INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL-RIO-GRANDENSE
Campus Pelotas

Computação Gráfica
(atualizada em: 13 jun. 2022)
glaucius@pelotas.ifsul.edu.br

OpenGL – Desenho de circunferências e círculos

Jack Elton Bresenham

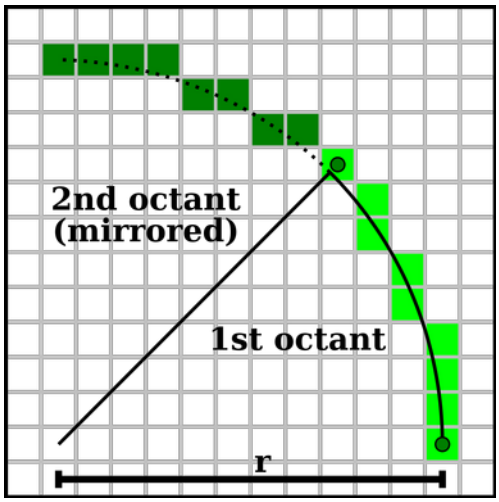
Nascido em 1937, em Clovis, Novo México, Jack Bresenham (1937*) (figura 1) é um professor de Ciência da Computação. Aposentou-se com 27 anos de serviço na IBM como um membro de equipe técnica sênior em 1987. Ensinou por 16 anos em Winthrop University e possui nove patentes. O Algoritmo de Bresenham para o desenho de circunferências, idealizado em 1962 é a sua inovação mais conhecida. Ele determina como será rasterizada uma reta entre dois determinados pontos em um plano, e é comumente utilizado para desenhar linhas na tela do computador. Foi um dos primeiros algoritmos desenvolvidos para a área da computação gráfica. (IT HISTORY SOCIETY, 2021).

Figura 1 – Jack Elton Bresenham

Fonte: Academia (2014).

Algoritmo de Bresenham para o Desenho de Circunferências

Figura 2 – Esquema do algoritmo para desenho de circunferências.



Fonte: BAHAR, Al Hizbul. Bresenham Circle Algorithm Using OpenGL and C++. 2021. Disponível em: https://engineersview.wordpress.com/2013/05/21/bresenham-circle-algorithm-using-opengl-and-c/ . Acesso em: 09 abr. 2021.

```
//----------------------------------------
vazio pontosCirc(inteiro x, y, x1, y1)
início
  desenha_pontos
    vértice(x1+x, y1+y)
    vértice(x1+y, y1+x)
    vértice(x1+y, y1-x)
    vértice(x1+x, y1-y)
    vértice(x1-x, y1-y)
    vértice(x1-y, y1-x)
    vértice(x1-y, y1+x)
    vértice(x1-x, y1+y)
  fim desenha_pontos
fim função


//----------------------------------------
vazio desenhaCirc(inteiro raio, x1, y1)
início
  inteiro x, y
  decimal d

  x = 0
  y = raio
  d = 5./4 - raio
  pontosCirc(x, y, x1, y1)
  enquanto y > x
  início
    se d < 0 então
```

```
      d = d + 2*x + 3
    senão
      d = d + 2*(x - y) + 5
      y = y - 1
    fim se
    x = x + 1
    pontosCirc(x, y, x1, y1)
  fim enquanto
fim função

//-----------------------------------------
vazio Desenha(vazio)
inteiro xc, yc

início
    ...
    desenhaCirc(r, xc, yc)
    ...
fim função
```

Algoritmo de Bresenham para o Desenho de Círculos Preenchidos

```
//-----------------------------------------
vazio pontosCircP(inteiro x, y, x1, y1)
início
  desenha_polígonos
    vértice(x1+x, y1+y)
    vértice(x1+y, y1+x)
    vértice(x1+y, y1-x)
    vértice(x1+x, y1-y)
    vértice(x1-x, y1-y)
    vértice(x1-y, y1-x)
    vértice(x1-y, y1+x)
    vértice(x1-x, y1+y)
  fim desenha_polígonos
fim função
```

CURVAS PARAMÉTRICAS

As curvas paramétricas podem ser geradas calculando-se poucos pontos e, a partir deles, definem-se os pontos restantes utilizando-se curvas simples e matematicamente fáceis de calcular.

Vantagens das Curvas Paramétricas Cúbicas

- Continuidade de posição.
- Passagem por pontos preespecificados.
- Continuidade entre segmentos.

Definição de uma curva paramétrica

Considere:

$P(t) = T \cdot C$

$P(t) = [\, x(t) \quad y(t) \,]$

$T = [\, t^3 \quad t^2 \quad t \quad 1 \,] = T(t)$

$$C = \begin{bmatrix} ax & ay \\ bx & by \\ cx & cy \\ dx & dy \end{bmatrix}$$ {matriz de coeficientes}

Portanto:

$P(t) = [\, ax.t^3 + bx.t^2 + cx.t + dx \quad ay.t^3 + by.t^2 + cy.t + dy \,]$

$P'(t) = [\, 3.ax.t^2 + 2.bx.t + cx \quad 3.ay.t^2 + 2.by.t + cy \,]$

$P'(t) = T' \cdot C = [\, x'(t) \quad y'(t) \,]$

$T' = [\, 3.t^2 \quad 2.t \quad 1 \quad 0 \,] = T'(t)$

Cúbica Paramétrica a 4 Coeficientes Por Eixo

É a paramétrica de mais baixa ordem capaz de assegurar compromisso entre as condições: posição e vetor tangente em cada extremidade em 3D, a partir da escolha de 4 coeficientes por eixo.

Formas Cúbicas Disponíveis:

Hermite: Posição e tangente (2 pontos e 2 ângulos).

Bézier: Posição e tangente indireta (4 pontos).

B-Spline: Posição aproximada, tangente e "curvatura".

Curva do tipo Hermite

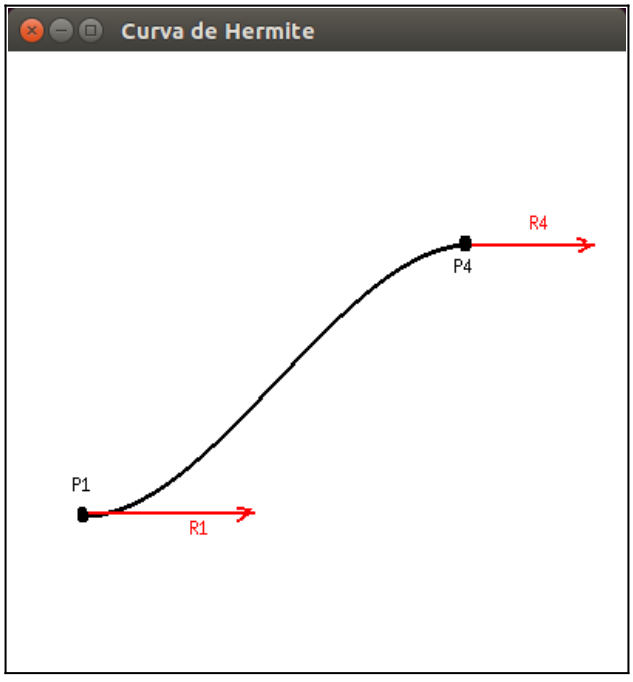
Idealizada com base no interpolador cúbico de Hermite, equações desenvolvidas na análise numérica, pelo matemático francês Charles Hermite (1822* - 1901+) (figura 3).

Figura 3 – Charles Hermite

Fonte: Britannica (2021).

É gerada a partir do fornecimento de duas posições e duas tangentes (P1 , P4 , R1 e R4). Ver figura 4.

Figura 4 – Curva paramétrica do tipo Hermite



Fonte: Autor.

Matriz de Coeficientes de Hermite

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

$X(t) = [\ (2.t^3 - 3.t^2 + 1) \quad (-2.t^3 + 3.t^2\ ) \quad (t^3 - 2.t^2 + t) \quad (t^3 - t^2\ )\ ]$

$X(t) = [\ xp1 \quad xp4 \quad xr1 \quad xr4\ ]$

Geometria de Hermite

$G_H = [\ P1 \quad P4 \quad R1 \quad R4\ ]$

$G_H = [\ (p1x, p1y) \quad (p4x, p4y) \quad (r1x, r1y) \quad (r4x, r4y)\ ]$

Definição Final da Curva de Hermite

$P(t) = X(t)\ .\ G_H$

$P\ i\ (x) = xp1*p1x + xp4*p4x + xr1*r1x + xr4*r4x$

$P\ i\ (y) = xp1*p1y + xp4*p4y + xr1*r1y + xr4*r4y$

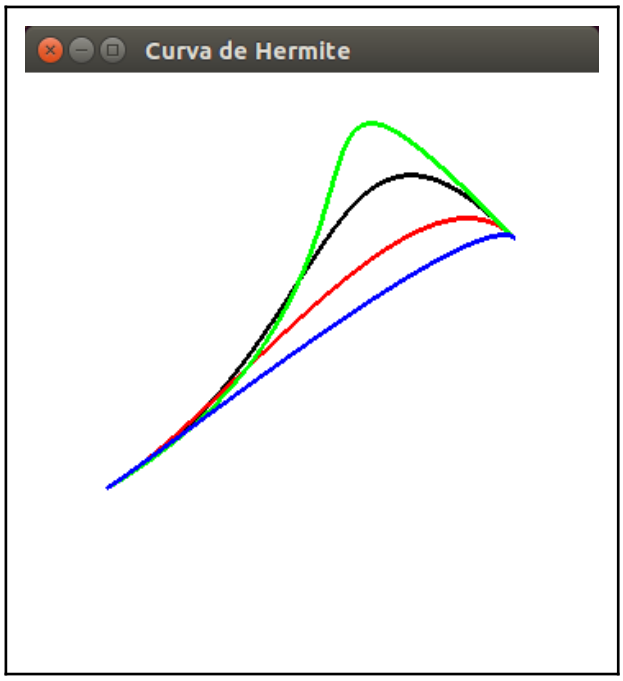
Curvatura da Curva

O parâmetro K (um número real) é utilizado para estabelecer a curvatura da curva, isto é, o quanto a curva se "entorta" sobre ela mesma. Também pode ser chamado de Constante de Proporcionalidade (figura 5).

Exemplo:

K = 1000

Figura 5 – Família de curvas de Hermite, para diferentes valores de K



Fonte: Autor.

Determinação das componentes dos vetores tangentes em P1 e P4

r1x = K*cos(ang1)

r1y = K*sin(ang1)

r4x = K*cos(ang4)

r4y = K*sin(ang4)

Algoritmo Hermite

```
// OBS: Utilize k=1000 para testar o algoritmo.

vazio curvaHermite ( decimal k, decimal ang1, decimal ang4,
                     inteiro p1x, inteiro p1y, inteiro p4x, inteiro p4y  )
inicio

        inteiro px, py, xmin, xmax, ymin, ymax, i
        decimal max, r1x, r1y, r4x, r4y, t, xp1, xp4, xr1, xr4

se (p1x > p4x)
        início
                xmin = p4x
```

```
                xmax = p1x
        fim
senão
        inicio
                xmin = p1x
                xmax = p4x
        fim

se (p1y > p4y)
        inicio
                ymin = p4y
                ymax = p1y
        fim
senão
        inicio
                ymin = p1y
                ymax = p4y
        fim

se (xmax - xmin > ymax - ymin)
        max = xmax - xmin
senão
        max = ymax - ymin

ang1 = (M_PI*ang1)/180
ang4 = (M_PI*ang4)/180

r1x = k * cos(ang1)
r1y = k * sin(ang1)
r4x = k * cos(ang4)
r4y = k * sin(ang4)

para(i=0; i<max; i++)
        início
                t = i / max
                xp1 = 2*t*t*t - 3*t*t + 1
                xp4 = - 2*t*t*t + 3*t*t

                xr1 = t*t*t - 2*t*t + t
                xr4 = t*t*t - t*t

                px = xp1 * p1x + xp4 * p4x + xr1 * r1x + xr4 * r4x
                py = xp1 * p1y + xp4 * p4y + xr1 * r1y + xr4 * r4y

                desenhaPonto( px, py )
        fim
fim
```

Curva do tipo Bézier

A curva de Bézier (1970) (figura 1) é uma curva polinomial expressa como combinação baricêntrica de alguns pontos representativos, chamados de pontos de controle. Foi desenvolvida para uso em aplicações CAD/CAM.

Figura 1 – Pierre Bézier

Fonte: Disponível em: <http://solidmodeling.org/awards/bezier-award/>. Acesso em: 15 dez. 2020.

Em (1) define-se a equação geral de uma curva Bézier, sendo que n é o número de pontos de controle menos um e i é o i-ésimo coeficiente da combinação. Por sua vez, t é o valor do parâmetro, que corresponde a um ponto da curva e B_i é o i-ésimo ponto de controle. O ponto na curva correspondente a t é dado por:

$$B(t) = \sum_{i=0}^{n} P_{i,n}(t).B_i = \sum_{i=0}^{n} \binom{n}{i}.(1-t)^{n-i}.t^i.B_i \quad (1)$$

sendo que (2) mostra a equação para determinação do coeficiente binomial.

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{n.(n-1).(n-2)\ldots(n-k+1)}{k!} \quad (2)$$

Curva de Bézier Cúbica

$$B(t) = (1-t)^3.B_0 + 3.t.(1-t)^2.B_1 + 3.t^2.(1-t).B_2 + t^3.B_3, t \in [0,1]. \quad (3)$$

As curvas de Bézier são utilizadas em diversas aplicações e formatos de imagem vetorial, como o SVG e formatos do *CorelDRAW*, por exemplo, sendo também muito utilizadas em modelagem tridimensional.

Equacionamento

$$x(t) = (1 - t)^3 . p_1x + 3.t.(1 - t)^2 . p_2x + 3.t^2.(1 - t) . p_3x + t^3 . p_4x \quad (4)$$

$$y(t) = (1 - t)^3 . p_1y + 3.t.(1 - t)^2 . p_2y + 3.t^2.(1 - t) . p_3y + t^3 . p_4y \quad (5)$$

Onde:

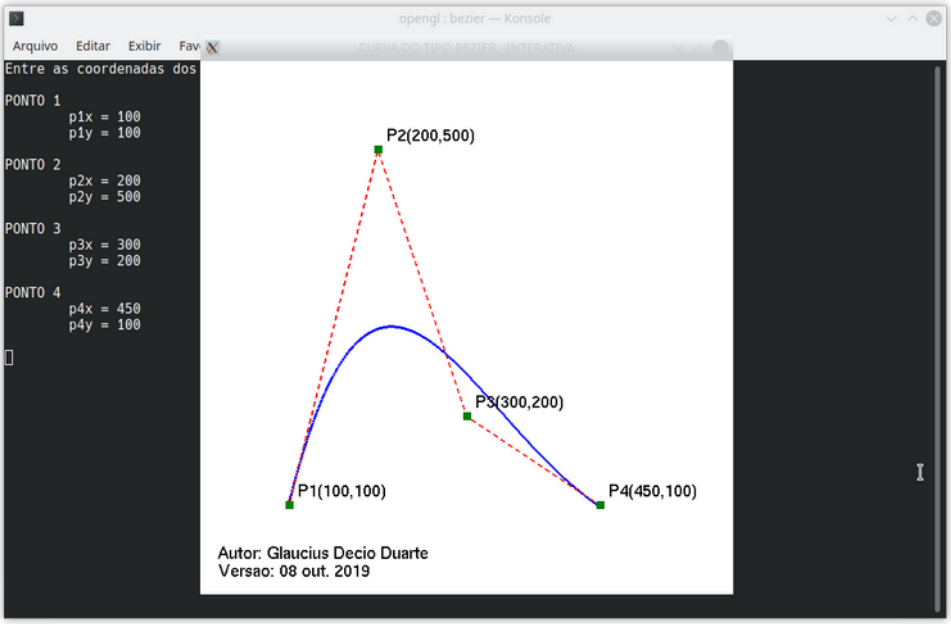
t : parâmetro (valores no intervalo real [0.0; 0.1]).

p1x, p2x, p3x, p4x : Abcissas dos pontos que definem a poligonal de controle da curva.

p1y, p2y, p3y, p4y : Ordenadas dos pontos que definem a poligonal de controle da curva.

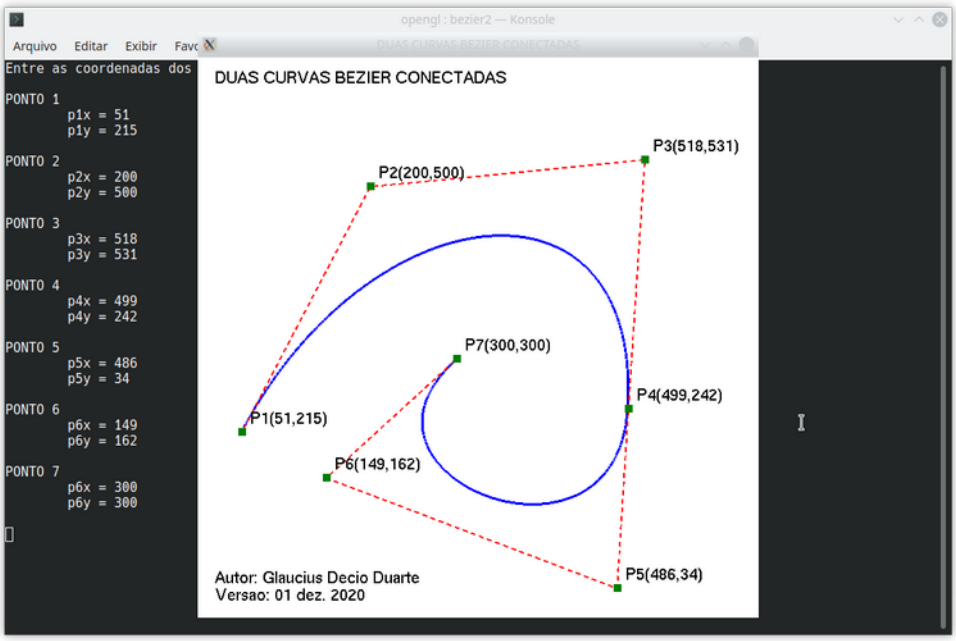
Exemplos

Figura 2 – Curva paramétrica do tipo Bézier



Fonte: Autor.

Figura 4 – Duas curvas Bézier conectadas.



Fonte: Autor.

Algoritmo Bézier

```
vazio curvaBezier ( inteiro p1x, inteiro p1y, inteiro p2x, inteiro p2y,
                    inteiro p3x, inteiro p3y, inteiro p4x, inteiro p4y )

inicio

      decimal max, t
      inteiro i, xt, yt

      se (p1x > p2x e p1x > p3x e p1x > p4x)
            xmax = p1x
      senão
            se (p2x > p1x e p2x > p3x e p2x > p4x)
                  xmax = p2x
            senão
                  se (p3x > p1x e p3x > p2x e p3x > p4x)
                        xmax = p3x
                  senão
                        se (p4x > p1x e p4x > p2x e p4x > p3x)
                              xmax = p4x


      se (p1y > p2y e p1y > p3y e p1y > p4y)
            ymax = p1y
      senão
            se (p2y > p1y e p2y > p3y e p2y > p4y)
                  ymax = p2y
            senão
                  se (p3y > p1y e p3y > p2y e p3y > p4y)
                        ymax = p3y
```

```
                  senão
                        se (p4y > p1y e p4y > p2y e p4y > p3y)
                              ymax = p4y

            se (p1x < p2x e p1x < p3x e p1x < p4x)
            xmin = p1x
      senão
            se (p2x < p1x e p2x < p3x e p2x < p4x)
                  xmin = p2x
            senão
                  se (p3x < p1x e p3x< p2x e p3x < p4x)
                        xmin = p3x
                  senão
                        se (p4x < p1x e p4x< p2x e p4x < p3x)
                              xmin = p4x

      se (p1y < p2y e p1y < p3y e p1y < p4y)
            ymin = p1y
      senão
            se (p2y < p1y e p2y < p3y e p2y < p4y)
                  ymin = p2y
            senão
                  se (p3y < p1y e p3y < p2y e p3y < p4y)
                        ymin = p3y
                  senão
                        se (p4y < p1y e p4y < p2y e p4y < p3y)
                              ymin = p4y

      se ((xmax – xmin) > (ymax – ymin))
            max = 3 * (xmax – xmin)
      senão
            max = 3 * (ymax – ymin)

      para (i=0; i<max; i++)
            início

                  t = i / max

                  xt = (1 – t)*(1 – t)*(1 – t) * p1x +
                        3*t*(1 – t)*(1 – t)*p2x +
                        3*t*t*(1 – t)*p3x + t*t*t*p4x
                  yt = (1 – t)*(1 – t)*(1 – t) * p1y +
                        3*t*(1 – t)*(1 – t)*p2y +
                        3*t*t*(1 – t)*p3y + t*t*t*p4y

                  desenhaPixel(xt, yt)

            fim
fim
```

Referências

ACADEMIA. **Jack Bresenham**. [online]. San Francisco: Academia.edu, 2014. Disponível em: https://independent.academia.edu/JackBresenham. Acesso em: 19 abr. 2021.

BÉZIER, P. **Emploi des machines a commande numerique**. Paris: Masson et Cie, 1970.

BRITANNICA. **Charles Hermite**: french mathematician. [online]. Chicago: Britannica Customer Support, 2021. Disponível em: https://www.britannica.com/biography/Charles-Hermite. Acesso em: 19 abr. 2021.

IT HISTORY SOCIETY. **Dr. Jack Elton Bresenham**: bio/description. [online]. Tiburon: IT History Society, 2021. Disponível em: https://www.ithistory.org/honor-roll/dr-jack-elton-bresenham. Acesso em: 19 abr. 2021.