Curvas de Bézier

DCC703 - Computação Gráfica (2024.2)
Prof. - Luciano Ferreira Silva
Aluno - Paulo Ferreira da Silva Júnior - 2019034400

Introdução

A rasterização de curvas de Bézier é amplamente utilizada em computação gráfica para representar curvas suaves através de um conjunto de pontos de controle. Existem diferentes abordagens para a construção dessas curvas, sendo as mais comuns:

1. **Equação Paramétrica**
2. **Algoritmo de De Casteljau**

Neste relatório, exploramos ambos os algoritmos, implementamos e analisamos seus desempenhos e diferenças fundamentais.

1. Equação Paramétrica

Descrição do Algoritmo

A equação paramétrica define a curva de Bézier a partir do polinômio de Bernstein. Para uma curva de Bézier cóbica (grau 3), a fórmula é:

$B(t) = (1-t)^3 P0+3(1-t)t^2 P1+3(1-t)t^2 P2+t^3 P3$

onde P0,P1,P2,P3 são os pontos de controle e tt varia entre 0 e 1.

Esse método permite calcular diretamente os pontos da curva sem subdivisão iterativa, tornando-se computacionalmente eficiente.

Código Implementado

```
def bernstein(n, i, t):
    """ Calcula o polinômio de Bernstein B_i^n (t) """
    return comb(n, i) * (t ** i) * ((1 - t) ** (n - i))

def bezier_generalized(t, P):
    """ Calcula pontos da curva Bézier de grau n usando a equação paramét
rica """
    n = len(P) - 1  # Grau da curva (n = número de pontos - 1)
    B = np.zeros(2)  # Inicializa coordenadas X e Y

    for i in range(n + 1):  # Para cada ponto de controle
        B += bernstein(n, i, t) * P[i]  # Soma as contribuições dos Bernstein

    return B
```

Resultado

attachment:f17e08d4-b0f0-4ade-94f7-0efe56fca456:2025-03-03-17-11-34.mp4

2. Algoritmo de De Casteljau

Descrição do Algoritmo

O algoritmo de De Casteljau é utilizado para a construção de curvas de Bézier por meio de interpolação linear sucessiva entre os pontos de controle. No código apresentado, ele é implementado de forma recursiva, subdividindo a curva até um nível de precisão determinado.

O processo segue os seguintes passos:

1. **Subdivisão Recursiva**: A cada chamada recursiva, o conjunto de pontos de controle P0,P1,P2,P3 é subdividido utilizando médias ponderadas para gerar novos pontos intermediários.

2. **Construção de Pontos Intermediários**:
   - Calcula-se os pontos médios entre cada par consecutivo dos pontos de controle.
   - Novos pontos intermediários são gerados aplicando o mesmo princípio aos pontos médios obtidos.
   - O último ponto intermediário P0xN3 pertence à curva.
3. **Critério de Parada**: O processo de subdivisão continua até que o valor de subdivisão t seja menor que um limiar (0.005), garantindo uma discretização adequada da curva.
4. **Armazenamento dos Pontos da Curva**: Os pontos pertencentes à curva de Bézier são coletados em uma lista e posteriormente utilizados para a plotagem.

A fórmula iterativa do algoritmo é dada por:

Pi(k)(t)=(1−t)Pi(k−1)+tPi+1(k−1

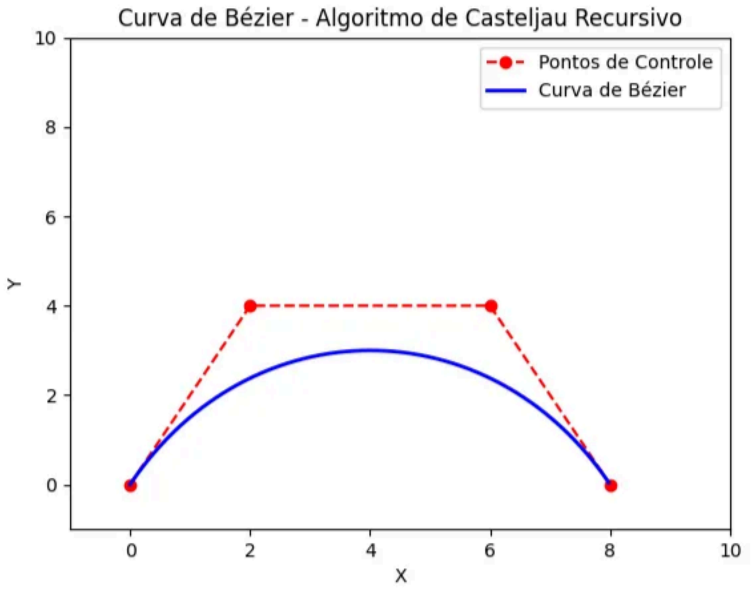
No código implementado, essa interpolação ocorre implicitamente através da subdivisão dos segmentos até que o critério de precisão seja atingido.

Código Implementado

```
def PMCurva(P0, P1, P2, P3):
    """
    Calcula os pontos intermediários da curva de Bézier usando o método d
e De Casteljau.
    Retorna os novos pontos para subdivisão.
    """
    # Primeira subdivisão
    P0xN1 = (P0 + P1) / 2
    P1xN1 = (P1 + P2) / 2
    P2xN1 = (P2 + P3) / 2

    # Segunda subdivisão
    P0xN2 = (P0xN1 + P1xN1) / 2
    P1xN2 = (P1xN1 + P2xN1) / 2

    # Terceira subdivisão (ponto na curva)
```

```
    P0xN3 = (P0xN2 + P1xN2) / 2

    return P0xN1, P1xN1, P2xN1, P0xN2, P1xN2, P0xN3

def casteljau_recursive(P0, P1, P2, P3, t, curve_points):
    """
    Algoritmo de De Casteljau recursivo para calcular pontos da curva de Bé
zier.
    """
    if t > 0.005:
        e = t / 2
        P0xN1, P1xN1, P2xN1, P0xN2, P1xN2, P0xN3 = PMCurva(P0, P1, P2, P
3)
        casteljau_recursive(P0, P0xN1, P0xN2, P0xN3, e, curve_points)
        casteljau_recursive(P0xN3, P1xN2, P2xN1, P3, e, curve_points)
    else:
        curve_points.append(P0)
```

Resultado



Comparativo Geral

Método	Eficiência Computacional	Precisão
Equação Paramétrica	Moderada	Moderada
De Casteljau	Alta	Alta

A equação paramétrica permite calcular diretamente os pontos da curva, sendo mais eficiente para curvas de grau fixo. Por outro lado, o algoritmo de De Casteljau é mais flexível e pode ser aplicado a curvas de qualquer grau, sendo mais intuitivo para manipulação interativa.

Conclusão

Os dois algoritmos são eficazes para gerar curvas de Bézier, cada um com suas vantagens. O método paramétrico é mais rápido para renderização fixa, enquanto De Casteljau é mais flexível para manipulação interativa. A escolha entre os dois depende da aplicação desejada.