

# CC 52B Computación Gráfica

Representación de curvas y superficies  
(incluyendo mallas de polígonos) para CG y  
aplicaciones de ciencias, ingeniería y medicina.

**Prof. María Cecilia Rivara**  
**mcrivara@dcc.uchile.cl**  
**Semestre 2006/2**

MCRivara/CG2006/2

## Representación / modelación de superficies

- Mallas de polígonos
- Superficies paramétricas (superficies curvas)
- Triangulaciones

MCRivara/CG2006/2

## Motivación en ingeniería

- En aplicaciones de la industria de entretenimientos (CG) no interesa la “modelación precisa” o “representación precisa” de objetos.
- En aplicaciones de ciencias, ingeniería y medicina, la obtención y manejo de modelos o representaciones con una precisión requerida es un problema complejo e importante

## Motivación en CG

- Curvas y superficies se usan en muchas aplicaciones de Computación Gráfica
- Muchos objetos del mundo real son intrínsecamente “suaves”: aplicaciones CAD en ingeniería, fonts de alta calidad, el camino de una cámara u objeto en secuencia animada.
- Dos tipos de modelación de curvas y superficies
  - modelación de objeto que existe (automóvil, montaña)
  - a partir de “cero”: se modela (construye) el objeto a través del proceso.

- Objeto que existe
  - descripción matemática puede no estar disponible
  - puede describirse con pedazos de planos, esferas u otras formas curvas fáciles describir
  - se requiere que puntos del modelo estén “cerca” de puntos del objeto real
- Objeto a partir de cero
  - se puede esculpir interactivamente
  - descripción matemática
  - descripción aproximada

## Tipos de modelos

- Modelación de superficies
  - Mallas de polígonos
  - superficies paramétricas
  - superficies polinómicas por pedazos
  - otros (tema muy amplio)
- Modelación de sólidos
 

representación de volúmenes “rodeados” totalmente por superficies (cubo , avión, edificio)

  - mallas de polígonos
    - » representan bien objetos poliédricos
    - » aproximan superficies curvas (no siempre es adecuado)

## Tipos de modelos (cont.)

- Modelación de curvas en el espacio
  - Curvas polinomiales paramétricas  
tres polinomios en función de un parámetro  $x = x(t)$ ,  $y = y(t)$ ,  $z = z(t)$
- Curvas por pedazos polinomiales
- Superficies polinomiales por pedazos bivariados (dos variables)  $x = x(u, v)$ ,  $y = y(u, v)$ ,  $z = z(u, v)$ 
  - los bordes de cada pedazo son curvas polinomiales paramétricas
  - se necesitan pocos pedazos (comparado con mallas de polígonos) para obtener precisión dada
  - algoritmos para manejar polinomios bivariados son más complejos que los algoritmos para mallas de polígonos
- Superficies cúbicas
  - definidas implícitamente por ecuación  $f(x, y, z) = 0$   
donde  $f$  es polinomio cúbico

MCRivara/CG2006/2

7

## Mallas (abiertas) de polígonos

- Colección de aristas, vértices y caras tales que
  - cada arista es compartida por dos polígonos
  - cada arista conecta dos vértices
  - cara es secuencia cerrada de aristas (o vértices)
  - cada vértice es compartido al menos por dos aristas

### Importante

- aristas, vértices y cara son elementos topológicos
- líneas (segmentos), puntos y polígonos son elementos geométricos

MCRivara/CG2006/2

8

- Mallas de polígonos pueden representarse de varias maneras distintas, cada una con ventajas y desventajas
  - es necesario elegir la representación más adecuada para cada aplicación evaluada
  - según criterios de espacio versus tiempo
  - operaciones típicas
    - encontrar aristas que inciden en un vértice
    - encontrar polígonos que compartan un vértice o una arista
    - encontrar aristas de un polígono
    - desplegar la malla
    - identificar errores en la representación

## Observaciones

- Cierta grado de redundancia es deseable: se obtienen soluciones más rápidas con más almacenamiento.
- En algunas aplicaciones mucha redundancia puede encarece el proceso.

## Representaciones de mallas de polígonos

1. Representación explícita ingenua (coordenadas duplicadas)

$$P = ((x_1, y_1, z_1), (x_2, y_2, z_2), \dots, (x_n, y_n, z_n))$$

No es útil en general

2. Polígonos definidos mediante punteros a lista de vértices

- cada vértice se guarda una vez
- se pueden cambiar fácilmente las coordenadas del vértice
- difícil encontrar polígonos que comparten una arista
- cada arista se dibuja dos veces al desplegarse

## Representaciones (cont.)

3. Representación de polígonos mediante punteros a lista de aristas

- conjunto de vértices (cada vértice aparece una vez)
  - conjunto de aristas (cada arista aparece una vez)
  - polígono: lista de punteros a aristas
  - arista:
    - apunta a sus dos vértices
    - apunta a 1 o 2 polígonos que la contienen
- polígono  $P = (E_1, E_2 \dots E_n)$
- arista  $E = (V_1, V_2, P_1, P_2)$

## Consistencia de Representaciones de mallas de polígonos

### Chequeos

- todos los polígonos son cerrados
- todas las aristas se usan al menos una vez y (en algunas aplicaciones) con un valor máximo
- cada vértice es referenciado al menos por dos aristas
- en algunas aplicaciones la malla debe ser conexa, o no tener hoyos, etc.

### Otros chequeos

- una arista no puede ser usada dos veces por el mismo polígono
- cada vértice es parte (al menos) de un polígono
- otros

**Observación:** La relación compartir arista es relación de equivalencia binaria. Permite particionar una malla en componentes conexas



## Curvas cúbicas por pedazos

- Se pegan exigiendo condiciones de continuidad entre los pedazos
- Un segmento de curva se define por restricciones en los puntos extremos, vectores tangentes y continuidad entre los segmentos  
polinomio cúbico  $\rightarrow$  4 coeficientes  
 $\Rightarrow$  se necesitan 4 restricciones por pedazo

## Tipos de curvas cúbicas por pedazos

- Hermite
  - dos puntos extremos y dos vectores tangentes
- Bézier
  - dos puntos extremos y otros dos puntos de control que controlan los vectores tangentes en los puntos extremos
- Varios tipos de splines definidos por 4 puntos de control

### Curva cúbica paramétrica

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

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

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

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

$$0 \leq t \leq 1$$

### Matricialmente

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

$$C = \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

$$T = [t^3 \ t^2 \ t \ 1]$$

Derivada de  $Q(t)$ : vector tangente paramétrico

$$\begin{aligned} \frac{d}{dt} Q(t) = Q'(t) &= \begin{bmatrix} \frac{dx(t)}{dt} & \frac{dy(t)}{dt} & \frac{dz(t)}{dt} \end{bmatrix} \\ &= [3t^2 \ 2t \ 1 \ 0] C \end{aligned}$$



## Tipos de continuidad

- Si los segmentos de curva se unen tienen continuidad geométrica  $G^0$
- Si las direcciones (pero no necesariamente las magnitudes) de los vectores tangentes son iguales en la juntura  $\Rightarrow$  continuidad geométrica  $G^1$  (frecuente en CAD)  
pendientes geométricas de los segmentos tangentes son iguales  
 $TV_1 = kTV_2$  con  $k > 0$
- Si los vectores tangentes de dos segmentos cúbicos son iguales (direcciones y magnitudes iguales)  $\Rightarrow$  continuidad  $C^1$

Derivada n-ésima  $\Rightarrow$  continuidad  $C^n$

**Observación**  $Q'(t)$  es velocidad de un punto en una curva con respecto al parámetro  $t$ ; la segunda derivada es la aceleración.

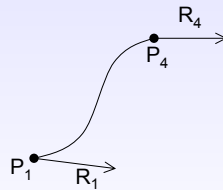
## ¿Cómo se introducen las restricciones?

- Cada pedazo de curva se escribe como:  
 $Q(t) = T C_{4 \times 4}$
- Se reescribe  $C$  en función de una base de los polinomios que son considerados  
 $C = M_{4 \times 4} G$   
 $G$ : vector de las restricciones geométricas  
ejemplo: puntos extremos y vectores tangentes  
 $G_x$ : vector columna de los componentes  $x$  del vector geométrico

## Curvas de Hermite

Restricciones en los puntos extremos  $P_1$  y  $P_4$  y en los vectores tangentes  $R_1$  y  $R_4$  en puntos extremos

Es necesario encontrar la base de Hermite  $M_H$



### Curva de Hermite

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

$$T = [t^3 \ t^2 \ t \ 1]$$

$$C_x = \begin{bmatrix} a_x \\ b_x \\ c_x \\ d_x \end{bmatrix}$$

$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x = TC_x = TM_H G_{H_x}$$

Condiciones (restricciones) que definen la curva en coordenadas  $x$

$$x(0) = P_{1x} = [0 \ 0 \ 0 \ 1] M_H G_{H_x}$$

$$x(1) = P_{4x} = [1 \ 1 \ 1 \ 1] M_H G_{H_x}$$

Además, como

$$x'(t) = [3t^2 \ 2t \ 1 \ 0] M_H G_{H_x} \Rightarrow$$

$$x'(0) = R_{1x} = [0 \ 0 \ 1 \ 0] M_H G_{H_x}$$

$$x'(1) = R_{4x} = [3 \ 2 \ 1 \ 0] M_H G_{H_x}$$

Las 4 Restricciones juntas  $\Rightarrow$

$$\begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix}_x = G_{H_x} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}}_A M_H G_{Hx}$$

Análogamente para y, z

$$G_{Hx} = A M_H G_{Hx} \quad / \text{premultiplicado por } A^{-1} \Rightarrow$$

$$A^{-1} G_{Hx} = M_H G_{Hx} \Rightarrow M_H = A^{-1}$$

$$M_H = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad \text{matriz única!}$$

Luego

$$Q(t) = [x(t) \ y(t) \ z(t)] = \underbrace{T M_H}_{B_H} G_H$$

y expandiendo

$$\begin{aligned} Q(t) = B_H G_H &= (2t^3 - 3t^2 + 1) P_1 + (-2t^3 + 3t^2) P_4 + \\ &\quad (t^3 - 2t^2 + t) R_1 + (t^3 - t^2) R_4 \\ &= P_1(t) P_1 + P_4(t) P_4 + \\ &\quad R_1(t) R_1 + R_4(t) R_4 \end{aligned}$$

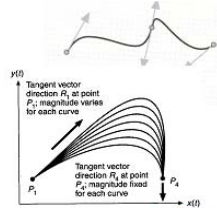
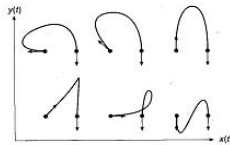
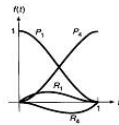
$P_1(t)$ ,  $P_4(t)$ ,  $R_1(t)$ ,  $R_4(t)$ : funciones de mezcla

# Hermite Curves

- Constraints on the endpoints

$$Q(t) = G_H \cdot M_H \cdot T = G_H \cdot B_H$$

$$= (2t^3 - 3t^2 + 1)P_1 + (-2t^3 + 3t^2)P_4 + (t^3 - 2t^2 + t)R_1 + (t^3 - t^2)R_4.$$

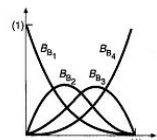
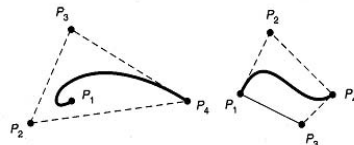


Christer Åhlund



# Bezier Curves

$$R_1 = Q'(0) = 3(P_2 - P_1), R_4 = Q'(1) = 3(P_4 - P_3)$$



Christer Åhlund



## Curvas de Bezier

- Se definen en función de  $P_1, P_2, P_3, P_4$  donde  $P_1$  y  $P_4$  son los extremos del segmento.
- $P_2$  y  $P_3$  son solo puntos de control.  $P_1$  y  $P_4$  son puntos de control e interpolación
- Los vectores tangentes en los puntos extremos se definen indirectamente en función los  $P_i$

$$\begin{aligned} R_1 &= Q'(0) = 3(P_2 - P_1) \\ R_4 &= Q'(1) = 3(P_4 - P_3) \end{aligned} \quad G_B = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} \text{ vector de la geometría}$$

## Curvas de Bezier (cont.)

$$G_H = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = M_{HB} G_B$$

$M_{HB}$ : matriz de traspaso de la geometría

Base de Hermite

$$M_B = M_H G_{HB} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$Q(t) = T M_B G_B = (1-t^3)P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t)P_3 + t^3P_4$$

Polinomios de Bernstein

## Curvas de Bezier (cont.)

### Propiedades

- Los polinomios de Bernstein son  $\geq 0$
- Su suma es uno para cada  $t$
- La curva de Bezier está contenida en el cierre convexo de los  $P_i$  (útil para reducir trabajo al hacer clipping)

## Splines

- Splines naturales (usados en métodos numéricos)
- B-Splines  $\Rightarrow$  Computación Gráfica

## Spline Cúbico natural

- Tiene continuidad  $C^0$ ,  $C^1$ ,  $C^2$ , más que Hermite y Bezier
- Los coeficientes de los polinomios en cada segmento dependen de todos los puntos de control.

Deventajas:

- cálculo  $\Rightarrow$  invertir matriz de  $(n + 1) \times (n + 1)$
- movimiento de un punto afecta toda la curva

## B-Spline cúbico

- Coeficientes de los segmentos de polinomios dependen de pocos puntos de control

Ventajas

- control local: movimiento de un punto afecta solo 4 segmentos cúbicos
- bajo tiempo de cálculo

## Notación especial B-Splines cúbicos

- $(m+1)$  puntos de control  $P_0, P_1, \dots, P_m$  con  $m \geq 3$
- $(m-1)$  nodos  $t_i$  (extremos de los parámetros de los segmentos de las curvas)  
 $t_3, t_4, \dots, t_{m+1}$  (UNIFORME:  $t_i = i-3$ )
- $(m-2)$  segmentos de curvas cúbica  $Q_i$  definida para  $t_i \leq t \leq t_{i+1}$

Ejemplo;  $m = 3$  (1 curva)

$Q_3$  definida para  $t_3 \leq t \leq t_4$ ,  $t_3 = 0$ ,  $t_4 = 1$  puntos de control:  $P_0, P_1, P_2, P_3$ ,

## Matricialmente B-Spline cúbico uniforme

- $Q_i(t) = T_i M_{BS} \cdot G_{BS} \quad t_i \leq t \leq t_{i+1}$

donde

$$T_i = [ (t-t_i)^3 \ (t-t_i)^2 \ (t-t_i) \ 1 ]$$

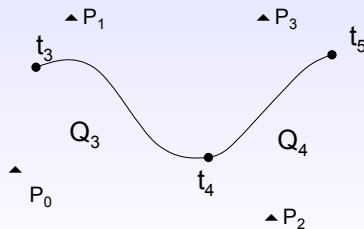
$$G_{BS} = \begin{bmatrix} P_i - 3 \\ P_i - 2 \\ P_i - 1 \\ P_i \end{bmatrix} \text{ vector de la geometría}$$

$$M_{BS} = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \text{ matriz de base}$$



## Propiedades

- cada segmento definido por 4 puntos
- cada punto de control influye sobre 4 curvas
- continuidad  $C^0$ ,  $C^1$ ,  $C^2$  entre  $Q_i$  y  $Q_{i-1}$
- para definir curva cerrada se repiten  $P_0$ ,  $P_1$ ,  $P_2$  al final de la curva
- se puede forzar a interpolar puntos de control repitiendo los puntos



## B-Spline general (no uniforme)

- Más general y más flexible
- Funciones de mezcla  $B_i(t)$

$$Q_i(t) = P_{i-3} B_{i-3}(t) + P_{i-2} B_{i-2}(t) + P_{i-1} B_{i-1}(t) + P_i B_i(t)$$

- $B_{i-3}$ ,  $B_{i-2}$ ,  $B_{i-1}$ ,  $B_i$  son polinomios de tercer grado y se construyen recursivamente en función de polinomios de grado 0, 1, 2.

## Superficies Paramétricas Bicúbicas

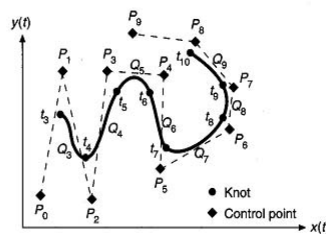
- parámetros  $s, t$
- Hermite, Bezier, B-Splines, NURBS
- Los pedazos de superficies (y las curvas) se pueden subdividir para mejorar la aproximación agregando NODOS
- NURBS: non-uniform rational B-Splines

TAREA: Usar las funciones de OpenGL relacionadas con curvas y superficies (OpenGL "avanzado")

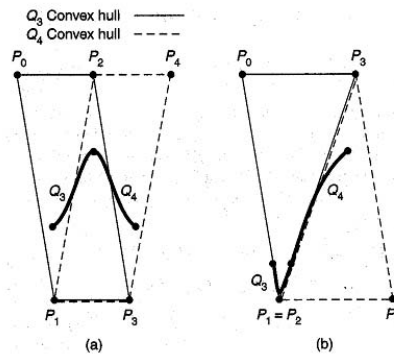
## Uniform, Nonrational B-Splines

$$Q_i(t - t_i) = G_{Bs_i} \cdot M_{Bs} \cdot T_i = G_{Bs_i} \cdot M_{Bs} \cdot T$$

$$= G_{Bs_i} \cdot B_{Bs} = P_{i-3} \cdot B_{Bs-3} + P_{i-2} \cdot B_{Bs-2} + P_{i-1} \cdot B_{Bs-1} + P_i \cdot B_{Bs0}$$



## Nonuniform, Nonrational B-Splines



Christer Åhlund



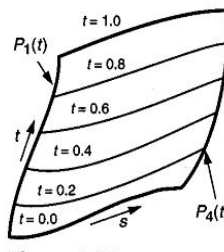
MCRivara/CG2006/2

37

## Parametric Bicubic Surfaces

Like parametric curves but in two dimensions

$$x(s, t) = G_{H_x}(t) \cdot M_H \cdot S = [P_1(t) \ P_4(t) \ R_1(t) \ R_4(t)] \cdot M_H \cdot S$$



Christer Åhlund



MCRivara/CG2006/2

38

## Resumen

- Las mallas de polígonos pueden ser poco adecuadas para objetos con caras curvas
- Curvas y superficies cúbicas paramétricas se usan ampliamente en CG y CAD
  - proveen control local
  - puede aproximar o interpolar los puntos de control
  - computacionalmente eficientes
  - refinamiento por subdivisión y adición de nodos
  - se transforman fácilmente, transformando los puntos de control
- Temas importantes: despliegue y rendering de superficies paramétricas

## Triangulaciones

- Caso especial de malla de polígonos
- Muy usada en CG y aplicaciones de ingeniería
- Modelación de superficies: 2D, 2½D, 3D; abiertas y cerradas
- Muy versátil para aproximar bien geometrías complejas con mallas no uniformes
- Cada triángulo (polígono) define exactamente un plano
- Más difícil de manejar que mallas de cuadriláteros (algoritmos, estructura de datos)
- Triangulaciones para CG: tema de gran desarrollo en los últimos años

## Problemas actuales en CG

- Tener modelos con distinta resolución de un objeto 3D: triangulaciones jerárquicas
- Dado un objeto 3D escaneado, obtener una representación que use el mínimo número de puntos necesario dados los requerimientos
- Representaciones comprimidos

OBS: En las aplicaciones se necesitan “buenas triangulaciones”

## Problemas algorítmicos con Triangulaciones (algunos)

- Triangularizar conjunto de puntos en 2D
- Triangularización óptima para algún criterio
- Triangularizar conjunto de puntos sobre una superficie en 3D (abierta o cerrada)
- Triangularización de volumen (puntos asociados a un objeto)
- Triangulación de un objeto dada su geometría
- Triangulación automática de “buena calidad” de un objeto

## Cont.

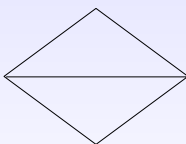
- Triangulaciones jerárquicas de un objeto
- Algoritmos de simplificación de triangulaciones
- Algoritmos de refinamiento de triangulaciones
- Algoritmos de mejoramiento de triangulaciones
- Triangulaciones no obtusas (o poco obtusas)

## Triangulación válida (acceptable o conforme)

$P = \{\text{vértices}\}$

$T = \{\text{triángulos}\}$

- dos triángulos se intersectan en vértice o arista común



## Algoritmos para triangularizar {puntos}

## Triangulación Delaunay

$T = \{\text{triángulos}\}$      $P = \{\text{vértices}\}$

- $T$  incluye todos los vértices de  $P$  y es triangulación válida
- Para todo triángulo  $t$   
 $C(t)$ : círculo circunscrito de  $t$  no contiene en su interior ningún vértice de  $P$

