

Computação Gráfica

Aula 12: Curvas e Animações

Kahoot

Kahoot!

Entrar em Kahoot.it : <https://kahoot.it/>

Curvas e Animações

Muitas vezes precisamos de curvas suaves:

- Caminhos de câmera
- Fontes de texto vetoriais
- CAD e outras modelagem de objetos

Caminhos de Câmeras

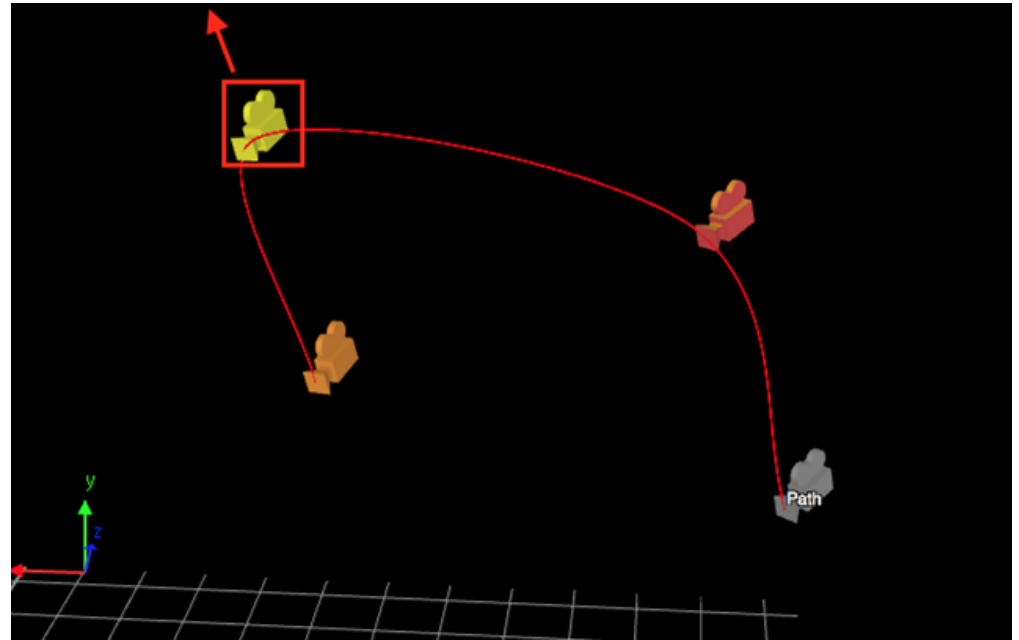
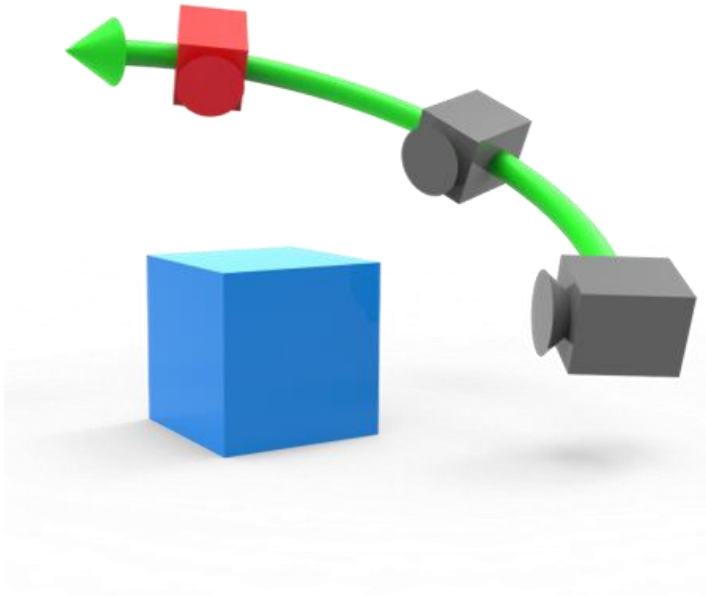


LUMION 10 AMAZING WALKTHROUGH OF AEROSPACE MUSEUM

<https://www.youtube.com/watch?v=KYteLM6ViBA>

Caminhos de Câmeras

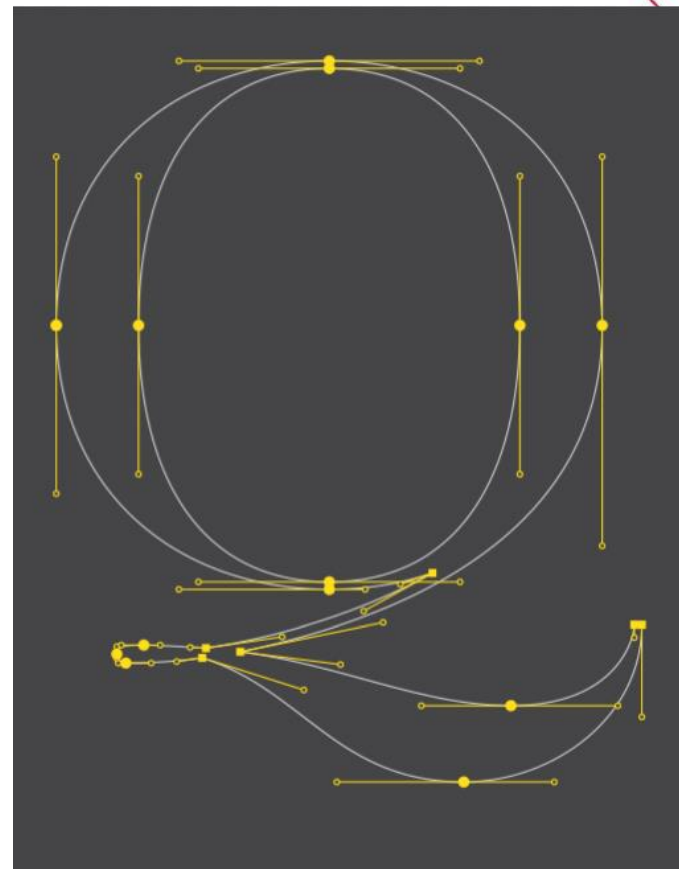
Definir pontos de controle e ir interpolando entre as posições desejadas (em geral posição e rotação).



Fontes de Texto Vetoriaiais

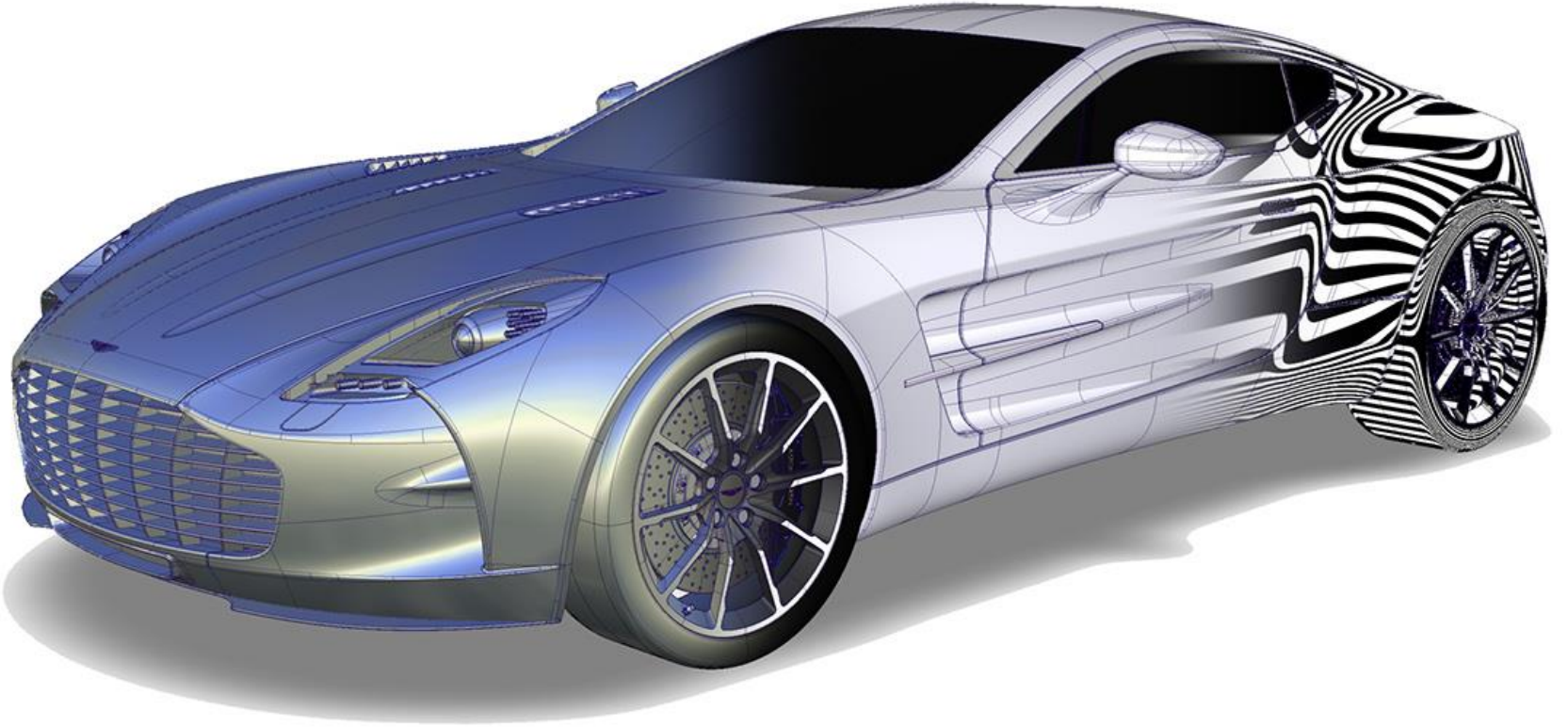
The Quick
Brown Fox
Jumps Over
The Lazy Dog

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz 0123456789



Fonte Baskerville – representada com Splines Cúbicas de Bézier

Design : CAD



Aston Martin One - 77 Surfacing - Alias, por Ankishu Gupta

Caminhos de Câmeras

As curvas envolvidas nesses caminhos devem ser **suaves**, como mostrado nos exemplos.

Mas o que significa ser suave do ponto de vista matemático?

Vamos relembrar!

Caminhos de Câmeras

O caminho da bolinha é uma curva suave? Por que?



NÃO!

A trajetória não é **contínua**.

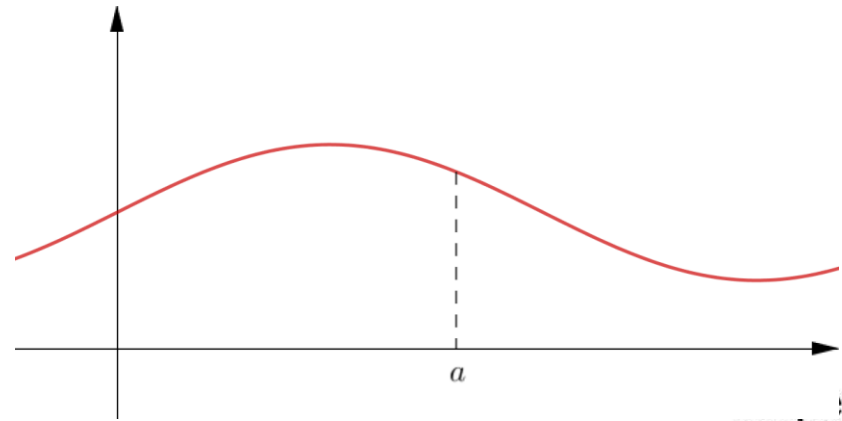
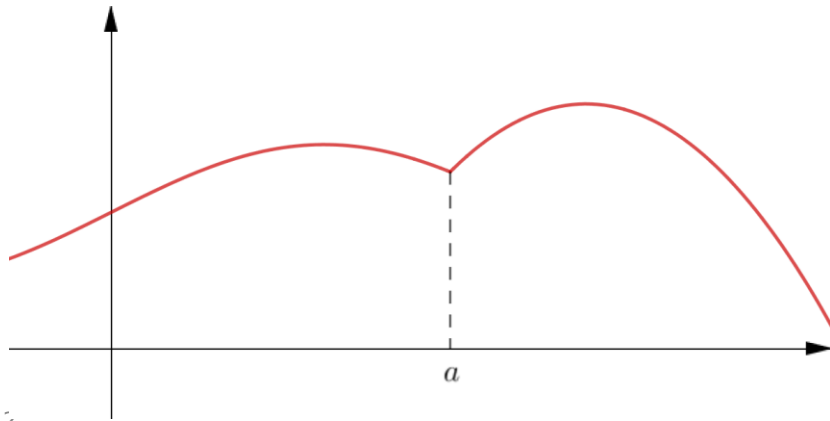
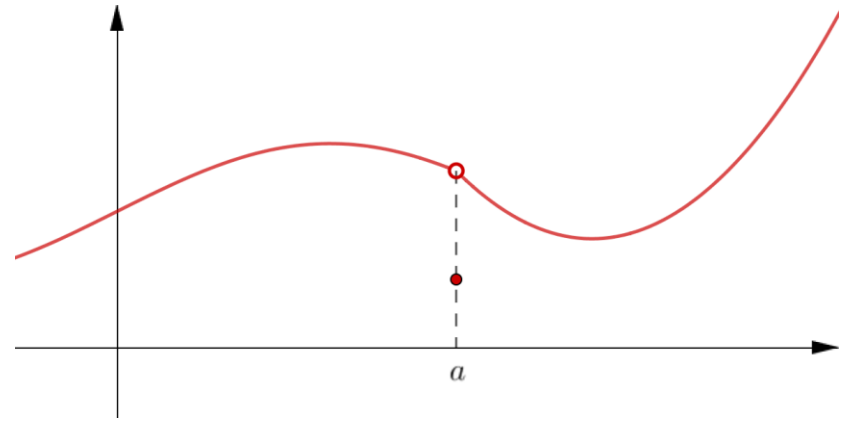
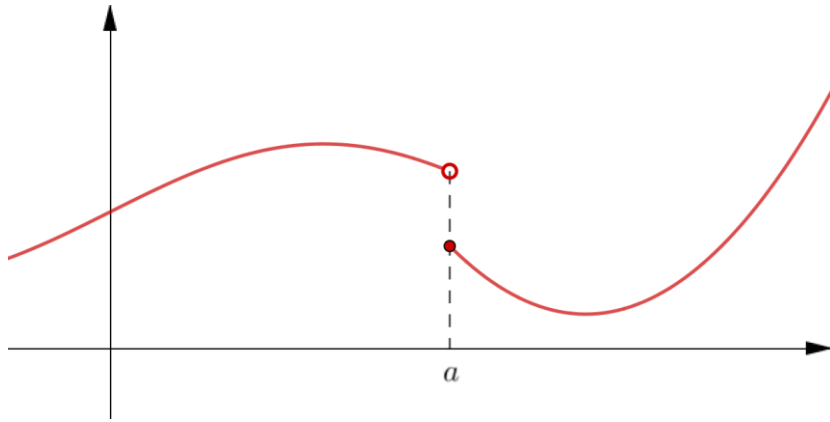
Dizemos que uma função f é contínua em um ponto $x = a$ quando:

$$1) \lim_{x \rightarrow a^-} f(x) = \lim_{x \rightarrow a^+} f(x)$$

$$2) \lim_{x \rightarrow a} f(x) = f(a)$$

Caminhos de Câmeras

Quais das figuras representam o gráfico de uma função contínua em $x = a$?



Caminhos de Câmeras

E esse caminho? É uma curva suave? Por que?



NÃO!

A trajetória não é **derivável**.

Dizemos que uma função f é derivável em um ponto $x = a$ quando o limite abaixo existe e resulta em um número finito:

$$\lim_{h \rightarrow 0} \frac{f(a + h) - f(a)}{h}$$

Caminhos de Câmeras

Uma forma equivalente de verificar se uma função é derivável no ponto $x = a$:

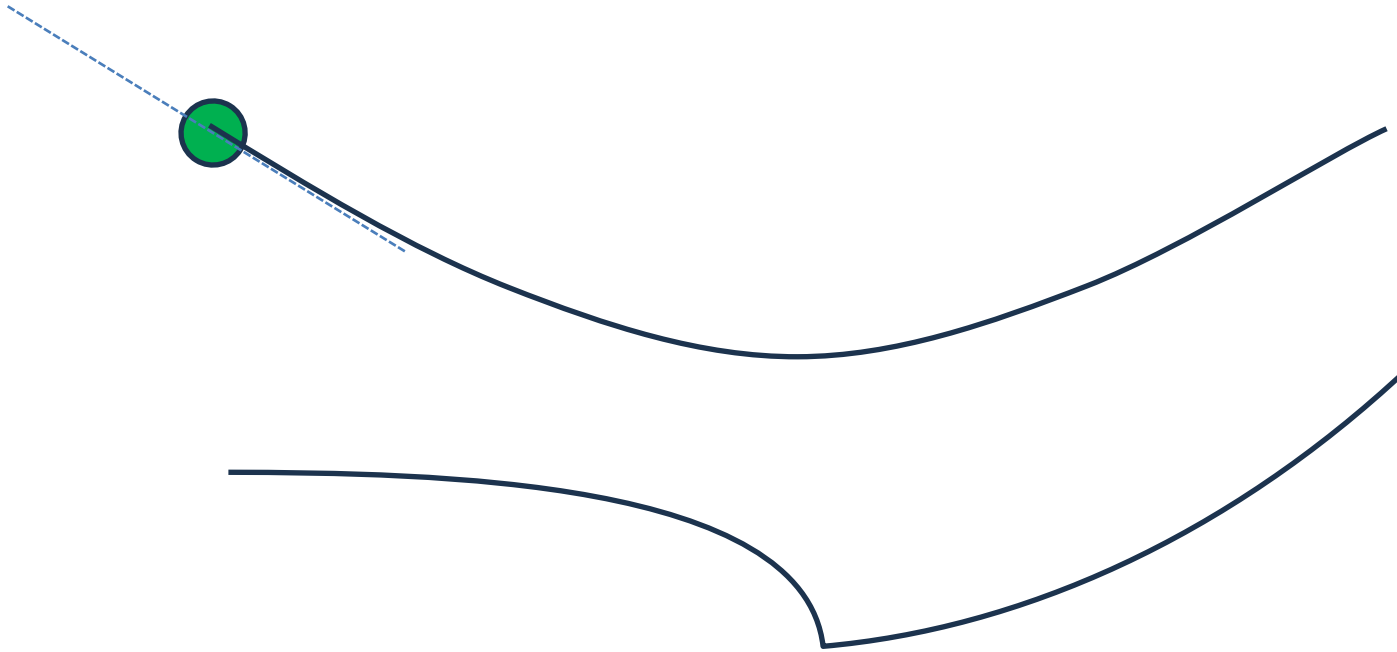
1) f é contínua em $x = a$.

2) Os limites laterais da derivada em $x = a$ são iguais.

$$\lim_{x \rightarrow a^-} f'(x) = \lim_{x \rightarrow a^+} f'(x)$$

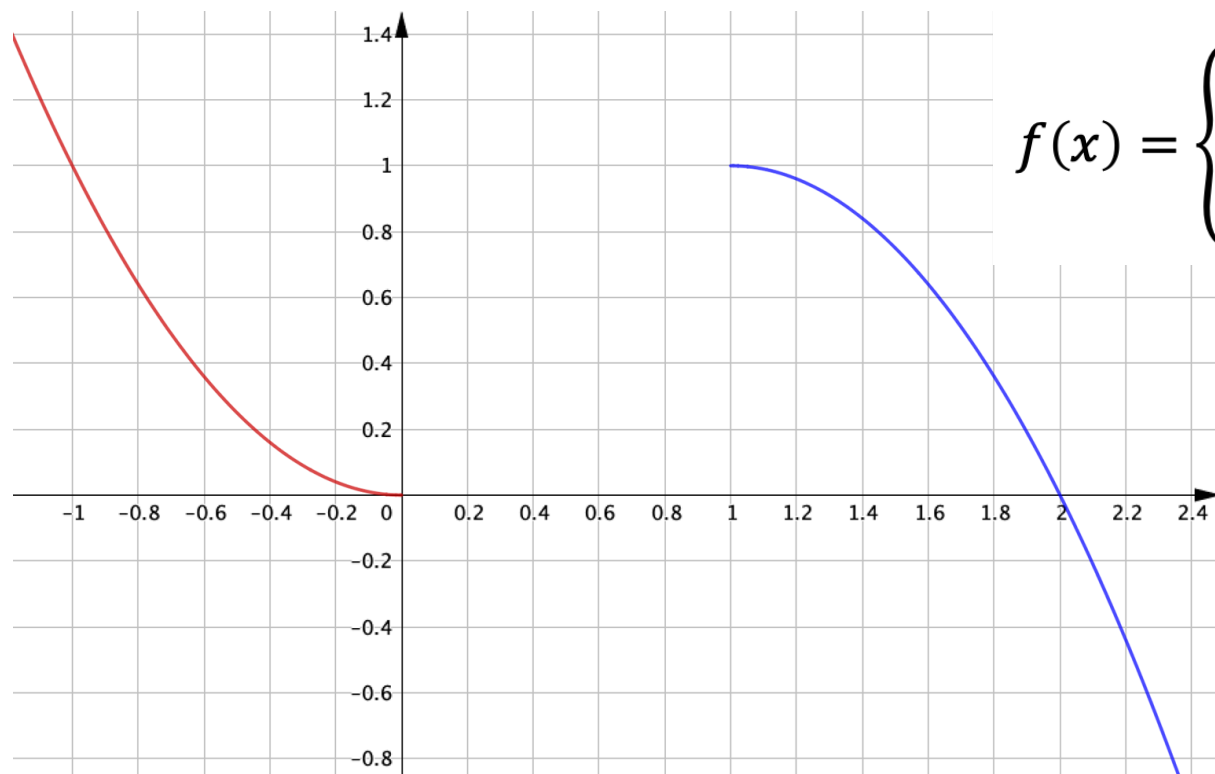
Caminhos de Câmeras

Em outras palavras, a reta tangente à curva não deve mudar bruscamente nas proximidades do ponto $x = a$.



Exercício

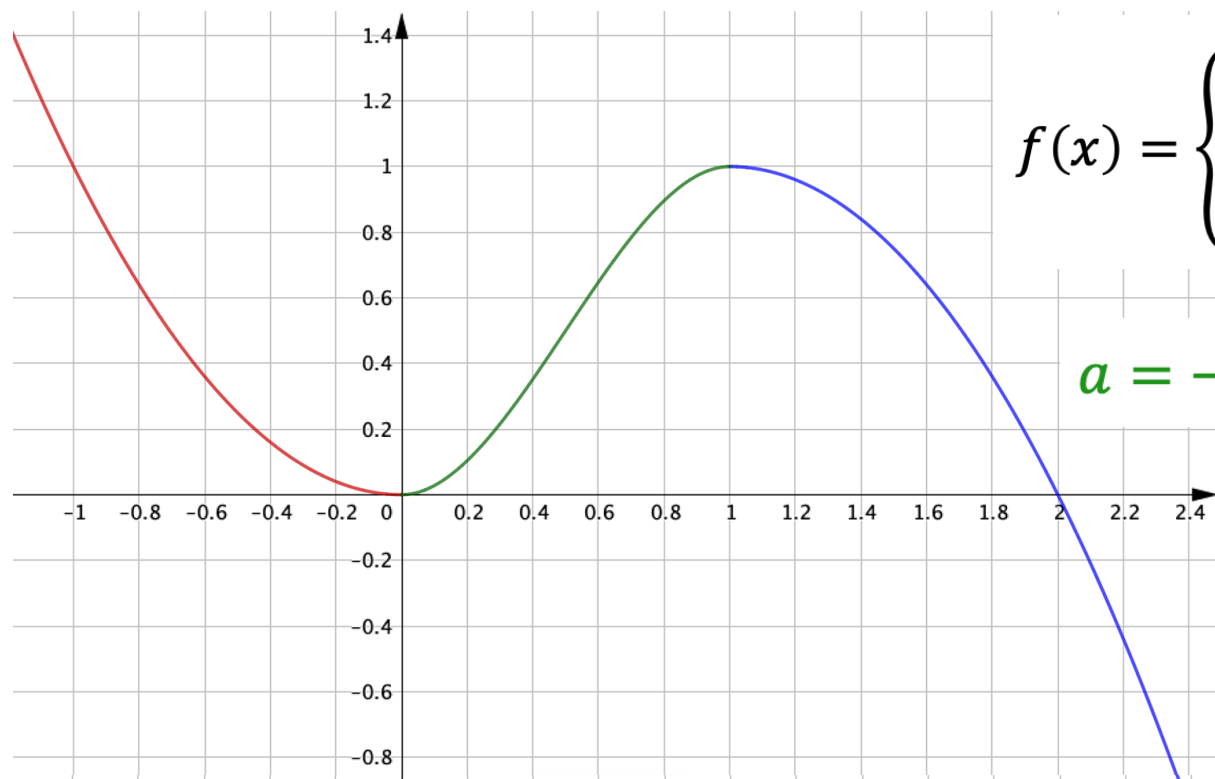
Defina uma função polinomial $p(x)$ no intervalo $[0,1]$ capaz de unir as duas curvas a seguir de forma suave.



$$f(x) = \begin{cases} x^2 & \text{se } x < 0 \\ 1 - (x - 1)^2 & \text{se } x > 1 \end{cases}$$

Exercício

Defina uma função polinomial $p(x)$ no intervalo $[0,1]$ capaz de unir as duas curvas a seguir de forma suave.



$$f(x) = \begin{cases} x^2 & \text{se } x < 0 \\ 1 - (x - 1)^2 & \text{se } x > 1 \end{cases}$$

$$a = -2, b = 3, c = 0 \text{ e } d = 0$$

$$p(x) = -2x^3 + 3x^2$$

Desenho Geométrico - Tangência e Concordância

Tangência

A tangência ocorre quando uma reta ou uma curva toca outra curva ou circunferência em um único ponto, chamado de ponto de tangência. As duas curvas têm o mesmo ponto em comum, e as inclinações naquele ponto são idênticas, garantindo que a transição entre elas seja suave. A tangência é importante porque ela indica que não há interrupção no "fluxo" entre as figuras no ponto de contato, ou seja, não há um "salto" ou ângulo agudo no ponto de conexão.

Concordância

A concordância refere-se ao processo de unir duas ou mais curvas de forma suave e contínua, sendo que as curvas de transição permitem uma mudança gradual entre elas. A suavidade pode ser medida:

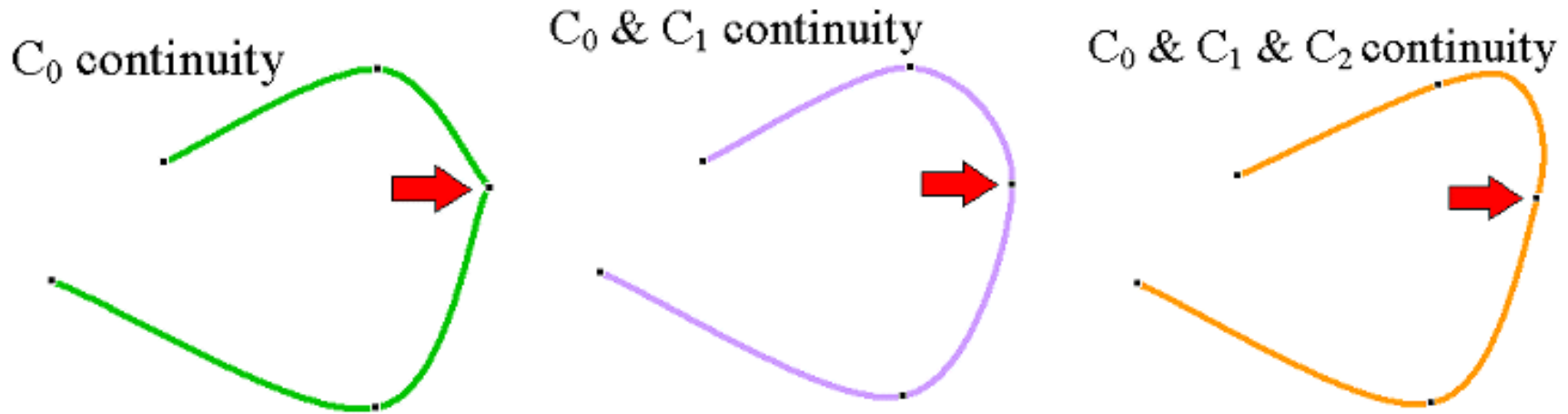
- **Primeira derivada:** se as inclinações (tangentes) são contínuas, garantindo a tangência.
- **Segunda derivada:** se a curvatura (a taxa de mudança da inclinação) é contínua, garantindo uma transição ainda mais suave.

Explicando Continuidade

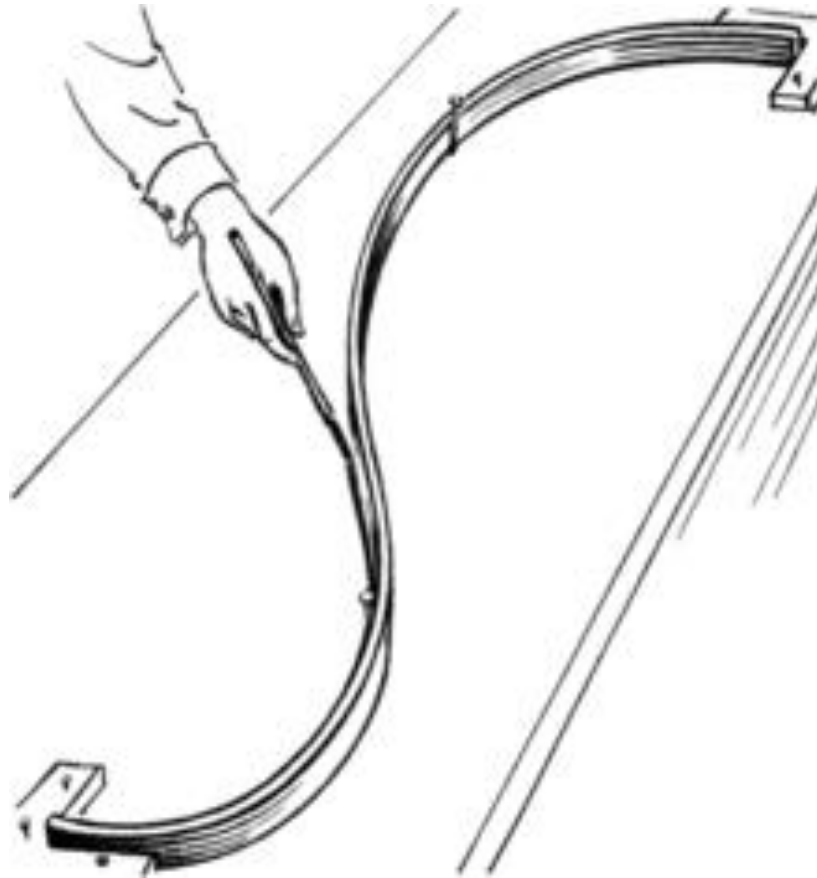
Continuidade C0: garante que os segmentos de curvas são interligados

Continuidade C1: garante que os segmentos de curvas tenham a mesma inclinação nos seus pontos de junção (mesma direção das tangentes)

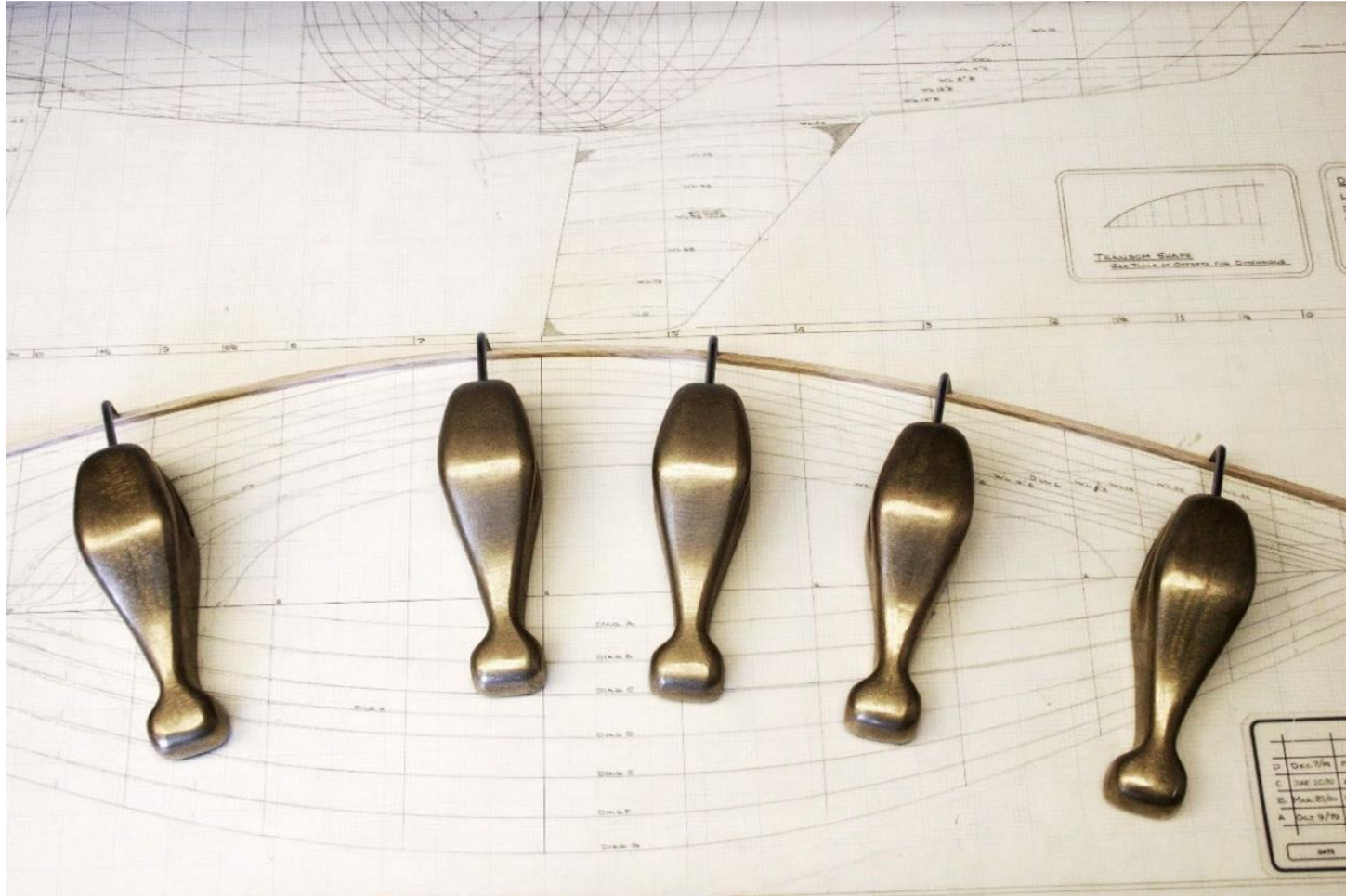
Continuidade C2: garante que os segmentos de curvas tenham a mesma curvatura nos pontos de junção (mesma direção e magnitude das tangentes)



Splines



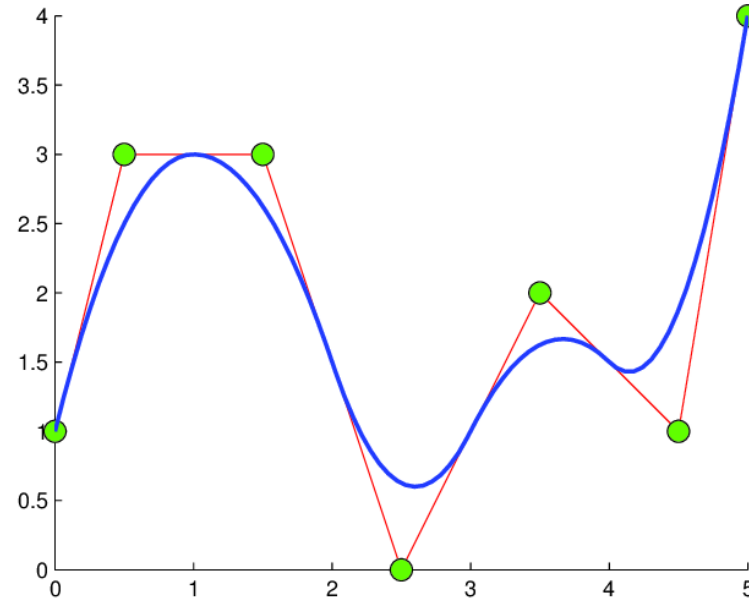
Spline de um desenhista



<http://www.alatown.com/spline-history-architecture/>

Splines

Uma spline é uma representação matemática para uma curva polinomial suave definida, por partes, através de uma sequência de pontos (chamados de pontos de controle).

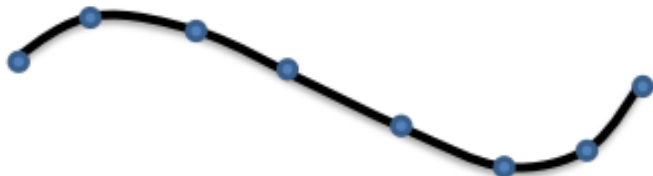


A quadratic ($p = 2$) B-spline curve with a uniform open knot vector $\Xi = \{0, 0, 0, 1, 2, 3, 4, 5, 5, 5\}$

Tópicos sobre Splines

Interpolação

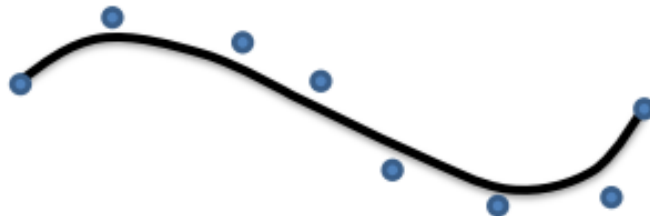
- Interpolação Cúbica de Hermite
- Interpolação Catmull-Rom



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

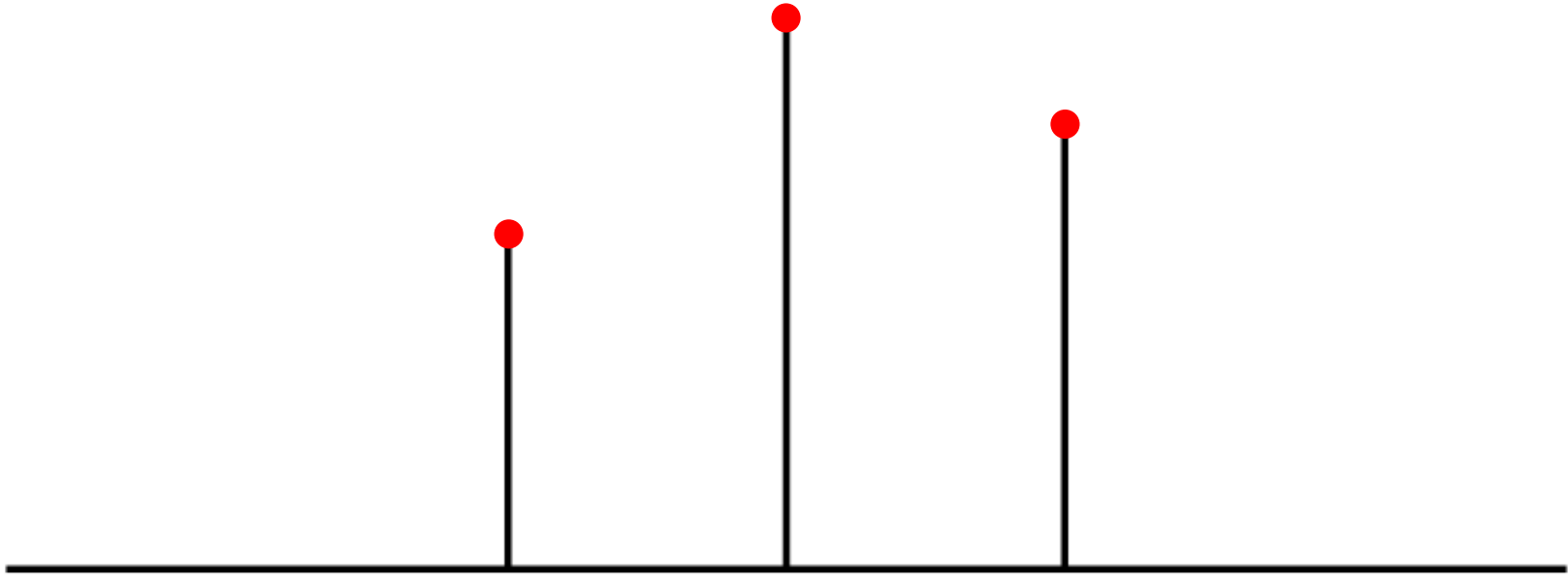
Aproximação

- Bezier (não veremos hoje)
- B-Spline (não veremos hoje)



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

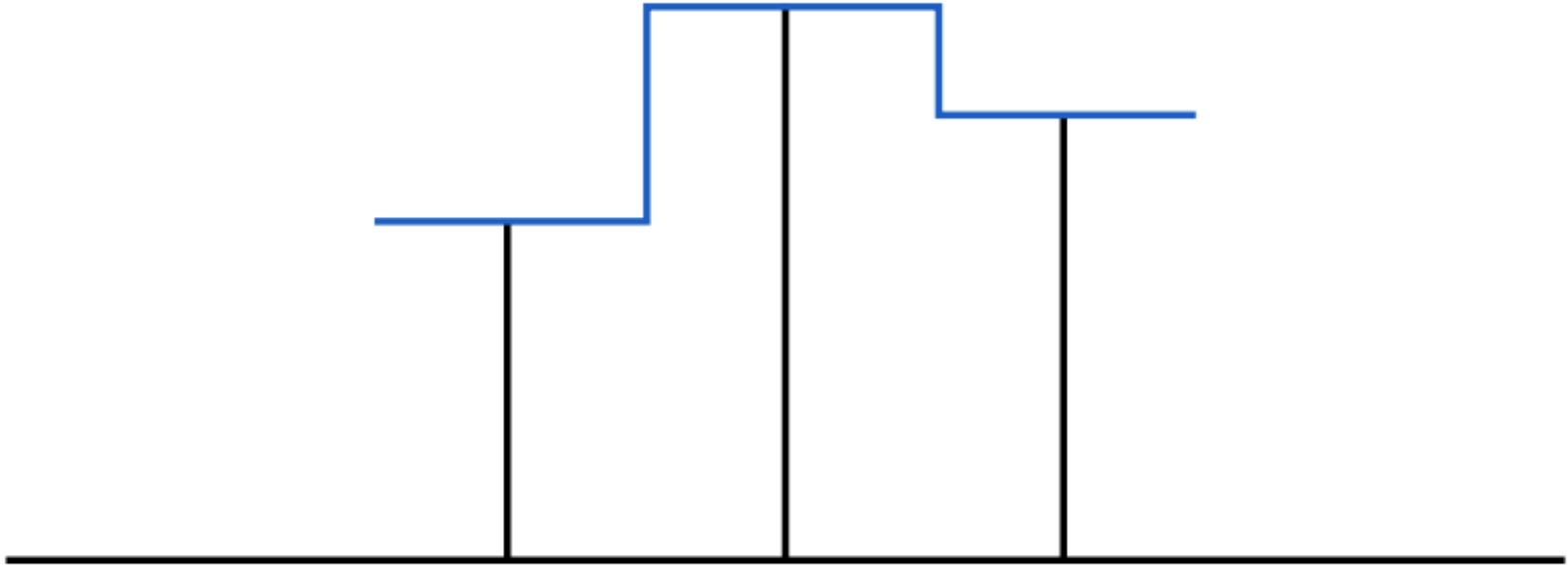
Objetivo Interpolar Valores



valores pontuais de uma função qualquer, como interpolar?

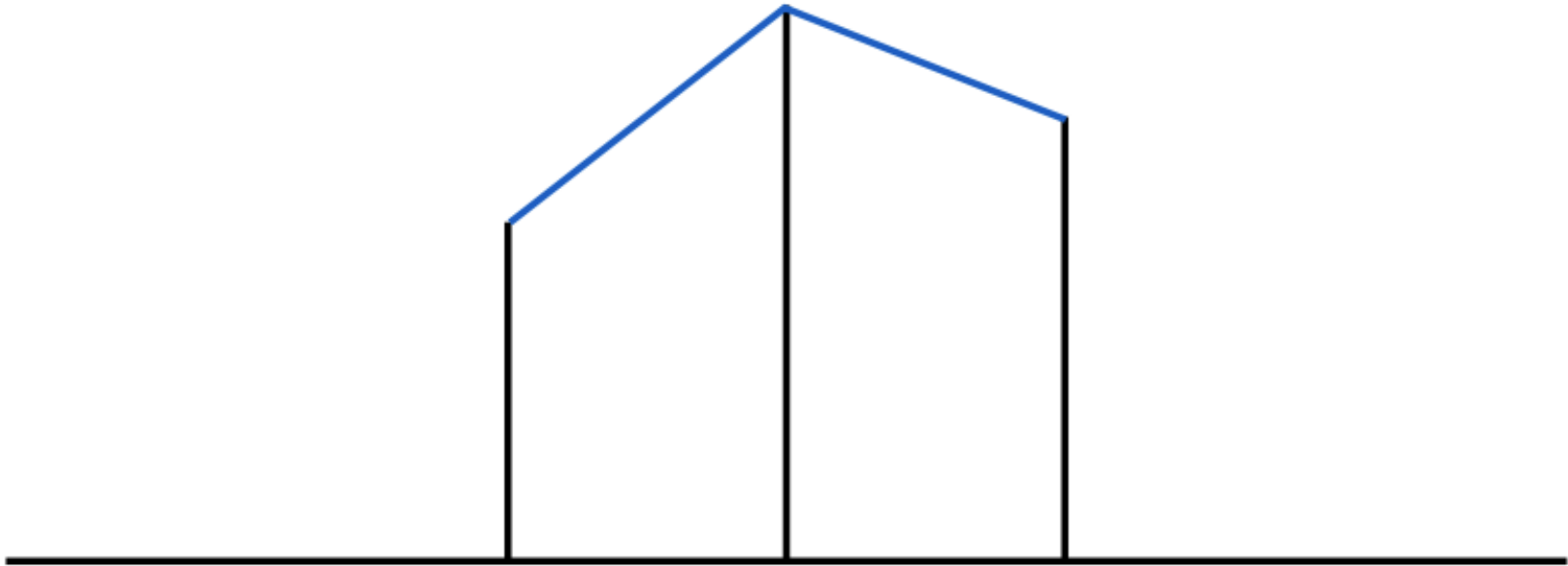
Interpolação do vizinho mais próximo

Problema: valores não são contínuos

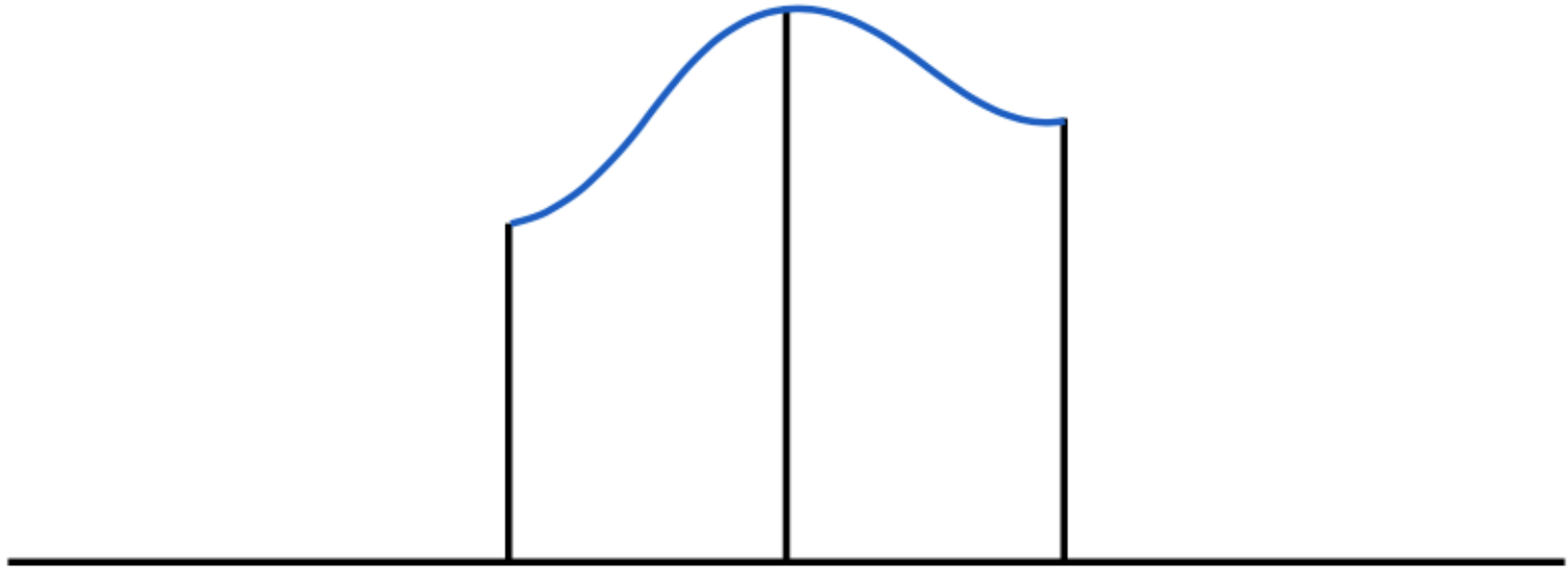


Interpolação Linear

Problema: derivações não contínuas



Interpolação Suave ?

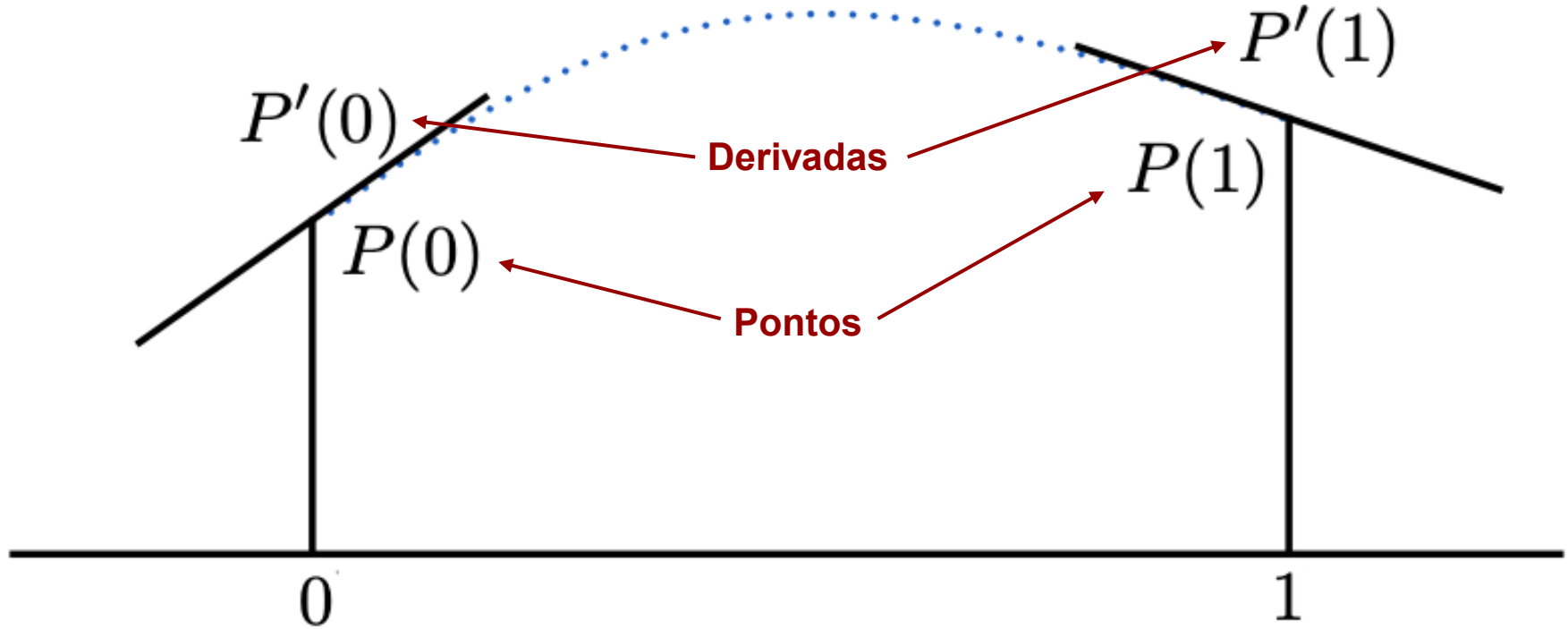


Interpolação Cúbica de Hermite



Charles Hermite cerca de 1887

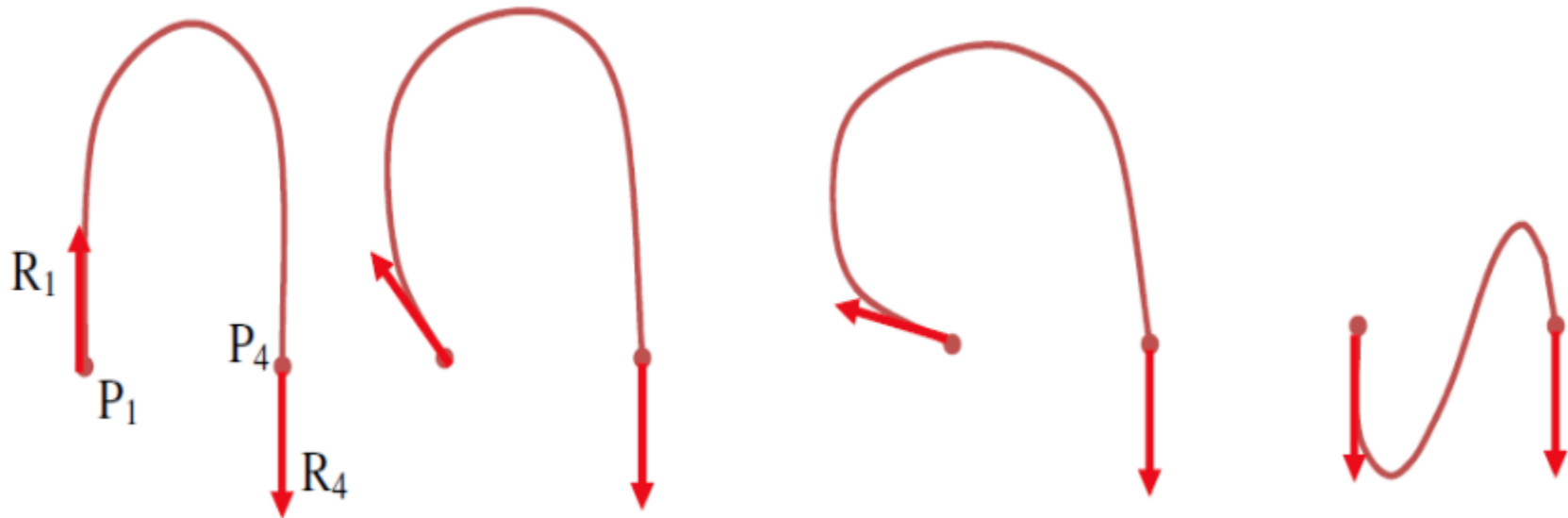
Interpolação Cúbica de Hermite



Entradas: valores e derivadas nos pontos de controle

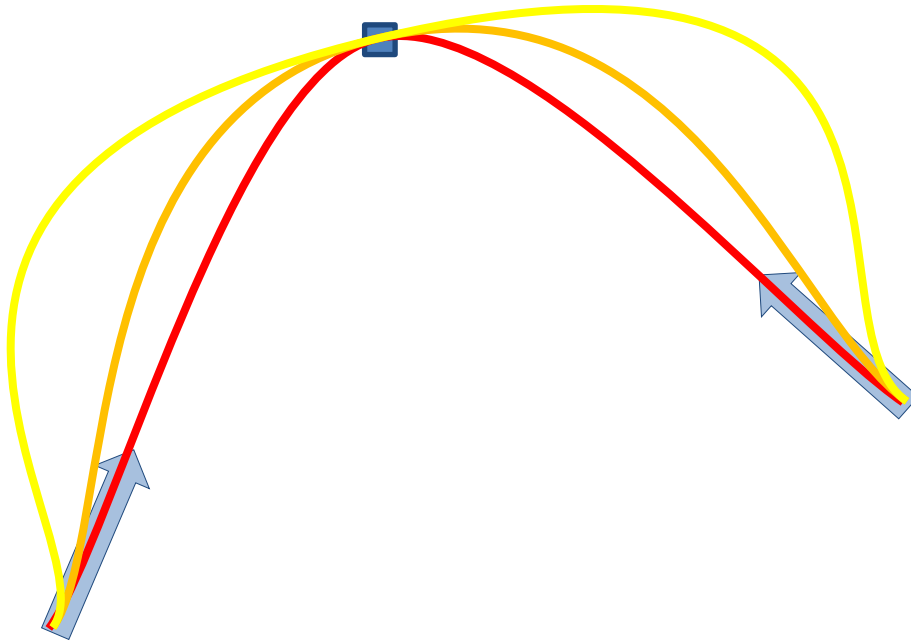
Curvas de Hermite

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



Curvas de Hermite

Curvas com os mesmos pontos iniciais e finais, apenas alterando a intensidade da tangente do ponto intermediário.



Interpolação Polinomial Cúbica

Polinômio Cúbico

$$P(t) = a t^3 + b t^2 + c t + d$$

Por que cúbico?

4 restrições de entrada : 4 graus de liberdade

$$P(0) = h_0$$

$$P(1) = h_1$$

$$P'(0) = h_2$$

$$P'(1) = h_3$$

Perceba que vamos trabalhar na faixa do 0 a 1.

Interpolação Polinomial Cúbica

Polinômio Cúbico

$$P(t) = a t^3 + b t^2 + c t + d$$

Derivando

$$P'(t) = 3a t^2 + 2b t + c$$

Configurando equações de restrição

$$P(0) = h_0 = d$$

$$P(1) = h_1 = a + b + c + d$$

$$P'(0) = h_2 = c$$

$$P'(1) = h_3 = 3a + 2b + c$$

Resolvendo os coeficientes polinomiais

$$h_0 = d$$

$$h_1 = a + b + c + d$$

$$h_2 = c$$

$$h_3 = 3a + 2b + c$$

Colocando na forma matricial

$$\begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

Mas o que precisamos é o contrário disso. Como fazer?

Resolvendo os coeficientes polinomiais

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$
$$= \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

(podem verificar se essas matrizes são inversas)

Forma Matricial para Função de Hermite

$$P(t) = a t^3 + b t^2 + c t + d$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Interpretação 1

Linhas da Matriz = Coeficientes da Fórmula

$$P(t) = a t^3 + b t^2 + c t + d$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Interpretação 2

Colunas da Matriz = ?

$$P(t) = a t^3 + b t^2 + c t + d$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

$$= \begin{bmatrix} 2t^3 - 3t^2 + 1 \\ -2t^3 + 3t^2 \\ t^3 - 2t^2 + t \\ t^3 - t^2 \end{bmatrix}^T \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Base da Função de Hermite

$$P(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} H_0(t) & H_1(t) & H_2(t) & H_3(t) \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

t^3

$$H_0(t) = 2t^3 - 3t^2 + 1$$

t^2

$$H_1(t) = -2t^3 + 3t^2$$

t

$$H_2(t) = t^3 - 2t^2 + t$$

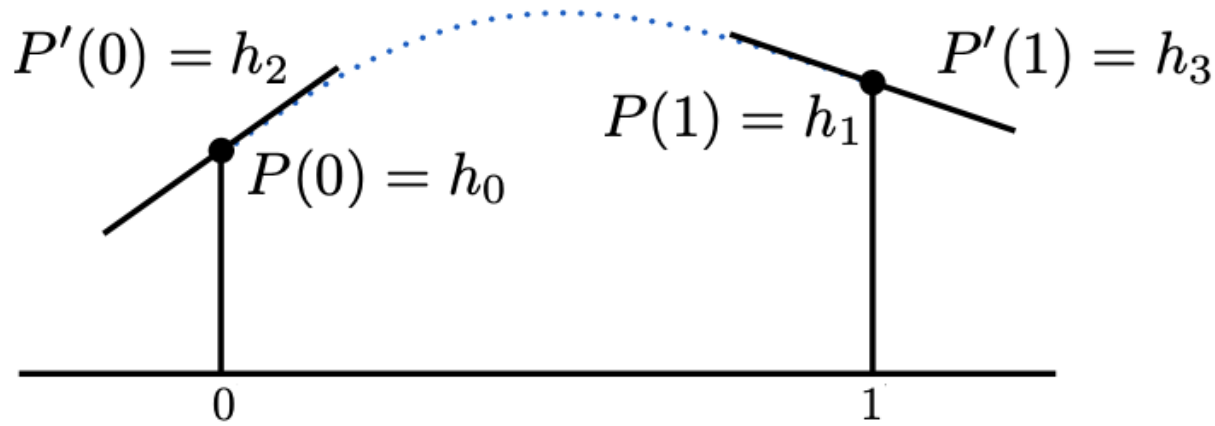
1

$$H_3(t) = t^3 - t^2$$

Termos polinomiais

Funções de base de Hermite
para polinômios cúbicos

Recapitulando: Interpolação Cúbica de Hermite



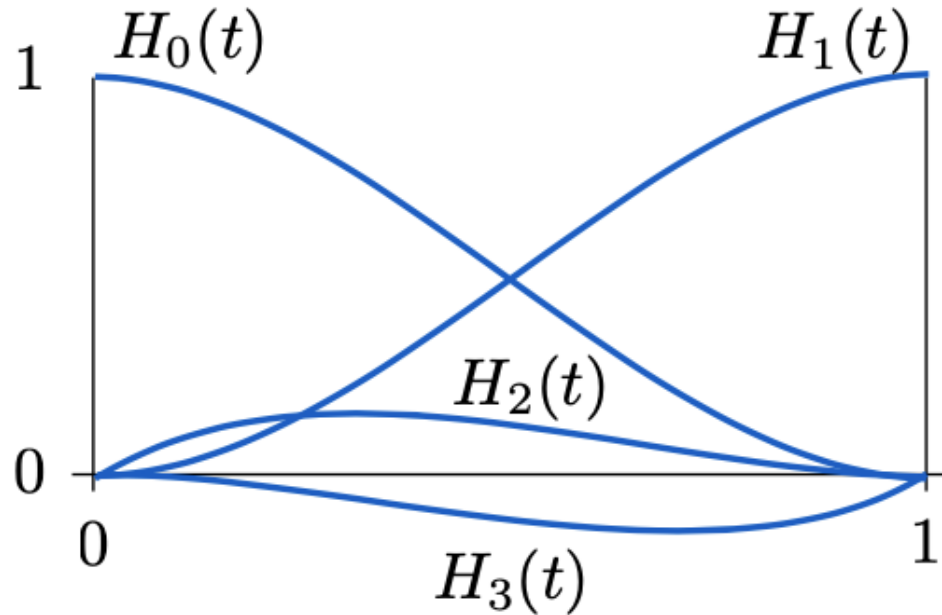
Entradas: valores e derivadas nos pontos de controle

Saída: polinômio cúbico que pode ser interpolado (de 0 a 1)

Solução: soma ponderada das funções interpoladoras de Hermite

$$P(t) = h_0 H_0(t) + h_1 H_1(t) + h_2 H_2(t) + h_3 H_3(t)$$

Funções Interpoladoras de Hermite



$$H_0(t) = 2t^3 - 3t^2 + 1$$

$$H_1(t) = -2t^3 + 3t^2$$

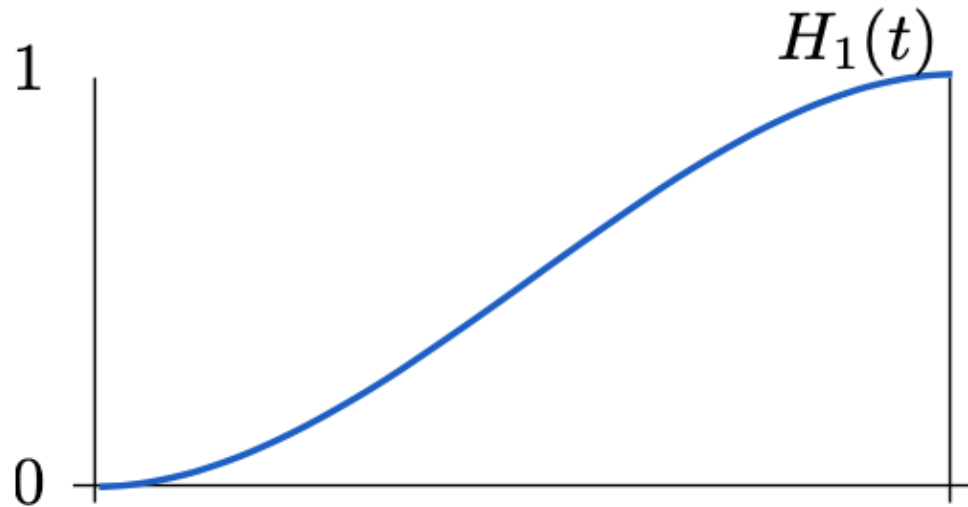
$$H_2(t) = t^3 - 2t^2 + t$$

$$H_3(t) = t^3 - t^2$$

Funções Simples

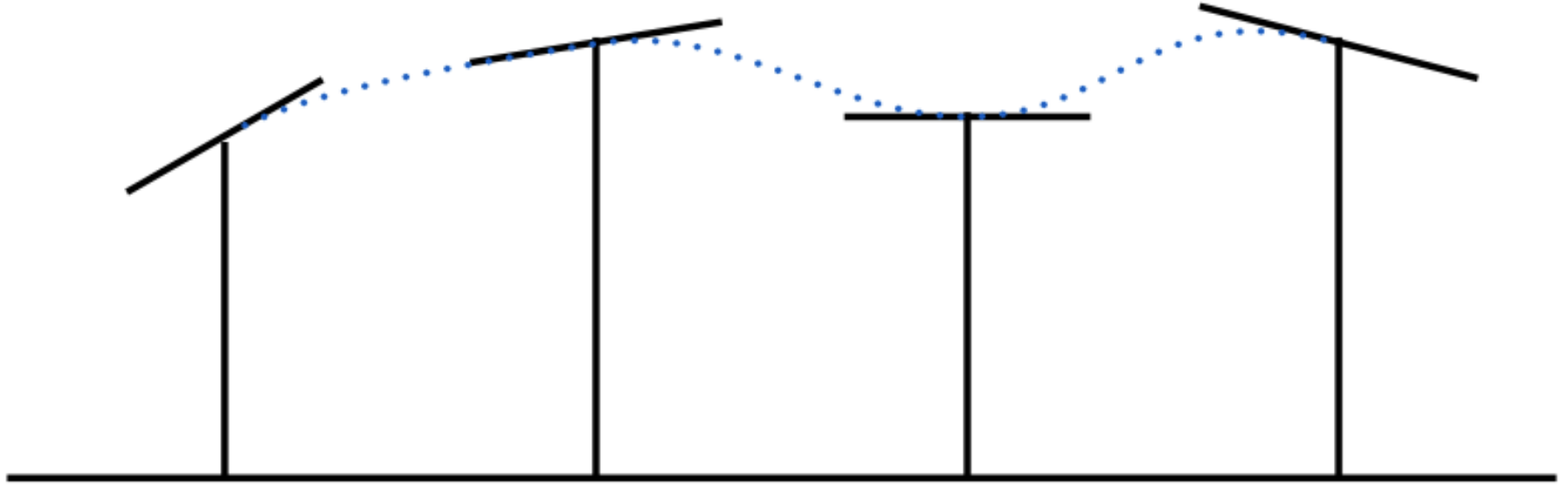
Uma função muito útil

Usada para iniciar e terminar animações (velocidades zero)



$$H_1(t) = -2t^3 + 3t^2 = t^2(3 - 2t)$$

Interpolação Spline de Hermite



Entradas: sequência de valores e derivadas

Interpolação Catmull-Rom



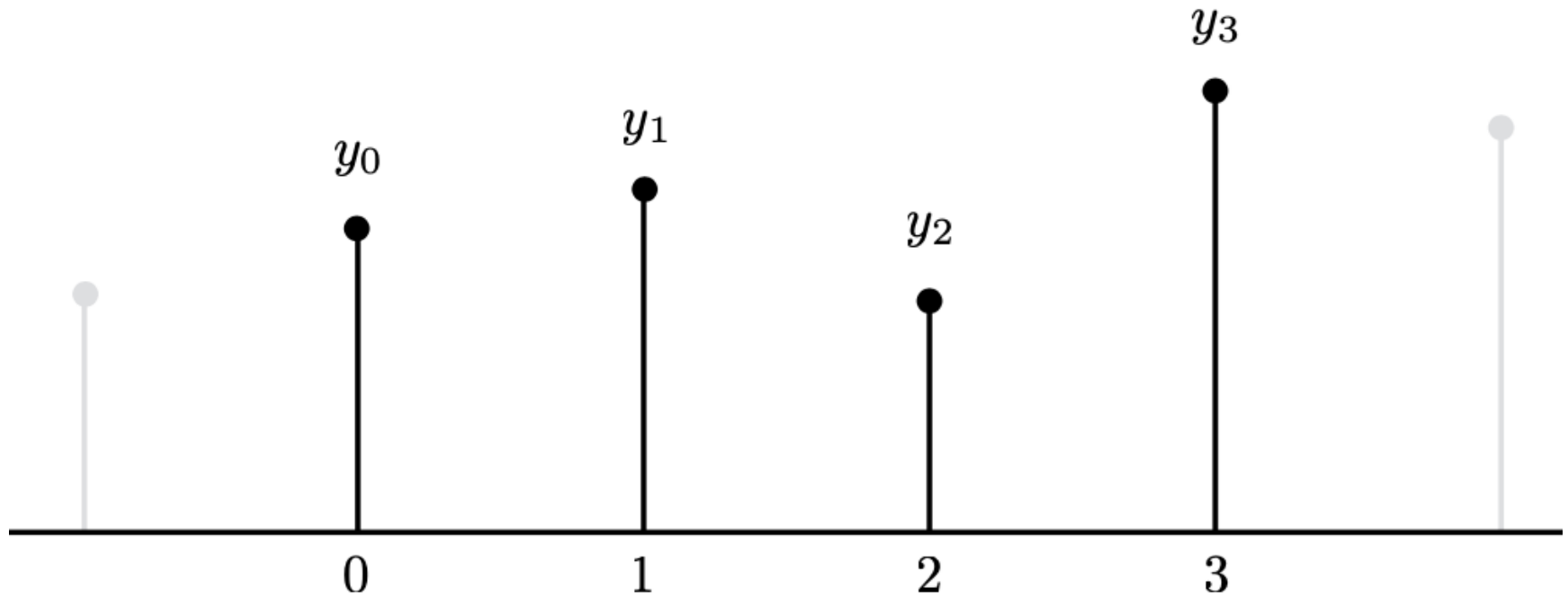
Ed Catmull



Raphael Rom

Interpolação Catmull-Rom

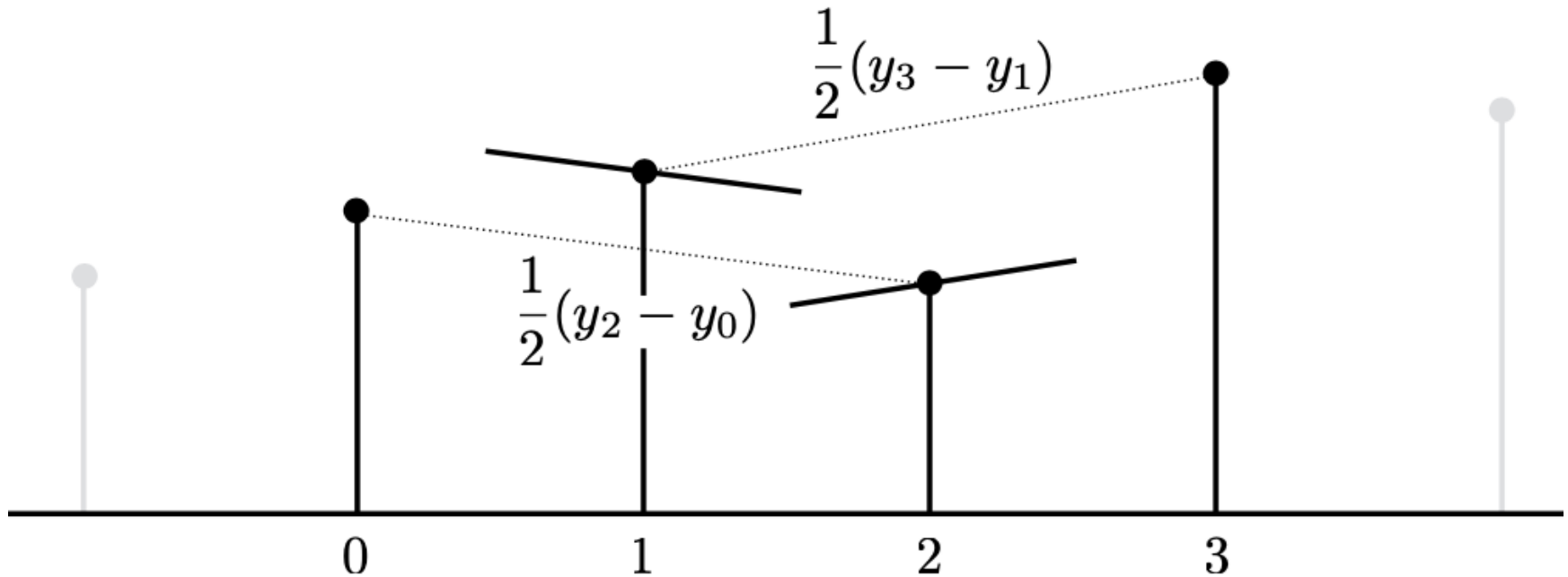
Entrada: sequência de valores



Interpolação Catmull-Rom

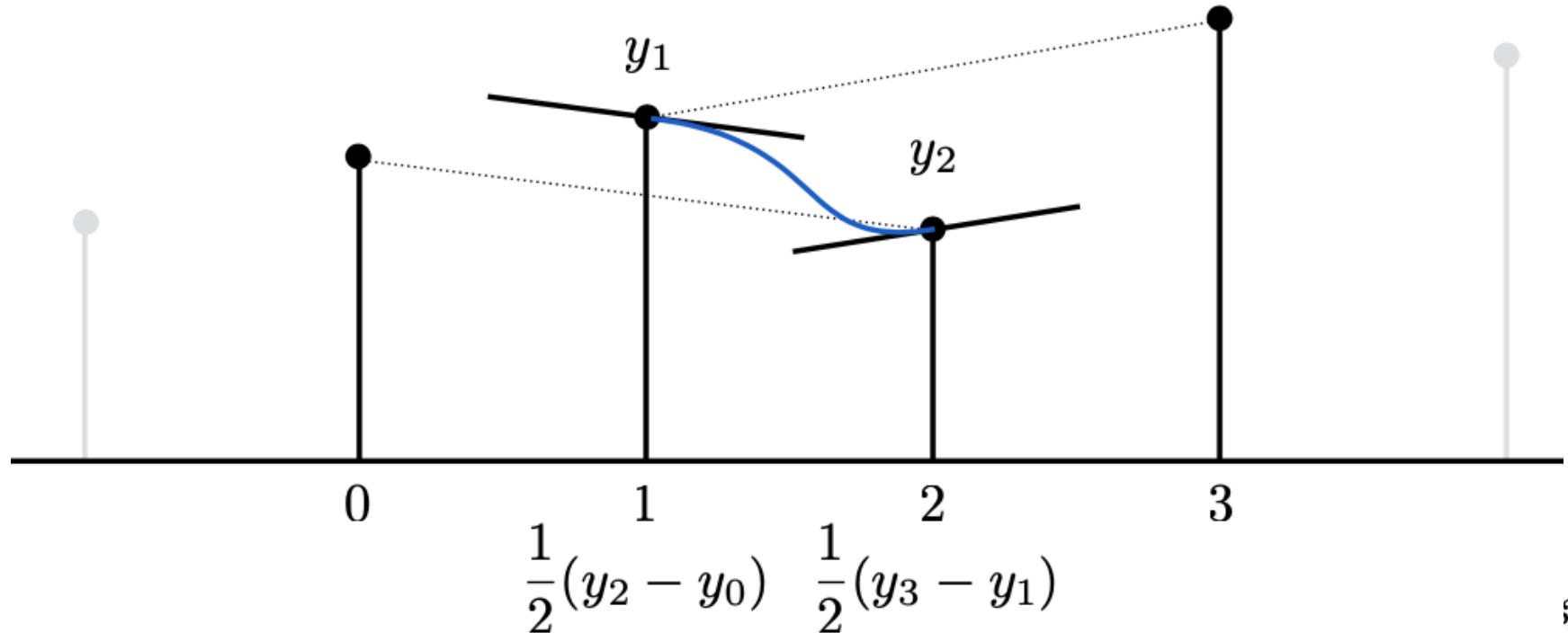
Regra para derivadas:

Combine a inclinação entre os valores anteriores e os próximos



Interpolação Catmull-Rom

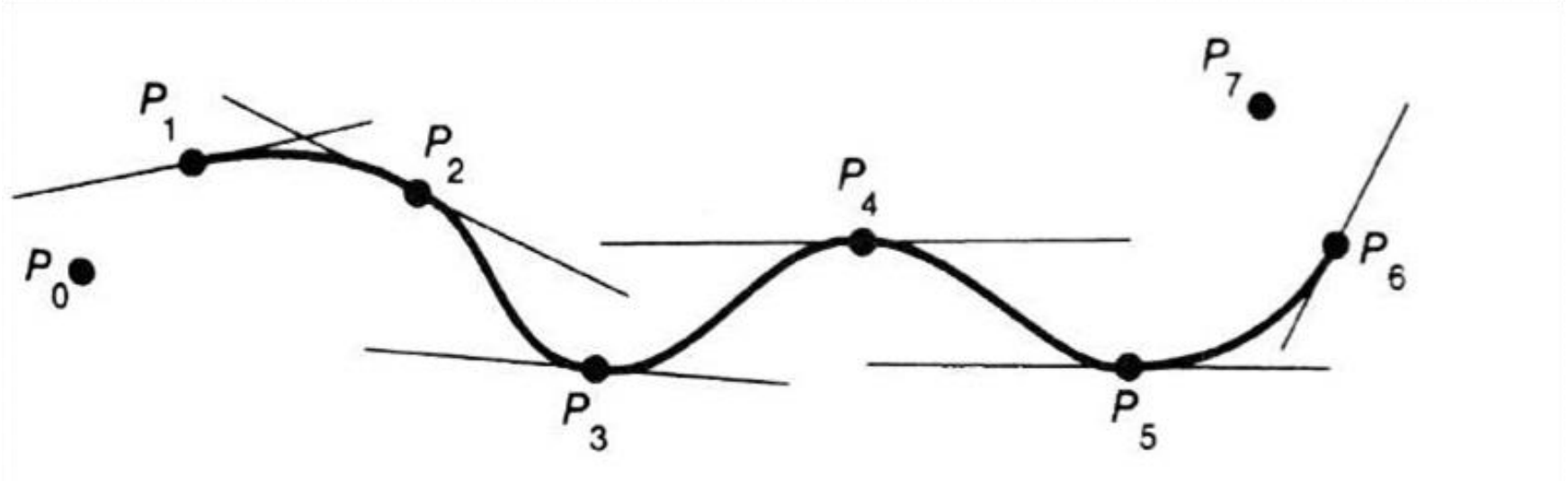
Então use a interpolação de Hermite



Spline de Catmull-Rom

Entrada: sequência de pontos

Saída: Spline que interpola todos os pontos com continuidade C^1

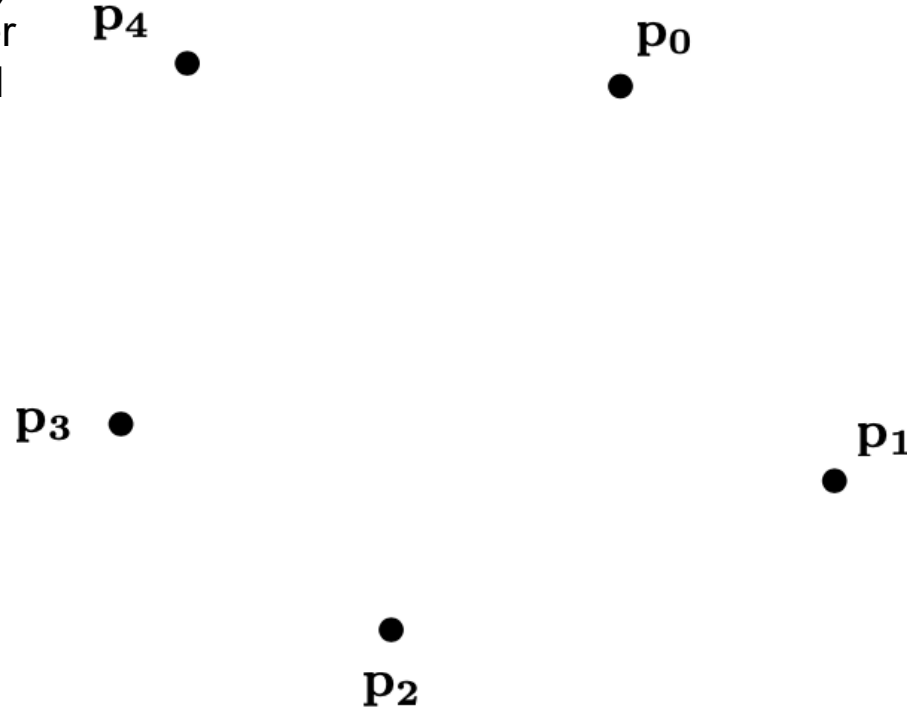


Interpolando Pontos e Vetores



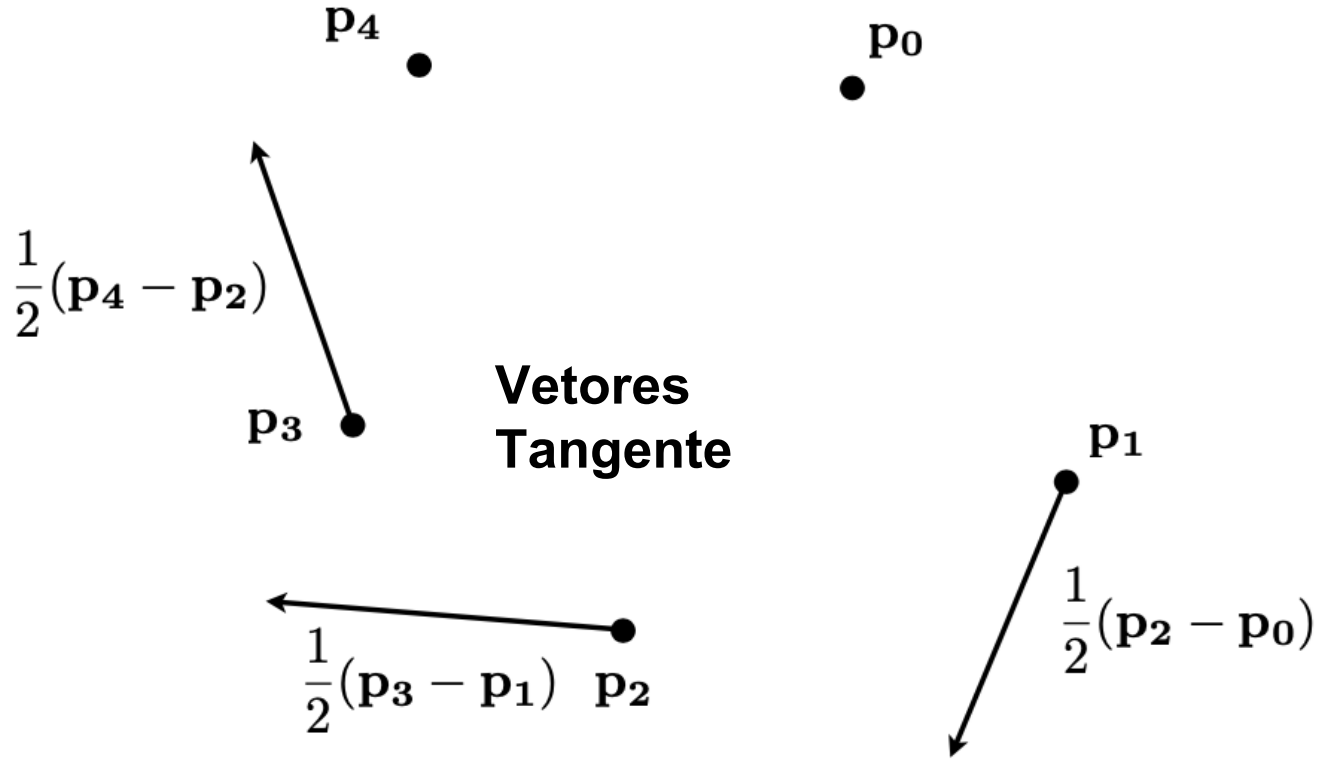
É possível interpolar pontos tão fácil quanto valores ?

Por exemplo: ponto $(0,1,3)$
no espaço 3D, ou qualquer
vetor geral na dimensão N



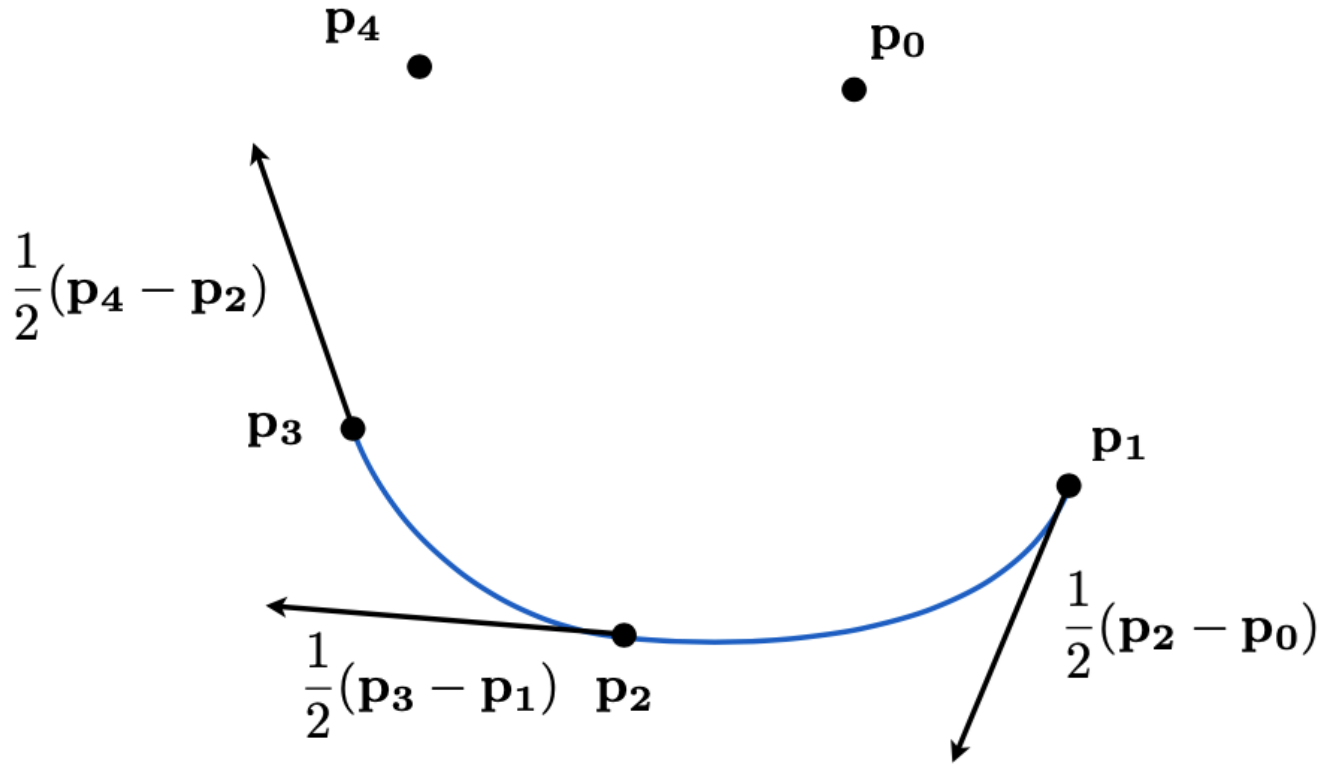
Pontos de controle da spline 3D de Catmull-Rom

É possível interpolar pontos tão fácil quanto valores ?



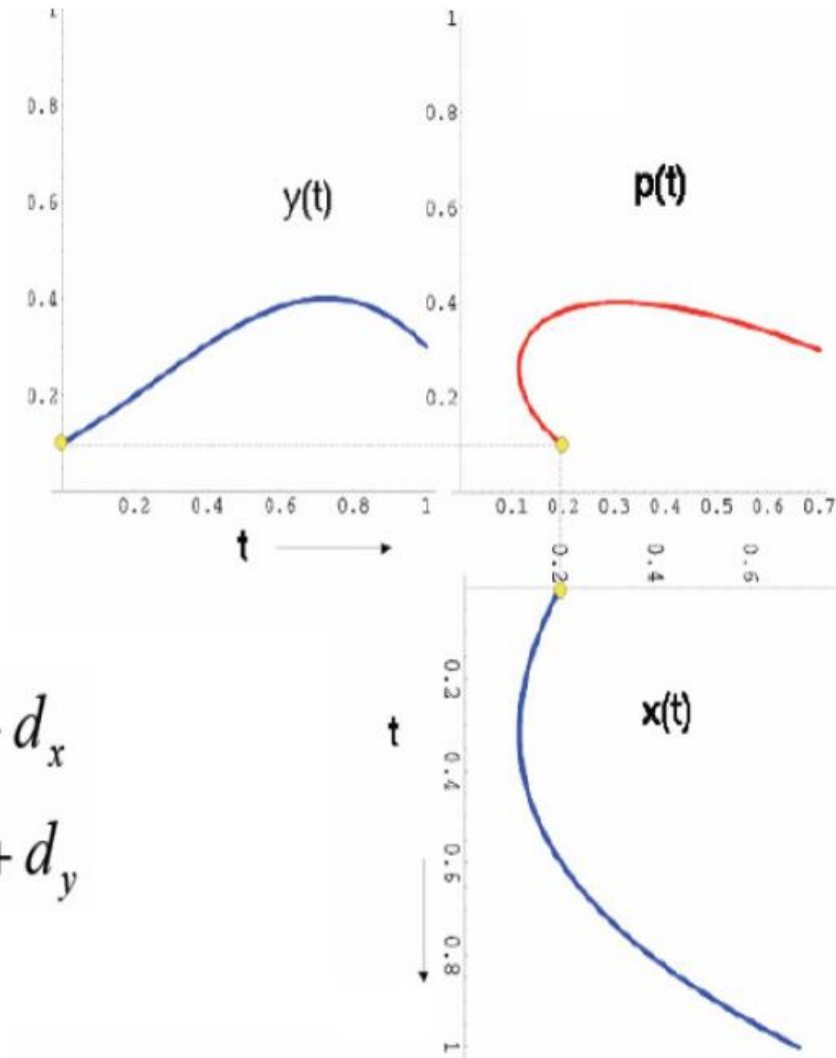
Vetores tangente 3D do Catmull-Rom

É possível interpolar pontos tão fácil quanto valores ?



Curvas do espaço 3D de Catmull-Rom

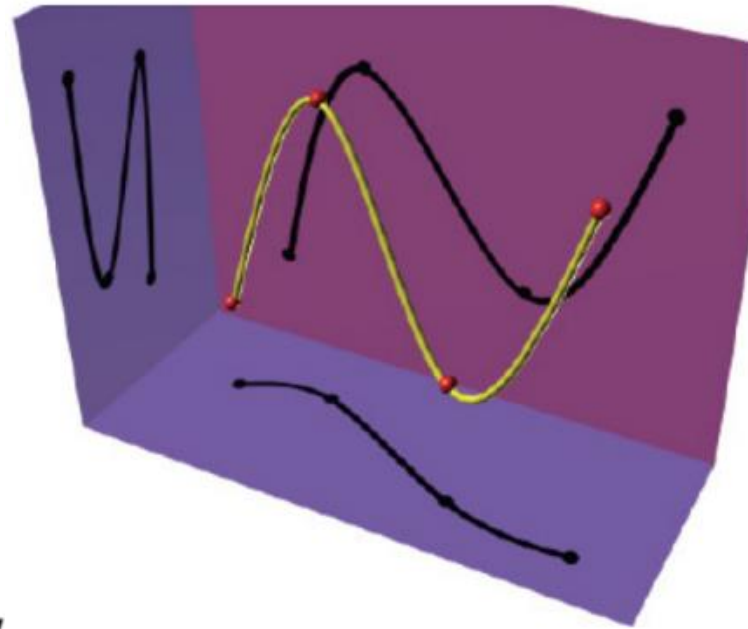
Curvas Paramétricas



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

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

Curvas Paramétricas



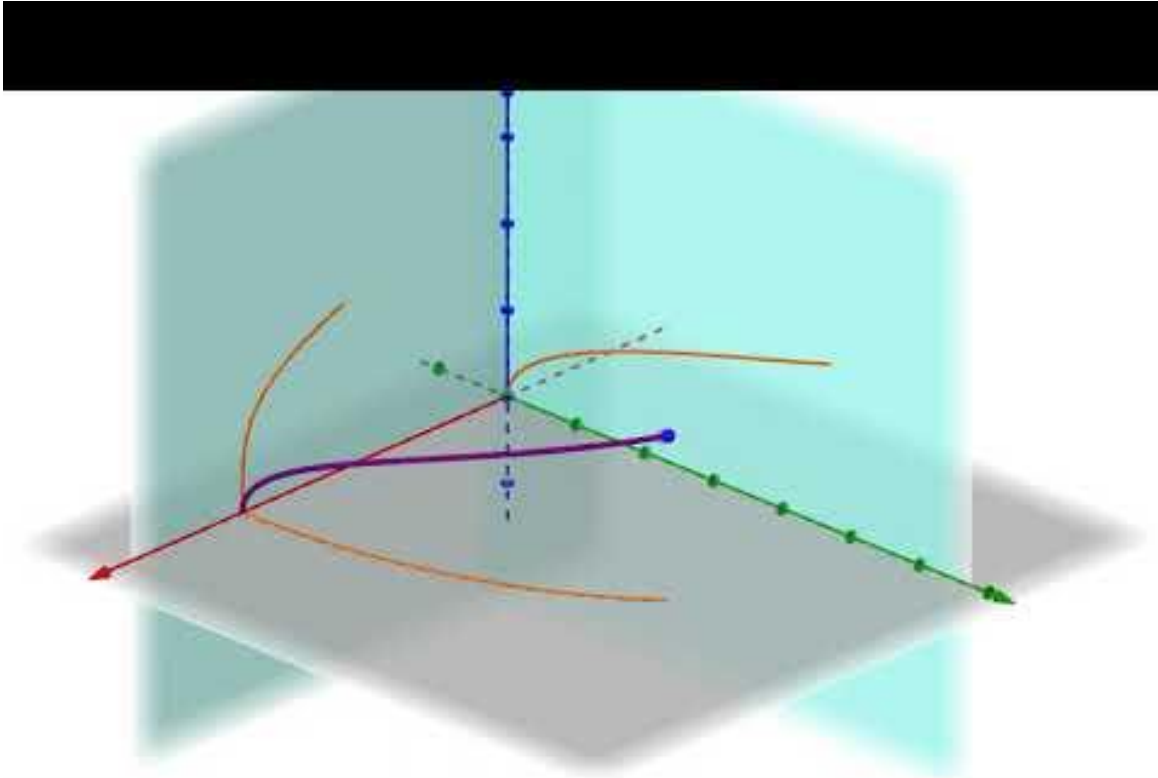
$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$

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

$$z(t) = a_z t^3 + b_z t^2 + c_z t + d_z$$

Curvas Paramétricas

Exemplo: $c(t) = (2 \cos t, 3 \sin t, \sqrt{t}); t \in [0, \frac{\pi}{2}]$



Forma Matricial para o Espaço de Curva Catmull-Rom

Use a forma matricial Hermite

- Pontos e tangentes dadas pelas regras Catmull-Rom

Pontos Hermite $\mathbf{h}_0 = \mathbf{p}_1$

$$\mathbf{h}_1 = \mathbf{p}_2$$

Tangentes Hermite $\mathbf{h}_2 = \frac{1}{2}(\mathbf{p}_2 - \mathbf{p}_0)$

$$\mathbf{h}_3 = \frac{1}{2}(\mathbf{p}_3 - \mathbf{p}_1)$$

$$P(t) = \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}^T \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

Matriz Hermite

Converte entradas
Catmull-Rom para
entradas Hermite

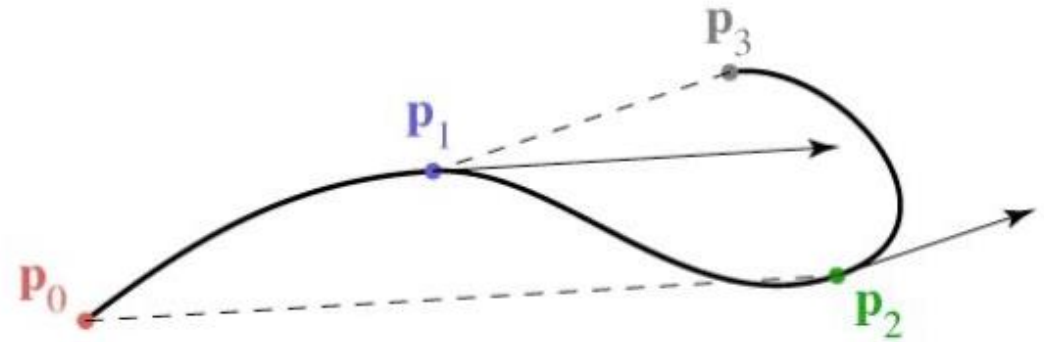
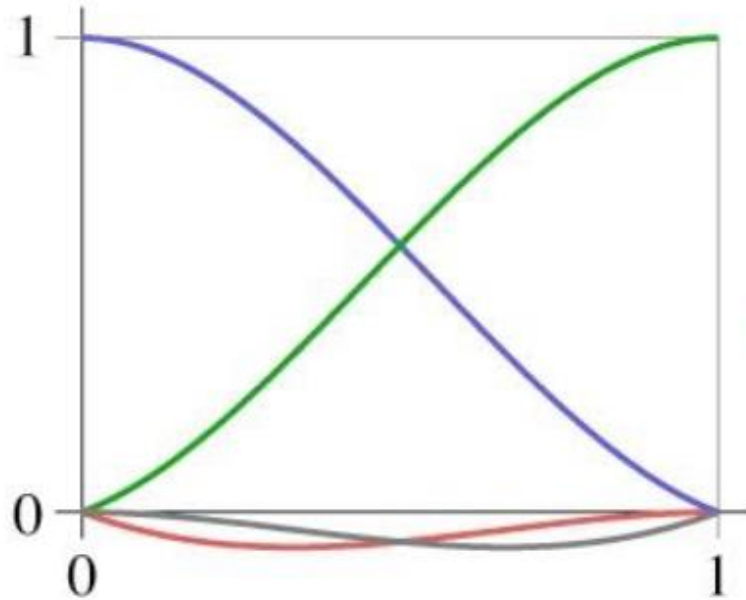
Forma Matricial para o Espaço de Curva Catmull-Rom

$$P(t) = \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}^T \cdot \begin{bmatrix} -\frac{1}{2} & \frac{3}{2} & -\frac{3}{2} & \frac{1}{2} \\ 1 & -\frac{5}{2} & 2 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

$$P(t) = C_{0(t)}\mathbf{p}_0 + C_{1(t)}\mathbf{p}_1 + C_{2(t)}\mathbf{p}_2 + C_{3(t)}\mathbf{p}_3$$

Colunas de matriz = funções interpoladoras Catmull-Rom

Funções Interpoladoras Catmull-Rom



Double Buffering

As imagens são organizadas em Buffers.

Tradicionalmente em computação gráfica temos:

- Front Buffer
- Back Buffer

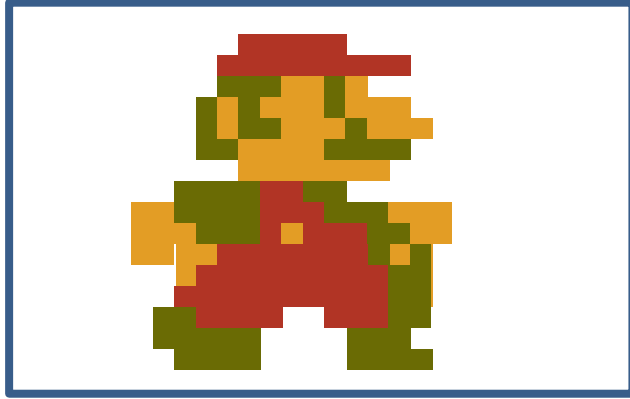
Durante a renderização as imagens são desenhadas no Back Buffer, enquanto as imagens exibidas vem do Front Buffer.

Na atualização de quadros, os buffers são trocados (Page Flipping). O Back Buffer é então limpo e se pode desenhar novamente nele.

Double Buffering

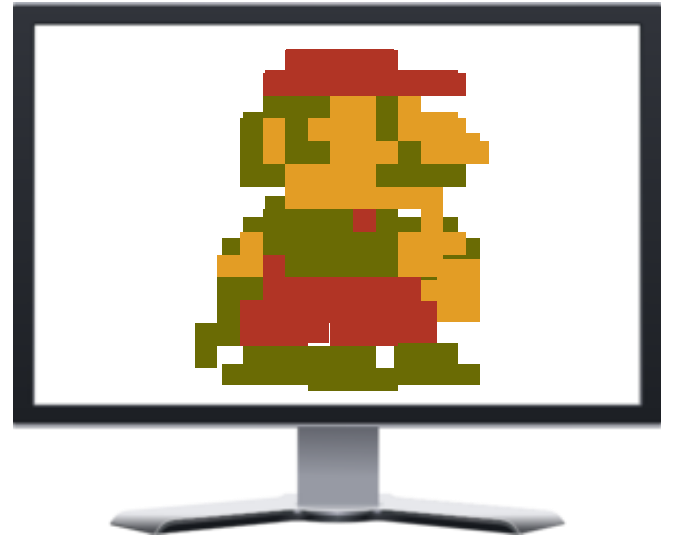
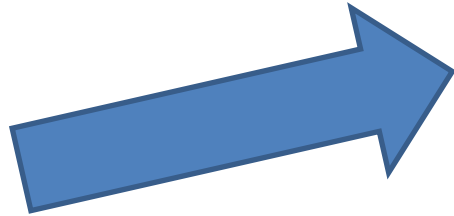
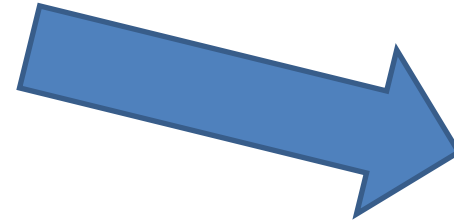
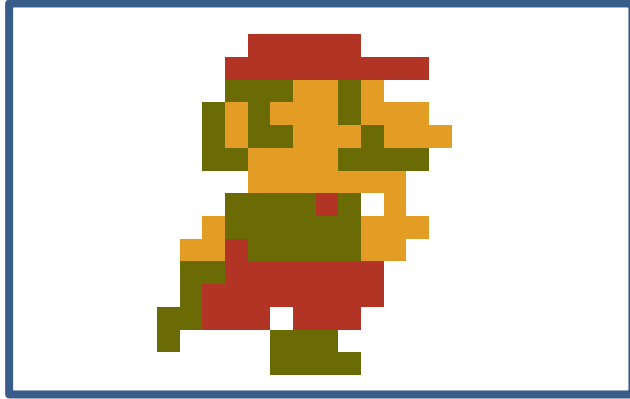
framebuffer 0

FRONT



framebuffer 1

BACK



Novos Nós X3D : TimeSensor

TimeSensor pode ser usado para:

- Condução de simulações e animações contínuas;
- Controlar atividades periódicas;
- Iniciar eventos de ocorrência única, como um despertador;

O ciclo de um nó TimeSensor dura **cycleInterval** segundos. Se, no final de um ciclo, o valor do **loop** for FALSE, a execução é encerrada. Já se o loop for TRUE no final de um ciclo, um nó dependente do tempo continua a execução no próximo ciclo. Deve retornar a fração de tempo passada em **fraction_changed**.

```
TimeSensor : X3DTimeDependentNode, X3DSensorNode {  
  SFTime  [in,out] cycleInterval  1      (0,∞)  
  SFBool  [in,out] enabled       TRUE  
  SFBool  [in,out] loop          FALSE  
  SFNode  [in,out] metadata      NULL  [X3DMetadataObject]  
  SFTime  [in,out] pauseTime     0      (-∞,∞)  
  SFTime  [in,out] resumeTime    0  
  SFTime  [in,out] startTime     0      (-∞,∞)  
  SFTime  [in,out] stopTime      0      (-∞,∞)  
  SFTime  [out]   cycleTime  
  SFTime  [out]   elapsedTime  
  SFFloat [out]   fraction_changed  
  SFBool  [out]   isActive  
  SFBool  [out]   isPaused  
  SFTime  [out]   time  
}
```

Novos Nós X3D : SplinePositionInterpolator

Interpola não linearmente entre uma lista de vetores 3D. O campo **keyValue** possui uma lista com os valores a serem interpolados, **key** possui uma lista respectiva de chaves dos valores em **keyValue**, a fração a ser interpolada vem de **set_fraction** que varia de zero a um. O campo **keyValue** deve conter exatamente tantos vetores 3D quanto os quadros-chave no **key**. O campo **closed** especifica se o interpolador deve tratar a malha como fechada, com uma transições da última chave para a primeira chave. Se os **keyValues** na primeira e na última chave não forem idênticos, o campo **closed** será ignorado. O resultado final é definido no **value_changed**.

```
SplinePositionInterpolator : X3DInterpolatorNode {  
  SFFloat [in]    set_fraction      (-∞,∞)  
  SFBool  [in,out] closed           FALSE  
  MFFloat [in,out] key              []   (-∞,∞)  
  MFVec3f [in,out] keyValue         []   (-∞,∞)  
  MFVec3f [in,out] keyVelocity      []   (-∞,∞)  
  SFNode  [in,out] metadata         NULL [X3DMetadataObject]  
  SFBool  [in,out] normalizeVelocity FALSE  
  SFVec3f [out]    value_changed  
}
```

Novos Nós X3D : OrientationInterpolator

Interpola rotações são absolutas no espaço do objeto e, portanto, não são cumulativas. Uma orientação representa a posição final de um objeto após a aplicação de uma rotação. Um OrientationInterpolator interpola entre duas orientações calculando o caminho mais curto na esfera unitária entre as duas orientações. A interpolação é linear em comprimento de arco ao longo deste caminho. Os resultados são indefinidos se as duas orientações forem diagonalmente opostas. O campo **keyValue** possui uma lista com os valores a serem interpolados, **key** possui uma lista respectiva de chaves dos valores em **keyValue**, a fração a ser interpolada vem de **set_fraction** que varia de zero a um. O campo **keyValue** deve conter exatamente tantas rotações 3D quanto os quadros-chave no **key**. O resultado final é definido no **value_changed**.

```
OrientationInterpolator : X3DInterpolatorNode {  
  SFFloat    [in]    set_fraction      (-∞,∞)  
  MFFloat    [in,out] key              []  (-∞,∞)  
  MFRotation [in,out] keyValue         []  [-1,1] or (-∞,∞)  
  SFNode     [in,out] metadata         NULL [X3DMetadataObject]  
  SFRotation [out]    value_changed  
}
```

Declaração ROUTE

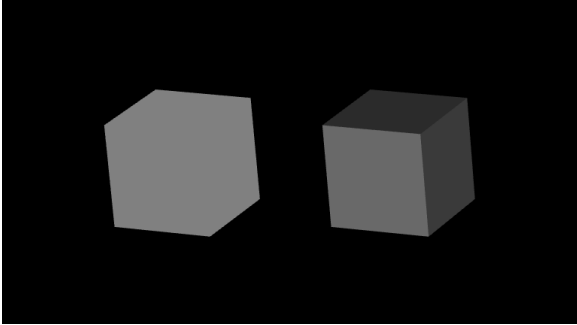
Em X3D as conexões entre campos de um nó para campos de outros nós usando são realizadas pela instrução ROUTE.

```
ROUTE <fromNodeName> <fromFieldName> <toNodeName> <toFieldName>
```

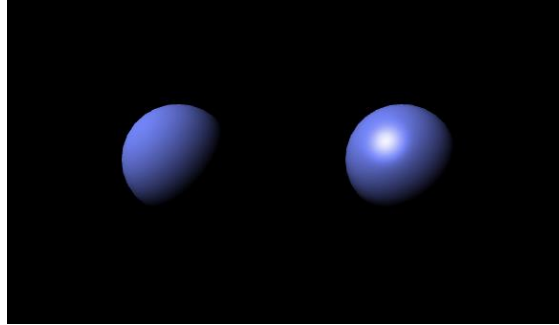
onde **<fromNodeName>** identifica o nó que irá gerar um evento, **<fromFieldName>** é o nome do campo do nó gerador do qual originará o evento, **<toNodeName>** identifica o nó que receberá um evento, e **<toFieldName>** identifica o campo no nó de destino que receberá o evento.

ROUTEs não são nós. A instrução ROUTE é uma construção para estabelecer caminhos de eventos entre campos especificados de nós.

Parte extra do projeto 1



caixas.x3d



esferas.x3d

<https://lpsoares.github.io/Renderizador/>

Computação Gráfica

Luciano Soares

<lpsoares@insper.edu.br>

Fabio Orfali

<fabioO1@insper.edu.br>