REPRESENTAÇÃO DE ARCOS POR CURVAS DE BEZIER

Jonas de Miranda Gomes
Instituto de Matemática Pura e Aplicada (IMPA)
Est. D. Castorina, 110 - Rio de Janeiro, RJ

1.0 - Introdução.

Não é necessário ressaltar a importância das curvas do tipo spline em Computação Gráfica, especialmente nas áreas de *modelagem* e *animação*. A uniformização dos tipos de curva presentes em um sistema gráfico facilita o trabalho de desenvolvimento ao longo da vida util do sistema, criando uma interface unica de programação. Apresentamos no presente trabalho um algoritmo que permite aproximar arcos de círculo no plano por *Curvas de Bezier* de grau 3. Uma aplicação desse fato pode ser vista na implementação da linguagem *Postscript*, onde os comandos *arc, arcn e curveto* são implementados utilizando curvas de Bezier de terceiro grau (ver [1]).

2.0 - Curvas de Bezier

A teoria das curvas de Bezier foi desenvolvida independentemente por P. de Casteljau em 1959, e por P. Bezier em 1962. Essa teoria utiliza o conceito de polinomios de Bernstein, que são largamente utilizados na Teoria da Aproximação. A relação entre os trabalhos de Bezier e Casteljau só foi descoberta em 1970 por A. Forrest (ver [2], [3]).

Considere $n+1$ pontos no plano $b_0, \dots b_n$, e sejam $B_0^n, \dots B_n^n$ os *polinomios de Bernstein* de grau n, definidos por

$$B_i^n(t) = \binom{n}{i}(1-t)^{n-i}t^i, \quad i = 0, 1, \dots, n$$

$t \in [0,1]$. A curva de Bezier de grau n é definida por

$$X(t) = b_0 B_0^n(t) + b_1 B_1^n(t) + \cdots + b_n B_n^n(t). \tag{1}$$

O polígono $b_0, \dots b_n$, é denominado *Polígono de Bezier*. É fácil de verificar diretamente que $X(0) = b_0$ e $X(1) = b_n$. No caso em que $n = 3$, o poligono de Bezier possue quatro lados, e a equação (1) se reduz a

$$X(t) = b_0(1-u)^3 + b_1 3u(1-u)^2 + b_2 3u^2(1-u) + b_3 u^3. \tag{2}$$

Uma propriedade importante das curvas de Bezier é a de subdivisão: *se dividirmos uma curva de Bezier de grau n em seu ponto médio, obtemos duas curvas de Bezier de grau n.* Para determinarmos o polígono de Bezier de cada uma das curvas da subdivisão, fazemos tres subdivisões sucessivas pelo ponto médio de cada lado do polígono de Bezier original. Cada um dos pontos da subdivisão é um vértice do polígono de Bezier, conforme indicado na figura 1. (ver [4]).

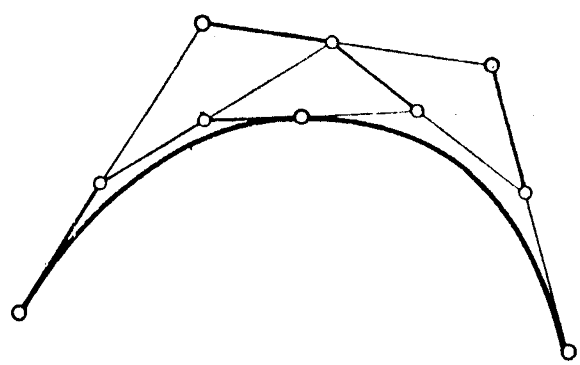

Fig. 1 - subdivisão do polígono de Bezier

Como a curva de Bezier está sempre contida no fecho convexo do polígono de Bezier (ver [4]), temos que no processo de subdivisão acima, cada um dos polígonos resultantes, l_0, l_1, l_2, l_3 e r_0, r_1, r_2, r_3 está mais próximo da curva de Bezier original. Obtemos então um algoritmo que permite aproximar uma curva de Bezier por polígonos. Este algoritmo é bastante estável, e para curvas de Bezier do terceiro grau, ele é muito eficiente. Pela sua importância vamos descreve-lo com mais detalhe. Dado um poligono de Bezier b_0, b_1, b_2, b_3, definimos a *tolerância* da aproximação como sendo o máximo da distância dos vertices b_1 e b_2 ao segmento $\overline{b_0 b_3}$ que liga os vértices extremos do polígono. Para um dado número real $\delta > 0$, aplicamos recursivamente o algoritmo de Subdivisão, até que a tolerância de cada polígono de Bezier seja $< \delta$, nesse caso substituimos cada poligono de Bezier pelo segmento que liga os extremos do poligono, e teremos a aproximação poligonal desejada da curva de Bezier.

O ponto médio da curva de Bezier no processo de subdivisão descrito acima é dado por

$$l_3 = r_3 = X(\frac{1}{2}).$$

Utilizando a equação (2) obtemos

$$l_3 = r_3 = \frac{1}{8}(b_0 + 3b_1 + 3b_2 + b_3). \qquad (3)$$

3.0 - Representação de arcos.

O problema que nos propomos a resolver nesta seção é o seguinte: *Dado um arco C de círculo com centro O, raio R e ângulo α, determinar uma curva de Bezier que seja uma aproximação para o arco dado.*

Para resolver o problema acima, basta determinar um polígono de Bezier a partir do arco dado. Como a curva de Bezier é invariante por transformações afins do plano, podemos supor que O é a origem do sistema de coordenadas, e $R = 1$. Alem disso os cálculos serão simplificados se tomarmos o eixo-x como sendo a bissetriz do ângulo α definido pelo arco, conforme indicado na figura 2.

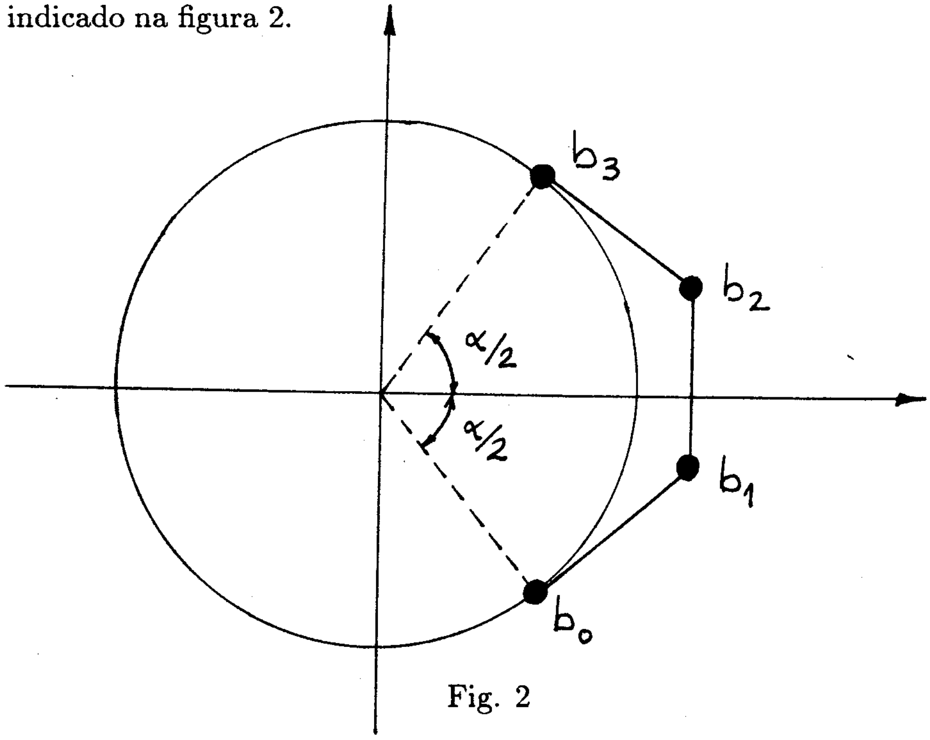

Fig. 2

O primeiro vértice do polígono de Bezier é o ponto b_0, e o último vértice coincide com o ponto b_3, e alem disso os lados inicial e final do polígono sao tangentes ao arco nos vértices inicial e final respectivamente. O problema consiste então em determinar os vértices b_1, e b_2 que definem o polígono.

Sejam

$$T = (sin\frac{\alpha}{2}, cos\frac{\alpha}{2}),$$

e

$$\overline{T} = (sin\frac{\alpha}{2}, -cos\frac{\alpha}{2})$$

os vetores tangentes ao arco nos pontos b_0 e b_3 respectivamente. Devemos calcular λ de modo que a curva de Bezier definida pelo polígono

$$\begin{aligned} b_0 &= (cos\frac{\alpha}{2}, -sin\frac{\alpha}{2}); \\ b_1 &= b_0 + \lambda T; \\ b_2 &= b_0 + \lambda\overline{T}; \\ b_3 &= (cos\frac{\alpha}{2}, sin\frac{\alpha}{2}), \end{aligned} \tag{4}$$

seja uma aproximação do arco dado, para isto exigiremos que o ponto médio M da curva de Bezier correspondente seja um ponto do arco, isto é

$$dist(M, O) = 1. \tag{5}$$

Substituindo as coordenadas dos vértices em (4) na equação (3), obtemos

$$M = \frac{1}{8}(8cos\frac{\alpha}{2} + 6\lambda sin\frac{\alpha}{2}, 0).$$

Substituindo o valor de M na equação acima em (5), após um cálculo direto, obtemos finalmente

$$\lambda = \frac{4(1 - cos\frac{\alpha}{2})}{3sin\frac{\alpha}{2}},$$

que é o valor procurado.

Como um exemplo, á fácil ver que para aproximar o arco de ângulo $\pi/2$, raio unitário, com centro na origem e cujo ponto inicial está no eixo-x, devemos utilizar o polígono de Bezier cujos vértices têm coordenadas dadas por

$$\begin{aligned} b_0 &= (1, 0); \\ b_1 &= (1, \frac{\sqrt{2}-1}{3}); \\ b_1 &= (\frac{\sqrt{2}-1}{3}, 1); \\ b_0 &= (0, 1). \end{aligned}$$

O algoritmo acima fornece uma boa aproximação de um arco para o caso em que $0 < \alpha < \pi$. Um problema interessante, e fundamental em algumas aplicações, é fazer um estudo do erro cometido na aproximação com o algoritmo acima. Erros menores certamente podem ser conseguidos utilizando curvas de Bezier de grau maior do que três.

4.0 - Agradecimentos

O algoritmo acima foi desenvolvido como parte de um projeto de implementação de um interpretador da linguagem Postscript na Globo Computação Gráfica. Meus agradecimentos a Luiz Velho pelas produtivas discussões durante o desenvolvimento do projeto.

REFERÊNCIAS

[1] Adobe Systems, *Postscript Language, Reference Manual*, Addison-Wesley, 1987.

[2] Pierre E. Bezier, *Emploi des machines à commande numerique*, Masson et Cie., Paris.

[3] A. Robin Forrest & Anne F. Pankhurst, *Numerical Control-Mathematics and Applications*, John Wiley and sons. London 1972 (Tradução de [2]).

[4] R. H. Bartels, J. C. Beatty & B. A. Barsky,*An introduction to the use of Splines in Computer Graphics.* SIGGRAPH 85 Course Notes. 1985.