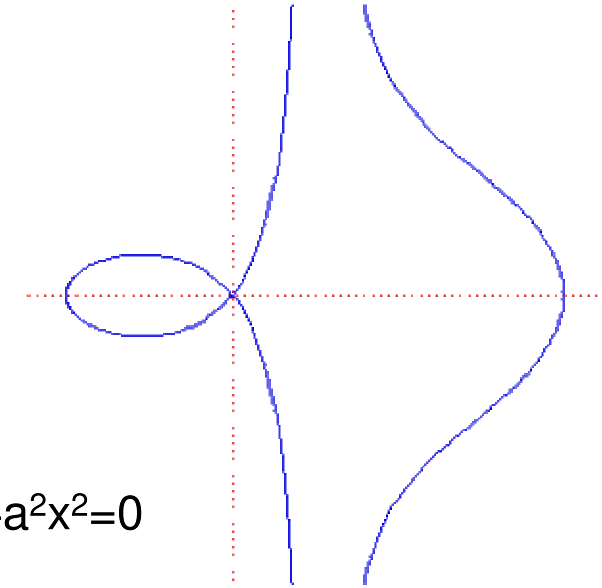
http://computacaografica.ic.uff.br/conteudocap3.html

https://www.matematica.pt/util/curvas.php



aula 20

IC/UFF - 2019

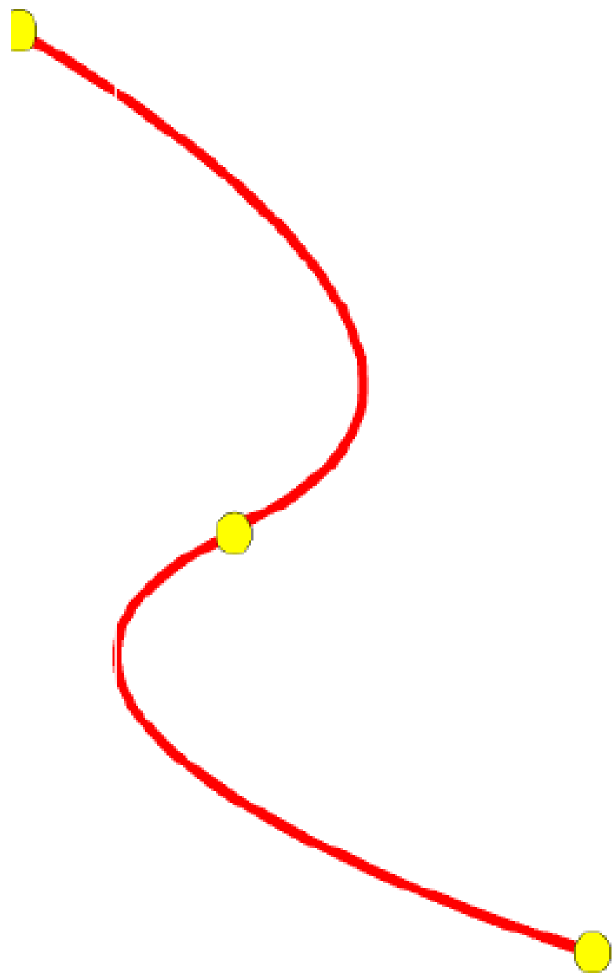
$(x-b)^2(x^2+y^2)-a^2x^2=0$

Curvas

A **Concoide** foi inventada por **Nicomedes** por volta do ano **200 A.C**. numa tentativa de duplicação do cubo através de métodos geométricos.

Elementos 1D

- Comprimento
- Distância ao inicio define a posição na curva
- Mas ela pode ser 2D e 3D



Curvas

- Formas de representação:
  - Procedural (não tem equação apenas algoritmo de geração:
    - exemplo *curvas fractais*)
  - Conjunto de pontos (digitalizados: x_i , y_i)
  - Por equações (analíticas):
    - Explicita : $y = f(x)$
    - Implícita : $x+y=0$
    - Paramétrica : $x= f(t)$, $y = f(t)$

Também podem ser

Classificadas de acordo com seus termos: linear (grau 1), quadrática (grau 2), cúbica (grau 3), transcendental (sin, cos, log, ...)

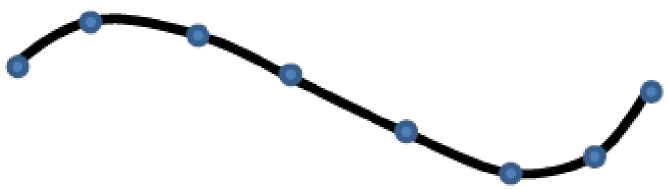
Representação analítica

- Não paramétrica e paramétrica
- Precisão sobre representação de ponto
- Armazenamento compacto
  - Centro do círculo e raio vs. pontos
- Ponto intermediário
  - Quaisquer pontos sobre a curva podem ser calculados
- Mais fácil gerar desenhos
- Mais fácil mudar a curvatura

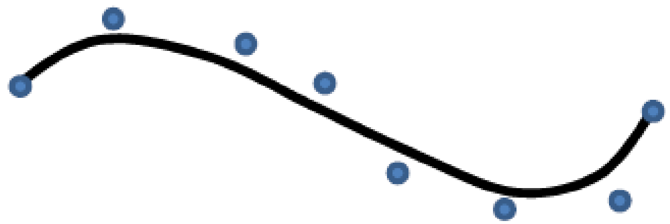
Representação analítica de curva definida por ponto

- Interpolação
  - Analiticamente definindo uma curva a partir de um conjunto de pontos conhecido
- Ajustada
  - Uma curva que passa por todos os pontos conhecidos
- Satisfatória
  - Uma curva que passa perto de pontos conhecidos

Interpolação X Aproximação

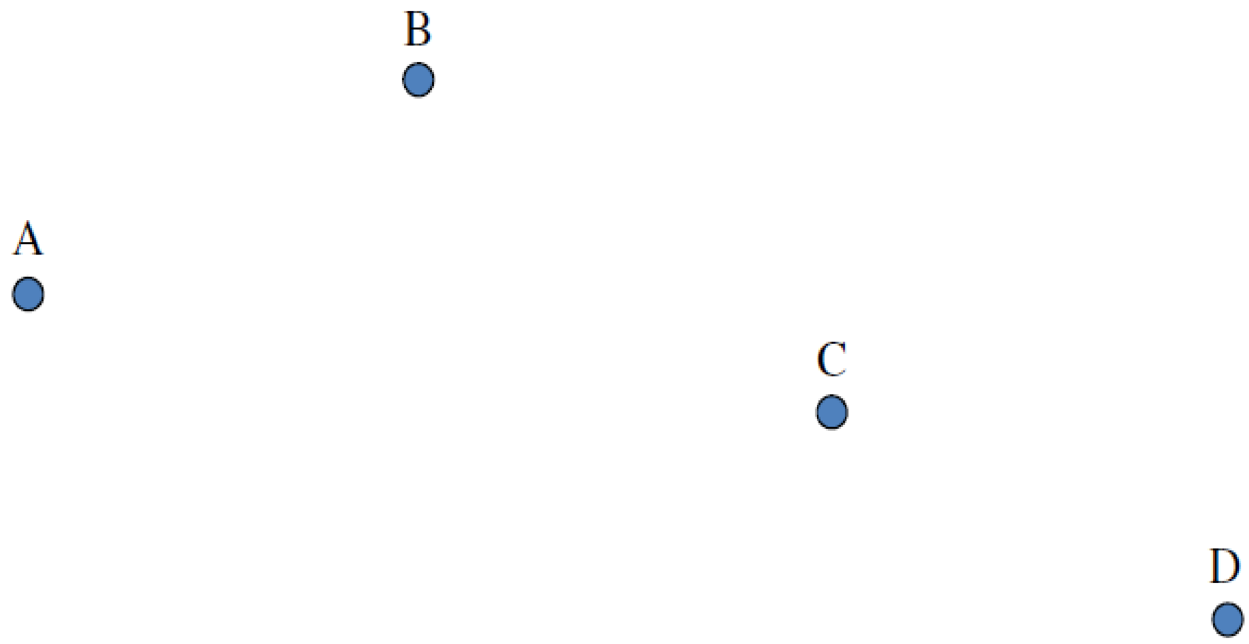


Na interpolação, a curva passa sobre todos os pontos definidos.

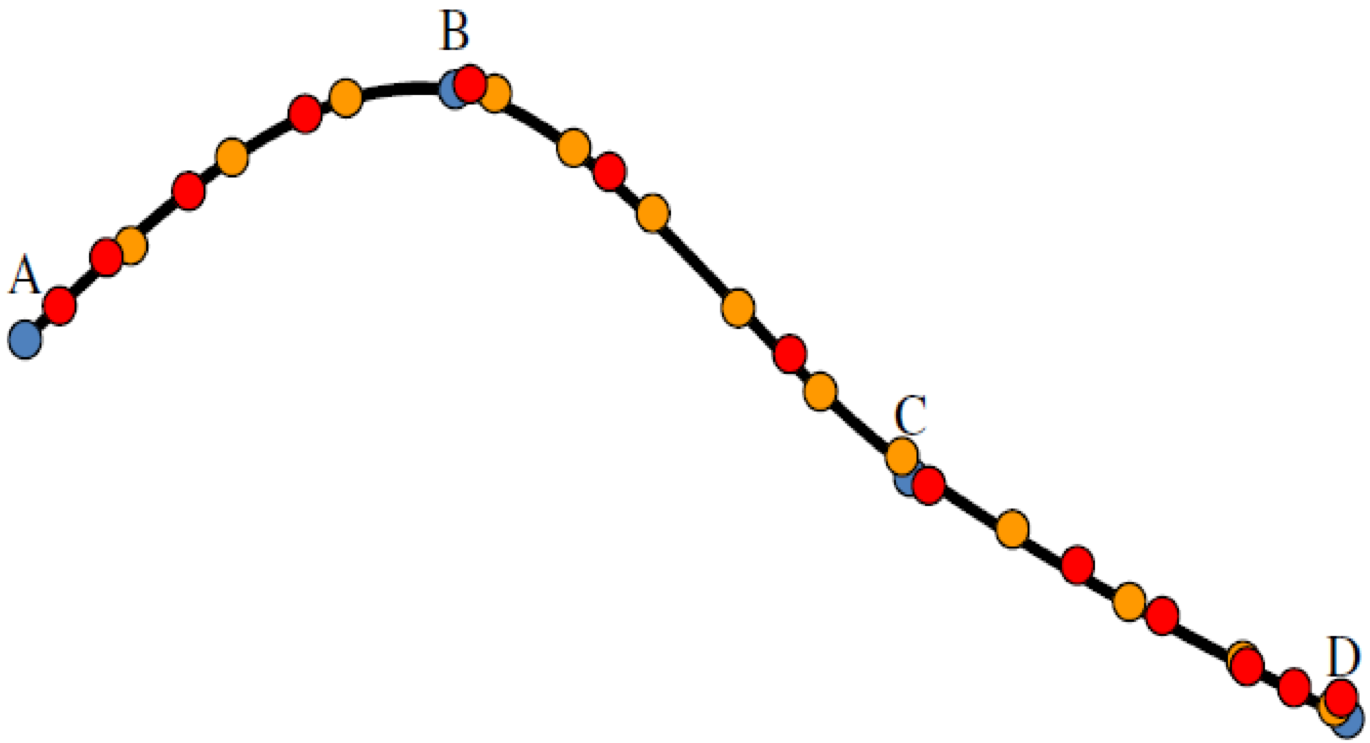


Na aproximação, a curva começa sobre o ponto inicial e termina sobre o final. Os demais pontos são aproximados.

Gerar uma curva **suave** que passe por **pontos** específicos



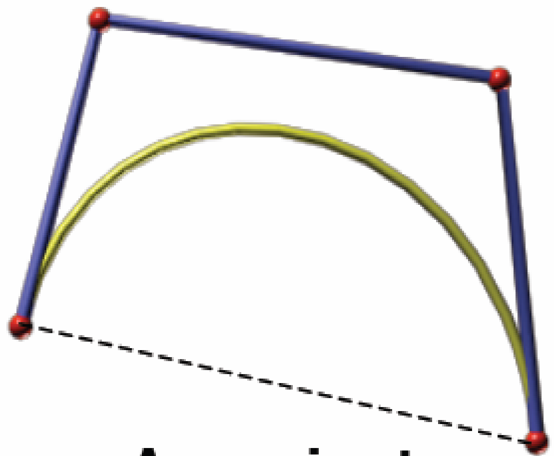
gerar uma curva no espaço, distribuindo **pontos** de maneira **suave**



Dado um número *n* de pontos para traçar uma curva:

- ***interpolar*** os pontos (curva passa *necessariamente* por todos os pontos)
- ***aproximar*** os pontos (pontos definem cobertura convexa (*convex hull)* da curva)

Interpolate



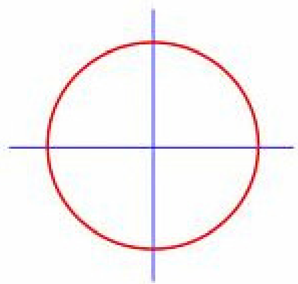
Approximate

Não paramétrica vs paramétrica

- Não paramétrica
  - Explícita y = f(x)
  - Implícita f(x, y) = 0
- Equação implícita de segundo grau geral

$$ax^2 + b2xy + cy^2 + 2dx + 2cy + f = 0$$

Exemplo circunferência representações não paramétricas



Explícita y = f(x)

$$y = \sqrt{1 - x^2}$$

$$y = -\sqrt{1 - x^2}$$

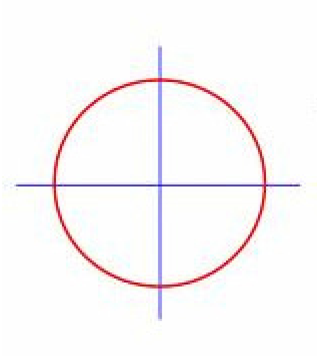
$$x^2 + y^2 - 1 = 0 \text{ or } x^2 + y^2 = 1$$

Implícita f(x, y) = 0

Exemplo : circunferência em representações paramétricas

$$f : [0, 2\pi] \longrightarrow \mathbb{R}^2,\ f(\theta) = (x(\theta), y(\theta))$$

$$\begin{cases} x = \sin\ \theta \\ y = \cos\theta \end{cases} \qquad \text{where } \theta \in [0, 2\pi]$$



$$\begin{cases} x = \dfrac{2t}{1+t^2} \\ y = \dfrac{1-t^2}{1+t^2} \end{cases} \qquad \text{where } t \in [0, 1]$$

E essas?

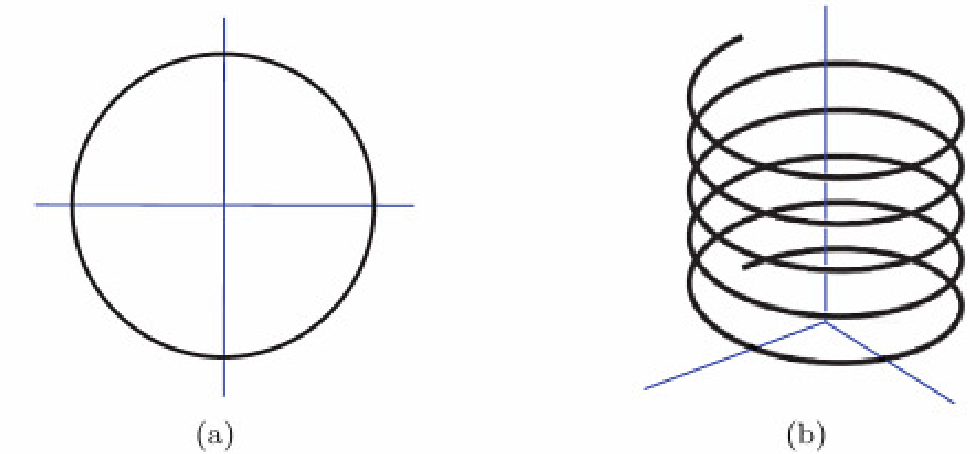
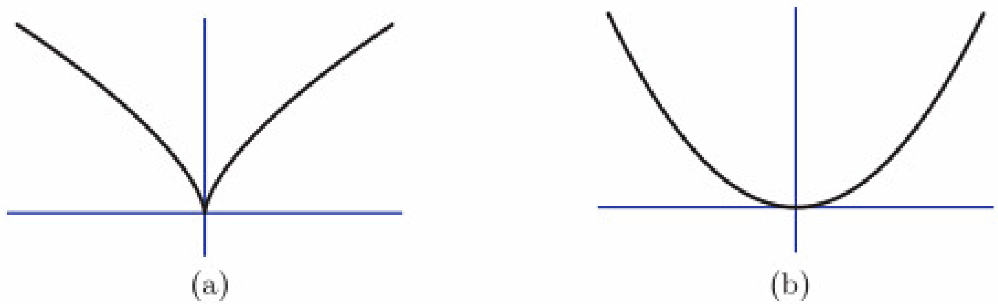


Fig. 1.1. (a) Image and (b) graph of $f(t) = (\cos t, \sin t)$.



. 1.2. (a) Cuspidal cubic $x^3 = y^2$ and (b) parabola $y = x^2$ as *images* of different ametrisations.

Explícita y = f(x)

Implícita f(x, y) = 0

Representação implícita

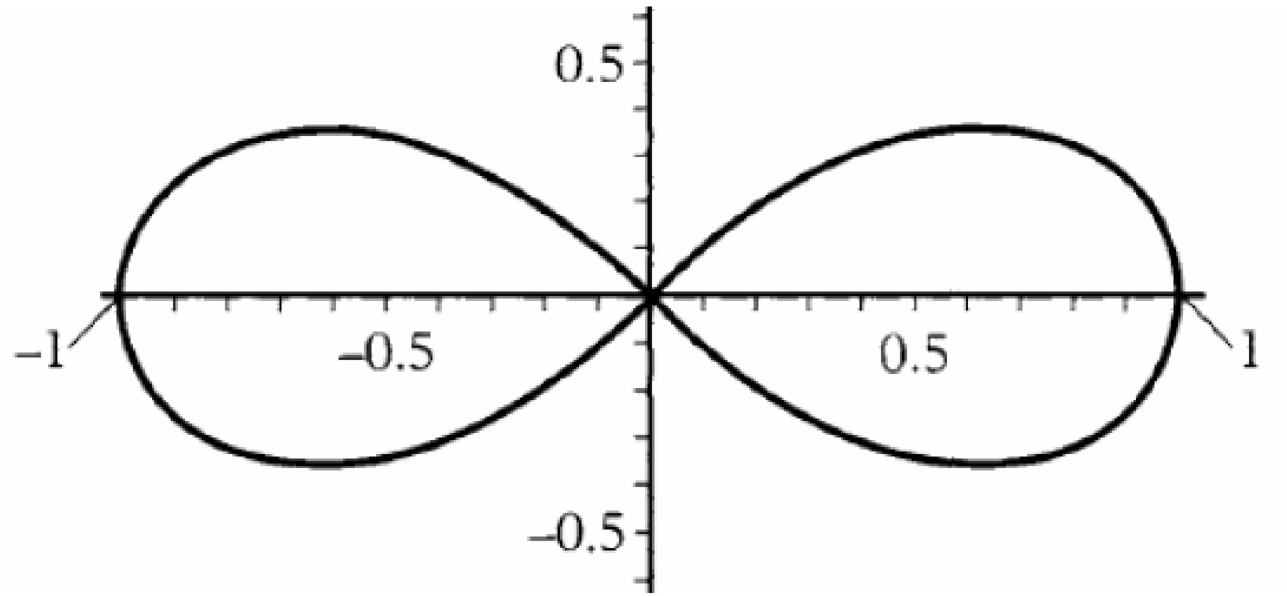
- Curva em 2D: $f(x,y) = 0$
  - Linha: $ax + by + c = 0$
  - Círculo: $x^2 + y^2 - r^2 = 0$
- Superfície em 3D: $f(x,y,z) = 0$
  - Plano: $ax + by + cz + d = 0$
  - Esfera: $x^2 + y^2 + z^2 - r^2 = 0$

Outros exemplos:

- Lemniscata de Bernoulli => símbolo infinito
- Quarto grau!

$(x^2+y^2)^2 - (x^2-y^2)^2 = 0$

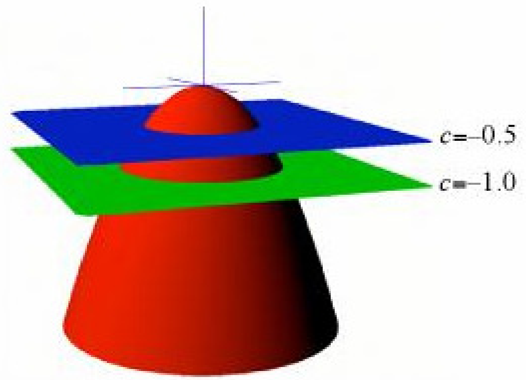
Implícita f(x, y) = 0



Curvas não paramétricas

- Linha
- Círculo
- Parábola
- Elipse
- Hipérbole

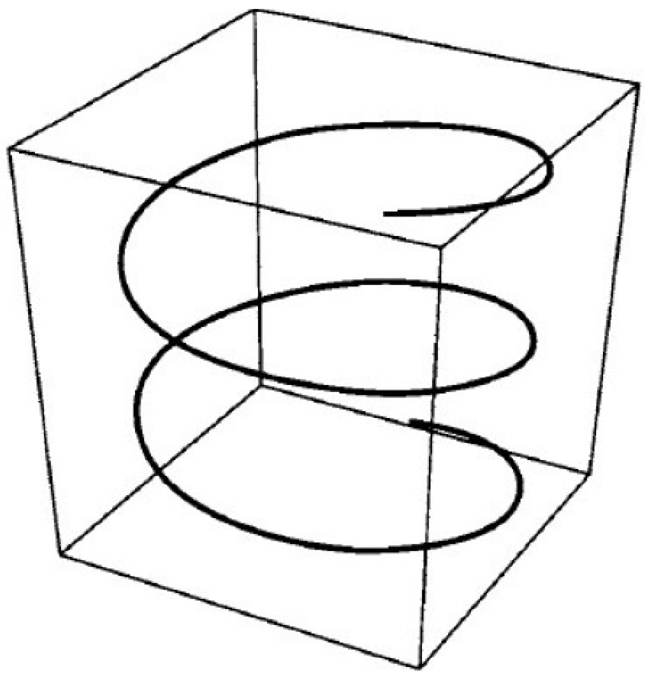
ı. The equation $-z = x^2 + y^2$ *explicitly* defines the paraboloid in $\mathbb{R}^3$.



Curvas paramétricas

– Pontos sobre uma curva são representados com uma função de um único parâmetro
  - $x = f(u)$, $y = g(u)$, $z = h(u)$
    - u : variável paramétrica

$x = \cos(t)$, $y = \sin(t)$, $z = t/5$

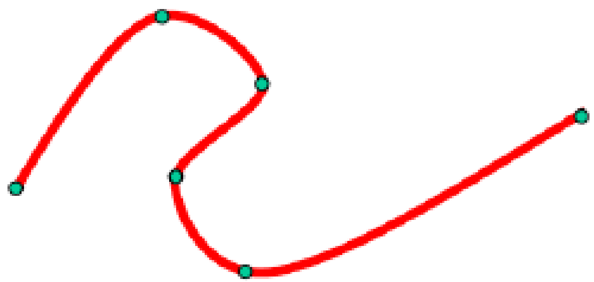


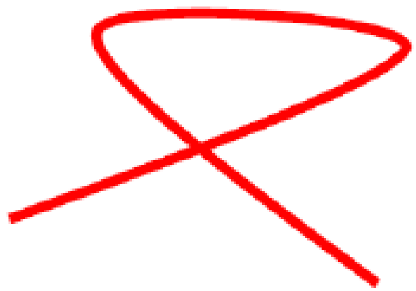
Peculiaridades das curvas em CG

Principais desvantagens das representações **não-paramétrica** em CG

- É difícil definir a equação não paramétrica de uma curva que passe por um conjunto de pontos pré-definidos.

- Não permite a representação de curvas com laços





Peculiaridades das curvas em CG

Para CG, representações paramétricas costumam ser as mais convenientes

Assim, genericamente, uma curva 3D é

- $Q(t)=[x(t) \quad y(t) \quad z(t)]$

$x(t)$, $y(t)$, $z(t)$ são chamadas de funções-base (*base functions*)

$$P(u) = (X(u), Y(u), Z(u))$$
$$0 \leq u \leq 1$$

$$P(u,v) = (X(u,v), Y(u,v), Z(u,v))$$
$$0 \leq u \leq 1$$
$$0 \leq v \leq 1$$

- As expressões paramétricas suportam declives infinitos, curvas fechadas ou multi-valor.

dy/dx = (dy/du) / (dx/du)

dy/dx = infinito => dx/du = 0

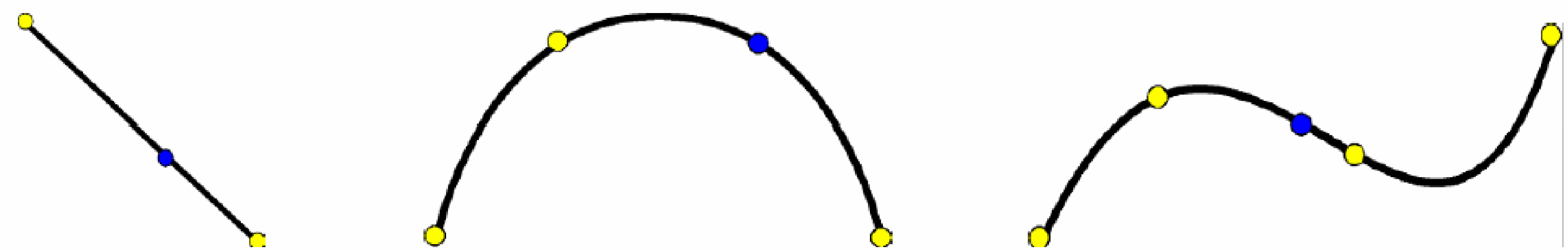
Reta na forma paramétrica

$$P(t) = P_0 + at$$

- $P_x = P_{x0} + at$
- $P_y = P_{y0} + at$
- $P_z = P_{z0} + at$

Parametrizando polinômios

$f(t) = at + b$ $\qquad$ $f(t) = at^2 + bt + c$ $\qquad$ $f(t) = at^3 + bt^2 + ct + d$



Linear $\qquad$ Quadrático $\qquad$ Cúbico

- Elementos geométricos definidos parametricamente são inerentemente limitados (0 <= u <= 1).

- As expressões paramétricas são facilmente traduzidas na forma de vectores e matrizes.

- Utilização de um só modelo matemático para representar qualquer curva ou superfície.

Porque as curvas em CG são especiais?

Curvas de Hermite, Splines e Bezier - > Formas livres

Peculiaridades das curvas em CG

Para CG, representações paramétricas costumam ser as mais convenientes

Assim, genericamente, uma curva 3D é

- $Q(t)=[x(t) \;\; y(t) \;\; z(t)]$

$x(t)$, $y(t)$, $z(t)$ são chamadas de funções-base (*base functions*)

$$P(u) = (X(u), Y(u), Z(u))$$ $\Rightarrow$ curvas em CG

$$0 \leq u \leq 1$$

$$P(u,v) = (X(u,v), Y(u,v), Z(u,v))$$ $\Rightarrow$ superfícies em CG

$$0 \leq u \leq 1$$

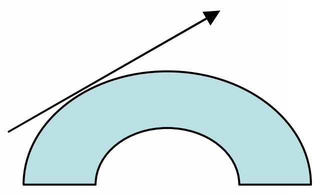
$$0 \leq v \leq 1$$

É positivo nas paramétricas:

- As expressões paramétricas suportam declives infinitos, curvas fechadas ou multi-valor.

dy/dx = (dy/du) / (dx/du) $\Rightarrow$

dy/dx = infinito => dx/du = 0



- Elementos geométricos definidos parametricamente são inerentemente limitados (0 <= u <= 1).

- As expressões paramétricas são facilmente traduzidas na forma de vectores e matrizes.

Elas permitem a:

- Utilização de um só modelo matemático para representar qualquer curva ou superfície.

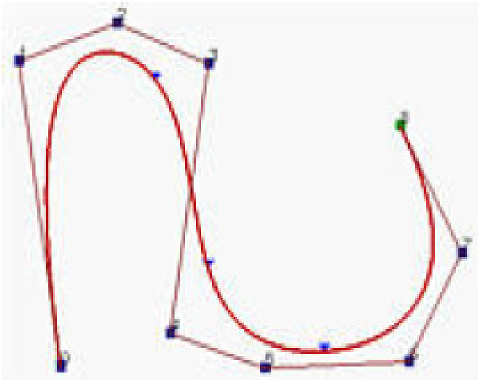
Peculiaridades das curvas em CG

A curva é definida através de um conjunto de **pontos de controle** que influenciam a forma da curva.

Os nós são pontos de controle que pertencem à curva.

A curva pode ser interpolada, passando nesse caso por todos os pontos de controle, ou pode ser aproximada, passando apenas em alguns pontos de controle ou mesmo nenhum.

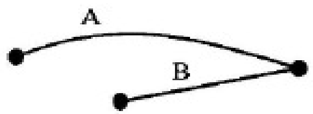
Os pontos de controle definem a fronteira de um polígono designado por *convex hull*.



as funções de base devem permitir que os desenhos tenham

Continuidade

- Duas ou mais curvas nos nós (conectando pontos) para formar uma curva contínua
- Tipos de continuidade
  - Continuidade de ponto
  - Continuidade de tangente
  - Continuidade de curvatura



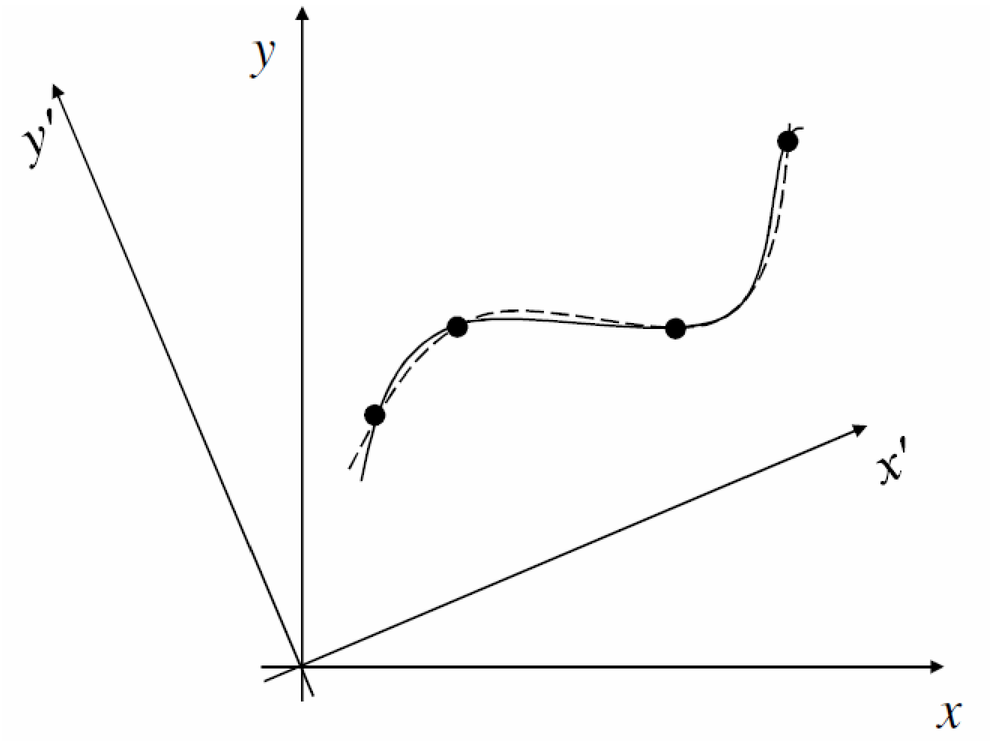
(a) Point continuity - C^0

(b) Tangent continuity - C^1

(c) Curvature continuity - C^2

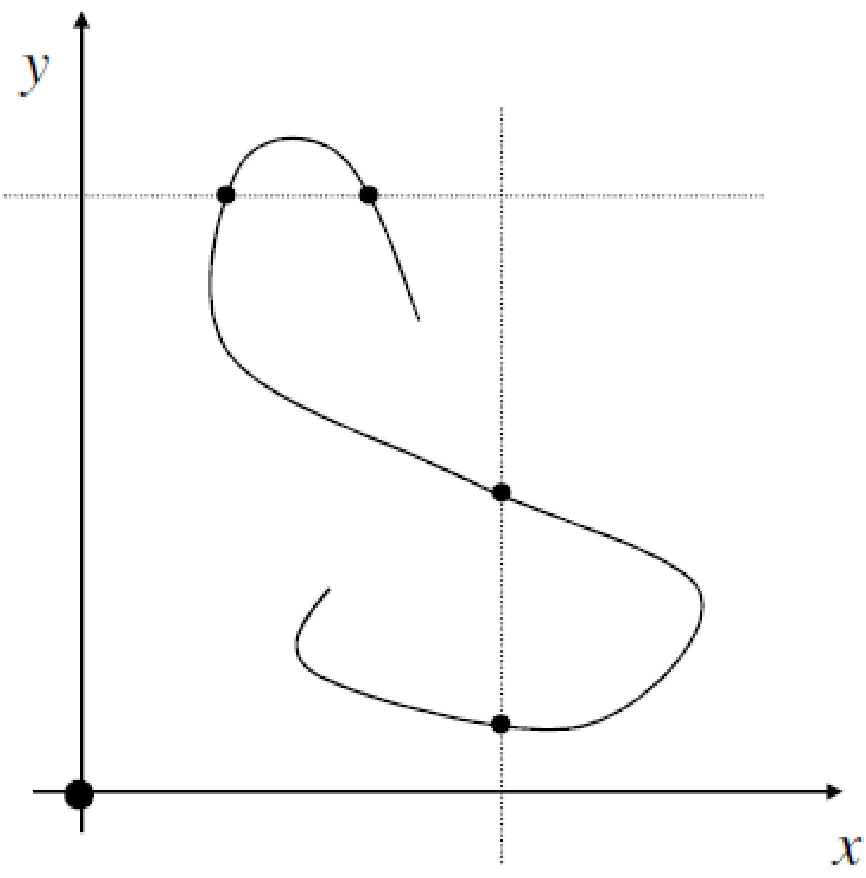
Propriedades desejáveis de curvas para modelagem em CG

Independência dos eixos usados



Propriedades desejáveis de curvas para modelagem em CG

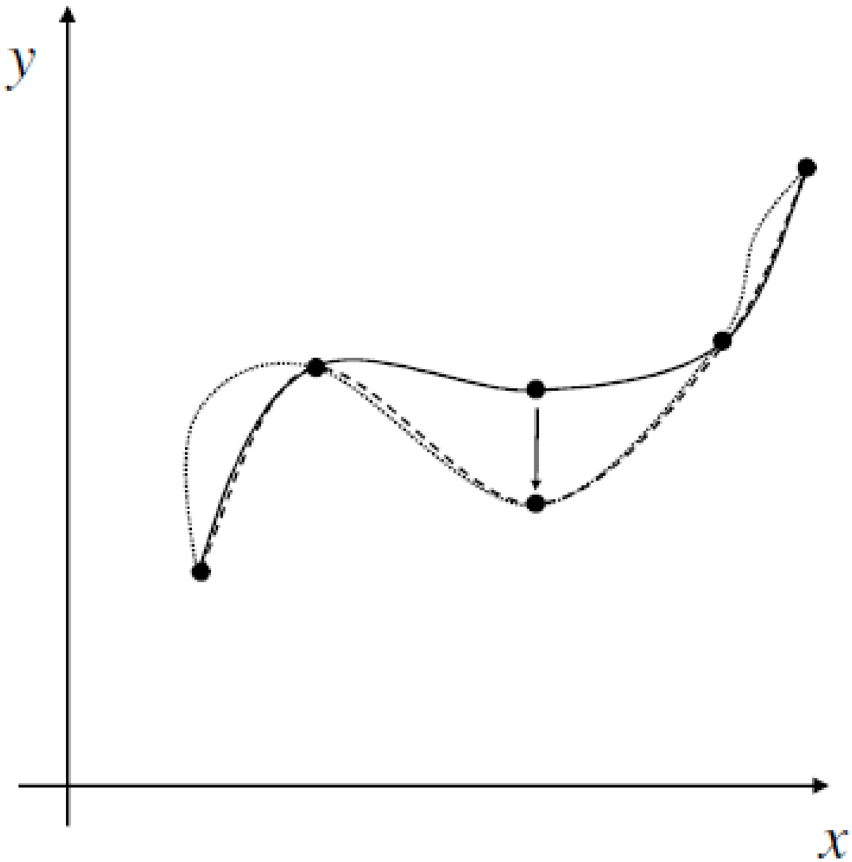
Deve poder ter Pontos com coordenadas Múltiplas em **x** ou **y**



Propriedades desejáveis de curvas para modelagem em CG

Devem ter uso **intuitivo** e poder ter **Controle local**:

i.e. possibilitar haverem **ajustes finos** (**e.g.** ao se alterar um trecho não altera toda a curva)

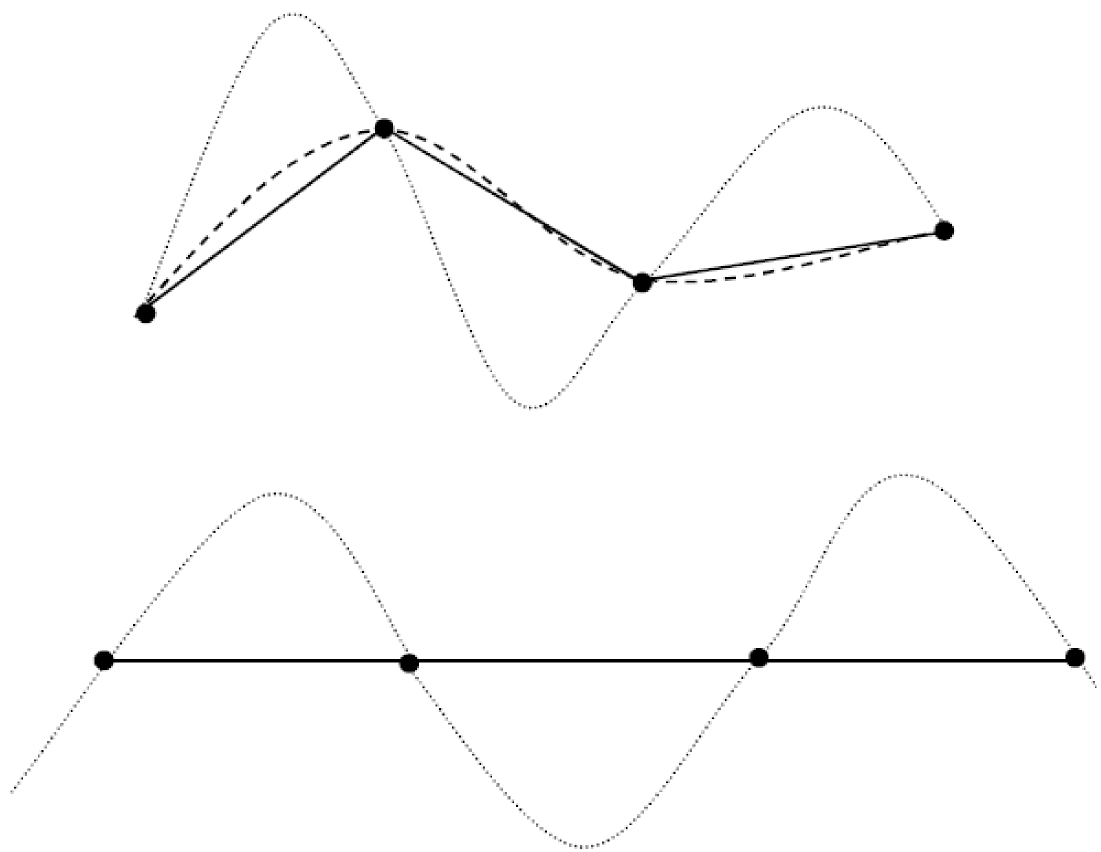


Propriedades desejáveis de curvas para modelagem em CG

- O número de pontos de

Controle local

  não deve estar associado **ao grau da curva** ou sua oscilação

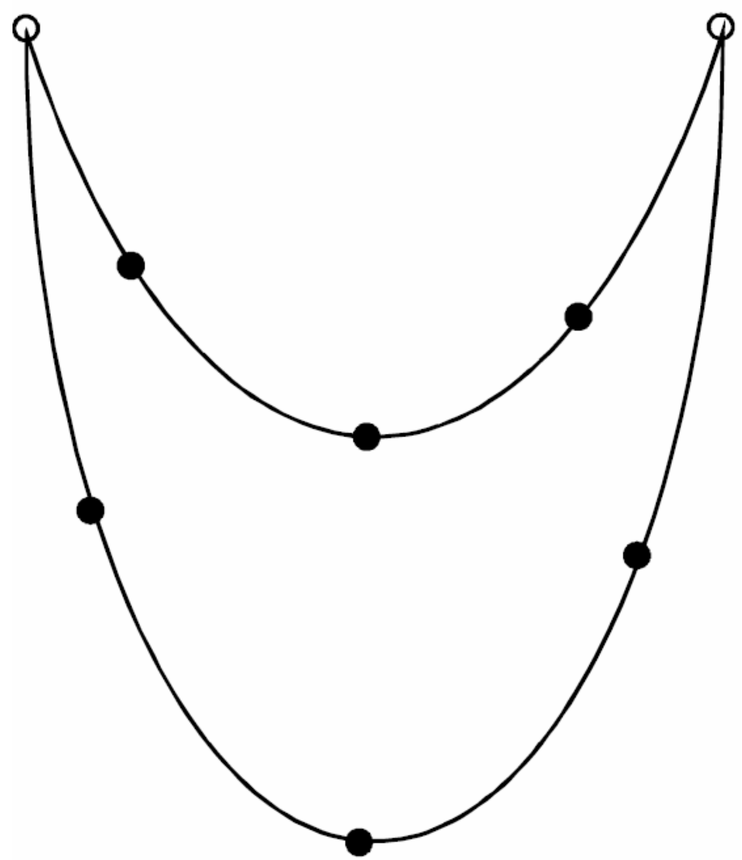


polinômio de grau elevado

formas com oscilações.

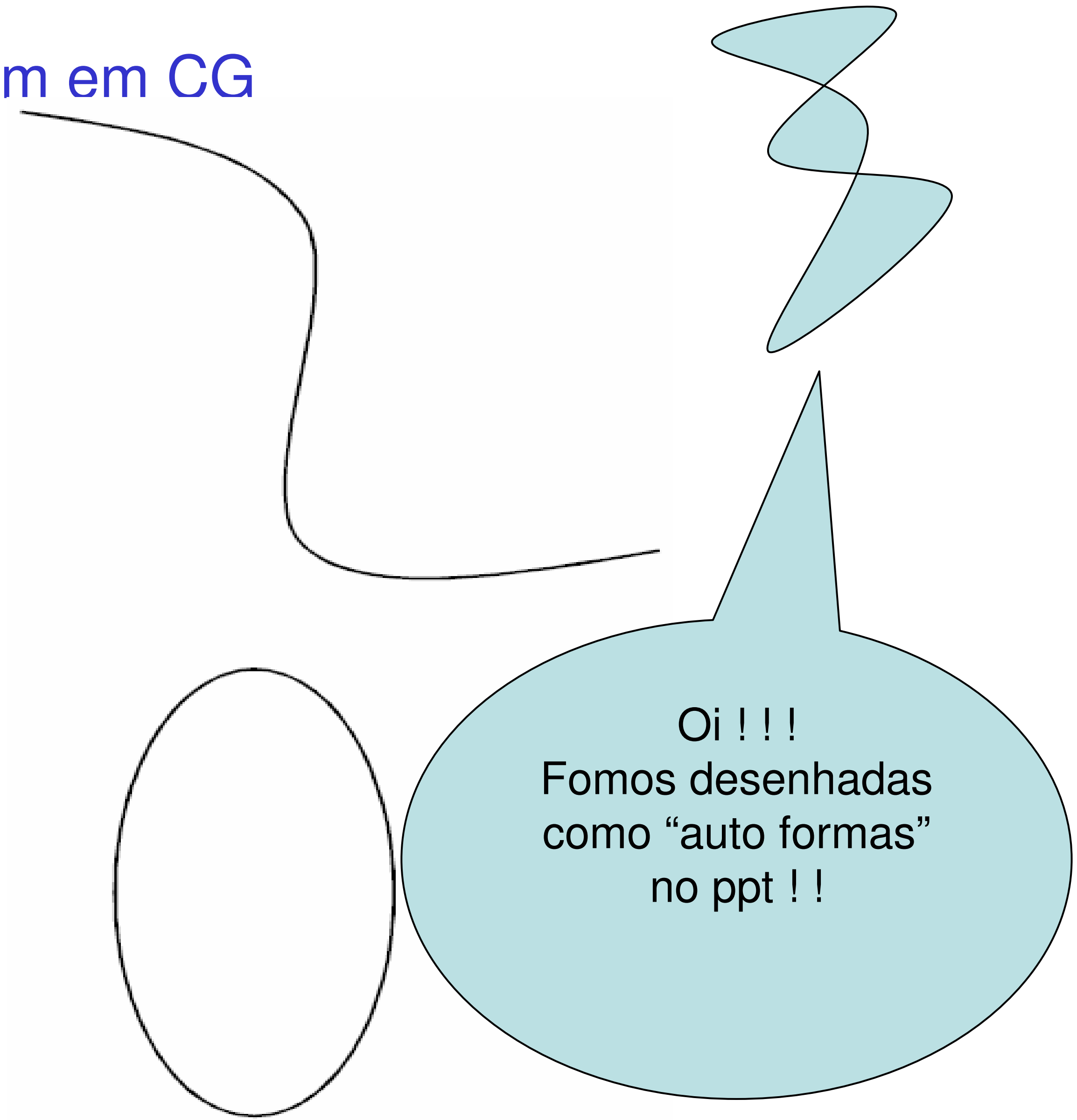
Propriedades desejáveis de curvas para modelagem em CG

- Ser possível representar diversos graus de continuidades que o usuário desejar



Propriedades desejáveis de curvas para modelagem em CG

- Ser possível representar curvas abertas, fechadas, com pontos de inflexão, etc. : ter a **versatilidade** que o usuário desejar



Propriedades desejáveis de curvas para modelagem em CG

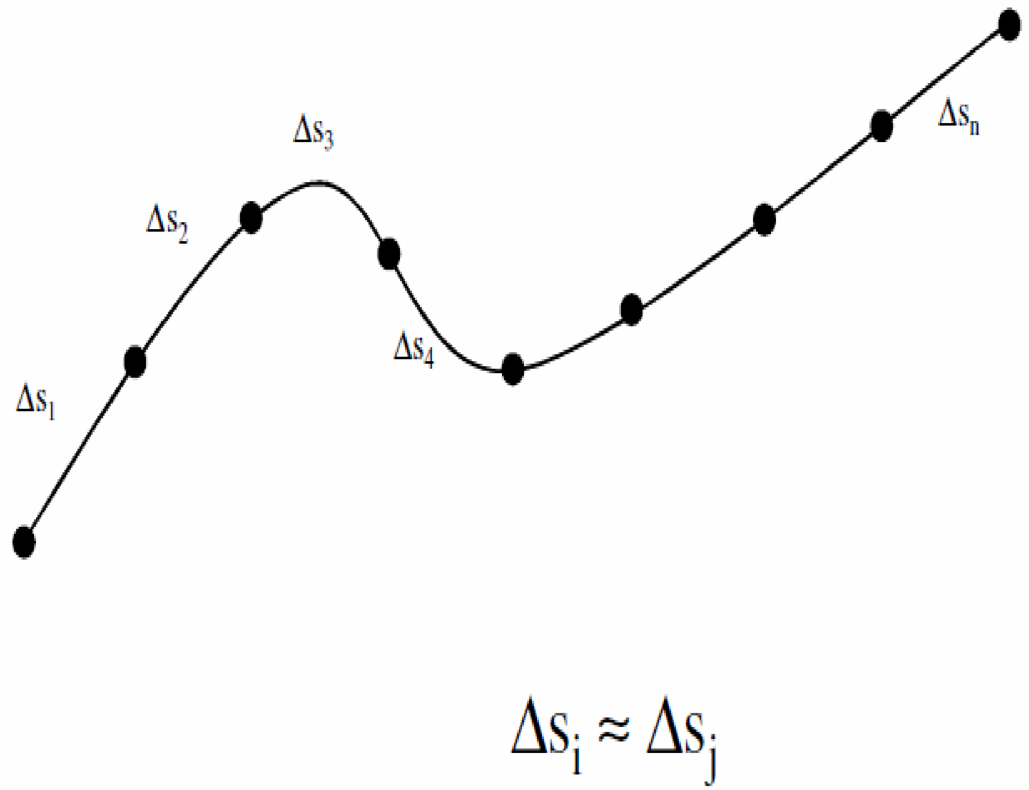
ter pontos

com distâncias ≈ constantes ao longo do seu

comprimento = parâmetro

uniformemente

distribuídos.



Solução desenvolvida para em CG

Curvas de formas livres

Representadas por uniões

Descritas por polinômios

Parametrizados

Até o grau 3

Com continuidade paramétrica

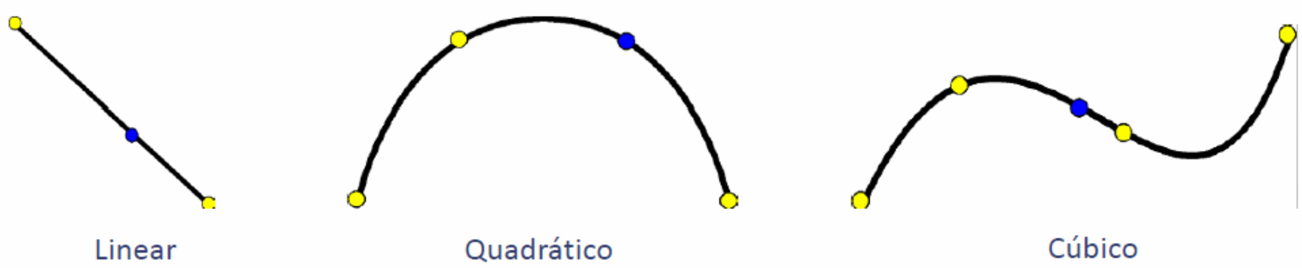
Porque polinômios até terceiro grau?

Quanto mais simples a equação da função de interpolação, mais rápida sua avaliação.

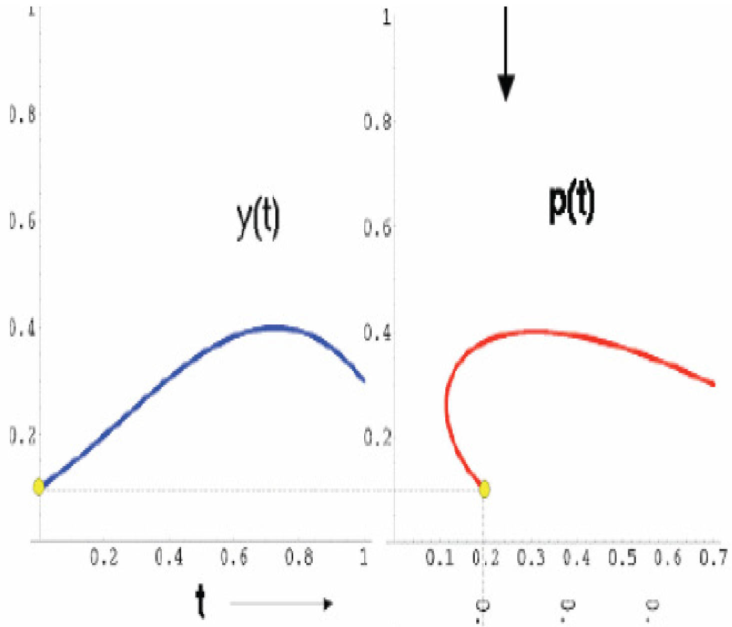
Polinômios são fáceis de avaliar
Grau menor que 3: Pouca flexibilidade.

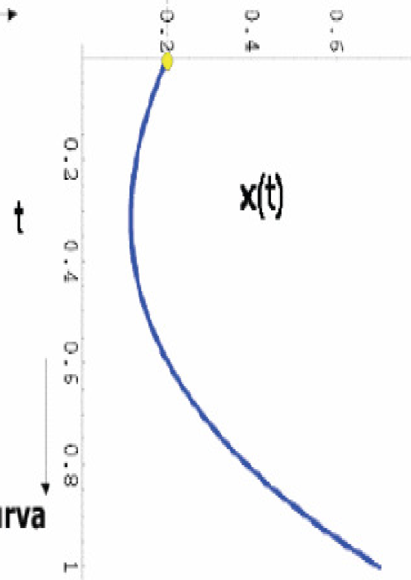
Grau maior que 3: Maior custo computacional com pouca vantagem prática.

$f(t) = at + b$ $\qquad$ $f(t) = at^2 + bt + c$ $\qquad$ $f(t) = at^3 + bt^2 + ct + d$



Curvas Paramétricas Cúbicas 2D





8 parâmetros para cada curva

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$

$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

Os valores de ax,bx, etc devem ser definidos para cada curva

Curva polinomial paramétrica:

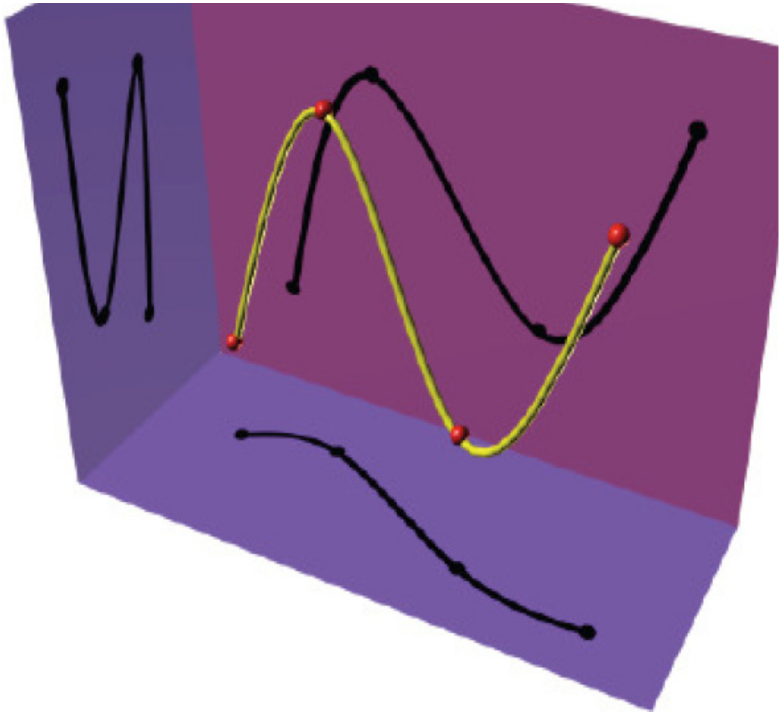
$$P(u) = c_0 + c_1 u + c_2 u^2 + \ldots + c_k u^k$$

Em 3D

$$\begin{cases} x(u) = c_{0x} + c_{1x} u + c_{2x} u^2 + \ldots + c_{kx} u^k \\ y(u) = c_{0y} + c_{1y} u + c_{2y} u^2 + \ldots + c_{ky} u^k \\ z(u) = c_{0z} + c_{1z} u + c_{2z} u^2 + \ldots + c_{kz} u^k \end{cases}$$

Em 3D

12 parâmetros para cada curva



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

De forma genérica

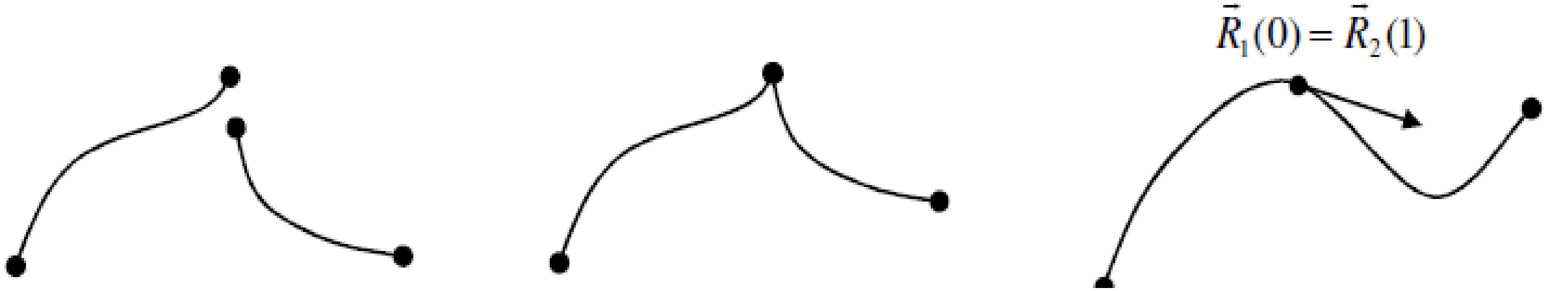
$$\mathrm{p}(u) = c_0 + c_1 u + c_2 u^2 + c_3 u^3 = \sum_{k=0}^{3} c_k u^k$$

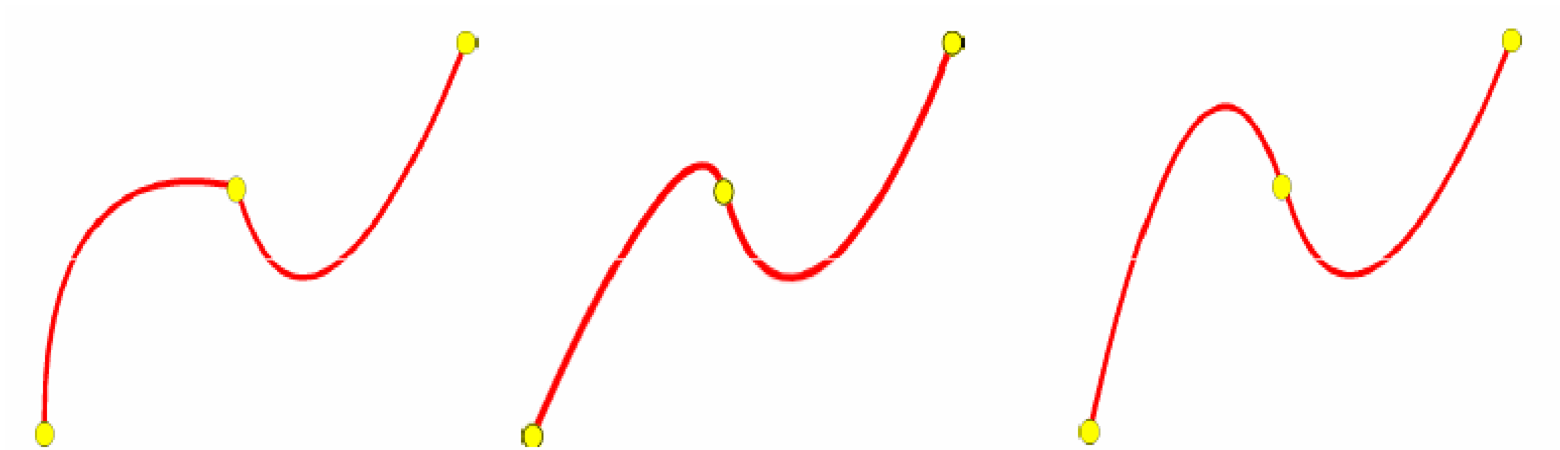
onde

$$\mathbf{c} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} 1 \\ u \\ u^2 \\ u^3 \end{bmatrix}, \qquad \mathbf{c}_k = \begin{bmatrix} c_{kx} \\ c_{ky} \\ c_{kz} \end{bmatrix}$$

Graus de Continuidade

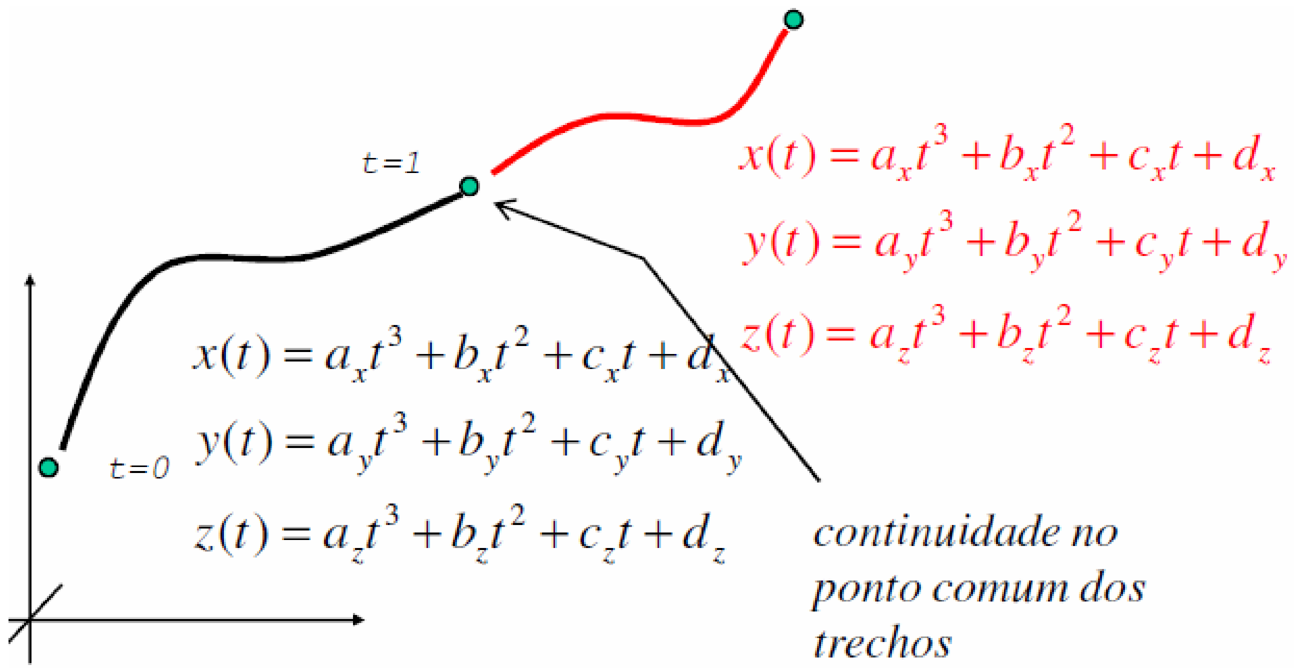
geométrica





Dois segmentos se encontram em um ponto	**Direção** das tangentes dos segmentos são iguais no ponto de junção	**Direção** e **magnitude** das tangentes dos segmentos são iguais no ponto de junção

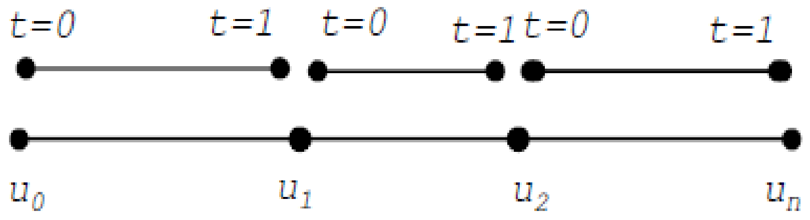
Com continuidade paramétrica



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

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

Parametrização



$t \in [0,1]$ *local*

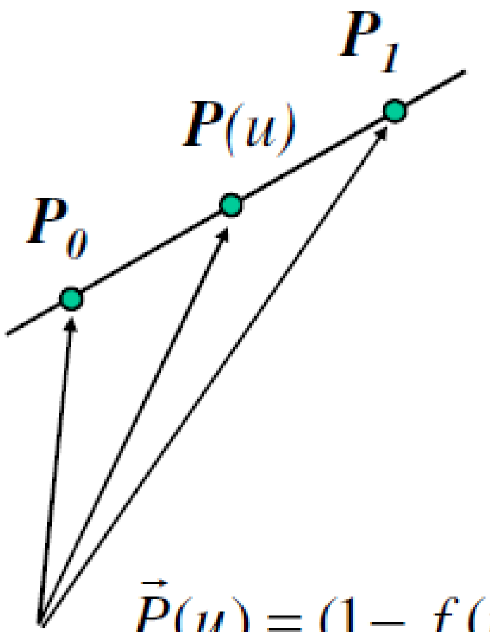
ou

$u \in [u_0, u_n]$ *global*

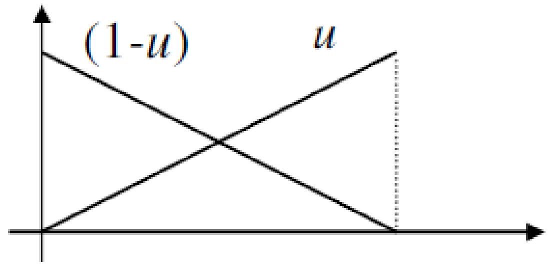
Requisitos para os parâmetros:

Parâmetro genérico: u

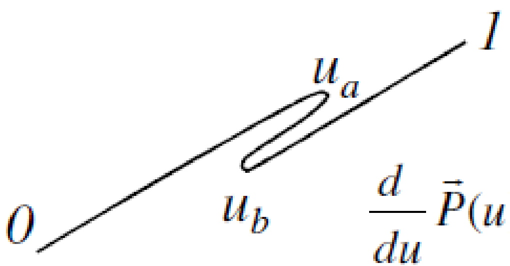
Parâmetro de comprimento: $s = s(u)$



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



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



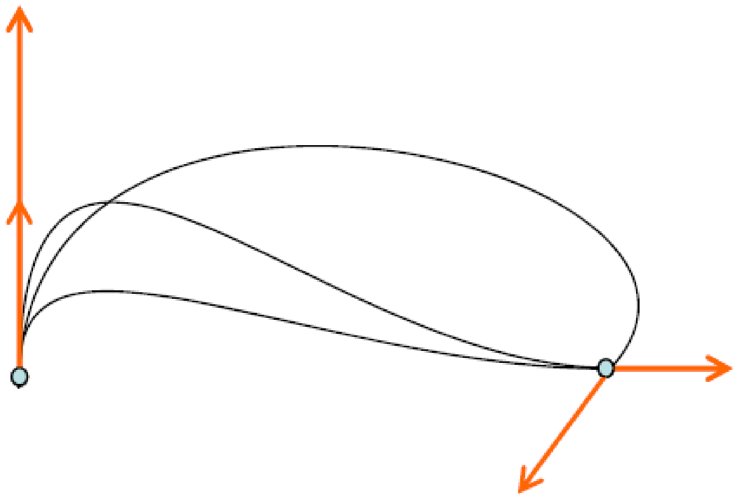
$$\frac{d}{du}\vec{P}(u) = 0$$

Com continuidade paramétrica

$$\text{Se } u_2 > u_1 \Rightarrow s(u_2) > s(u_1)$$

Curvas de Hermite

- Ao invés de modelar a curva a partir de um polígono de controle (Bézier), especifica-se pontos de controle e vetores tangentes nesses pontos
- Vantagem: é fácil emendar várias curvas bastando especificar tangentes iguais nos pontos de emenda
- Exemplos (cúbicas):



Curvas de Hermite

https://pt.wikipedia.org/wiki/Charles_Hermite

Criadas por Charles Hermite (1822-1901)

tem como idéia básica o comando por Vetores

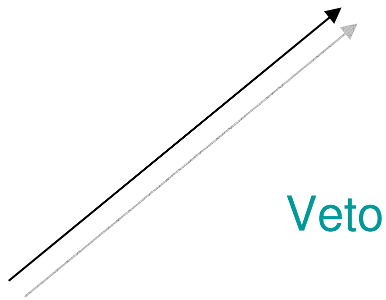
mas qual tipo de vetores?? :

Na matemática - um elemento com de um espaço vetorial

Em Física – em oposição as grandezas escalares, algo que se caracteriza por ter intensidade, sentido , direção e ponto de aplicação em engenharia e outras ciências

Computação – arranjo unidimensional - estrutura de dados utilizada no contexto da programação.

Epidemiologia - um agente de disseminação de doenças infecto-contagiosas



Vetor livre

Curva de Hermite

- A curva de Hermite é determinada por restrições no primeiro e último pontos
- O usuário especifica o primeiro (P_1) e o último pontos (P_4) bem como os vetores tangentes a P_1 e P_4, chamados R_1 e R_4

pontos de controle = *Pi*

Curvas de Hermite

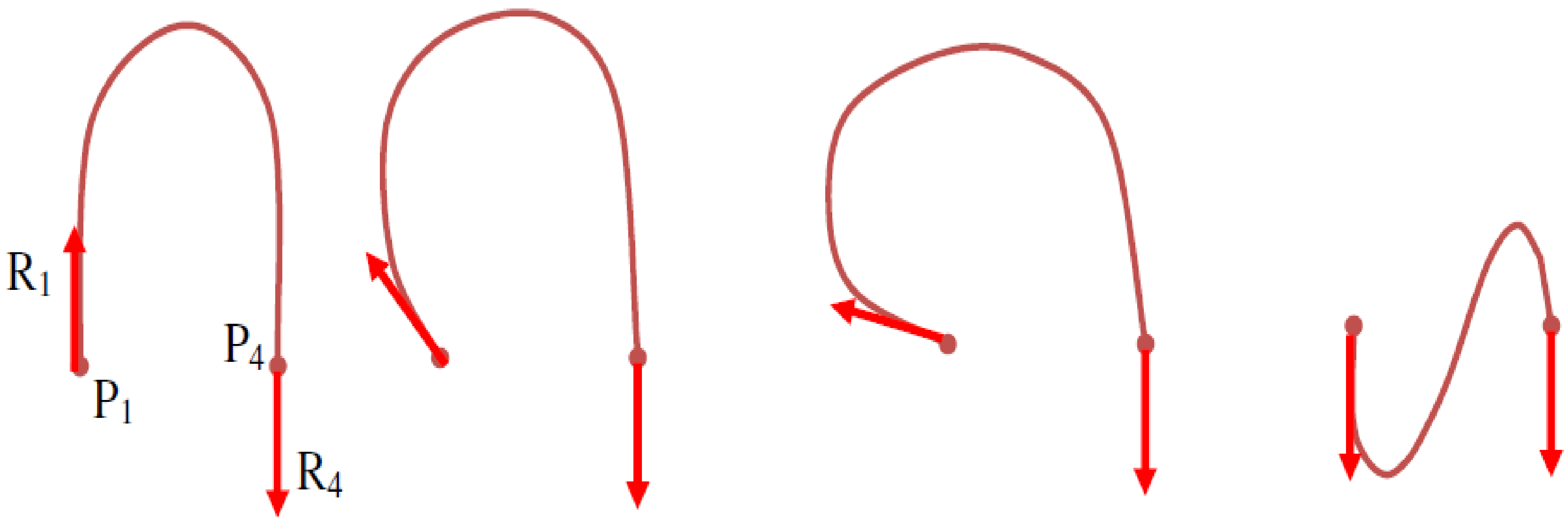
Definida a partir de restrições no ponto inicial e no ponto final.

– Os pontos propriamente ditos: P0 e P1

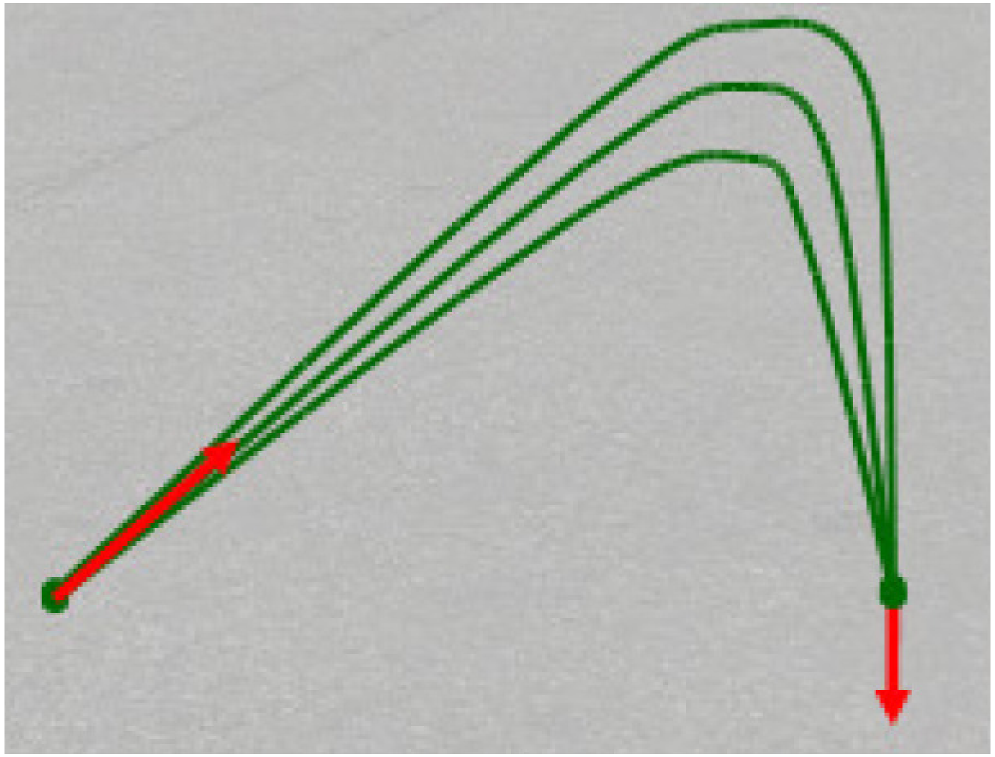
– Vetores tangentes nestes pontos: V0 e V1

pontos de controle = *Pi*

Curvas de Hermite com os mesmos pontos iniciais e finais, apenas alterando a direção da tangente



Curvas de Hermite com os mesmos pontos iniciais e finais, apenas alterando a intensidade da tangente



Vantagens

- Bem fácil de implementar ☺
- Adequada para aplicações onde seja útil definir a curva em função dos vetores tangentes
- Passa nos pontos de controle (interpolação)

Desvantagens

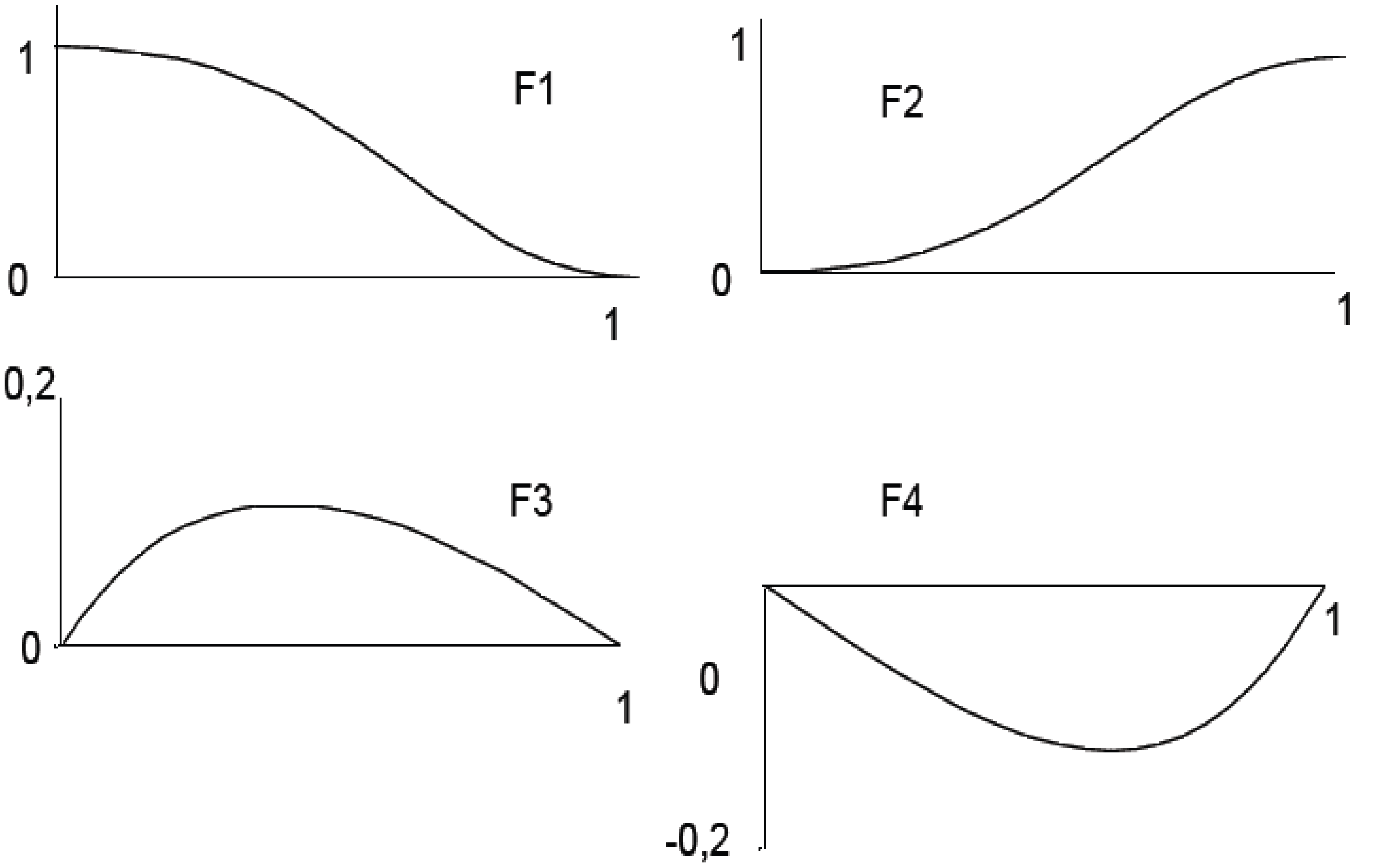
- Não garante, de forma automática, a continuidade entre os segmentos de curva
  - É necessário os vetores tangentes terem a mesma direção e sentido
- Não permite controle local
  - Alteração de um ponto de controle altera toda a curva

Forma matricial

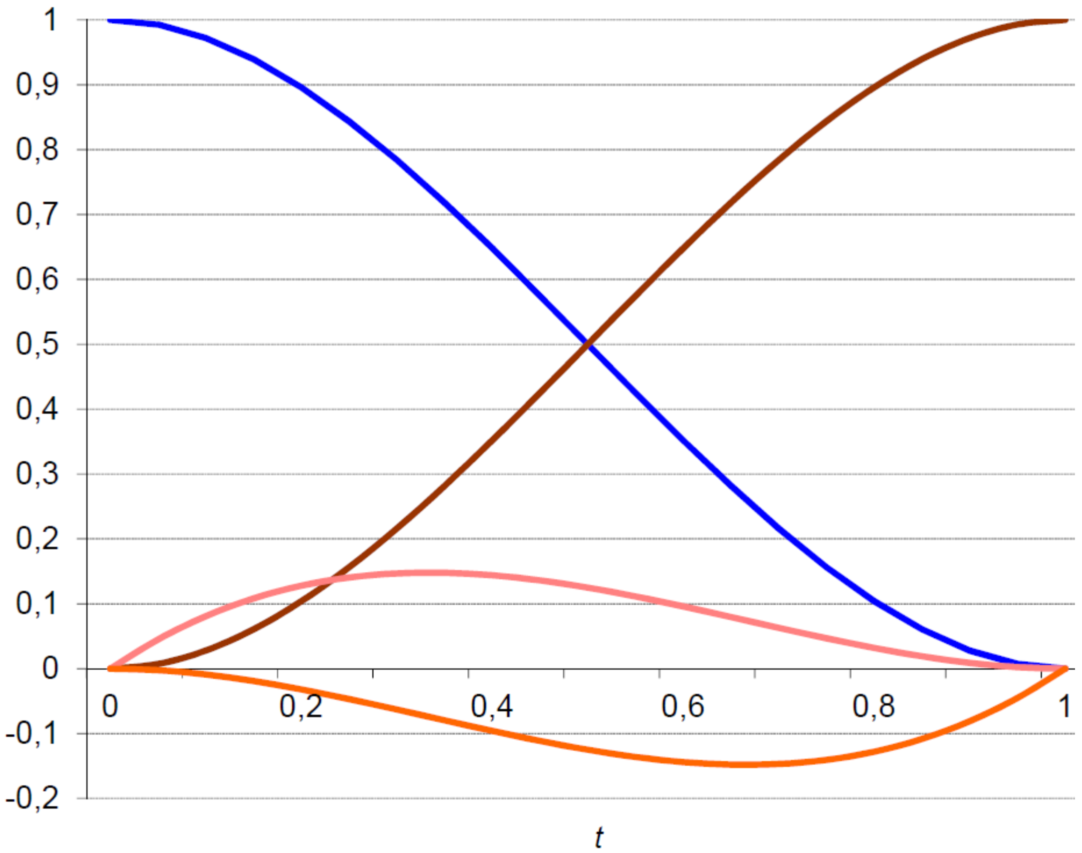
$$P(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P(0) \\ P(1) \\ P'(0) \\ P'(1) \end{bmatrix}$$

pontos de controle = *P(0) e P(1)*

Funções de mistura de Hermite



Funções de mistura ou funções interpoladoras de Hermite



Funções de Mistura:

$F1(u) = (2u^3 - 3u^2 + 1)$

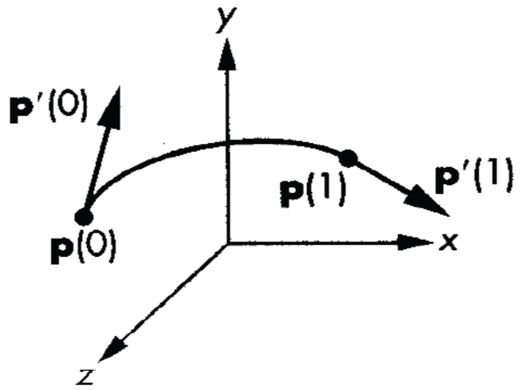
$F2(u) = (-2u^3 + 3u^2)$

$F3(u) = (u^3 - 2u^2 + u)$

$F4(u) = (u^3 - u^2)$

$$p(u) = F1.p(0) + F2.p(1) + F3.p'(0) + F4.p'(1)$$

pontos de controle = *p e p´*



pontos de controle = *p e p´*

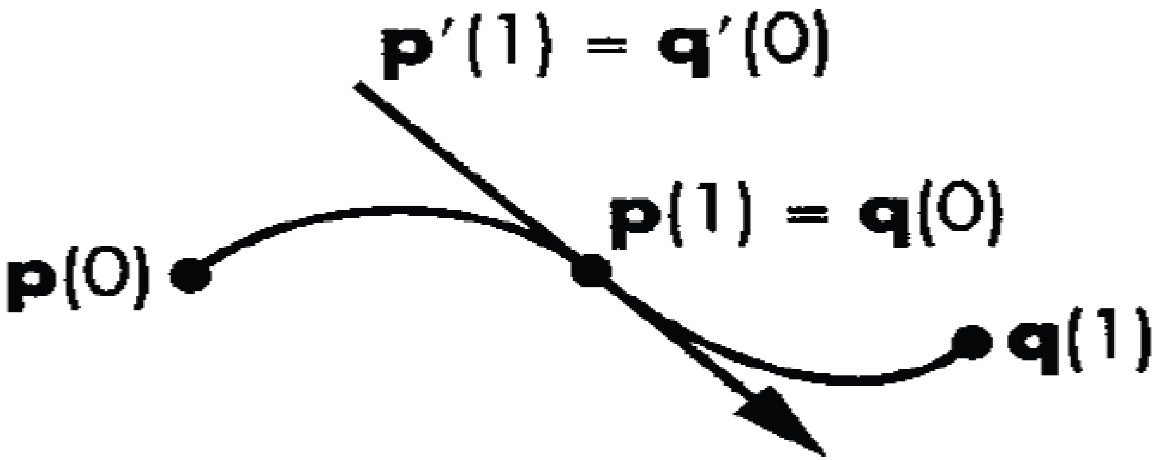
Definindo a curva de Hermite cúbica

$$\mathbf{p}(u) = \mathbf{u}^T \mathbf{c} = \mathbf{u}^T \mathbf{M}_H \mathbf{p}$$

$$M_H = A^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}$$

portanto $\mathrm{c} = \mathrm{M}_H \mathrm{p}$

Como fica a curva formada pela união de 2 no ponto de união



pontos de controle = *p para a curva 1 e q para a curva 2*

$$Q_H(t) = TM_H G_H$$

pontos de controle = *Pi e Ri*

G_H é

$$G_H = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}$$

G_{Hx} é a componente *x* de G_H:

$$G_{Hx} = \begin{bmatrix} P_{1x} \\ P_{4x} \\ R_{1x} \\ R_{4x} \end{bmatrix}$$

G_{Hy} é a componente y de G_H e
G_{Hz} é a componente z de G_H

Próxima aula

- Curvas de Bezier e Splines
- Superficies
- Trabalho 2 – parte 2