CS770/870 Spring 2017 Curve Generation

• Primary resources used in preparing these notes:

1. Foley, van Dam, Feiner, Hughes, Phillips, **Introduction to Computer Graphics**, Addison-Wesley, 1993.
2. Angel, **Interactive Computer Graphics — A top-down approach with OpenGL**, Addison-Wesley, 1998.
3. Wikipedia for several images.

Curve and Surface Generation

- Fundamental applications
  - Curve/surface fitting
    - given sample data, determining a curve or surface that approximates that data in a compact form
  - Modeling: designing a curve or surface
    - for designing and producing parts
    - for realistic image generation

2D Parametric Curves

- $\mathbf{f}(t)=[x(t),y(t)]$
  - linear: $x(t) = x_0 + t(x_1-x_0) = a_x + b_x t$
    $y(t) = y_0 + t(y_1-y_0) = a_y + b_y t$
  - quadratic: $x(t) = a_x + b_x t + c_x t^2$
    $y(t) = a_y + b_y t + c_y t^2$
  - cubic: $x(t) = a_x + b_x t + c_x t^2 + d_x t^3$
    $y(t) = a_y + b_y t + c_y t^2 + d_y t^3$

3D Parametric Curves

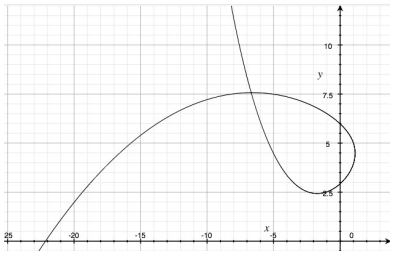
- $\mathbf{f}(t)=[x(t),y(t),z(t)]$
  - linear: $x(t) = x_0 + t(x_1-x_0) = a_x + b_x t$
    $y(t) = y_0 + t(y_1-y_0) = a_y + b_y t$
    $z(t) = z_0 + t(z_1-z_0) = a_z + b_z t$
  - quadratic: $x(t) = a_x + b_x t + c_x t^2$
    $y(t) = a_y + b_y t + c_y t^2$
    $z(t) = a_z + b_z t + c_z t^2$
  - cubic: $x(t) = a_x + b_x t + c_x t^2 + d_x t^3$
    $y(t) = a_y + b_y t + c_y t^2 + d_y t^3$
    $y(t) = a_z + b_z t + c_z t^2 + d_z t^3$

Notation

- Cubic: $x(t) = a_x + b_x t + c_x t^2 + d_x t^3$
  $y(t) = a_y + b_y t + c_y t^2 + d_y t^3$
  $y(t) = a_z + b_z t + c_z t^2 + d_z t^3$
- Let $M = \begin{vmatrix} a_x & b_x & c_x & d_x \\ a_y & b_y & c_y & d_y \\ a_z & b_z & c_z & d_z \end{vmatrix}$ $T = \begin{vmatrix} 1 \\ t \\ t^2 \\ t^3 \end{vmatrix}$
- Then $C(t) = [x(t)\ y(t)\ z(t)]^T = MT$

What good is a cubic polynomial?

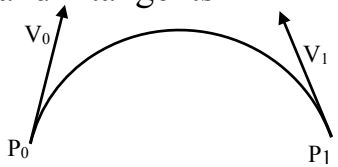
- Consider
  $x(t) = 0.2t^3 - 2t^2 + t + 1$
  $y(t) = -0.3t^3 - 0.2t^2 - 2t + 5$
  $-3 < t < 3$



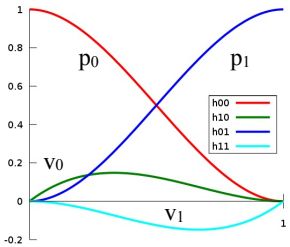
- Simple polynomial; simple to plot
- But, how can we predict how to use it?
- How to determine coefficients for a particular desired curve or surface?
  - piecewise curves and surfaces
  - sample points
  - control points

Hermite Interpolation

- Each segment defined by 2 points and 2 tangents
  - There are 4 *blending* functions (cubic polynomials), t = [0,1], $p_0(t)$, $p_1(t)$, $v_0(t)$, $v_1(t)$.



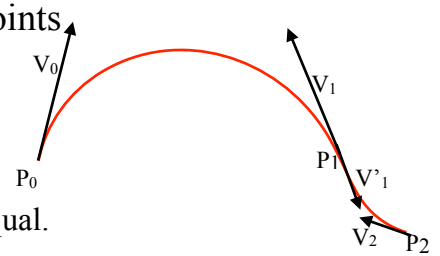
- Blending functions



$[x(t),y(t)]^T = P_0 p_0(t) + P_1 p_1(t) + V_0 v_0(t) + V_1 v_1(t)$

Hermite Interpolation-2

- Add P_2 and V_2 to get longer curve (for t=[1,2])
  - Get C^0 continuity of joined segments at P_1
  - G^1 continuity (1st deriv.) at P_1 if V_1 is parallel to V'_1
- Multiple joined segments make longer curves
- Curve *interpolates* all the points

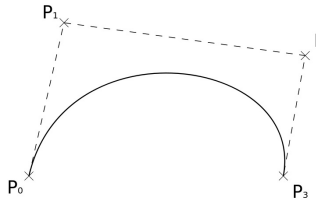


C^1 *parametric* continuity: tangents equal.

G^1 *geometric* continuity: tangents parallel.

Bezier Curves

- Replace explicit tangents at interpolating points with extra *control points* that define tangents at ends, but also modify intermediate points
- Blending functions are all >=0

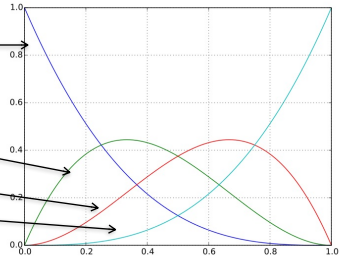


$b_0(t) = (1-t)^3$

$b_1(t) = 3t(1-t)^2$

$b_2(t) = 3t^2(1-t)$

$b_s(t) = t^3$



- $b(t) = P_0b_0(t)+P_1b_1(t)+P_2b_2(t)+P_3b_3(t)$

Bezier Matrix formulation

- Can compactly express the blending function operation as matrix operations

$$M_b = \begin{vmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 1 \\ -1 & 3 & -3 & 1 \end{vmatrix} \qquad T = \begin{vmatrix} 1 \\ t \\ t^2 \\ t^3 \end{vmatrix}$$

- $b(t) = PM_bT$

where

$$P = \begin{vmatrix} x_0 & x_1 & x_2 & x_3 \\ y_0 & y_1 & y_2 & y_3 \\ z_0 & x_1 & z_2 & z_3 \end{vmatrix}$$

$P_0=(x_0\ y_0\ z_0)$
$P_1=(x_1\ y_1\ z_1)$
$P_2=(x_2\ y_2\ z_2)$
$P_3=(x_3\ y_3\ z_3)$

Uniform B-Splines

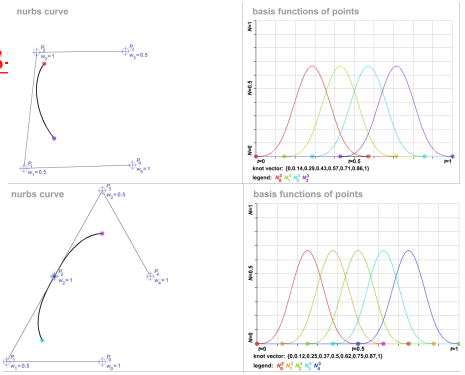
- B-splines are better than Bezier curves for joining multiple short curve segments
- Each piece of a B-spline curve depends on only a few control points, and there is better control over continuity between the pieces
- Key difference: B-spline curve segment share *multiple* control points
  - Cubic B-spline has 4 control points/segment
  - segment *i* shares 3 CP with segment *i+1*

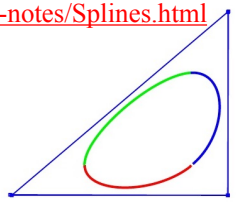
Uniform Cubic B-spline Curve

- 4 Control points define a curve segment
- 5 control points define 2
- Can replicate end points to get closed curves or curves that start at first and/or last CP

http://geometrie.foretnik.net/files/NURBS.

www.ibiblio.org/e-notes/Splines.html





Unif B-Spline Matrix formulation

- Can compactly express the blending function operation as matrix operations

$$M_B = \frac{1}{6}\begin{vmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{vmatrix} \qquad T = \begin{vmatrix} 1 \\ t \\ t^2 \\ t^3 \end{vmatrix}$$

- $b(t) = PM_BT$

where

$$P = \begin{vmatrix} x_0 & x_1 & x_2 & x_3 \\ y_0 & y_1 & y_2 & y_3 \\ z_0 & x_1 & z_2 & z_3 \end{vmatrix} \qquad \begin{aligned} P_0&=(x_0\ y_0\ z_0) \\ P_1&=(x_1\ y_1\ z_1) \\ P_2&=(x_2\ y_2\ z_2) \\ P_3&=(x_3\ y_3\ z_3) \end{aligned}$$

Uniform B-Spline Blending Curves

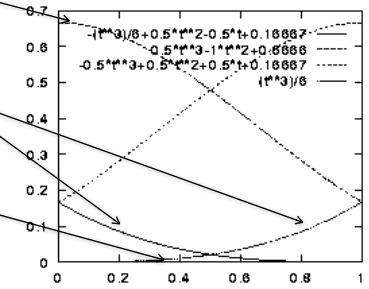
- Blending functions are all >=0

$b_0(t) = (1-t)^3$

$b_1(t) = 4-6t^2+3t^3$

$b_2(t) = 1+3t+3t^2-3t^2$

$b_s(t) = t^3$



- $b(t) = P_0b_0(t)+P_1b_1(t)+P_2b_2(t)+P_3b_3(t)$

Non-uniform B-Splines

- Uniform B-Splines distribute the *t* parameter values equally for all segments.
  - Given 4 segments *t* can range from (0,0.25) for the 1st, (0.25,0.5) for the 2nd, etc.
  - Very easy for *t* to be (0,1) for 1st, (1,2) for 2nd, etc.
  - Either way, the "join" positions are called *knots*.
- Knots can be defined at any *ordered* set of *t* values
  - *http://www.ibiblio.org/e-notes/Splines/none.html*
  - *http://geometrie.foretnik.net/files/NURBS-en.swf*

Resources

- NURBS: non-uniform rational B-Splines demo
  http://nurbscalculator.in
  http://geometrie.foretnik.net/files/NURBS-en.swf
- Cubic Bezier spline patch
  https://www.ibiblio.org/e-notes/Splines/bezier3d.html