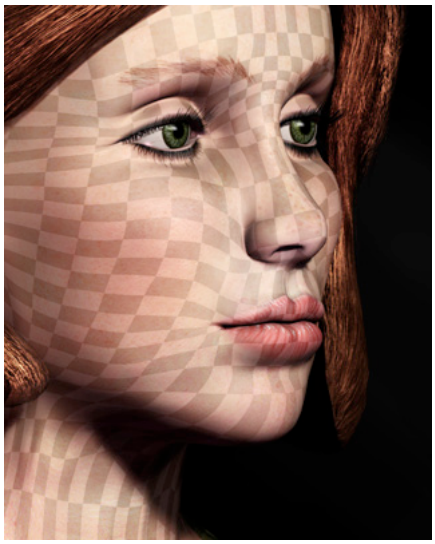
Curvas e Superfícies Paramétricas



Exemplo de superfícies NURBS

Curvas e Superfícies

- Para aplicações de CG normalmente é mais conveniente adotar a forma paramétrica
- Independente do sistema de coordenadas
- Representação Flexível (grande classes de objetos podem ser representados)

Curvas Paramétricas

- Curva genérica em 3D

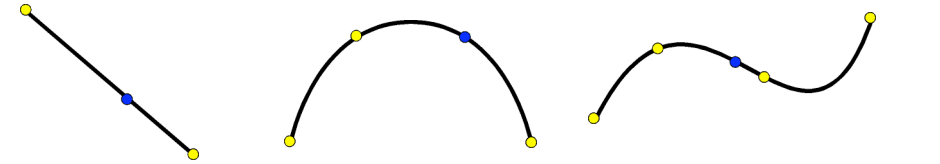
$$Q(t) = [x(t) \qquad y(t) \qquad z(t)]$$

- x(t), y(t) e z(t) são denominadas Funções-Base (*Base Functions*)

Funções-Base

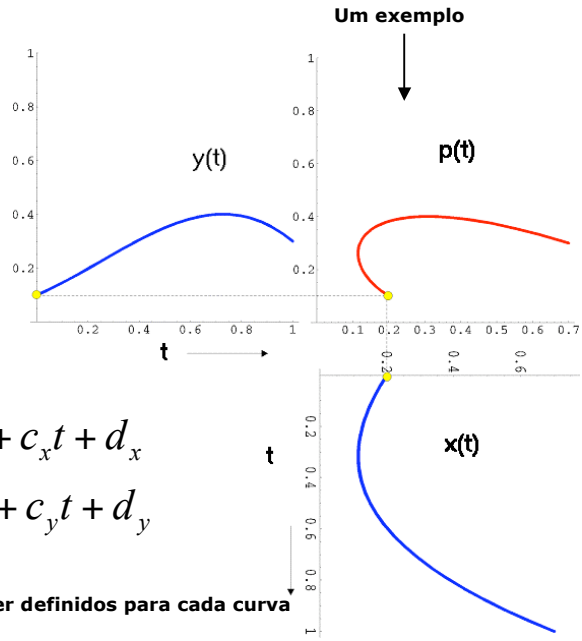
- Normalmente polinômios de grau 3
- Porque grau 3?

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



Linear $\qquad$ **Quadratic** $\qquad$ **Cubic**

Curvas Paramétricas Cúbicas 2D

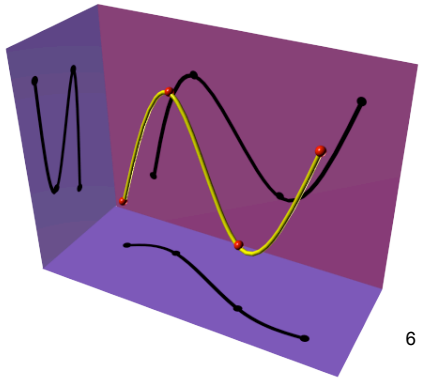


$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$
$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

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

Em 3D

$$\left.\begin{aligned} x(t) &= a_x t^3 + b_x t^2 + c_x t + d_x \\ y(t) &= a_y t^3 + b_y t^2 + c_y t + d_y \\ z(t) &= a_z t^3 + b_z t^2 + c_z t + d_z \end{aligned}\right\}$$



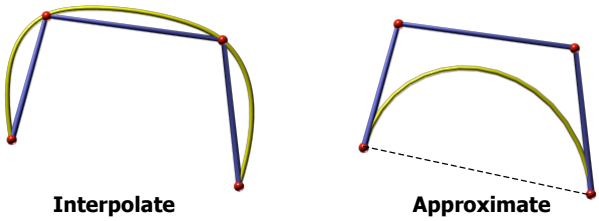
Em 3D 12 valores são necessários

Para o usuário, como especificar estas curvas?

- Sabemos que cada conjunto de 12 valores representa uma curva diferente
  - Diretamente através dos 12 valores (ax,bx,cx,...,cz,dz). Não há intuição sobre os valores e a curva correspondente
  - Indiretamente, procurando facilitar a tarefa para o usuário
- Diferença entre pontos que APROXIMAM uma curva e pontos que INTERPOLAM

Aproximação x Interpolação

- 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)



Representando curvas por aproximação

- Cada polinômio cúbico tem 4 coeficientes, logo 4 condições são necessárias para especificas este polinômio
- Variação em como os pontos são especificados
  - Hermite
  - Bézier
  - B-splines
  - NURBS and β-splines

Especificação matricial da curva

- Matriz de Geometria (**G**) e Matriz Base (**M**)

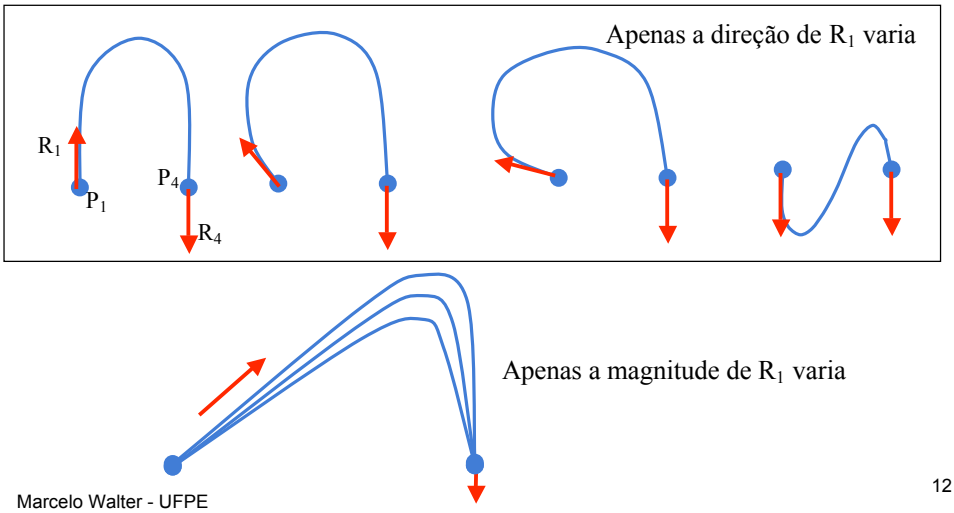
$$Q(t) = \begin{bmatrix} x(t) & y(t) & z(t) \end{bmatrix} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \begin{bmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \end{bmatrix}$$

$$Q(t) = TMG$$

Curvas 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

Exemplos de Curvas Hermite



Como converter os dados do usuário na curva Hermite?

- Precisamos encontrar M_H e G_H

$$Q_H(t) = TM_H G_H$$

Vetor Geometria Hermite

- 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:

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

Curvas Hermite

- A matriz base da Hermite M_H é calculada levando em consideração G_H
- Conforme visto anteriormente:

$$x(t) = T \cdot M_H \cdot G_{Hx}$$
$$y(t) = T \cdot M_H \cdot G_{Hy}$$
$$z(t) = T \cdot M_H \cdot G_{Hz}$$

Demonstraremos apenas para uma coordenada (x). Para as outras segue o mesmo desenvolvimento.

Curvas Hermite

- As restrições em x(0) e x(1) são encontradas por substituição direta:

$$x(0) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \cdot M_H \cdot G_{Hx} = P_{1x}$$
$$x(1) = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \cdot M_H \cdot G_{Hx} = P_{4x}$$

Esta é a matriz T com os valores substituídos

Curvas Hermite

- As restrições dos vetores tangentes em x(0) e x(1) são encontrados por derivadas:

$$x'(t) = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{Hx}$$

- Logo:

$$x'(0) = R_{1x} = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{Hx}$$

- e

$$x'(1) = R_{4x} = \begin{bmatrix} 3 & 2 & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{Hx}$$

Curvas Hermite

- As 4 restrições podem ser escritas na forma matricial:

$$\begin{bmatrix} P_{1x} \\ P_{4x} \\ R_{1x} \\ R_{4x} \end{bmatrix} = G_{Hx} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \cdot M_H \cdot G_{Hx}$$

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

Porque M_H é a inversa desta matriz?

Funções de Mistura Hermite

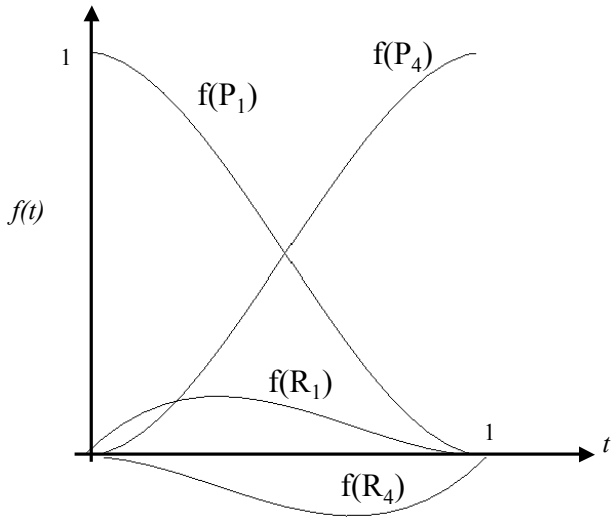
- Nós podemos ler a equação da seguinte forma:

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

- *"Q(t) é a soma ponderada dos elementos de G"*

$$\begin{aligned} Q(t) = T \cdot M_H . G_H = B_H \cdot G_H = & \left(2t^3 - 3t^2 + 1\right)P_1 + \\ & \left(-2t^3 + 3t^2\right)P_4 + \\ & (t^3 - 2t^2 + t)R_1 + \\ & (t^3 - t^2)R_4 \end{aligned}$$

Funções de Mistura Hermite



Curvas Bézier

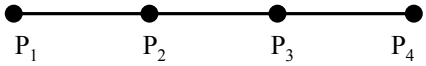
- Obteremos as curvas Bezier a partir de Hermite
- As curvas de Bézier são definidas por pontos P_1 e P_4 e também por dois pontos intermediários: P_2 e P_3
- Existe uma relação entre Hermite e Bézier: os vetores tangentes iniciais e finais são determinados pelos vetores P_1P_2 e P_3P_4 e são relacionados com R_1 e R_4 na Hermite da seguinte forma:

$$R_1 = p'(0) = 3(P_2 - P_1)$$

$$R_4 = p'(1) = 3(P_4 - P_3)$$

Curvas Bézier

- A razão para usar a constante 3 do slide anterior:
  - Considere a curva designada pelos pontos: (0,0), (0,1), (0,2), (0,3).

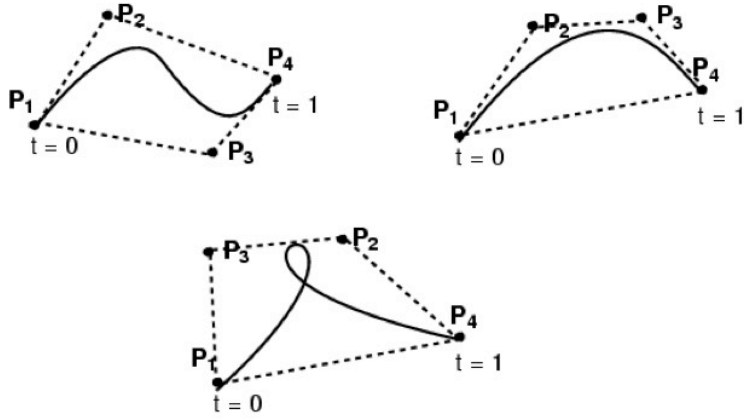


  - A definição desta curva é: $p(t) = P_1 + t(P_4 - P_1)$
  - Logo:
  - Se a velocidade for constante:

$$p'(t) = P_4 - P_1$$

$$R_1 = p'(0) = P_4 - P_1 = 3(P_2 - P_1) \qquad R_4 = p'(1) = P_4 - P_1 = 3(P_4 - P_3)$$

Exemplos de Curvas Bezier



Curvas Bézier

- O vetor G_B por definição: $G_B = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$

- A matriz M_{HB} define a relação geométrica entre Hermite e Bézier:

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

De onde saíram os valores desta matriz?

Matriz de Base Bézier

- Para encontrar M_B, considere:

$$\begin{aligned} Q_H(t) &= T \cdot M_H \cdot G_H \\ &= T \cdot M_H \cdot (M_{HB} \cdot G_B) \quad \textbf{Pelo slide anterior} \\ &= T \cdot (M_H \cdot M_{HB}) \cdot G_B \\ &= T \cdot M_B \cdot G_B \end{aligned}$$

- Logo:

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

Multiplicação de duas matrizes conhecidas

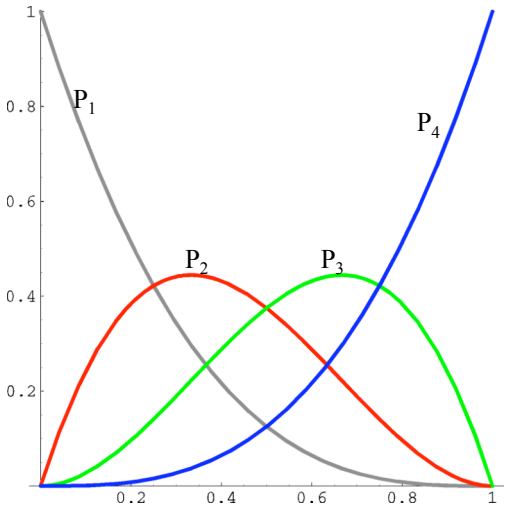
Curva Bézier

- Logo:

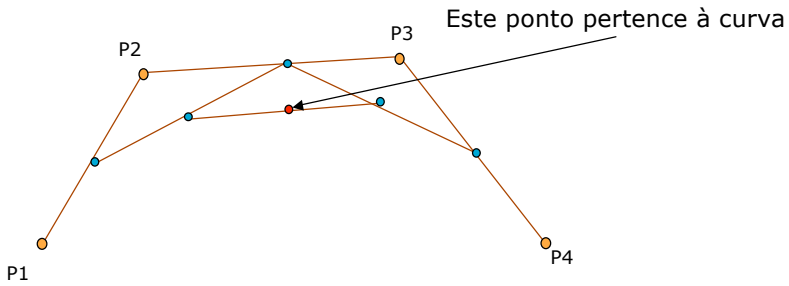
$$\begin{aligned} Q_B(t) = T \cdot M_B \cdot G_B = \; & (1-t)^3 P_1 + \\ & 3t(1-t)^2 P_2 + \\ & 3t^2(1-t) P_3 + \\ & t^3 P_4 \end{aligned}$$

- São conhecidos como *Bernstein Polynomials*

Bernstein Polynomials



Casteljau
Obter a curva Bezier iterativamente

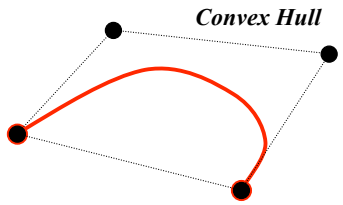


Os pontos da curva são os últimos pontos gerados no processo de subdivisão

Propriedade de *Convex Hull*

- A curva de Bézier está completamente dentro do maior polígono convexo, formado pelos pontos de controle

Você consegue imaginar uma situação onde esta propriedade seria útil?



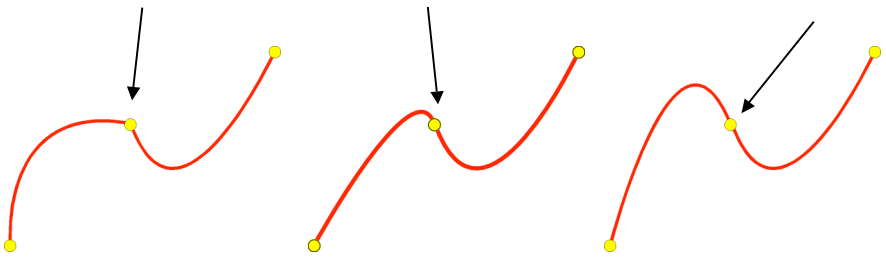
Montando uma curva mais complexa

- Geralmente as aplicações necessitam curvas mais complexas do que apenas dois pontos de inflexão
- Solução é unir segmentos de curvas

Continuidade

- Para assegurar a continuidade entre segmentos de curva, definem-se restrições de continuidade
- 2 tipos de continuidade (n = grau de continuidade):
  - *Continuidade paramétrica*, denotada por C^n
  - *Continuidade geométrica*, denotada por G^n

Exemplos de Continuidade



Continuidade Geométrica G0	Continuidade Geométrica G1	Continuidade Paramétrica C1
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

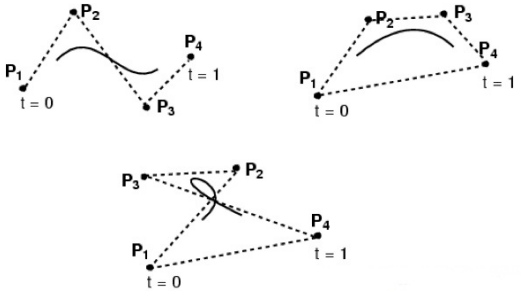
Manipulando curvas Bezier

- O que acontece ao editarmos qualquer ponto de uma curva Bezier?
- A curva toda se modifica, não há controle local
- Esta é uma desvantagem das curvas Bezier que levou a buscar outras possibilidades de representação

B-Splines

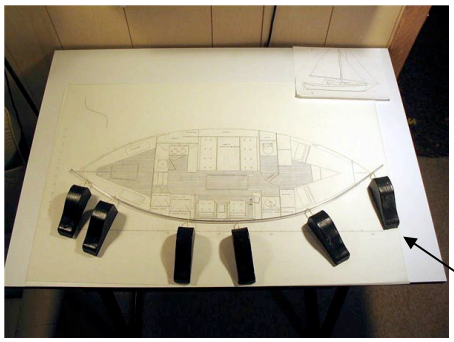
- Porque são melhores? Porque fornecem **suporte local** e mais controle (ao contrário da Bézier que por mínima mudança nos pontos de controle, gera uma nova curva)

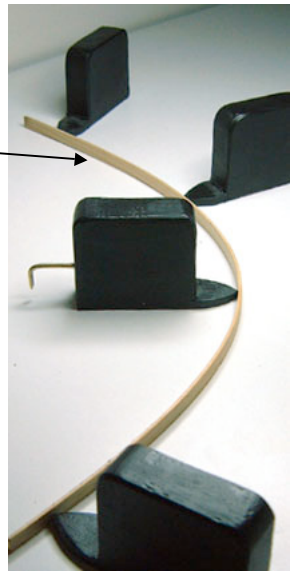
- Genericamente:
  - Para m+1 pontos de controle
    - M>=3 P0,P1,..,Pn
  - Teremos curvas com m-2 segmentos
    - Q3,Q4,..,Qm



De onde saiu o nome Splines :-)

- Construindo barcos





Estes blocos de madeira são chamados "ducks"

B-Splines

- Como definir as funções de mistura para as B-Splines???
  - Considerando as propriedades desejadas:convex hull, continuidade nas junções…
  - Com um pouco de matemática chega-se a:

B-Splines Uniformes

$$Q_{BS}(t) = \frac{(1-t)^3}{6} P_{i-3} + \frac{3t^3 - 6t^2 + 4}{6} P_{i-2} +$$

$$\frac{-3t^3 + 3t^2 + 3t + 1}{6} P_{i-1} + \frac{t^3}{6} P_i$$

$$0 <= t <= 1$$

Observem as diferenças na especificação dos pontos

B-Splines Uniformes

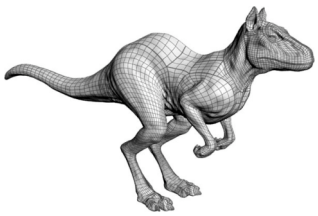
- Significa que a variável paramétrica está espaçada em intervalos uniformes (t=0.1, 0.2, 0.3, etc)
- Cada um dos *m-2* segmentos é definido por 4 dos *m+1* pontos de controle
- Segmento $Q_i = P_{i-3}, P_{i-2}, P_{i-1}, P_i$
- GS=[P_{i-3} P_{i-2} P_{i-1} P_i] 3<=i<=m

B-Splines Não-Uniformes

- O intervalo entre valores da variável paramétrica não é necessariamente uniforme…
- Logo as funções de *blending* não são as mesmas para cada segmento…. Maior flexibilidade

NURBS

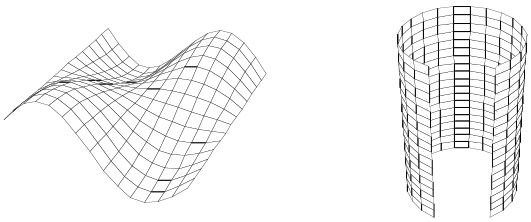
- ***Non-uniform rational B-splines***
- B-spline não-uniforme racional
- *Rational* significa que os segmentos de curva são expressos por razões entre polinômios cúbicos
- Ainda Maior flexibilidade
- Ex: x(t) = y(t)/w(t)



NURBS - 90 patches

Superfícies Paramétricas

- Idéia de *multiplicação* de 2 curvas
- A informação geométrica que define uma curva passa a ser ela própria uma função de uma variável paramétrica



Superfícies paramétricas

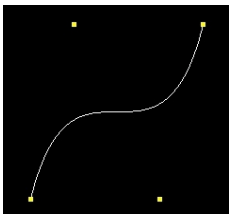
- A forma geral de uma superfície 3D na sua representação paramétrica é:

$$f(u,v) = (f_x(u,v), f_y(u,v), f_z(u,v))$$

Curvas em OpenGL

Definição dos pontos de controle

Habilitando o uso

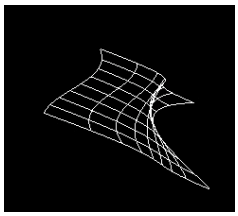


```
GLfloat ctrlpoints[4][3] = {
        { -4.0, -4.0, 0.0}, { -2.0, 4.0, 0.0},
        {2.0, -4.0, 0.0}, {4.0, 4.0, 0.0}};

void init(void)
{
   glClearColor(0.0, 0.0, 0.0, 0.0);
   glShadeModel(GL_FLAT);
   glMap1f(GL_MAP1_VERTEX_3, 0.0, 1.0, 3, 4, &ctrlpoints[0][0]);
   glEnable(GL_MAP1_VERTEX_3);
}

void display(void)
{
   int i;

   glClear(GL_COLOR_BUFFER_BIT);
   glColor3f(1.0, 1.0, 1.0);
   glBegin(GL_LINE_STRIP);
      for (i = 0; i <= 30; i++)
         glEvalCoord1f((GLfloat) i/30.0);
   glEnd();
   /* The following code displays the control points as dots. */
   glPointSize(5.0);
   glColor3f(1.0, 1.0, 0.0);
   glBegin(GL_POINTS);
      for (i = 0; i < 4; i++)
         glVertex3fv(&ctrlpoints[i][0]);
   glEnd();
   glFlush();

}
```

Desenho propriamente dito

Superfícies em OpenGL

Definição dos pontos de controle



A biblioteca GLU implementa também NURBS!

```
GLfloat ctrlpoints[4][4][3] = {
    {{-1.5, -1.5, 4.0}, {-0.5, -1.5, 2.0},
     {0.5, -1.5, -1.0}, {1.5, -1.5, 2.0}},
    {{-1.5, -0.5, 1.0}, {-0.5, -0.5, 3.0},
     {0.5, -0.5, 0.0}, {1.5, -0.5, -1.0}},
    {{-1.5, 0.5, 4.0}, {-0.5, 0.5, 0.0},
     {0.5, 0.5, 3.0}, {1.5, 0.5, 4.0}},
    {{-1.5, 1.5, -2.0}, {-0.5, 1.5, -2.0},
     {0.5, 1.5, 0.0}, {1.5, 1.5, -1.0}}
};

void display(void)
{
   int i, j;

   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glColor3f(1.0, 1.0, 1.0);
   glPushMatrix ();
   glRotatef(85.0, 1.0, 1.0, 1.0);
   for (j = 0; j <= 8; j++) {
      glBegin(GL_LINE_STRIP);
      for (i = 0; i <= 30; i++)
         glEvalCoord2f((GLfloat)i/30.0, (GLfloat)j/8.0);
      glEnd();
      glBegin(GL_LINE_STRIP);
      for (i = 0; i <= 30; i++)
         glEvalCoord2f((GLfloat)j/8.0, (GLfloat)i/30.0);
      glEnd();
   }
   glPopMatrix ();
   glFlush();
}
```

Desenho propriamente dito