas de Bézier fazem com que elas sejam perfeitas para o uso em CAD
  - Interpolação dos pontos finais: A curva Bézier $P(t)$ baseada nos pontos de controle $P_0, P_1, \ldots, P_L$ sempre interpola os pontos P_0 e P_L.
  - Invariância afim: para aplicar uma transformação afim T em todos os pontos $P(t)$ da curva, transformamos os pontos de controle uma vez, e usamos os novos pontos para recriar a curva transformada Q(t) em qualquer *t*.

$$Q(t) = \sum_{k=0}^{L} T(P_k) B_k^L(t) = T\left(\sum_{k=0}^{L} P_k B_k^L(t)\right)$$

Propriedades das curvas Bézier

- Exemplo: Uma curva Bezier é baseada em quatro pontos de controle $P_0,\ldots, P_3$. Os pontos são rotacionados, escalados e transladados para as novas posições Q_k.
- A curva Bezier resultante para Q_k é desenhada. Ela é idêntica ao resultado da transformação da curva Bezier original.



Propriedades das curvas Bézier

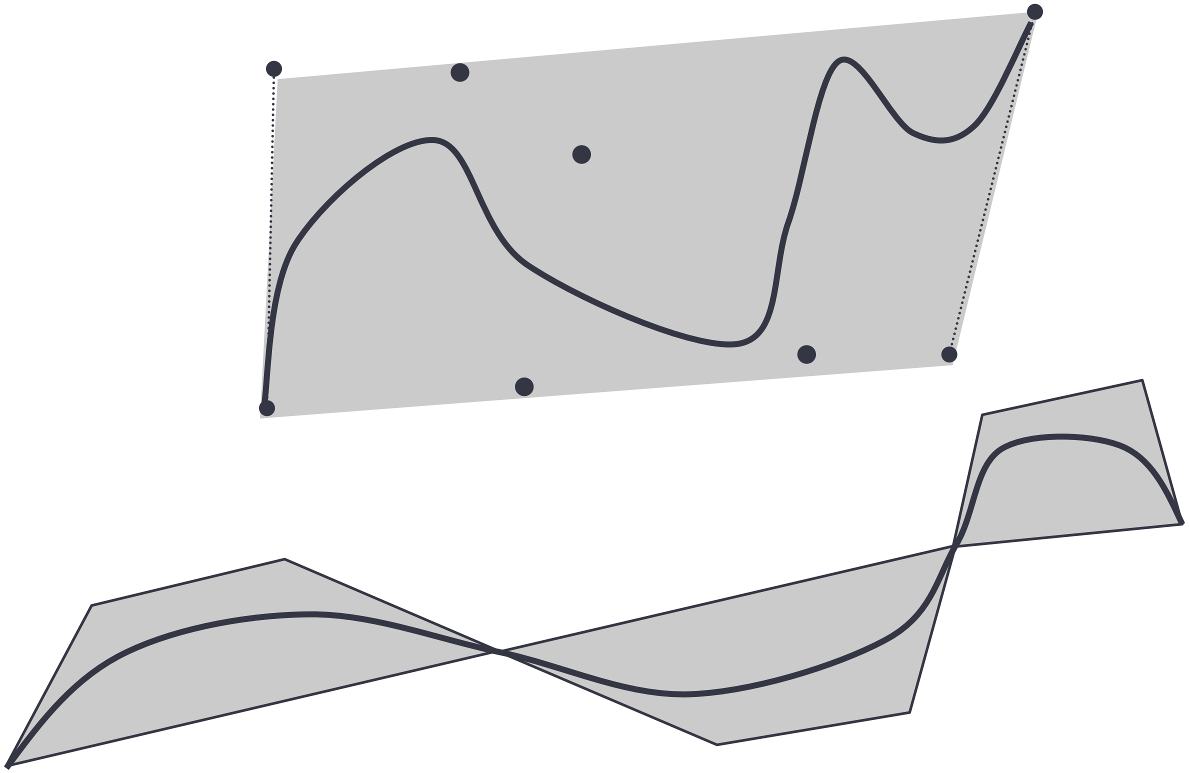
- Uma curva Bézier *P*(*t*), nunca deixa o seu fecho convexo
- O fecho convexo do conjunto de pontos V_0, V_1,.. ., V_i é o conjunto de todas as suas *combinações convexas*; isto é, o conjunto de todos os pontos dados por

$$\boxed{P = \sum_{i=0}^{n} \alpha_i \vec{V}_i \quad com \quad \sum_{i=0}^{n} \alpha_i = 1}$$

onde cada α_i é positivo, e a soma é igual 1.

Fecho convexo

$$P = \sum_{i=0}^{n} \alpha_i \vec{V}_i \quad com \quad \sum_{i=0}^{n} \alpha_i = 1$$



Desenhando curvas Bézier

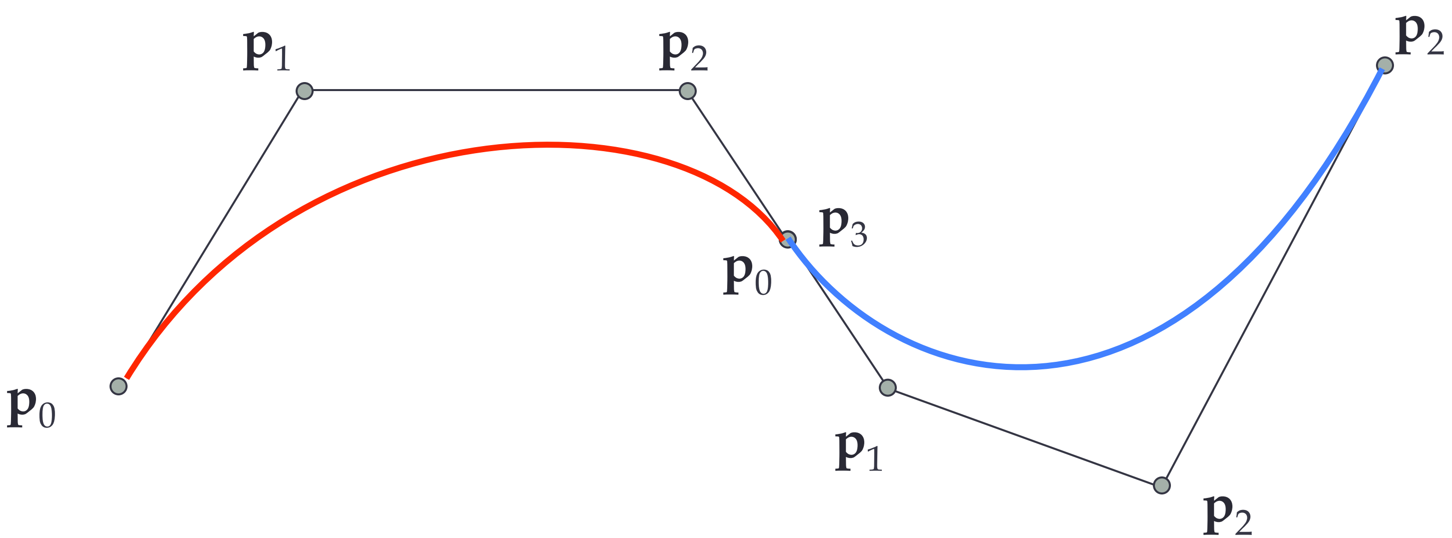
- Curva normalmente é aproximada por uma linha poligonal
- Pontos podem ser obtidos avaliando a curva em t = $t_1, t_2 \ldots t_k$
  - Avaliar os polinômios de Bernstein
  - Usar o algoritmo recursivo de De Casteljau
- Quantos pontos?
  - Mais pontos em regiões de alta curvatura
- Idéia: subdividir recursivamente a curva em trechos até que cada trecho seja aproximadamente "reto"

Curvas Longas

- Curvas Bézier com k pts de controle são de grau k–1
  - Curvas de grau alto são difíceis de desenhar
  - Complexas
  - Sujeitas a erros de precisão
- Normalmente, queremos que pontos de controle tenham efeito local
- Em curvas Bézier, todos os pontos de controle têm efeito global
- Solução:
  - Emendar curvas polinomiais de grau baixo
  - Relaxar condições de continuidade

Emendando curvas Bézier

- Continuidade C^0: último ponto da primeira = primeiro ponto da segunda
- Continuidade C^1: C^0 e segmento $\mathbf{p}_2\mathbf{p}_3$ da primeira com mesma direção e comprimento que o segmento $\mathbf{p}_0\mathbf{p}_1$ da segunda
- Continuidade C^2: C^1 e + restrições sobre pontos $\mathbf{p}_1$ da primeira e $\mathbf{p}_2$ da segunda



60

Tarefa de casa

- Leitura livro-texto
  - Shirley and Marschner. Fundamentals of Computer Graphics, CRC Press, 3rd Ed. 2010
  - Capítulo 15 até seção 15.6.1