CS4620/5620: Lecture 26

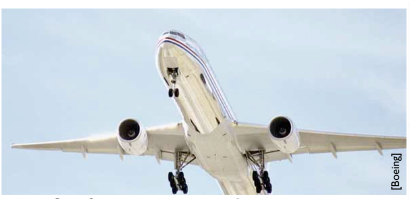
Splines

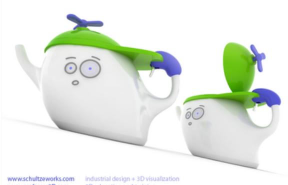
Announcements

- PPA 1 grading: Thu and Fri
  - Signup or send us mail

Motivation: smoothness

- In many applications we need smooth shapes
  - that is, without discontinuities

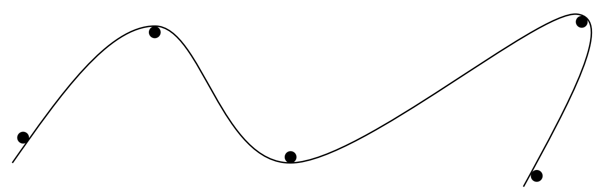




- So far we can make
  - things with corners (lines, squares, triangles, …)
  - circles and ellipses (only get you so far!)

Classical approach

- Pencil-and-paper draftsmen also needed smooth curves
- Origin of "spline": strip of flexible metal
  - held in place by pegs or weights to constrain shape
  - traced to produce smooth contour



Translating into usable math

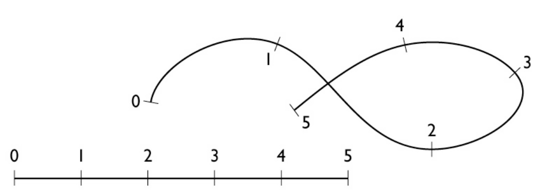
- Smoothness
  - in drafting spline, comes from physical curvature minimization
  - in CG spline, comes from choosing smooth functions
    - usually low-order polynomials
- Control
  - in drafting spline, comes from fixed pegs
  - in CG spline, comes from user-specified *control points*

Defining spline curves

- At the most general they are parametric curves

$$S = \{\mathbf{p}(t) \,|\, t \in [0, N]\}$$

- Generally *p*(*t*) is a piecewise polynomial
  - the discontinuities are at the integers



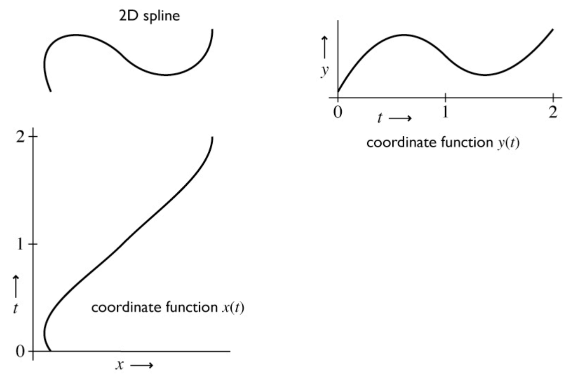
Defining spline curves

- Generally $p(t)$ is a piecewise polynomial
  - the discontinuities are at the integers
  - e.g., a cubic spline has the following form over [k, k + 1]:

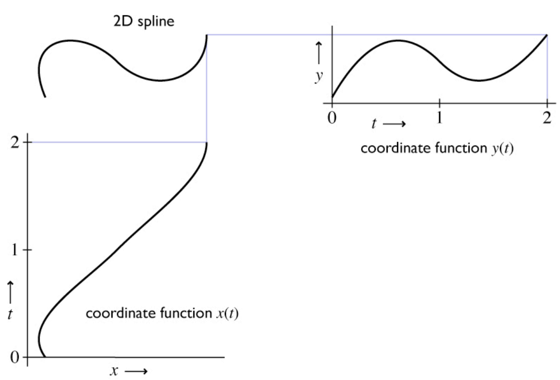
$$x(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$
$$y(t) = a_y t^3 + b_y t^2 + c_y t + d_y$$

  - Coefficients are different for every interval

Coordinate functions

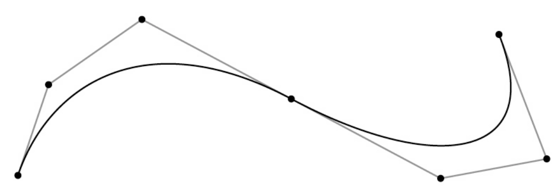


Coordinate functions



Control of spline curves

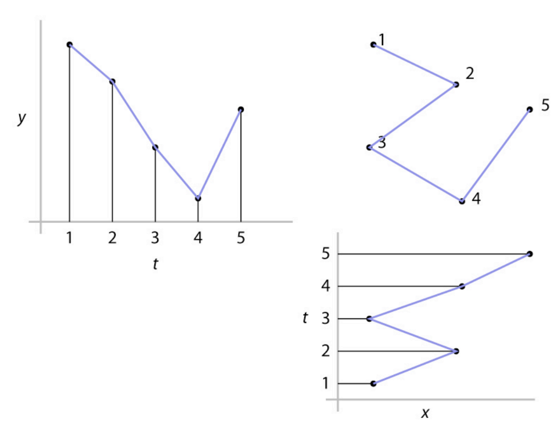
- Specified by a sequence of control points
- Shape is guided by control points (aka control polygon)
  - interpolating: passes through points
  - approximating: merely guided by points



How splines depend on their controls

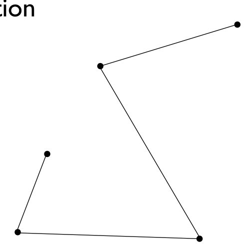
- Each coordinate is separate
  - the function $x(t)$ is determined solely by the x coordinates of the control points
  - this means 1D, 2D, 3D, … curves are all really the same
- Spline curves are **linear** functions of their controls
  - moving a control point two inches to the right moves $x(t)$ twice as far as moving it by one inch
  - $x(t)$, for fixed t, is a linear combination (weighted sum) of the control points' x coordinates
  - $\mathbf{p}(t)$, for fixed t, is a linear combination (weighted sum) of the control points

Splines as reconstruction



Trivial example: piecewise linear

- This spline is just a polygon
  - control points are the vertices
- But we can derive it anyway as an illustration
- Each interval will be a linear function
  - $x(t) = at + b$
  - constraints are values at endpoints
  - $b = x_0$; $a = x_1 - x_0$
  - this is linear interpolation



Trivial example: piecewise linear

- Vector formulation

$$x(t) = (x_1 - x_0)t + x_0$$
$$y(t) = (y_1 - y_0)t + y_0$$
$$\mathbf{p}(t) = (\mathbf{p}_1 - \mathbf{p}_0)t + \mathbf{p}_0$$

- Matrix formulation

$$\mathbf{p}(t) = \begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \end{bmatrix}$$

Trivial example: piecewise linear

- Basis function formulation
  - regroup expression by **p** rather than *t*

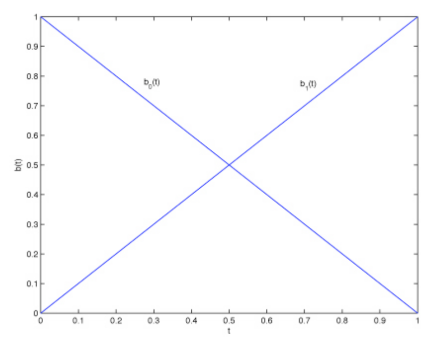
$$\mathbf{p}(t) = (\mathbf{p}_1 - \mathbf{p}_0)t + \mathbf{p}_0$$
$$= (1 - t)\mathbf{p}_0 + t\mathbf{p}_1$$

  - interpretation in matrix viewpoint

$$\mathbf{p}(t) = \left(\begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \right) \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \end{bmatrix}$$

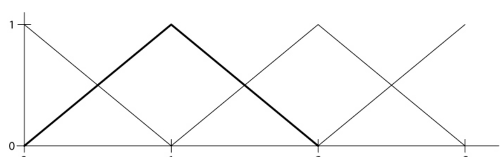
Trivial example: piecewise linear

- Vector blending formulation: "average of points"
  - blending functions: contribution of each point as *t* changes



Trivial example: piecewise linear

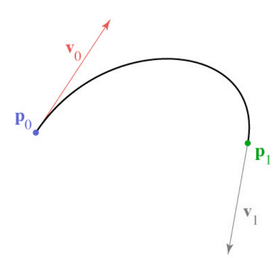
- Basis function formulation: "function times point"
  - basis functions: contribution of each point as *t* changes



  - can think of them as blending functions glued together
  - this is just like a reconstruction filter!

Hermite splines

- Less trivial example
- Form of curve: piecewise cubic
- Constraints: endpoints and tangents (derivatives)



Hermite splines

- Solve constraints to find coefficients

$$\begin{aligned} x(t) &= at^3 + bt^2 + ct + d \\ x'(t) &= 3at^2 + 2bt + c \\ x(0) &= x_0 = d \\ x(1) &= x_1 = a + b + c + d \\ x'(0) &= x'_0 = c \\ x'(1) &= x'_1 = 3a + 2b + c \end{aligned}$$

$$\begin{aligned} d &= x_0 \\ c &= x'_0 \\ a &= 2x_0 - 2x_1 + x'_0 + x'_1 \\ b &= -3x_0 + 3x_1 - 2x'_0 - x'_1 \end{aligned}$$

Hermite splines

- Matrix form is much simpler

$$\mathbf{p}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{v}_0 \\ \mathbf{v}_1 \end{bmatrix}$$

  - coefficients = rows
  - basis functions = columns

Longer Hermite splines

- Can only do so much with one Hermite spline
- Can use these splines 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
- To avoid discontinuity, match derivatives at junctions
  - this produces a C^1 curve

Hermite splines

- Hermite basis functions