

Modelagem Geométrica I

Descrever e processar problemas
com técnicas de Geometria Analítica

Motivação

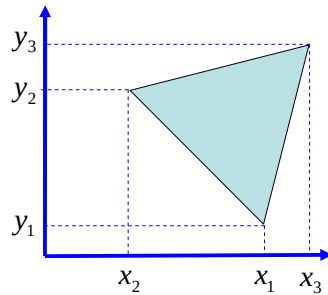
Tarefa 1

- Desenhe um triângulo equilátero de lado igual a 3cm.
- Compare o seu desenho com o dos seus colegas. São todos iguais?
- Explique o que você observou e o que falta na especificação do item (1) para garantir a **univocidade** do resultado?
- Proponha uma **representação não-ambígua**.

Sistema de Referência

Sequência de pontos em coordenadas cartesianas

$$(x_1, y_1), (x_3, y_3), (x_2, y_2)$$



Possíveis inferências geométricas:

Comprimento do lado: $\sqrt{\begin{pmatrix} x_3 - x_1 \\ y_3 - y_1 \end{pmatrix} \cdot \begin{pmatrix} x_3 - x_1 \\ y_3 - y_1 \end{pmatrix}}$

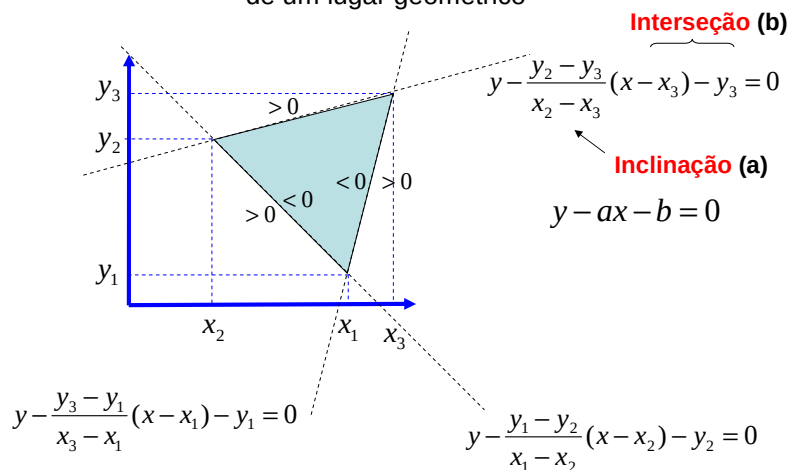
Orientação no plano: $Base = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \end{pmatrix}; Altura = \begin{pmatrix} y_2 - y_1 \\ -(x_2 - x_1) \end{pmatrix}$

Pertinência/Continência de um ponto:

$$y - \frac{y_3 - y_1}{x_3 - x_1}(x - x_1) - y_1 \leq 0 \quad y - \frac{y_1 - y_2}{x_1 - x_2}(x - x_2) - y_2 \leq 0 \quad y - \frac{y_2 - y_3}{x_2 - x_3}(x - x_3) - y_3 \leq 0$$

Representação Implícita

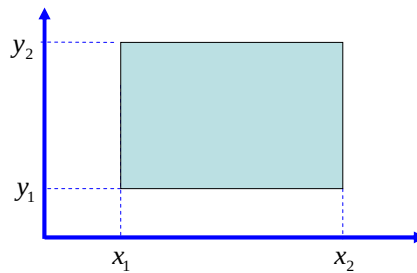
Função que estabelece a relação entre as coordenadas dos pontos de um lugar geométrico



Motivação

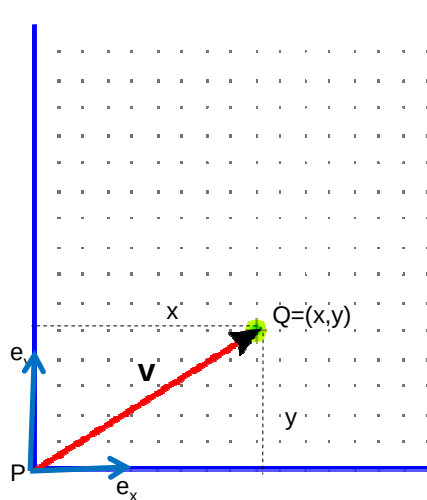
Tarefa 2

- Utilize a representação implícita na forma $y-ax-b=0$ para descrever os pontos pertencentes ao retângulo. Qual é a inclinação da reta suporte de cada lado?



- Há alternativas com uma abrangência descritiva maior?

Pontos e Vetores-posição



Ponto: $Q = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

$$P = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

“Identificação”

Vetor-posição: “diferença” entre 2 pontos

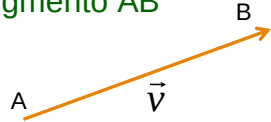
$$\vec{v} = Q - P = \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

“Grandeza geométrica”

Representação Paramétrica

Coordenadas dos pontos do lugar geométrico especificadas em termos de um conjunto de parâmetros

Segmento AB



$$\begin{aligned} P(\alpha) &= A + \alpha \vec{v} \\ &= A + \alpha(B - A) \\ &= \alpha B + (1 - \alpha)A \end{aligned}$$

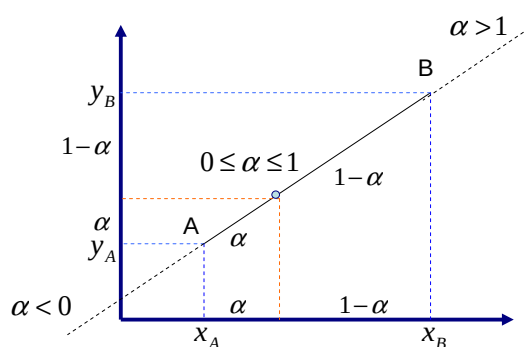
$$\vec{v} = \overrightarrow{AB} = B - A$$

pode ser representado, sem ambigüidade, pela sequência de pontos {A,B}

$$P(\alpha) = \begin{pmatrix} x(\alpha) \\ y(\alpha) \end{pmatrix} = \alpha \begin{pmatrix} x_B \\ y_B \end{pmatrix} + (1 - \alpha) \begin{pmatrix} x_A \\ y_A \end{pmatrix}$$

Parâmetro

Combinação Baricêntrica



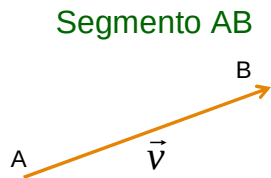
$$P(\alpha) = \alpha \begin{pmatrix} x_A \\ y_A \end{pmatrix} + (1 - \alpha) \begin{pmatrix} x_B \\ y_B \end{pmatrix}$$

$$P(\alpha_1, \alpha_2) = \alpha_1 \begin{pmatrix} x_A \\ y_A \end{pmatrix} + \alpha_2 \begin{pmatrix} x_B \\ y_B \end{pmatrix}$$

$$\alpha_1 + \alpha_2 = 1$$

Coordenadas baricêntricas (em relação aos pontos extremos A e B): uma outra alternativa para identificar um ponto.

Combinação Convexa



$$\vec{v} = \overrightarrow{AB} = B - A$$

pode ser representado, sem
ambigüidade, pela sequência
de pontos {A,B}

$$P(\alpha_1, \alpha_2) = \alpha_1 B + \alpha_2 A$$

Se $\alpha_1 + \alpha_2 = 1$ e

$$0 \leq \alpha_1, \alpha_2 \leq 1$$

então $P(\alpha_1, \alpha_2)$

são pontos do segmento $\overline{AB}$

Motivação

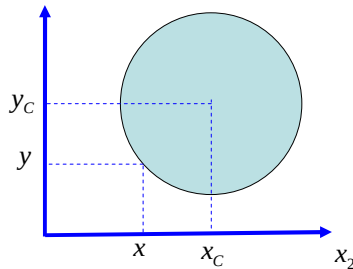
Tarefa 3

- Utilize uma sequência de pontos para descrever o lugar geométrico dos pontos de uma circunferência de raio igual a 2 cm.
- Quantos pontos são necessários para descrever com **exatidão** a circunferência?
- Há alguma alternativa mais **concisa** para representar a circunferência?
- Como “desenhar” esta circunferência?

Representação Implícita

$$(x - x_c)^2 + (y - y_c)^2 = R^2 = 4$$

Possíveis inferências geométricas:



Perímetro: $2\pi R$

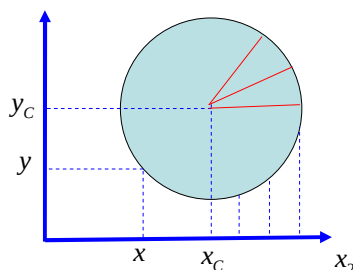
Área: πR^2

Pertinência/Continência de um ponto:

$$(x - x_c)^2 + (y - y_c)^2 \leq 4$$

Representação Paramétrica

$$(x - x_c)^2 + (y - y_c)^2 = R^2 = 4$$



Pontos com coordenadas x ou y igualmente espaçados não implicam em distribuição uniforme (mesma variação angular) de amostras sobre a circunferência!



Parametrizar as coordenadas em termos do ângulo θ de cada arco

$$x = R \cos \theta$$

$$y = R \sin \theta$$

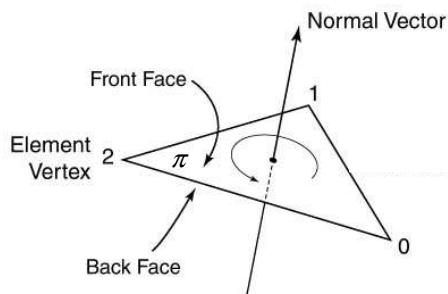
Coordenadas Polares

Resumo

- Distintas formas para representar pontos no espaço
 - Coordenadas cartesianas
 - Coordenadas polares
 - Coordenadas baricêntricas
- Distintas formas para especificar os pontos de um lugar geométrico
 - Sequência orientada de vértices
 - Funções
 - Paramétricas
 - Implícitas
- Distintos algoritmos de inferência geométrica
 - Propriedades de interesse para Computação Gráfica: **vetores-posição e vetores normais.**

Representações de um triângulo

Alternativas de Representação:



1. Sequência orientada:

$$P_0, P_1, P_2$$

$$\vec{n} = \frac{(P_0 - P_2) \times (P_1 - P_0)}{|P_0 - P_2| |P_1 - P_0|}$$

2. Implícita

$$f = \vec{n} \cdot (P - P_2) = 0 \text{ e domínio de } x, y, z$$

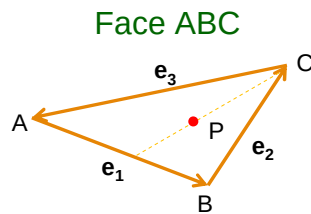
$$\vec{n} = \nabla f$$

3. Paramétrica

$$P(\alpha, \beta) = \alpha P_0 + \beta P_1 + (1 - \alpha - \beta) P_2$$

$$\vec{n} = \frac{\frac{\partial P}{\partial \alpha} \times \frac{\partial P}{\partial \beta}}{|\frac{\partial P}{\partial \alpha}| |\frac{\partial P}{\partial \beta}|}$$

Combinação Baricêntrica



$$\begin{aligned} \mathbf{e}_1 &= \mathbf{B} - \mathbf{A} \\ \mathbf{e}_2 &= \mathbf{C} - \mathbf{B} \\ \mathbf{e}_3 &= \mathbf{A} - \mathbf{C} \end{aligned}$$

Pode ser codificado,
sem ambigüidade,
pela seqüência de
pontos $\{A, B, C\}$

Pontos interiores do triângulo

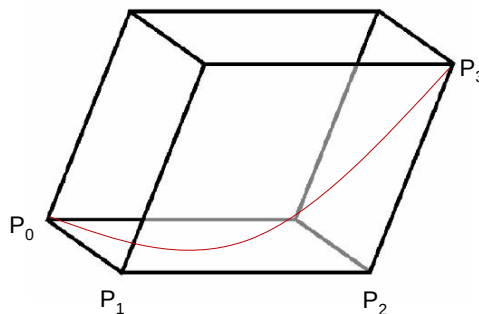
$$\begin{aligned} P(\alpha_1, \lambda) &= \lambda(\alpha_1 \mathbf{B} + (1 - \alpha_1) \mathbf{A}) + (1 - \lambda) \mathbf{C} \\ &= \lambda \alpha_1 \mathbf{B} + \lambda(1 - \alpha_1) \mathbf{A} + (1 - \lambda) \mathbf{C} \end{aligned}$$

$$\begin{aligned} P(\alpha, \beta, \gamma) &= \alpha \mathbf{B} + \beta \mathbf{A} + \gamma \mathbf{C} \\ \alpha + \beta + \gamma &= 1 \\ 0 \leq \alpha, \beta, \gamma &\leq 1 \end{aligned}$$

↑
Coordenadas baricêntricas

Curvas de Bézier

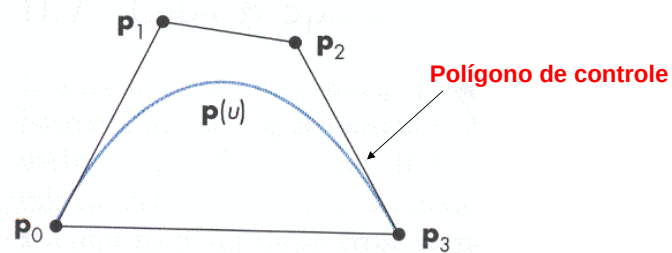
Conceito Original: curvas
fixas num paralelepípedo
que controla a sua forma



Curva de Bézier

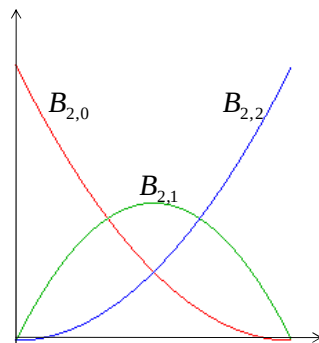
$$P(t) = \sum_{i=0}^n P_i J_{n,i}(t) \quad J_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i} \quad t \in [0,1]$$

$$\sum J_{n,i}(t) = 1, J_{n,i}(t) \geq 0 \Rightarrow \text{Combinação convexa de } P_i$$



Curvas de Bézier Quadráticas

$$P(t) = \sum_{i=0}^n P_i J_{2,i}(t) \quad J_{2,i}(t) = \binom{2}{i} t^i (1-t)^{2-i}$$

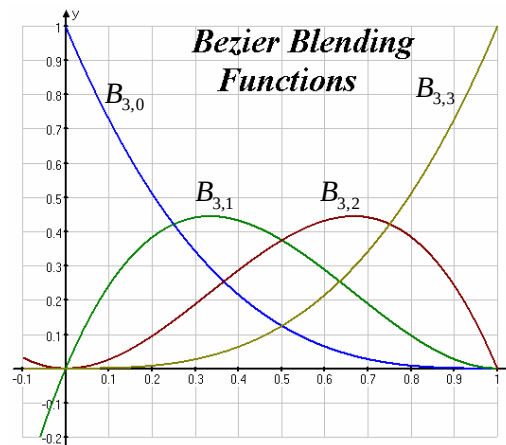


$$\begin{aligned} B_{2,0}(t) &= (1-t)^2 \\ B_{2,1}(t) &= 2t(1-t) \\ B_{2,2}(t) &= t^2 \end{aligned}$$

Curvas de Bézier Cúbicas

$$P(t) = \sum_{i=0}^n P_i J_{3,i}(t) \quad J_{3,i}(t) = \binom{3}{i} t^i (1-t)^{3-i}$$

$$\begin{aligned} B_{3,0}(t) &= (1-t)^3 \\ B_{3,1}(t) &= 3t(1-t)^2 \\ B_{3,2}(t) &= 3t^2(1-t) \\ B_{3,3}(t) &= t^3 \end{aligned}$$



Outras Funções de Bernstein

<http://i33www.ira.uka.de/applets/mocca/html/noplugin/curves.html>

Propriedades

- É polinomial. Se a quantidade de pontos de controle é $(n+1)$, o grau do polinômio é n .
- Acompanha a forma do polígono de controle. Está no fecho convexo dos pontos de controle.
- Influência global de cada ponto de controle.
- Pontos extremos da curva e do polígono coincidem.
- Tangente nos pontos extremos coincidem com os segmentos extremos do polígono de controle.
- A curva não oscila mais que o polígono de controle.
- São invariantes sob transformações afins.

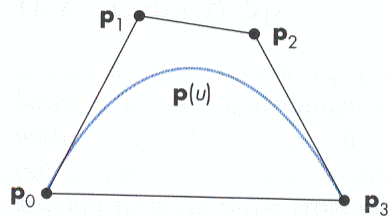
Exemplos

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

Polígono de Controle e Fecho Convexo

<http://i33www.ira.uka.de/applets/mocca/html/noplugin/RenderBezierC/AppRenderBezierC/index.html>

Notação Matricial



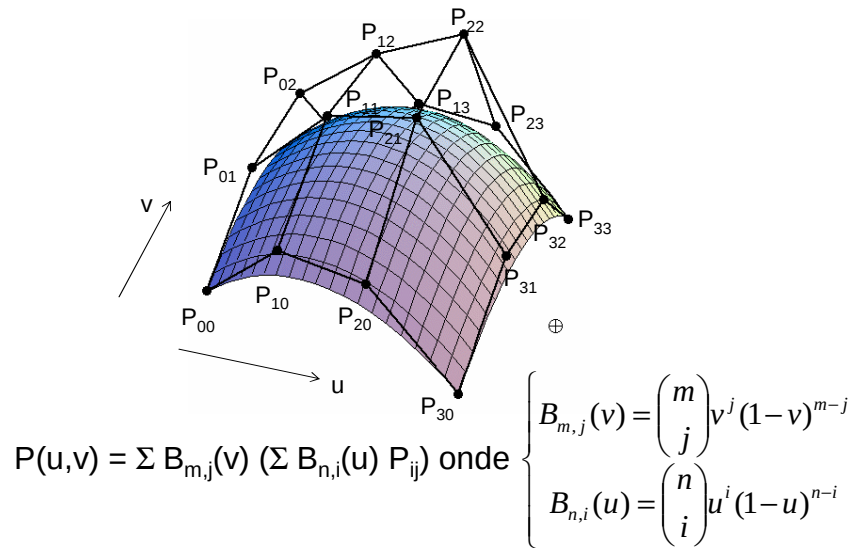
$$P(t) = B_{3,0}(t) P_0 + B_{3,1}(t) P_1 + B_{3,2}(t) P_2 + B_{3,3}(t) P_3$$

$$= (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$

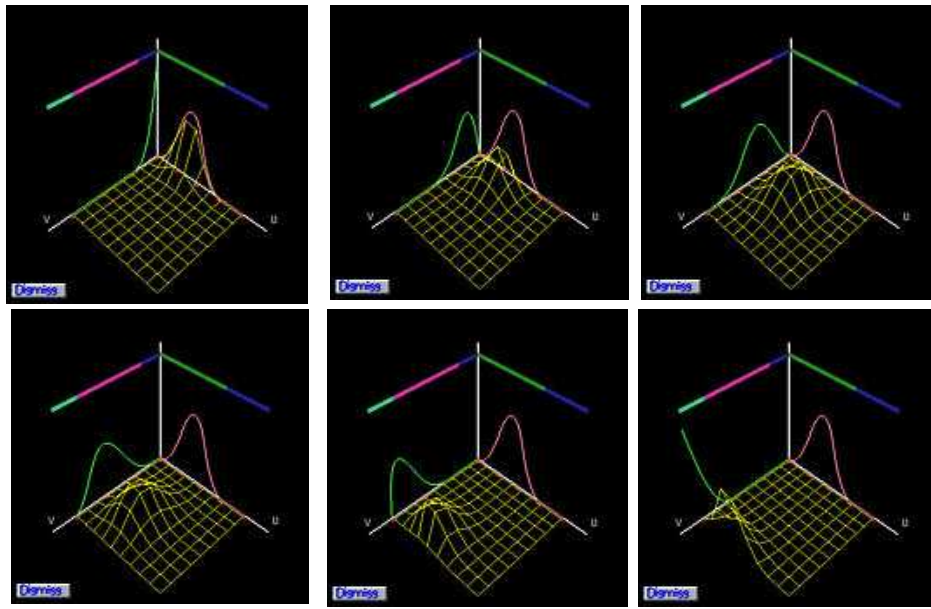
$$= (1-3t+3t^2-t^3)P_0 + (3t-6t^2+3t^3) P_1 + (3t^2-3t^3) P_2 + t^3 P_3$$

$$= \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ y_0 & y_1 & y_2 & y_3 \\ z_0 & z_1 & z_2 & z_3 \end{pmatrix} \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

Superfícies de Bézier



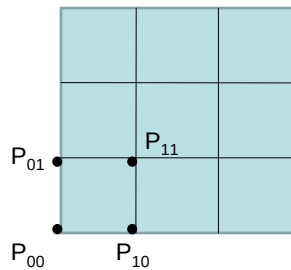
Funções de Bernstein Bi-dimensionais



Exemplos de Superfícies de Bézier Biquadráticas

<http://i33www.ira.uka.de/applets/mocca/html/noplugin/TPBezierSurface/AppTPBezierSurface/index.html>

Superfícies de Bézier Bicúbicas



Pontos de controle na borda:
p.ex.: P_{00} , P_{01} , P_{10}

Vetores-tangente na borda
p.ex.: $3(P_{10} - P_{00})$, $3(P_{01} - P_{00})$

Vetores-torsão (derivada mista) na borda
p.ex.: $9(P_{00} - P_{01} - P_{10} - P_{11})$

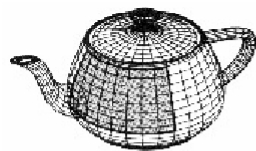
Se $9(P_{00} - P_{01} - P_{10} - P_{11}) = 0$
 $P_{00} - P_{01} - P_{10} = P_{11}$

Exemplos de Superfícies de Bézier Bicúbicas

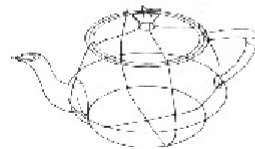
Uma implementação do algoritmo descrito em Watt

<http://www.nbb.cornell.edu/neurobio/land/OldStudentProjects/cs490-96to97/anson/BezierPatchApplet/>

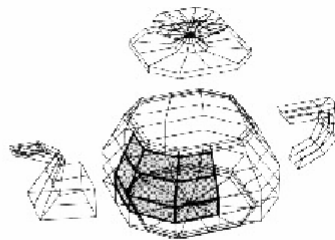
Superfícies de Bézier Bule de Utah



single shaded patch



Patch edges



wireframe of the control points

<http://www.holmes3d.net/graphics/teapot/>

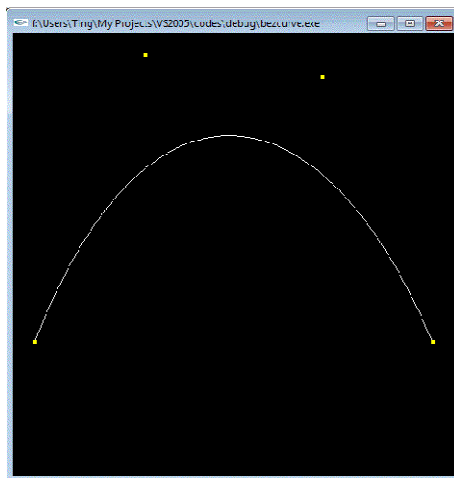
Interface do OpenGL – Curvas de Bézier

```
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);
    glMapGrid1f(20,0.0,1.0);
}

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();
}
```

Exemplo



```
GLfloat ctrlpoints[4][3] = {
    { -4.5, -4.0, 0.0},
    { -2.0, 4.5, 0.0},
    { 2.0, 4.0, 0.0},
    { 4.5, -2.0, 0.0}};
```

Código-fonte: <http://www.opengl.org/resources/code/samples/redbook/bezcurve.c>

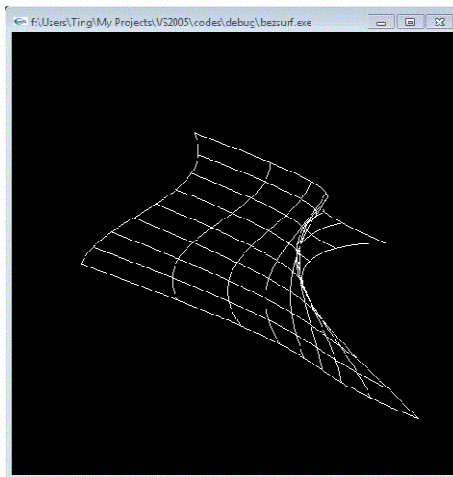
Interface do OpenGL – Superfícies de Bézier

```
void init(void)
{
    glClearColor (0.0, 0.0, 0.0, 0.0);
    glMap2f(GL_MAP2_VERTEX_3, 0, 1,
            3, 4, 0, 1, 12, 4,
            &ctrlpoints[0][0][0]);
    glEnable(GL_MAP2_VERTEX_3);
    glMapGrid2f(20, 0.0, 1.0, 20, 0.0, 1.0);
    glEnable(GL_DEPTH_TEST);
    glShadeModel(GL_FLAT);
}

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++)
            glVertex2f(((GLfloat)i/30.0,
            (GLfloat)j/8.0);
        glEnd();
        glBegin(GL_LINE_STRIP);
        for (i = 0; i <= 30; i++)
            glVertex2f(((GLfloat)i/8.0,
            (GLfloat)j/30.0);
        glEnd();
    }
    glPopMatrix ();
    glFlush();
}
```

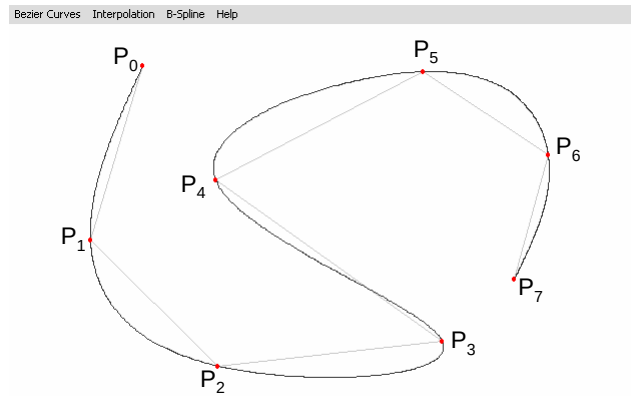
Exemplo



```
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}}
};
```

Código-fonte: <http://www.opengl.org/resources/code/samples/redbook/bezsurf.c>

Curvas de Bézier “Complexas”

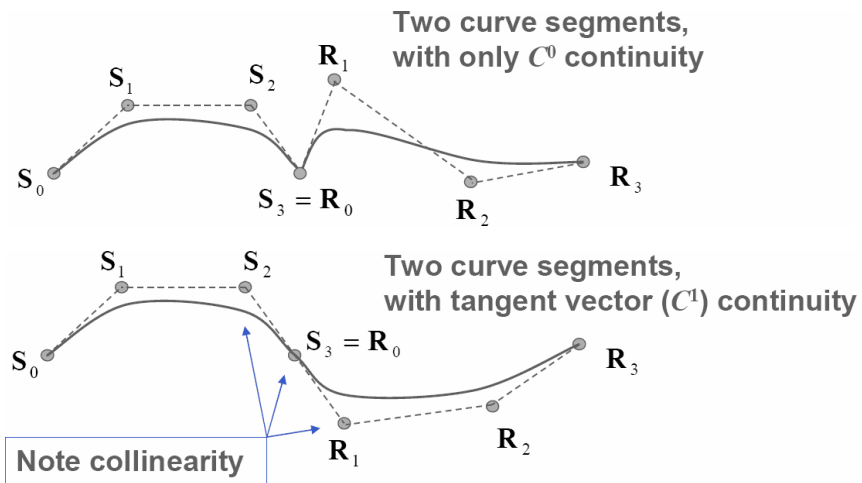


Propriedades

- É polinomial. Se a quantidade de pontos de controle é $(n+1)$, o grau do polinômio é n .
- Acompanha a forma do polígono de controle. Está no fecho convexo dos pontos de controle.
- Influência global de cada ponto de controle.
- Pontos extremos da curva e do polígono coincidem.
- Tangente nos pontos extremos coincidem com os segmentos extremos do polígono de controle.
- A curva não oscila mais que o polígono de controle.
- São invariantes sob transformações afins.

Splines de Curvas de Bézier

Concatenação de curvas S e R



Continuidade

Diferenciabilidade:

C^0 : $S_3 = R_0$

C^1 : primeiras derivadas são iguais

C^2 : primeiras e segundas derivadas são iguais

C^n : as primeiras n derivadas são iguais

Continuidade Geométrica:

G^0 : $S_3(1) = R_0(0)$

G^1 : $\partial S(1)/\partial t_1 = k_1 \partial R(0)/\partial t_2$

G^2 : $\partial S(1)/\partial t_1 = k_1 \partial R(0)/\partial t_2$; $\partial^2 S(1)/\partial t_1^2 = k_2 \partial^2 R(0)/\partial t_2^2$

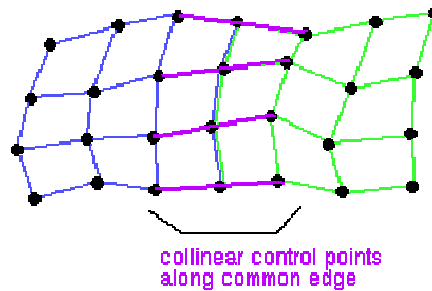
G^n : as primeiras n derivadas são iguais

Continuidade

Continuidade Geométrica:

$$G^0: S(1, v_1) = R(0, v_2)$$

$$G^1: \partial S(1, v_1) / \partial v_1 = k_1 \partial R(0, v_2) / \partial v_2$$



Splines de Base (B-Splines)

➤ pontos de controle: m

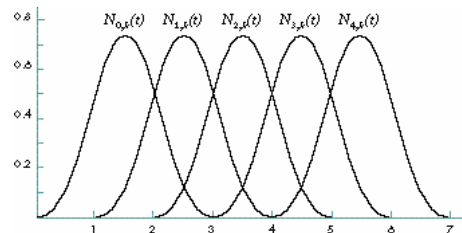
➤ ordem: k (grau: $n=k-1$)

➤ vetor de nós: $\{t_0, t_1, t_2, \dots, t_{m+1+n}\}$

➤ $(m-n+1)$ "segmentos de curvas"

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

$$N_{i,0}(t) = \begin{cases} 1, & \text{se } t \in [t_i, t_{i+1}) \\ 0, & \text{em outros intervalos} \end{cases}$$



$$N_{i,k}(t) = \frac{(t - t_i) N_{i,k-1}(t)}{(t_{i+k-1} - t_i)} + \frac{(t_{i+k} - t) N_{i+1,k-1}(t)}{(t_{i+k} - t_{i+1})}$$

$n=2$
3 segmentos

Funções de Base Constantes e Lineares

➤vetor de nós: $\{0,1,2,3,4,5,6\}$

$$N_{0,0}(t) = 1, t \in [0,1)$$

$$N_{0,1}(t) = t, t \in [0,1)$$

$$N_{1,0}(t) = 1, t \in [1,2)$$

$$N_{0,1}(t) = (1-t), N_{1,1}(t) = (t-1), t \in [1,2)$$

$$N_{2,0}(t) = 1, t \in [2,3)$$

$$N_{1,1}(t) = (3-t), N_{2,1}(t) = (t-2), t \in [2,3)$$

$$N_{3,0}(t) = 1, t \in [3,4)$$

$$N_{2,1}(t) = (4-t), N_{3,1}(t) = (t-3), t \in [3,4)$$

$$N_{4,0}(t) = 1, t \in [4,5)$$

$$N_{3,1}(t) = (5-t), N_{4,1}(t) = (t-4), t \in [4,5)$$

$$N_{5,0}(t) = 1, t \in [5,6)$$

$$N_{4,1}(t) = (6-t), t \in [5,6)$$

Funções de Base Quadráticas

➤vetor de nós: $\{0,1,2,3,4,5,6\}$

$$N_{0,2}(t) = t^2, t \in [0,1)$$

$$N_{0,2}(t) = \frac{t}{2}(1-t) + \frac{3-t}{2}(t-1), N_{1,2}(t) = \frac{(t-1)^2}{2}, t \in [1,2)$$

$$N_{0,2}(t) = \frac{(3-t)^2}{2}, N_{1,2}(t) = \frac{(t-1)(3-t)}{2} + \frac{(4-t)(t-2)}{2}, N_{2,2}(t) = \frac{(t-2)^2}{2}, t \in [2,3)$$

$$N_{1,2}(t) = \frac{(4-t)^2}{2}, N_{2,2}(t) = \frac{(t-2)(4-t)}{2} + \frac{(5-t)(t-3)}{2}, N_{3,2}(t) = \frac{(t-3)^2}{2}, t \in [3,4)$$

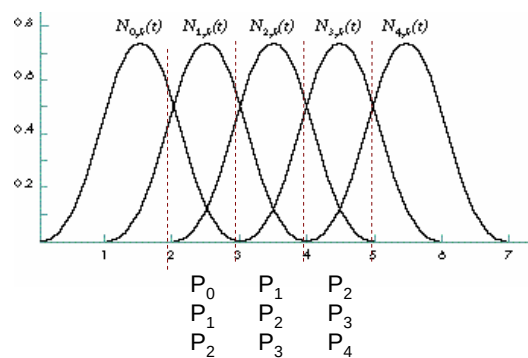
$$N_{2,2}(t) = \frac{(5-t)^2}{2}, N_{3,2}(t) = \frac{(t-3)(5-t)}{2} + \frac{(6-t)(t-4)}{2}, t \in [4,5)$$

$$N_{3,2}(t) = \frac{(6-t)^2}{2}, t \in [5,6)$$

Exemplos de Funções de Base

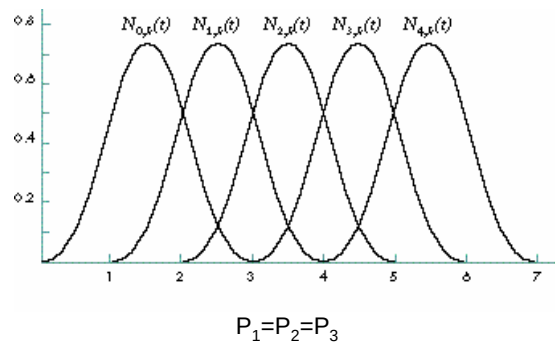
<http://i33www.ira.uka.de/applets/mocca/html/noplugin/BSplineBasis/AppBSplineBasis/index.html>

Influência Local de Pontos de Controle



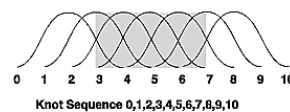
Multiplicidade de Pontos

Interpolação da Curva: multiplicidade = $n+1$



Influência de Vetor de Nós

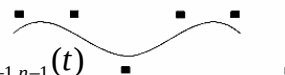
Spline de base uniforme



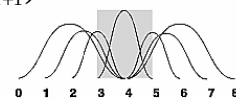
Basis Functions

$$N_{i,n}(t) = \frac{(t - t_i)N_{i,n-1}(t)}{(t_{i+n} - t_i)} + \frac{(t_{i+n+1} - t)N_{i+1,n-1}(t)}{(t_{i+n+1} - t_{i+1})}$$

$0/0=0$



Example of a Curve



Basis Functions

Spline de base não-uniforme



Example of a Curve

Propriedades de *Splines* de Base

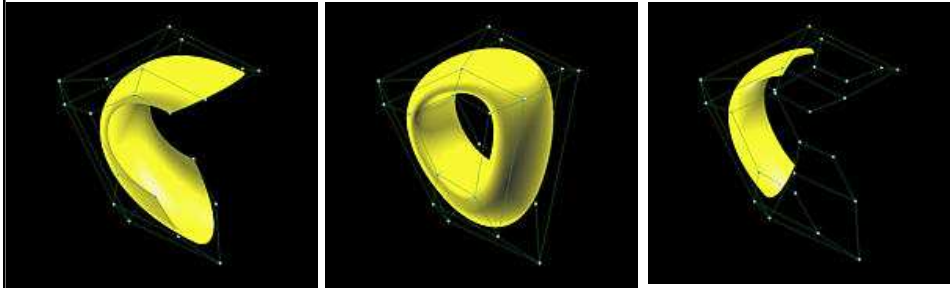
- É polinomial. O grau do polinômio não depende da quantidade de pontos de controle.
 - Acompanha a forma do polígono de controle. Está no **fecho convexo** dos pontos de controle de cada segmento de curva.
 - **Influência local** de cada ponto de controle.
 - A curva não oscila mais que o polígono de controle.
 - São **invariantes sob transformações afins**.
 - Multiplicidade m de um ponto “puxa” a curva para ele. Para $m=k$, curva interpola o ponto.
- Diferenciabilidade C^{n-1} .
- Multiplicidade m de nós altera a diferenciabilidade C^{k-m} .

Exemplos

<http://i33www.ira.uka.de/applets/mocca/html/noplugin/IntBSpline/AppSubdivision/index.html>

Representação de Superfícies

Superfícies de Base



Interface do OpenGL – Curvas B-Splines

```

void init(void)
{
    glClearColor (0.0, 0.0, 0.0, 0.0);

    theNurb = gluNewNurbsRenderer();

    gluNurbsProperty(theNurb, GLU_SAMPLING
        _METHOD, GLU_PATH_LENGTH);
    gluNurbsProperty(theNurb,
        GLU_PARAMETRIC_TOLERANCE, 0.1);
    gluNurbsCallback(theNurb, GLU_ERROR,
        nurbsError);
}

void CALLBACK nurbsError(GLenum errorCode)
{
    const GLubyte *estring;

    estring = gluErrorString(errorCode);
    fprintf (stderr, "Nurbs Error: %s\n", estring);
    exit (0);
}

void display(void)
{
    int i, j;

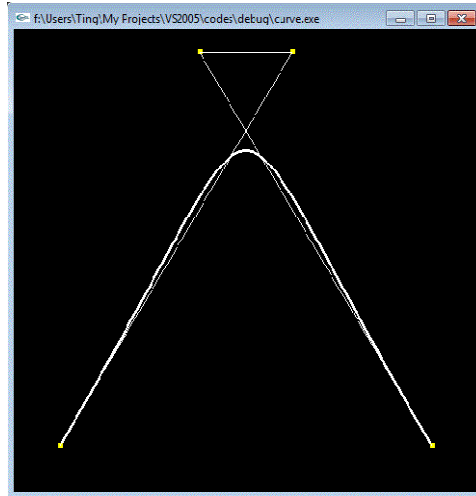
    glClear(GL_COLOR_BUFFER_BIT
        GL_DEPTH_BUFFER_BIT);

    glLineWidth(3.0);
    glColor3f(1.0, 1.0, 1.0);
    gluBeginCurve(theNurb);
    gluNurbsCurve(theNurb,
        8, knots,
        3, &ctlpoints[0][0],
        4, GL_MAP1_VERTEX_3);
    gluEndCurve(theNurb);

    if (showPoints) {
        draw_control_polygon();
    }

    glFlush();
}
    
```

Exemplo



```
GLfloat ctpoints[4][3] = {
    { -4.0, -4.0, 0.0 }, { 1.0, 4.5, 0.0 },
    { -1.0, 4.5, 0.0 }, { 4.0, -4.0, 0.0 } };
```

```
GLfloat knots[8] = { 0.0, 0.0, 0.0, 0.0, 1.0,
    1.0, 1.0, 1.0 };
```

Código-fonte: <http://www.dca.fee.unicamp.br/courses/IA725/1s2006/program/samples.html>

Interface do OpenGL – Superfícies B-Splines

```
GLUnurbsObj *theNurb;
void init(void)
{
    GLfloat mat_diffuse[] = { 0.7, 0.7, 0.7, 1.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
    GLfloat mat_shininess[] = { 100.0 };

    glClearColor (0.0, 0.0, 0.0, 0.0);
    glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
    glMaterialfv(GL_FRONT, GL_SPECULAR,
        mat_specular);
    glMaterialfv(GL_FRONT, GL_SHININESS,
        mat_shininess);
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_AUTO_NORMAL);
    glEnable(GL_NORMALIZE);

    init_surface();

    theNurb = gluNewNurbsRenderer();
    gluNurbsProperty(theNurb, GLU_SAMPLING_TOLERANCE,
        25.0);
    gluNurbsProperty(theNurb, GLU_DISPLAY_MODE,
        GLU_OUTLINE_POLYGON);
    gluNurbsCallback(theNurb, GLU_ERROR,
        nurbsError);
}

void display(void)
{
    int i, j;

    glClear(GL_COLOR_BUFFER_BIT |
        GL_DEPTH_BUFFER_BIT);
    glPushMatrix();
    glRotatef(330.0, 1.0, 0.0, 0.0);
    glScalef (0.5, 0.5, 0.5);

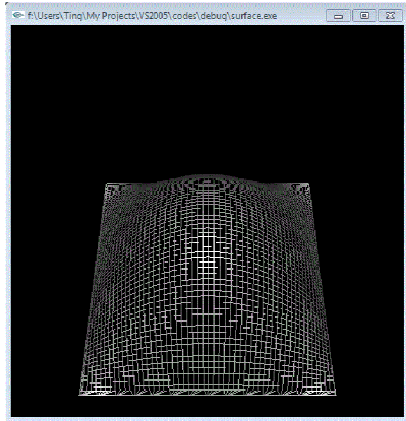
    gluBeginSurface(theNurb);
    gluNurbsSurface(theNurb,
        8, knots, 8, knots,
        4 * 3, 3, &ctpoints[0][0][0],
        4, 4, GL_MAP2_VERTEX_3);
    gluEndSurface(theNurb);

    glPopMatrix();
    glFlush();
}

void CALLBACK nurbsError(GLenum errorCode)
{
    const GLubyte *estring;

    estrng = gluErrorString(errorCode);
    fprintf (stderr, "Nurbs Error: %s\n", estrng);
    exit (0);
}
```

Exemplo



```
GLfloat knots[8] = {0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0};  
/*  
 * Initializes the control points of the surface to a  
 * small hill.  
 * The control points range from -3 to +3 in x, y, and  
 * z  
 */  
void init_surface(void)  
{  
    int u, v;  
    for (u = 0; u < 4; u++) {  
        for (v = 0; v < 4; v++) {  
            ctlpoints[u][v][0] = 2.0*((GLfloat)u - 1.5);  
            ctlpoints[u][v][1] = 2.0*((GLfloat)v - 1.5);  
  
            if ( (u == 1 || u == 2) && (v == 1 || v == 2))  
                ctlpoints[u][v][2] = 3.0;  
            else  
                ctlpoints[u][v][2] = -3.0;  
        }  
    }  
}
```

Código-fonte: <http://www.opengl.org/resources/code/samples/redbook/surface.c>