2D Spline Curves

CS 4620 Lecture 28
Administration

• A4 and PPA2 demos
  – Together on Monday
  – Please sign up
de Casteljau’s algorithm

• A recurrence for computing points on Bézier spline segments:

\[ p_{0,i} = p_i \]
\[ p_{n,i} = \alpha p_{n-1,i} + \beta p_{n-1,i+1} \]

• Cool additional feature: also subdivides the segment into two shorter ones
Recursive algorithm

```c
void DrawRecBezier (float eps) {
    if Linear (curve, eps)
        DrawLine (curve);
    else
        SubdivideCurve (curve, leftC, rightC);
        DrawRecBezier (leftC, eps);
        DrawRecBezier (rightC, eps);
}
```
Evaluating by subdivision

– Recursively split spline
  • stop when polygon is within epsilon of curve

– Termination criteria
  • distance between control points
  • distance of control points from line
  • angles in control polygon
Cubic Bézier splines

- Very widely used type, especially in 2D
  - e.g. it is a primitive in PostScript/PDF
- Nice de Casteljau recurrence for evaluation
Chaining spline segments

- Can only do so much with a single polynomial
- Can use these functions as segments of a longer curve
  - curve from $t = 0$ to $t = 1$ defined by first segment
  - curve from $t = 1$ to $t = 2$ defined by second segment

\[ f(t) = f_i(t - i) \quad \text{for } i \leq t \leq i + 1 \]

- To avoid discontinuity, match derivatives at junctions
  - this produces a $C^1$ curve
Continuity

- Smoothness can be described by degree of continuity
  - zero-order ($C^0$): position matches from both sides
  - first-order ($C^1$): tangent matches from both sides
  - second-order ($C^2$): curvature matches from both sides
  - $G^n$ vs. $C^n$
Continuity

- Parametric continuity ($C$) of spline is continuity of coordinate functions
  - $f_1'(1) = f_2'(0)$
- Geometric continuity ($G$) is continuity of the curve itself
  - $f_1'(1) = k f_2'(0)$ for some $k$
  - Derivatives have same direction, but may have diff magnitude
    - Generally $G$ is less restrictive than $C$
    - Can be $G^1$ but not $C^1$ when the tangent vector changes length
- Neither form of continuity is guaranteed by the other
  - Can be $C^1$ but not $G^1$ when $p(t)$ comes to a halt (next slide)
Geometric vs. parametric continuity

2D spline

coordinate function $x(t)$

coordinate function $y(t)$
Properties
Control

• Local control
  – changing control point only affects a limited part of spline
  – without this, splines are very difficult to use
  – many likely formulations lack this
    • polynomial fits
Trivial example: piecewise linear

- Basis function formulation: “function times point”
  - basis functions: contribution of each point as $t$ changes

  ![Diagram of piecewise linear function]

- can think of them as blending functions glued together
Control

• Convex hull property
  – convex hull = smallest convex region containing points
    • think of a rubber band around some pins
  – some splines stay inside convex hull of control points
    • make clipping, culling, picking, etc. simpler
Convex hull

- If basis functions are all positive, the spline has the convex hull property
  - we require them to sum to 1

- if any basis function is ever negative, no convex hull prop.
Affine invariance

• Transforming the control points is the same as transforming the curve
  – true for all commonly used splines
  – extremely convenient in practice…
Affine invariance

- Basis functions associated with points should always sum to 1

\[
p(t) = b_0 p_0 + b_1 p_1 + b_2 v_0 + b_3 v_1
\]
\[
p'(t) = b_0 (p_0 + u) + b_1 (p_1 + u) + b_2 v_0 + b_3 v_1
\]
\[
= b_0 p_0 + b_1 p_1 + b_2 v_0 + b_3 v_1 + (b_0 + b_1) u
\]
\[
= p(t) + u
\]
Chaining spline segments

- Hermite curves are convenient because they can be made long easily
- Bézier curves are convenient because their controls are all points
  - but it is fussy to maintain continuity constraints
  - and they interpolate every 3rd point, which is a little odd
- We derived Bézier from Hermite by defining tangents from control points
  - a similar construction leads to the interpolating Catmull-Rom spline
Hermite to Catmull-Rom

- Have not yet seen any interpolating splines
- Would like to define tangents automatically
  - use adjacent control points
  - end tangents: extra points or zero
Hermite to Catmull-Rom

- Tangents are \((p_{k+1} - p_{k-1}) / 2\)

  - scaling based on same argument about collinear case

\[
p_0 = q_k
\]

\[
p_1 = q_k + 1
\]

\[
v_0 = 0.5(q_{k+1} - q_{k-1})
\]

\[
v_1 = 0.5(q_{k+2} - q_K)
\]

\[
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
-0.5 & 0 & 0.5 & 0 \\
0 & -0.5 & 0 & 0.5
\end{bmatrix}
\begin{bmatrix}
q_{k-1} \\
q_k \\
q_{k+1} \\
q_{k+2}
\end{bmatrix}
\]
Hermite splines

- Matrix form is much simpler

\[ f(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ t_0 \\ t_1 \end{bmatrix} \]

- coefficients = rows
- basis functions = columns
Hermite to Catmull-Rom

- Tangents are \( \frac{(p_{k+1} - p_{k-1})}{2} \)
- scaling based on same argument about collinear case
  
  \[
  p_0 = q_k
  \]
  
  \[
  p_1 = q_k + 1
  \]
  
  \[
  v_0 = 0.5(q_{k+1} - q_{k-1})
  \]
  
  \[
  v_1 = 0.5(q_{k+2} - q_K)
  \]

\[
\begin{bmatrix}
  a \\
  b \\
  c \\
  d
\end{bmatrix}
= \begin{bmatrix}
  2 & -2 & 1 & 1 \\
  -3 & 3 & -2 & -1 \\
  0 & 0 & 1 & 0 \\
  1 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  -0.5 & 0 & 0.5 & 0 \\
  0 & -0.5 & 0 & 0.5
\end{bmatrix}
\begin{bmatrix}
  q_{k-1} \\
  q_k \\
  q_{k+1} \\
  q_{k+2}
\end{bmatrix}
\]
Catmull-Rom basis
Catmull-Rom splines

- Our first example of an interpolating spline
- Like Bézier, equivalent to Hermite

- First example of a spline based on just a control point sequence
- Does not have convex hull property
B-splines

• We may want more continuity than $C^1$
• We may not need an interpolating spline
• B-splines are a clean, flexible way of making long splines with arbitrary order of continuity
Cubic B-spline basis
Deriving the B-Spline

- Approached from a different tack than Hermite-style constraints
  - Want a cubic spline; therefore 4 active control points
  - Want $C^2$ continuity
  - Turns out that is enough to determine everything
Efficient construction of any B-spline

- B-splines defined for all orders
  - order $d$: degree $d - 1$
  - order $d$: $d$ points contribute to value
- One definition: Cox-deBoor recurrence

\[
\begin{align*}
    b_1 &= \begin{cases} 
        1 & 0 \leq u < 1 \\
        0 & \text{otherwise}
    \end{cases} \\
    b_d &= \frac{t}{d-1} b_{d-1}(t) + \frac{d-t}{d-1} b_{d-1}(t-1)
\end{align*}
\]
B-spline construction, alternate view

- Recurrence
  - ramp up/down
- Convolution
  - smoothing of basis fn
  - smoothing of curve
Cubic B-spline matrix

\[ f_i(t) = \left[ t^3 \quad t^2 \quad t \quad 1 \right] \cdot \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_{i-1} \\ p_i \\ p_{i+1} \\ p_{i+2} \end{bmatrix} \]
Bézier matrix

\[ f(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix} \]

– note that these are the Bernstein polynomials

\[ b_{n,k}(t) = \binom{n}{k} t^k (1 - t)^{n-k} \]

and that defines Bézier curves for any degree
Cubic B-spline basis
Over many segments

Uniform BSplines
B-spline

• All points are same, no special points
• Basis functions are the same
Converting spline representations

• All the splines we have seen so far are equivalent
  – all represented by geometry matrices

\[ p_S(t) = T(t)M_SP_S \]

• where \( S \) represents the type of spline
  – therefore the control points may be transformed from one type to another using matrix multiplication

\[ P_1 = M_1^{-1}M_2P_2 \]

\[ p_1(t) = T(t)M_1(M_1^{-1}M_2P_2) = T(t)M_2P_2 = p_2(t) \]
Other types of B-splines

• Nonuniform B-splines
  – discontinuities not evenly spaced
  – allows control over continuity or interpolation at certain points
  – e.g. interpolate endpoints (commonly used case)

• Nonuniform Rational B-splines (NURBS)
  – ratios of nonuniform B-splines: $x(t) / w(t); y(t) / w(t)$
  – key properties:
    • invariance under perspective as well as affine
    • ability to represent conic sections exactly