
Curvas y superficies en 2D y 3D

Índice

- Curvas en 2D y 3D
 - Introducción
 - Interpolación lineal
 - Interpolación por Curvas Spline
 - Interpolación de Hermite
 - Aproximación por Curvas de Bézier
 - Aproximación por Curvas B-Spline
- Superficies en 3D
 - Interpolación bilineal
 - Parches bicúbicos

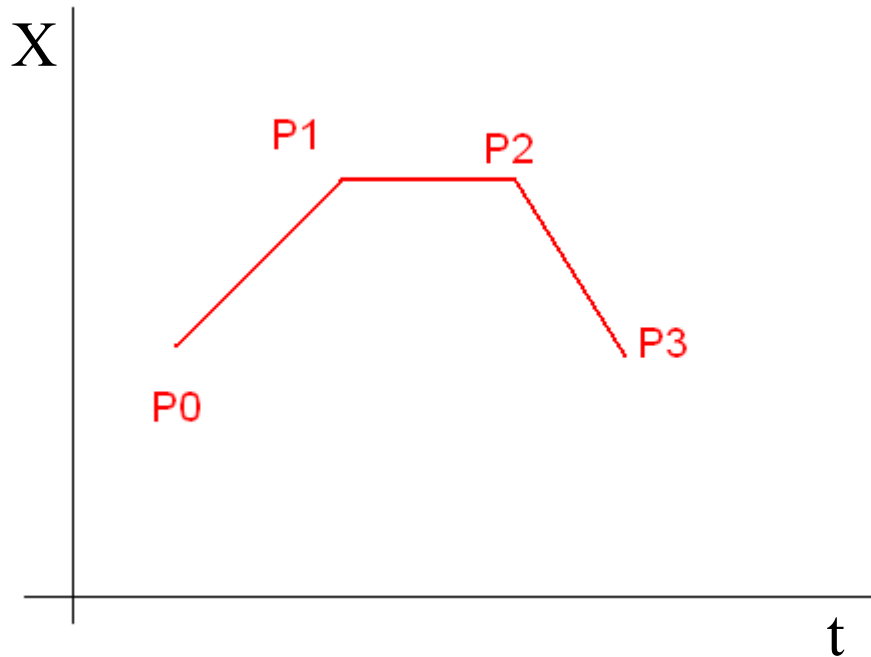
Introducción

- **Representación paramétrica vs implícita o explícita**
 - paramétrica es más flexible
 - curvas componentes: $C(t) = (x(t), y(t))$
- **Interpolación vs. aproximación: puntos de control**
 - interpolación: curva ha de pasar por una serie de puntos (importancia de los puntos)
 - aproximación: curva según unos puntos de control (importancia de curva)
- **Propiedades deseables:**
 - representación paramétrica
 - suave: C^n , en curvas componentes
 - sin oscilaciones (*wiggles*)
 - local: cambio de un punto afecta a entorno reducido
 - fácil de calcular: poco coste computacional

Interpolación lineal

- Dado un conjunto de puntos se interpolan usando rectas entre ellos.
- Sencillo.
- La curva es continua pero no sus derivadas.
- Curva local: la modificación de un punto afecta a dos intervalos.

Interpolación lineal



Entre dos puntos se define una línea recta.

$$X(t) = m_x t + b_x \text{ (explícita)}$$

$$X(t) - X_0 = m(t - t_0) \text{ (pto. pendiente)}$$

Con las condiciones

$$X(t=0) = X_0$$

$$X(t=1) = X_1$$

Para el primer intervalo y la primera coordenada.

Splines Cúbicos

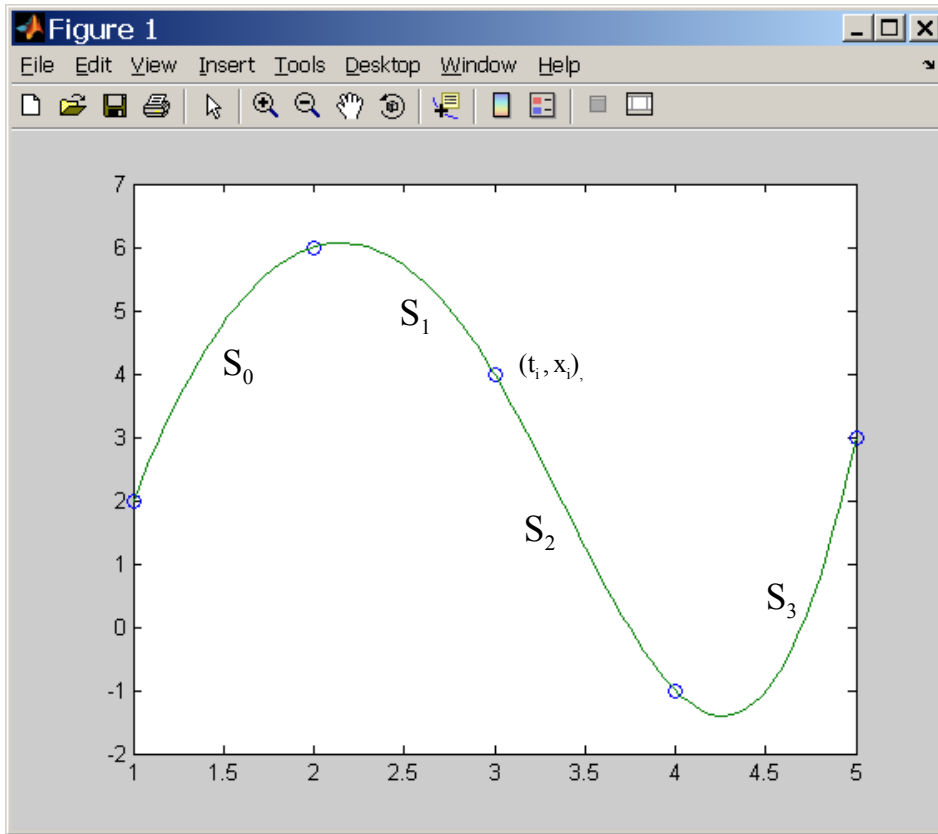
- Interpolan entre dos puntos utilizando un polinomio de grado 3
- Los polinomios de grado 3 son los de menor grado que permiten la existencia de un punto de inflexión.
- Intentan evitar oscilaciones y complejidad de interpolación polinómica, al aumentar el número de puntos
- Si el polinomio es de grado m , 3 en este caso, se puede imponer que la curva global sea continua hasta el orden $m-1$ (C^{m-1}), en este caso: grado 2. Es decir, podemos imponer que sea continua la curva, la primera y la segunda derivada, es decir, curvas suaves.

Splines Cúbicos

- **Spline cúbico $S(t)$ que pasa por los $n+1$ puntos P_0, P_1, \dots, P_n para los valores del parámetro $\{t_0, t_1, \dots, t_n\}$**
- Se buscan n polinomios cúbicos S_i (grado 3), uno para cada “tramo” de la curva, definida en el intervalo $[t_i, t_{i+1}]$, que empalmen con continuidad C^2 (2ª derivada) en cada valor del parámetro t_i
- Condiciones: $2n + n - 1 + n - 1 = 4n - 2$
 $S_i(t_i) = x_i ; S_i(t_{i+1}) = x_{i+1} \quad i=0, 1, \dots, n-1$ $2n$ condiciones Continuidad curva
- $S_i'(t_{i+1}) = S_{i+1}'(t_{i+1}) \quad i=0, 1, \dots, n-2$ $n-1$ condiciones Cont. prim. Deriv.
- $S_i''(t_{i+1}) = S_{i+1}''(t_{i+1}) \quad i=0, 1, \dots, n-2$ $n-1$ condiciones Cont. Seg. Deriv.
- Incógnitas: $4n$
 $S_i(t) = a_i + b_i(t-t_i) + c_i(t-t_i)^2 + d_i(t-t_i)^3$
- Faltan 2 condiciones extra

Splines Cúbicos

- Natural Cubic Splines:
 - Condiciones extra: derivada segunda nula en los extremos, t_0 , t_n
 - $S''_0(t_0)=S''_{n-1}(t_n)=0$



$$\begin{aligned}
 S_0 &= a_0 + b_0(t-t_0) + c_0(t-t_0)^2 + d_0(t-t_0)^3 \\
 S_1 &= a_1 + b_1(t-t_1) + c_1(t-t_1)^2 + d_1(t-t_1)^3 \\
 S_2 &= a_2 + b_2(t-t_2) + c_2(t-t_2)^2 + d_2(t-t_2)^3 \\
 S_3 &= a_3 + b_3(t-t_3) + c_3(t-t_3)^2 + d_3(t-t_3)^3
 \end{aligned}
 \left. \vphantom{\begin{aligned} S_0 \\ S_1 \\ S_2 \\ S_3 \end{aligned}} \right\} 16 \text{ inc\u00f3gnitas}$$

$$\begin{aligned}
 S_0(t_0) &= x_0 & S'_0(t_1) &= S'_1(t_1) \\
 S_0(t_1) &= x_1 & S'_1(t_2) &= S'_2(t_2) \\
 S_1(t_1) &= x_1 & S'_2(t_2) &= S'_3(t_3) \\
 S_1(t_2) &= x_2 & S''_0(t_1) &= S''_1(t_1) \\
 S_2(t_2) &= x_2 & S''_1(t_2) &= S''_2(t_2) \\
 S_2(t_3) &= x_3 & S''_2(t_3) &= S''_3(t_3) \\
 S_3(t_3) &= x_3 & & \\
 S_3(t_4) &= x_4 & &
 \end{aligned}
 \left. \vphantom{\begin{aligned} S_0(t_0) \\ S_0(t_1) \\ S_1(t_1) \\ S_1(t_2) \\ S_2(t_2) \\ S_2(t_3) \\ S_3(t_3) \\ S_3(t_4) \end{aligned}} \right\} 14 \text{ ecuaciones}$$

Imponiendo las condiciones de Natural Cubic Spline $S''_0(t_0) = S''_3(t_4) = 0$ obtenemos 2 ecuaciones adicionales que nos permiten resolver el sistema.

Splines cúbicos

- Curva interpolante, con segmentos polinómicos (curvas componentes)
- Representación paramétrica o cartesiana
- Suave: C^2 (curvas componentes)
- Sin oscilaciones: grado cúbico de los polinomios evita oscilaciones
- No local: el cambio de un punto afecta a los polinomios de todos segmentos (ver el sistema $(n+1) \times (n+1)$)
- Relativamente fácil de calcular: el sistema de ecuaciones es tridiagonal
- Se ha sacrificado la suavidad (no mucho) para evitar las oscilaciones
- Para hacer las curvas locales, hay que eliminar el requerimiento de que interpole

Interpolación de Hermite

- Al igual que en la interpolación por Splines, partimos de una serie de $n+1$ puntos P_0, \dots, P_n , que se interpolan por medio de una curva cúbica por “trozos” o “tramos”.
- Se busca una función de interpolación $H_i(t)$ que sea cúbica en cada subintervalo y que interpole a la curva y a su primera derivada en los puntos que introduce el usuario.
- La función $H_i(t)$ queda determinada en forma única por estas condiciones y su cálculo requiere de la solución de n *sistemas lineales* de tamaño 4×4 cada uno.
- La desventaja de la interpolación de Hermite es que requiere de la disponibilidad de las primeras derivadas, lo cual no es el caso en muchas aplicaciones.

Interpolación de Hermite

Para cada intervalo entre dos puntos que introduce el usuario $\mathbf{P}_i(t_i, x_i)$ y $\mathbf{P}_{i+1}(t_{i+1}, x_{i+1})$ con tangente en el punto inicial \mathbf{m}_i y en el punto final \mathbf{m}_{i+1} , el polinomio interpolador se expresa de la siguiente forma:

$$H_i(s) = (2s^3 - 3s^2 + 1)x_i + (s^3 - 2s^2 + s)m_i + (-2s^3 + 3s^2)x_{i+1} + (s^3 - s^2)m_{i+1}$$

- donde $s \in [0, 1]$, variable auxiliar $s = \frac{t - t_i}{t_{i+1} - t_i}$
- Nota: Obsérvese que en el polinomio aparecen los valores de x y m . Atención: t_i es la abscisa y x_i la ordenada: $\mathbf{P}_i(t_i, x_i)$

Interpolación de Hermite

- Como cada subintervalo comparte tangentes con los vecinos hay varias técnicas para definir los valores de las tangentes intermedias en caso de que no se proporcionen como datos del problema.
 - Cardinal spline
 - Catmull-Rom spline
 - Kochanek-Bartels spline

Interpolación de Hermite

- En Informática Gráfica los más utilizados son los splines de **Catmull-Rom**. Sobre todo cuando lo que se desea es interpolar movimiento de forma suave entre varios frames. Por ejemplo cuando la cámara se está moviendo y tenemos unos frames clave y queremos interpolar el movimiento entre ellos.
- La ventaja es que es fácil de calcular, garantizan la posición en los frames clave (es una interpolación) y garantizan que la tangente de la curva generada es continua a lo largo de múltiples segmentos.

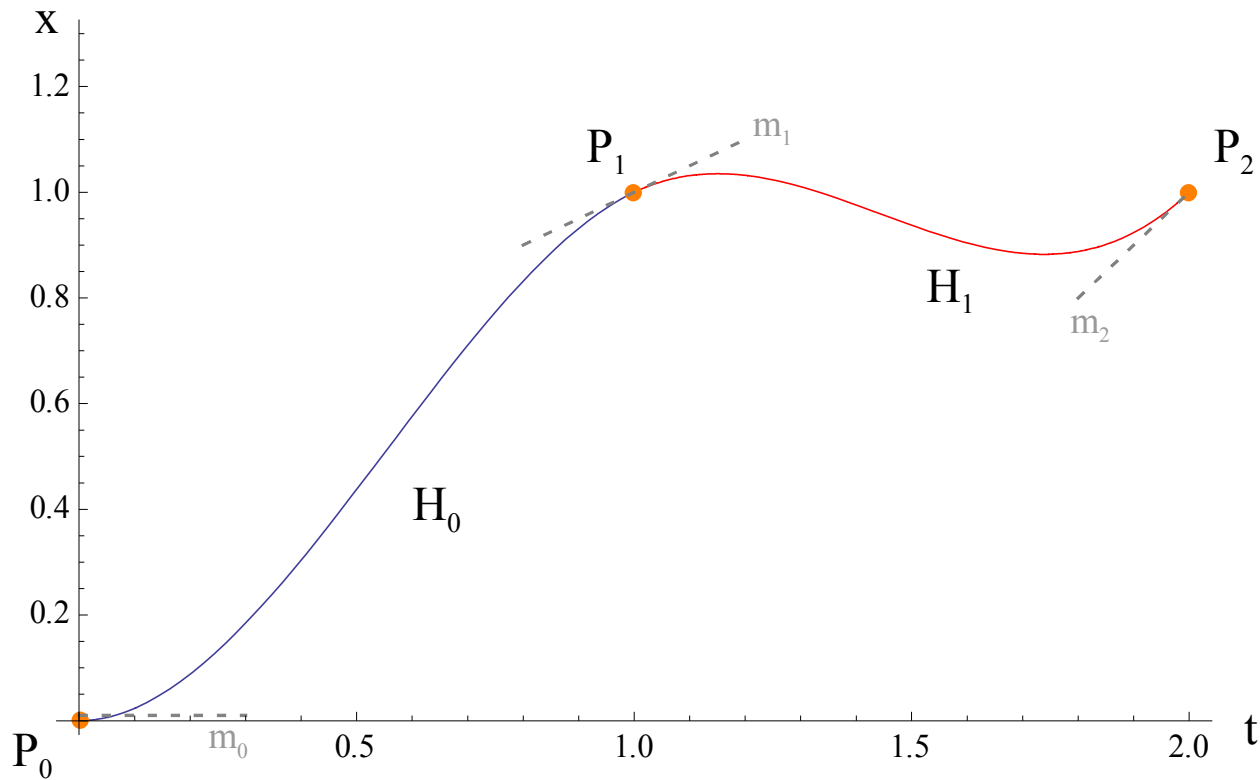
Interpolación de Hermite

Dados $(n+1)$ puntos $\mathbf{P}_0, \dots, \mathbf{P}_n$, para interpolarlos con n segmentos de curva de tipo polinomios cúbicos de Hermite se divide la curva global en intervalos cada uno de los cuales empieza en un punto (t_i, x_i) y termina en (t_{i+1}, x_{i+1}) siendo la tangente inicial \mathbf{m}_i y la final \mathbf{m}_{i+1} y se definen las tangentes por medio de la fórmula:

$$m_i = \frac{1}{2}(x_{i+1} - x_{i-1})$$

- Nota: deben definirse \mathbf{m}_0 y \mathbf{m}_n como parte del diseño.

- Ejemplo: $P_0(0,0)$, $P_1(1,1)$, $P_2(2,1)$
- $m_0 = 0$, $m_1 = 0.5$, $m_2 = 1$
- $H_0(s) = 2.5s^2 - 1.5s^3$, $s = t$
- $H_1(s) = 1.5s^3 - 2s^2 + 0.5s + 1$, $s = t-1$



Demo

- Saltos en el ejemplo cuando la interpolación del movimiento es lineal, movimiento suave en la interpolación de catmull-rom. Demo disponible en:
- <http://www.mvps.org/directx/articles/catmull/>

Curvas de Bézier

- Es un sistema desarrollado hacia los años setenta del siglo XX, para el trazado de dibujos técnicos, en el diseño aeronáutico y de automóviles. Su denominación es en honor a Pierre Bézier quien ideó un método de descripción matemática de las curvas que se comenzó a utilizar con éxito en los programas de CAD.
- Posteriormente, los inventores del PostScript, introdujeron en ese código el método de Bézier para la generación del código de las curvas y los trazados.
- Applet sencillo para generación de curvas de Bézier:

<http://www.cs.technion.ac.il/~cs234325/Applets/applets/bezier/GermanApplet.html>

Curvas de Bézier

- Se denomina curva Bézier asociada a los $(n + 1)$ puntos P_0, P_1, \dots, P_n a la curva parametrizada, definida para $t \in [0, 1]$, cuyos puntos vienen dados mediante la siguiente expresión

$$(x(t), y(t)) = \sum_{i=0}^n B_{i,n}(t) P_i$$

- en la que los $B_{i,n}(t)$ son los polinomios de Bernstein de grado n .
- Los puntos P_0, P_1, \dots, P_n que determinan una curva de Bézier se denominan puntos de control, y la poligonal que los une es el polígono Bézier o B-polígono.

Curvas de Bézier

- Los polinomios de Bernstein de grado n , que denotamos por $B_{0,n}(t)$, $B_{1,n}(t)$, ..., $B_{n,n}(t)$, vienen dados por la expresión:

$$B_{i,n}(t) = \frac{n!}{i!(n-i)!} (1-t)^{n-i} t^i$$

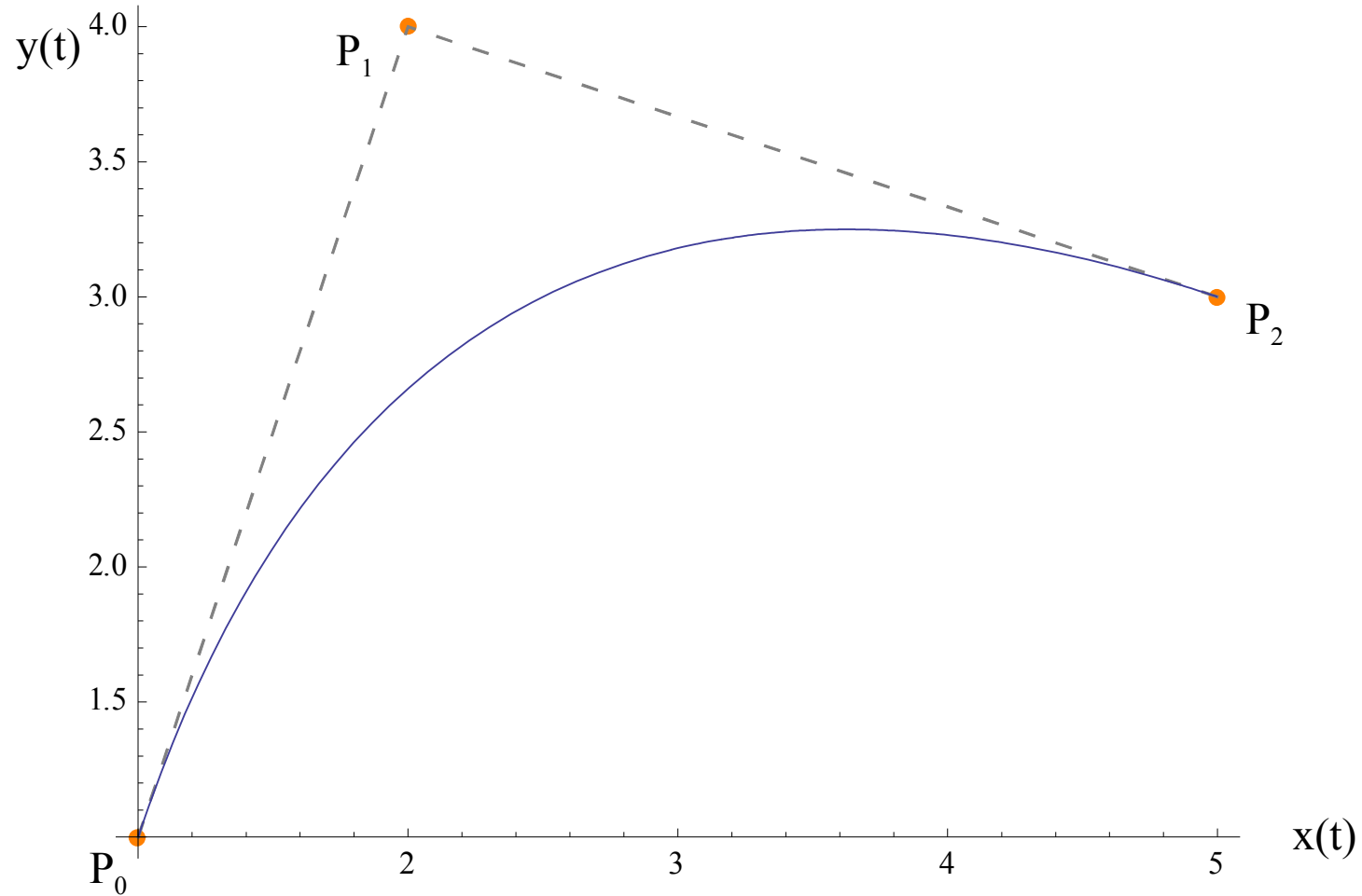
Curvas de Bézier

- $B_{0,1}(t) = (1 - t)$, $B_{1,1}(t) = t$;
- $B_{0,2}(t) = (1-t)^2$, $B_{1,2}(t) = 2(1-t)t$, $B_{2,2}(t) = t^2$;
- $B_{0,3}(t) = (1-t)^3$, $B_{1,3}(t) = 3(1-t)^2t$, $B_{2,3}(t) = 3(1-t)t^2$,
 $B_{3,3}(t) = t^3$.
- A medida que aumenta el número de puntos, aumenta el grado de la curva.
- Si se limita a 4 el número de puntos de control el polinomio es de grado 3.

Curvas de Bézier

- Ejemplo de curva de Bézier:
- Dados los puntos $P_0=(1, 1)$, $P_1=(2, 4)$, $P_2=(5, 3)$, la curva Bézier asociada tiene las siguientes ecuaciones paramétricas
- $x(t) = B_{0,2}(t) + 2B_{1,2}(t) + 5B_{2,2}(t) = 1+2t+2t^2$
- $y(t) = B_{0,2}(t) + 4B_{1,2}(t) + 3B_{2,2}(t) = 1+6t-4t^2$

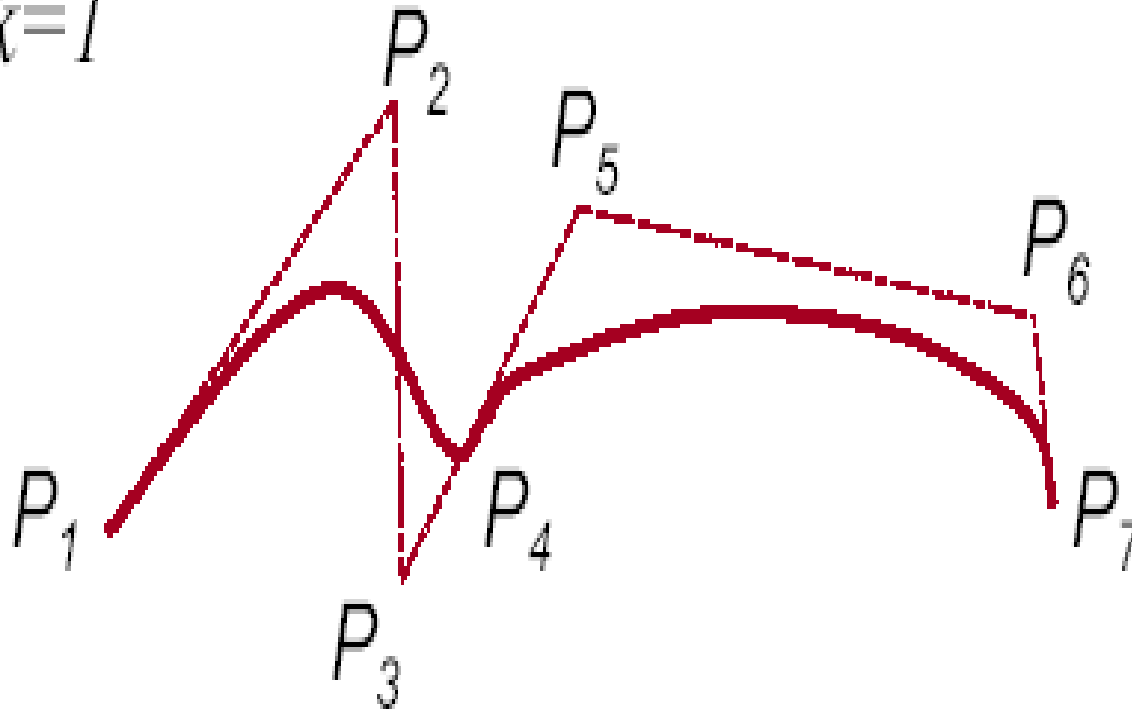
Curvas de Bézier



Curvas de Bézier

- La curva de Bézier empieza en P_0 y termina en P_n
- El vector tangente a la curva $P(t)$ en el punto P_0 tiene la dirección del vector P_0P_1
- El vector tangente a la curva $P(t)$ en el punto P_n tiene la dirección del vector $P_{n-1}P_n$.
- La modificación de un punto de control afecta a toda la curva que define.

■ C^1 si $k=1$



- En este caso se tienen 7 puntos, se toman 2 subconjuntos, de P_1 a P_4 y de P_4 a P_7 .
- Aproximando cada conjunto de 4 puntos con una curva Bézier de orden 3, obtenemos una curva continua
- Si el segmento P_3 - P_4 tiene la misma pendiente que P_4 - P_5 (es decir, si el cociente de pendientes es $k=1$), la curva global es C^1

Curvas B-Spline

- Generalización de las curvas de Bézier, especialmente indicada si el número de puntos es muy grande y queremos encontrar una curva de aproximación de grado no muy alto.
- Dado un conjunto de puntos P_0, \dots, P_n , obtenemos una curva de aproximación compuesta por varios tramos, y las ecuaciones de cada tramo están influenciadas solamente por k vértices del polígono de control siendo k (orden de la B-Spline) un parámetro elegido a voluntad por el diseñador y, lógicamente, $k \leq n + 1$

Curvas B-Spline

Expresión explícita:
$$P(t) = \sum_{i=0}^n N_{i,k}(t) P_i$$

Los parámetros que intervienen en una curva B-Spline se enumeran a continuación:

- $\mathbf{P}_0, \dots, \mathbf{P}_n$, $n+1$ vértices o puntos de control
- $\mathbf{N}_{i,k}$: funciones B-Spline básicas de orden k
- \mathbf{d} : grado de las B-Spline básicas (elección usual, $d=3$)
- \mathbf{k} : orden de la B-Spline: $k = d+1$
- \mathbf{N}° de tramos: $n-d+1$
- **Suavidad** global de la curva: $C^{k-2} = C^{d-1}$

Curvas B-Spline

Es necesario definir un vector de nodos para el cálculo de las funciones B-Spline básicas:

- **m**: número de nodos, $m = n+k+1$
- **Vector de nodos** $\{t_i\} = \{t_0, \dots, t_{n+k}\}$

La elección más habitual del vector de nodos asegura que la B-Spline pasa por P_0 y P_n , y asigna un peso uniforme a cada vértice o punto de control. Viene dada por:

$$t_i = \begin{cases} 0 & \text{si } i < k \\ i-k+1 & \text{si } k \leq i \leq n \\ n-k+2 & \text{si } i > n \end{cases}$$

Curvas B-Spline

$$P(t) = \sum_{i=0}^n N_{i,k}(t) P_i$$

$N_{i,k}$ Funciones B-Spline básicas de orden k . Son funciones de ponderación polinómicas a trozos, se definen por recurrencia por medio de la fórmula de cálculo de Cox-de Boor:

$$N_{i,1}(t) = \begin{cases} 1 & t_i \leq t < t_{i+1} \\ 0 & \text{resto} \end{cases}; \quad \text{orden 1}$$

$$N_{i,j}(t) = \left(\frac{t - t_i}{t_{i+j-1} - t_i} \right) N_{i,j-1}(t) + \left(\frac{t_{i+j} - t}{t_{i+j} - t_{i+1}} \right) N_{i+1,j-1}(t); \quad \text{orden } 1 < j \leq k; \quad t \in [t_i, t_{i+j}]; \quad i + j \leq n + k$$

Si hay nodos repetidos se tiene una indeterminación $0/0$ que se resuelve como 0

Propiedades que cumplen las funciones B-Spline básicas:

$$\sum_{i=0}^n N_{i,j}(t) = 1; \quad N_{i,j}(t) \geq 0$$

Propiedades

- No interpolan (salvo en P_0, P_n , si así se especifica)
- Paramétricas $P(t) = (x(t), y(t))$
- Suavidad C^{k-2} : k es el orden de la B-spline
- No oscilan
- Locales
- Difíciles de calcular salvo casos especiales con fórmula matricial: B-Splines uniformes, Bézier
- Mayor flexibilidad: elección de nodos permite más tipos de curvas

Cálculo de B-Splines

Descripción detallada sobre los cálculos y algoritmos asociados a las curvas B-Spline.

- http://personal.us.es/esplebrue/ampcap5a_0506.pdf

Applet Java para la simulación de B-Splines

- <http://www.engin.umd.umich.edu/CIS/course.des/cis577/projects/BSP/welcome.html>

Ejemplo

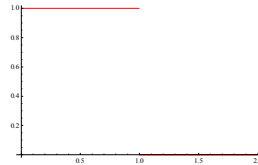
- Dados $P_0(0,0)$, $P_1(1,1)$, $P_2(3,0)$, $P_3(4,2)$, calcular la curva B-Spline de grado 2 determinada por dichos puntos
- Parámetros: $n=3$ (4 puntos), $d=2$ (grado), $k=3$ (orden)
- La curva estará formada por $n-d+1=2$ tramos
- Su suavidad global será C^1
- Existirán $m=n+k+1=7$ nodos
- El vector de nodos es $t_0 = t_1 = t_2 = 0$, $t_3 = 1$, $t_4 = t_5 = t_6 = 2$
- La curva B-Spline pedida es $P(t) = (x(t), y(t)) = \sum_{i=0}^3 N_{i,3}(t) P_i$
- Para calcular $N_{i,3}$ es necesario calcular primero $N_{i,1}$, $N_{i,2}$ por la fórmula recursiva de Cox-de Boor.

Ejemplo

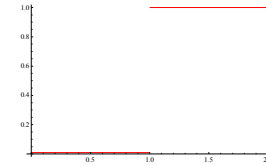
Cálculo de B-Spline básicas de orden 1:

$$N_{01} = N_{11} = 0; N_{41} = N_{51} = 0$$

$$N_{21} = \begin{cases} 1 & \text{en } [0,1] \\ 0 & \text{en } [1,2] \end{cases}$$



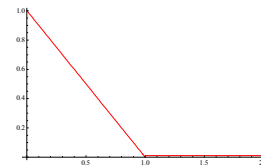
$$N_{31} = \begin{cases} 0 & \text{en } [0,1] \\ 1 & \text{en } [1,2] \end{cases}$$



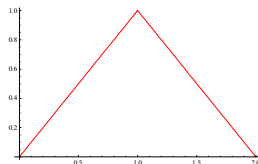
Cálculo de B-Spline básicas de orden 2:

$$N_{02} = N_{42} = 0 \text{ (0/0 se resuelve como 0)}$$

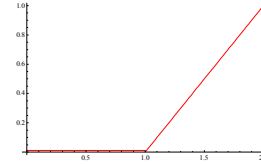
$$N_{12} = \begin{cases} 1-t & \text{en } [0,1] \\ 0 & \text{en } [1,2] \end{cases}$$



$$N_{22} = \begin{cases} t & \text{en } [0,1] \\ 2-t & \text{en } [1,2] \end{cases}$$

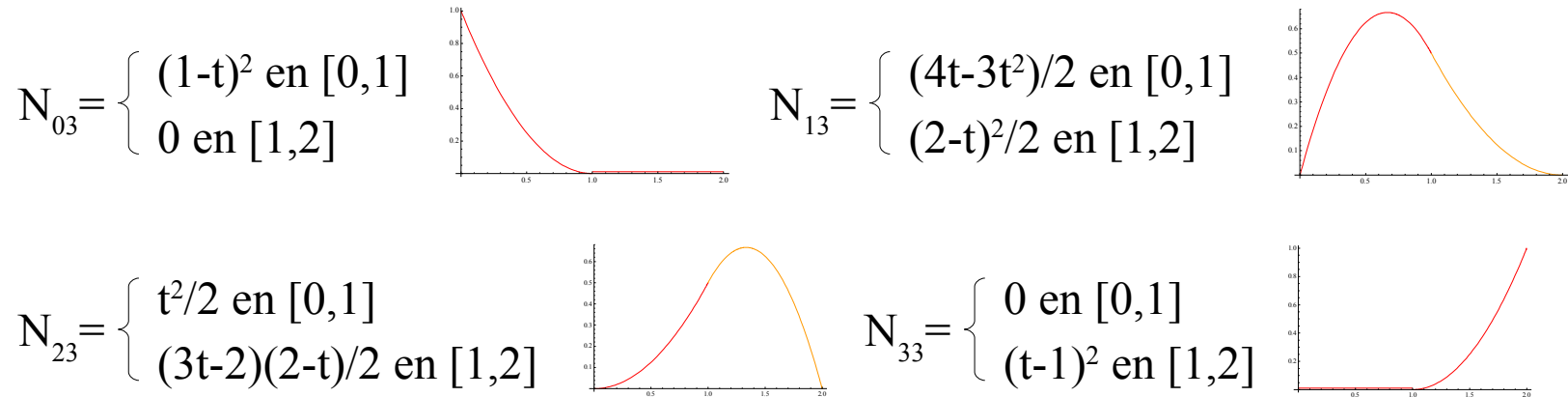


$$N_{32} = \begin{cases} 0 & \text{en } [0,1] \\ t-1 & \text{en } [1,2] \end{cases}$$



Ejemplo

Cálculo de B-Spline básicas de orden 3:



Las ecuaciones explícitas de la curva son:

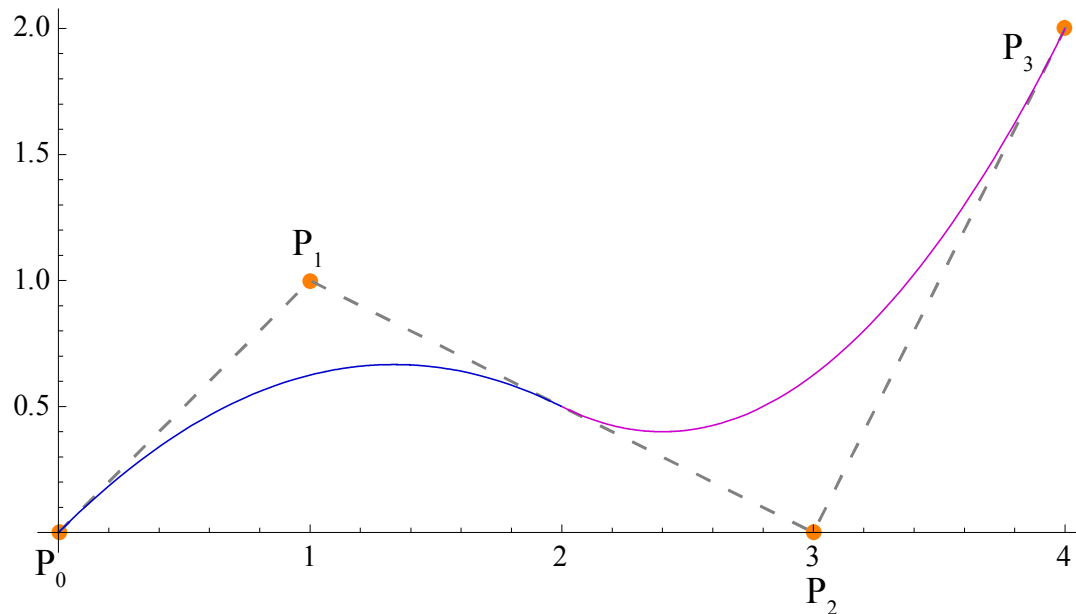
$$P(t) = (x(t), y(t)) = \sum_{i=0}^3 N_{i,3}(t) P_i = N_{0,3} P_0 + N_{1,3} P_1 + N_{2,3} P_2 + N_{3,3} P_3$$

$$(x(t), y(t)) = (N_{1,3} + 3 N_{2,3} + 4 N_{3,3}, N_{1,3} + 2 N_{3,3})$$

Ejemplo

Representación gráfica:

$$x(t) = \begin{cases} 2t & \text{en } [0,1] \\ 2t & \text{en } [1,2] \end{cases} \quad y(t) = \begin{cases} (4t-3t^2)/2 & \text{en } [0,1] \\ 5t^2/2-6t+4 & \text{en } [1,2] \end{cases}$$



- Suavidad global C^1
- P_0, P_1, P_2 determinan el primer tramo de la B-Spline (azul)
- P_1, P_2, P_3 determinan el segundo tramo de la B-Spline (morado)
- Si se modifica P_0 , p.ej., sólo se ve afectado el primer tramo de la B-Spline, el 2º no cambia, y se mantiene la suavidad C^1 global

Superficies 3D. Interpolación Bilineal

- La *interpolación bilineal* calcula los puntos intermedios mediante la siguiente ecuación:
- $V(x, y) = ax + by + cxy + d$
- donde x e y representan la distancia al punto o esquina superior izquierda del entorno.
- Los coeficientes a, b, c, d deben calcularse de modo que se cumplan las siguientes igualdades:
- $V(x = 0, y = 0) = P(i, j)$
- $V(x=1, y=0) = P(i+1, j)$
- $V(x = 0, y = 1) = P(i, j + 1)$
- $V(x=1, y=1) = P(i + 1, j + 1)$

Superficies en 3D

- **Concepto de parche (patch)**
- Superficie como curva con puntos de control en una curva

$$S(s, t) = \sum_{j=0}^n P'_j(s) b_j(t) \quad ; \quad P'_j(s) = \sum_{i=0}^m P_{ij} b_i(s)$$

$$S(s, t) = \sum_{i=0}^m \sum_{j=0}^n P_{ij} b_i(s) b_j(t) = \sum_{i=0}^m \sum_{j=0}^n P_{ij} b_{ij}(s, t)$$

Superficies en 3D

- Funciones base son producto de funciones base de curvas: $b_{ij}(s,t)=b_i(s)b_j(t)$
- Matriz (malla) $m \times n$ de puntos de control: P_{ij} , $i=0,\dots,m$; $j=0,\dots,n$

$$\begin{array}{c} \xrightarrow{t} \\ s \downarrow \begin{array}{cccc} P_{00} & P_{01} & \dots & P_{0n} \\ P_{10} & P_{11} & \dots & P_{1n} \\ \dots & \dots & \dots & \dots \\ P_{m0} & P_{m1} & \dots & P_{mn} \end{array} \end{array}$$