Bezier and Spline Curves and Surfaces

Ed Angel
Professor of Computer Science, Electrical and Computer Engineering, and Media Arts
University of New Mexico
Objectives

• Introduce the Bezier curves and surfaces
• Derive the required matrices
• Introduce the B-spline and compare it to the standard cubic Bezier
Beziers's Idea

• In graphics and CAD, we do not usually have derivative data

• Bezier suggested using the same 4 data points as with the cubic interpolating curve to approximate the derivatives in the Hermite form
p_1 \text{ located at } u=1/3

\begin{align*}
p'(0) &\approx \frac{p_1 - p_0}{1/3} \\
\text{slope } p'(0) &\approx \frac{p_1 - p_0}{1/3} \\
p_0
\end{align*}

p_2 \text{ located at } u=2/3

\begin{align*}
p'(1) &\approx \frac{p_3 - p_2}{1/3} \\
\text{slope } p'(1) &\approx \frac{p_3 - p_2}{1/3} \\
p_3
\end{align*}
Equations

Interpolating conditions are the same

\[ p(0) = p_0 = c_0 \]
\[ p(1) = p_3 = c_0 + c_1 + c_2 + c_3 \]

Approximating derivative conditions

\[ p'(0) = 3(p_1 - p_0) = c_0 \]
\[ p'(1) = 3(p_3 - p_2) = c_1 + 2c_2 + 3c_3 \]

Solve four linear equations for \( c = M_B p \)
Bezzer Matrix

\[ \mathbf{M}_B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \]

\[ p(u) = \mathbf{u}^T \mathbf{M}_B \mathbf{p} = \mathbf{b}(u)^T \mathbf{p} \]

blending functions
Blending Functions

\[
b(u) = \begin{bmatrix}
(1-u)^3 \\
3u(1-u)^2 \\
2u^2(1-u) \\
u^3
\end{bmatrix}
\]

Note that all zeros are at 0 and 1 which forces the functions to be smooth over (0,1)
Bernstein Polynomials

- The blending functions are a special case of the Bernstein polynomials

$$b_{kd}(u) = \frac{d!}{k!(d-k)!} u^k (1-u)^{d-k}$$

- These polynomials give the blending polynomials for any degree Bezier form
  - All zeros at 0 and 1
  - For any degree they all sum to 1
  - They are all between 0 and 1 inside (0,1)
Convex Hull Property

• The properties of the Bernstein polynomials ensure that all Bezier curves lie in the convex hull of their control points.
• Hence, even though we do not interpolate all the data, we cannot be too far away.
Bezïer Patches

Using same data array $P = [p_{ij}]$ as with interpolating form

$$p(u, v) = \sum_{i=0}^{3} \sum_{j=0}^{3} b_i(u) b_j(v) p_{ij} = u^T M_B P M_B^T v$$

Patch lies in convex hull
Analysis

- Although the Bezier form is much better than the interpolating form, we have the derivatives are not continuous at join points
- Can we do better?
  - Go to higher order Bezier
    - More work
    - Derivative continuity still only approximate
    - Supported by OpenGL
  - Apply different conditions
    - Tricky without letting order increase
B-Splines

- **Basis splines**: use the data at \( p = [p_{i-2} \ p_{i-1} \ p_i \ p_{i-1}]^T \) to define curve only between \( p_{i-1} \) and \( p_i \).
- Allows us to apply more continuity conditions to each segment.
- For cubics, we can have continuity of function, first and second derivatives at join points.
- Cost is 3 times as much work for curves
  - Add one new point each time rather than three.
- For surfaces, we do 9 times as much work.
Cubic B-spline

\[ p(u) = u^T M_S p = b(u)^T p \]

\[
M_S = \begin{bmatrix}
1 & 4 & 1 & 0 \\
-3 & 0 & 3 & 0 \\
3 & -6 & 3 & 0 \\
-1 & 3 & -3 & 1 \\
\end{bmatrix}
\]
Blending Functions

\[ b(u) = \frac{1}{6} \begin{bmatrix} (1-u)^3 \\ 4 - 6u^2 + 3u^3 \\ 1 + 3u + 3u^2 - 3u^2 \\ u^3 \end{bmatrix} \]

convex hull property
B-Spline Patches

\[ p(u, v) = \sum_{i=0}^{3} \sum_{j=0}^{3} b_i(u) b_j(v) p_{ij} = u^T M_S P M_S^T v \]

defined over only 1/9 of region
Splines and Basis

• If we examine the cubic B-spline from the perspective of each control (data) point, each interior point contributes (through the blending functions) to four segments.

• We can rewrite \( p(u) \) in terms of the data points as

\[
p(u) = \sum B_i(u) p_i
\]

defining the basis functions \( \{B_i(u)\} \)
Basis Functions

In terms of the blending polynomials

\[ B_i(u) = \begin{cases} 
0 & u < i - 2 \\
b_0(u + 2) & i - 2 \leq u < i - 1 \\
b_1(u + 1) & i - 1 \leq u < i \\
b_2(u) & i \leq u < i + 1 \\
b_3(u - 1) & i + 1 \leq u < i + 2 \\
0 & u \geq i + 2 
\end{cases} \]
Generalizing Splines

• We can extend to splines of any degree
• Data and conditions to not have to given at equally spaced values (the knots)
  - Nonuniform and uniform splines
  - Can have repeated knots
    • Can force spline to interpolate points
• Cox-deBoor recursion gives method of evaluation
NURBS

- **Nonuniform Rational B-Spline** curves and surfaces add a fourth variable \( w \) to \( x, y, z \)
  - Can interpret as weight to give more importance to some control data
  - Can also interpret as moving to homogeneous coordinate

- Requires a perspective division
  - NURBS act correctly for perspective viewing

- Quadrics are a special case of NURBS