CPSC 424
Bézier Curves

Syllabus

Curves in 2D and 3D
- Implicit vs. Explicit vs. Parametric curves
- Bézier curves, de Casteljau algorithm
- Continuity
- B-Splines
- Subdivision Curves

Properties of Curves and Surfaces

Surfaces/Meshes/Advanced Topics
Clicker Test

Do you have a clicker?

A. Hardware
B. Mobile
C. No

Curves & Surfaces as Parametric Functions

Concept:
• Curve as function of artificial “time” parameter $t$

2D curve:
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} F_x(t) \\ F_y(t) \end{pmatrix} =: F(t); F : \mathbb{R} \mapsto \mathbb{R}^2$$

3D curve:
$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} F_x(t) \\ F_y(t) \\ F_z(t) \end{pmatrix} =: F(t); F : \mathbb{R} \mapsto \mathbb{R}^3$$
Parametric Curves

**Advantage:**
- Arbitrary curves in arbitrary dimensions

**Still a problem:**
- Unintuitive
  - *Try to find a formula for a specific curve you have in mind!*
- Hard to program with
  - *Deal with arbitrary mathematical functions*

**Solution:**
- Restrict yourself to specific class of functions

Spline Curves

*Description = basis functions + coefficients*

\[
F(t) = \sum_{i=0}^{n} P_i B_i(t) = (x(t), y(t))
\]

\[
x(t) = \sum_{i=0}^{n} P_i^x B_i(t)
\]

\[
y(t) = \sum_{i=0}^{n} P_i^y B_i(t)
\]

- Same basis functions for all coordinates
Polynomial Curves

Polynomial Curves:
- Restrict to polynomial functions of degree ≤ m:
  \[ x = \sum_{i=0}^{m} b_i t^i \]
- Note: \( b_i \) are vectors!
- Example curve in 2D:
  \[
  \begin{pmatrix}
  x \\
  y 
  \end{pmatrix}
  = \sum_{i=0}^{m} \begin{pmatrix}
  b_{x,i} \\
  b_{y,i}
  \end{pmatrix} t^i
  \]

Advantages:
- Computationally easy to handle
  - \( b_0 \ldots b_m \) uniquely describe curve (finite storage, easy to represent)

Disadvantages:
- Not all shapes representable
  - Partially fix with piecewise functions later (splines)
- Still not very intuitive
  - Fix: represent polynomials in different basis
Assign GEOMETRIC meaning to coefficients (base)

- Approximate/interpolate set of positions, derivatives, etc..

Parametric Curves

Commonly used classes:

- Polynomials
  - Bézier curves, Hermite interpolation etc.
- Piecewise polynomials
  - B-splines
- Rational and piecewise-rational curves
  - Rational Bézier curves, rational B-splines (NURBS)
Interpolate “Control” Points: Lagrange Polynomials

*Use points we want to interpolate as controls*

- Polynomial degree = number of input points

  ![Lagrange Polynomials](https://www.ibiblio.org/e-notes/Splines/lagrange.html)

Basis Functions: Lagrange Polynomials

- Given: m+1 parameter values \( t_0 \ldots t_m \)
- Define

\[
L_i^m(t) := \prod_{j=0 \ldots m, j \neq i} \frac{t - t_j}{t_i - t_j}; i = 0 \ldots m
\]

- Clear from definition:
  - All \( L_i^m \) are polynomials of degree \( m \)
  - \( L_i^m(t_k) = \begin{cases} 1; i = k \\ 0; \text{else} \end{cases} \)
  - In particular, all \( L_i^m \) are linearly independent!
Lagrangr Polynomials (cont)

• $L_i^m$ are **linearly independent** & there are $m+1$ of them - basis for polynomials of degree up to $m$
• Can write any polynomial of degree up to $m$ as

$$F(t) = \sum_{i=0}^{m} L_i^m(t_j) \cdot b_i$$

• In addition, we have for all $i$: $F(t_i) = b_i$
  – *In other words, the polynomial interpolates the points $(t_i, b_i)$*

Lagrange Polynomials

• [https://www.ibiblio.org/e-notes/Splines/lagrange.html](https://www.ibiblio.org/e-notes/Splines/lagrange.html)
  • Oscillates unpredictably 😞
Other Option: Hermite Curves

Geometrically-oriented coefficients
- 2 positions + 2 tangents

Require $F(0)=P_0, F(1) = P_1, F'(0)=T_0, F'(1)=T_1$

Define basis function per requirement

$$F(t) = P_0 h_{00}(t) + P_1 h_{01}(t) + T_0 h_{10}(t) + T_1 h_{11}(t)$$
Hermite Basis Functions

\[ F(t) = P_0 h_{00}(t) + P_1 h_{01}(t) + T_0 h_{10}(t) + T_1 h_{11}(t) \]

To enforce \( C(0) = P_0 \), \( C(1) = P_1 \), \( C'(0) = T_0 \), \( C'(1) = T_1 \), basis should satisfy

\[ h_{ij}(t), \ j = 0,1, \ t \in [0,1] \]

<table>
<thead>
<tr>
<th>curve</th>
<th>( F(0) )</th>
<th>( F(1) )</th>
<th>( F'(0) )</th>
<th>( F'(1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_{00}(t) )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( h_{01}(t) )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( h_{10}(t) )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( h_{11}(t) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Hermite Cubic Basis

Can satisfy with cubic polynomials as basis

\[ h_{ij}(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0 \]

Obtain - solve 4 linear equations in 4 unknowns for each basis function

\[ h_{ij}(t), \ j = 0,1, \ t \in [0,1] \]

<table>
<thead>
<tr>
<th>curve</th>
<th>( F(0) )</th>
<th>( F(1) )</th>
<th>( F'(0) )</th>
<th>( F'(1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_{00}(t) )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( h_{01}(t) )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( h_{10}(t) )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( h_{11}(t) )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Hermite Cubic Basis

Four polynomials that satisfy the conditions

\[ h_{00}(t) = t^2(2t - 3) + 1 \quad h_{01}(t) = -t^2(2t - 3) \]
\[ h_{10}(t) = t(t - 1)^2 \quad h_{11}(t) = t^2(t - 1) \]

https://codepen.io/liorda/pen/KrvBwr

Bézier Curves

**Definition:**

- Bézier curve is a polynomial curve that uses **Bernstein polynomials** as basis

\[ F(t) = \sum_{i=0}^{m} b_i B_i^m(t) \]

- \( b_i \) are called **control points** of Bézier curve

- Control polygon obtained by connecting control points with line segments
Bernstein Polynomials

\[ B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i} ; i = 0..m; t \in [0,1], \]

\[ \binom{m}{i} = \frac{m!}{(m-i)!i!} \]

Clicker Question

\[ B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i} ; i = 0..m; t \in [0,1], \]

\[ \binom{m}{i} = \frac{m!}{(m-i)!i!} \]

**What is the value of** \( B_0^m \) **at** t=0?  
A. Depends on m  
B. 1  
C. 0
Bernstein Polynomials

\[ B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}; \quad i = 0..m; \quad t \in [0,1], \]

\[ \binom{m}{i} = \frac{m!}{(m-i)!i!} \]

• Graph for degree m=1:

• Graph for m=2:

• Graph for m=3:
Bernstein Polynomials

\[ B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}; i = 0..m; t \in [0,1] \]

Properties:

• \( B_i^m(t) \) is a polynomial of degree \( m \)

• \( B_i^m(t) \geq 0 \) for \( t \in [0,1] \); \( B_0^m(0) = 1; B_i^m(0) = 0 \) for \( i \neq 0 \)

• \( B_i^m(t) = B_{m-i}^m(1-t) \)

Bernstein Polynomials

\[ B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}; i = 0..m; t \in [0,1] \]

Properties:

• \( B_i^m(t) \) has exactly one maximum in the interval \( 0..1 \). It is at \( t=i/m \) (proof: compute derivative…)

• \( \) W/o proof: all \((m+1)\) functions \( B_i^m \) are linearly independent
  – Thus they form a basis for all polynomials of degree \( \leq m \)
Bernstein Polynomials

More properties

• \[ \sum_{i=0}^{m} B_i^m(t) = (t + (1 - t))^m = 1 \]
  – (proof: apply Binomial Theorem to definition)

• \[ B_i^m(t) = t \cdot B_{i-1}^{m-1}(t) + (1 - t) \cdot B_i^{m-1}(t) \]
  – (proof on board)

• Important (later) for fast evaluation algorithm of Bézier curves (de Casteljau algorithm)

Properties of Bézier Curves
(Pierre Bézier, Renault, about 1970)

Easy to see:

• Endpoints \( b_0 \) and \( b_m \) of control polygon interpolated & corresponding parameter values are t=0 and t=1

Without proof for the moment (will be easier to show later):

• Bézier curve is tangential to control polygon at endpoints
• Curve lies within convex hull of control points
• Curve is affine invariant
• There is a fast, recursive evaluation algorithm
Mini Bonus [1%]

Prove that Bezier (Hermite, and Lagrange) curves are affine invariant

- Affine invariant: invariant under linear transformations + translation
- Post your proof as private post on piazza
- First 5 correct respondents will get 1% toward final grade