CSC 433/533 Computer Graphics

Alon Efrat
Credit: Joshua Levine

Lecture 26 Curves

Dec. 2, 2019

Today's Agenda

- Reminders:
  - A07 questions?
  - TCEs! Already at 33.33%!
- Goals for today: start discussing curves and curve modeling

Demo

https://www.geogebra.org/m/jUwMnfQk

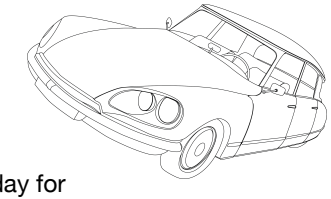
Building Complex Shapes from Primitives

- Our basic primitives (points, line segments, triangles) are great for *rendering*, but hard to work with for *modeling*
- What we can do right now:
  - Polygonal shapes with “corners” (triangles, rectangles, etc.)
  - Simple implicit shapes (circles, spheres, conics) that we can write equations for.
- In computer graphics though, we often need smooth shapes!

Inspiration: Drafting

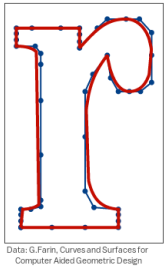
- How to draw a smooth curve on a piece of paper?
  - Use flexible metal strip, hold in place with weights (called “ducks”)
  - Trace to draw the curve

Applications

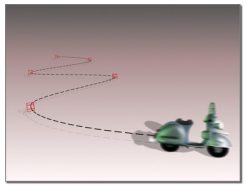


- We use the techniques from today for modeling complex shapes
- Also for modern typography
- And for designing smooth paths for animation

Font as a B-spline curve



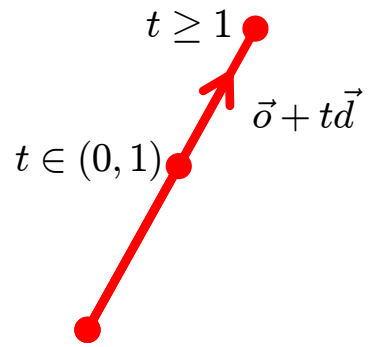
Data: G.Farin, Curves and Surfaces for Computer Aided Geometric Design



Autodesk

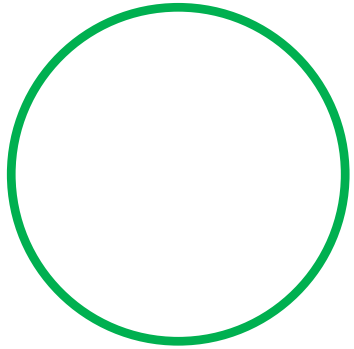
Parametric Curves

Parameterized Line



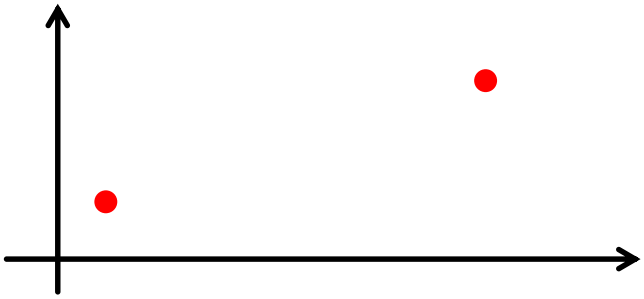
We've seen it before!

Parameterized Circle



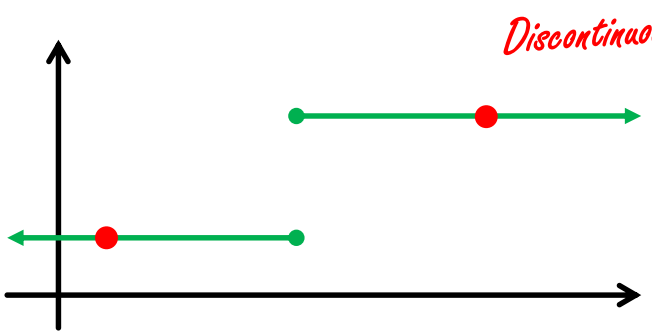
$$\vec{\gamma}(t) = (\cos t, \sin t)$$

1D Problem: Interpolation



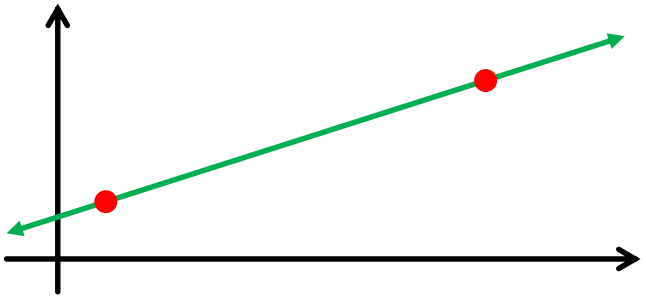
How do you fill the space?

1D Problem: Interpolation



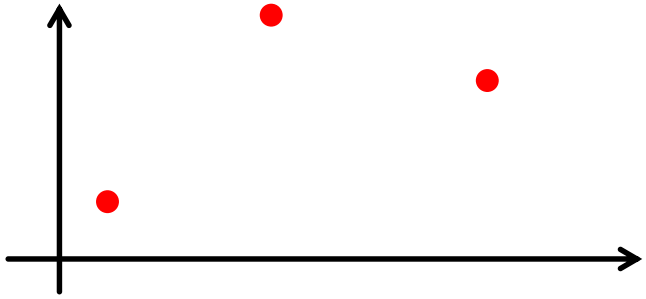
Nearest-neighbor interpolation

1D Problem: Interpolation



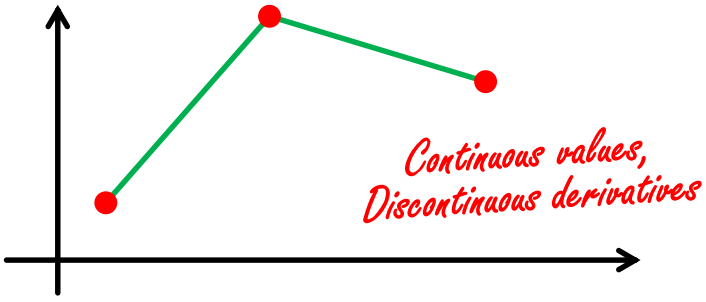
Linear interpolation

1D Problem: Interpolation



Three points?

1D Problem: Interpolation

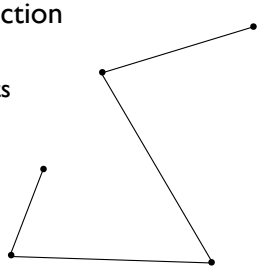


Piecewise linear

Matrix Form,
Blending Functions

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

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

- Matrix formulation

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

Trivial example: piecewise linear

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

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

  - interpretation in matrix viewpoint

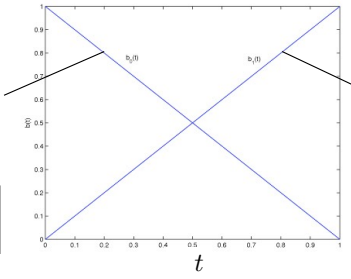
$$\mathbf{f}(t) = \left(\begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \right) \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \end{bmatrix}$$

blending function, $b_i(t)$

Trivial example: piecewise linear

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

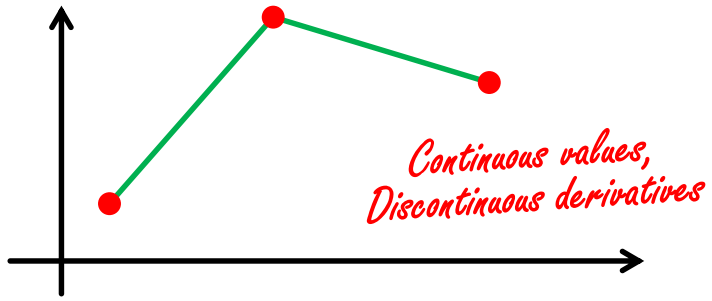
$\mathbf{b}_0(t) = 1 - t$ $\qquad$ $\mathbf{b}_1(t) = t$



$$\mathbf{f}(t) = \sum_{i=0}^{n} \mathbf{p}_i b_i(t)$$

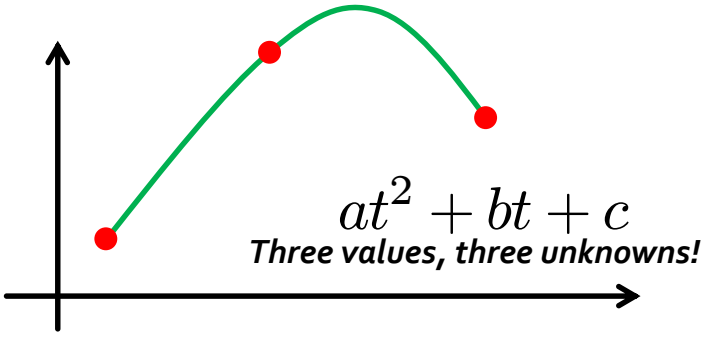
Hermite Interpolation

1D Problem: Interpolation



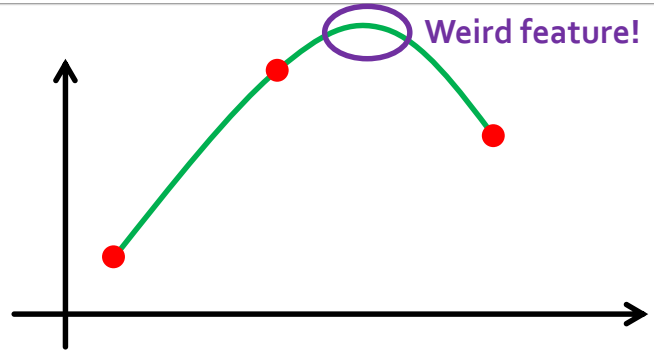
Piecewise linear

1D Problem: Interpolation



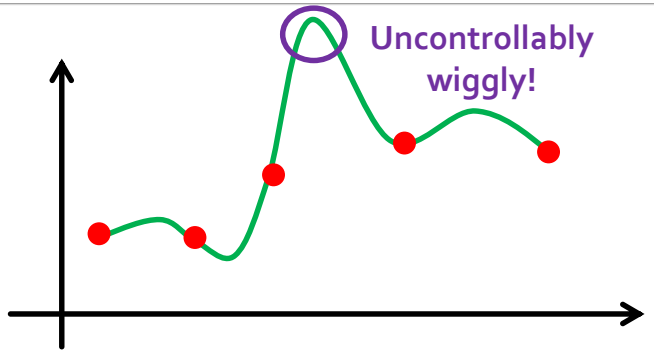
Parabola

1D Problem: Interpolation



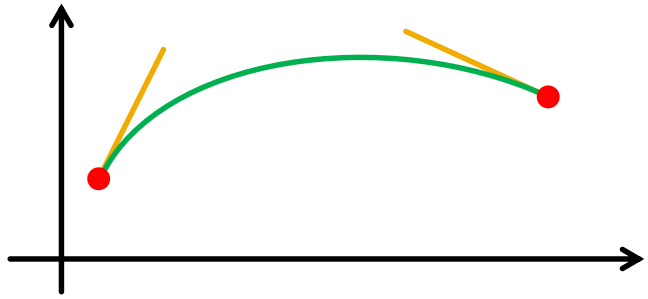
Parabola

Problems with Polynomials



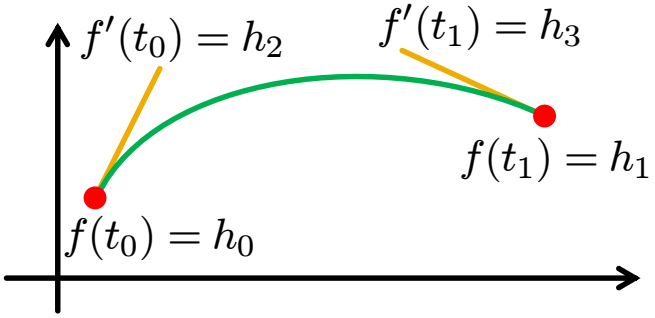
k* points yields degree *k

New Idea

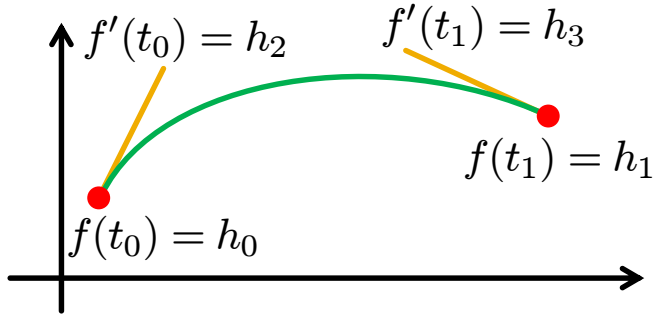


Specify points and slopes

Degrees of Freedom?



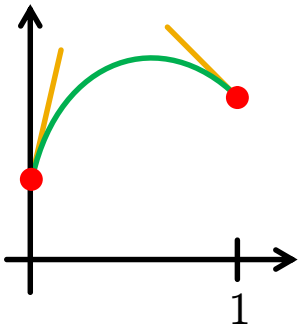
Degrees of Freedom?



Four: Two points, two slopes

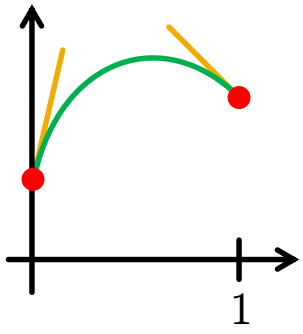
Cubic Hermite Interpolation

$$P(t) = at^3 + bt^2 + ct + d$$
$$P'(t) = 3at^2 + 2bt + c$$



Cubic Hermite Interpolation

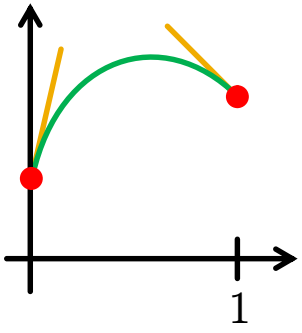
$$P(t) = at^3 + bt^2 + ct + d$$
$$P'(t) = 3at^2 + 2bt + c$$



$$P(0) = d$$
$$P(1) = a + b + c + d$$
$$P'(0) = c$$
$$P'(1) = 3a + 2b + c$$

Cubic Hermite Interpolation

$$P(t) = at^3 + bt^2 + ct + d$$
$$P'(t) = 3at^2 + 2bt + c$$



$$P(0) = d = h_0$$
$$P(1) = a + b + c + d = h_1$$
$$P'(0) = c = h_2$$
$$P'(1) = 3a + 2b + c = h_3$$

Matrix Representation

$$P(0) = d = h_0$$
$$P(1) = a + b + c + d = h_1$$
$$P'(0) = c = h_2$$
$$P'(1) = 3a + 2b + c = h_3$$

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{pmatrix}$$

Matrix Representation

Invert matrix

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{pmatrix}$$

$$\begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

Matrix Representation

$$\begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

$$a = 2h_0 - 2h_1 + h_2 + h_3$$

$$b = -3h_0 + 3h_1 - 2h_2 - h_3$$

$$c = h_2$$

$$d = h_0$$

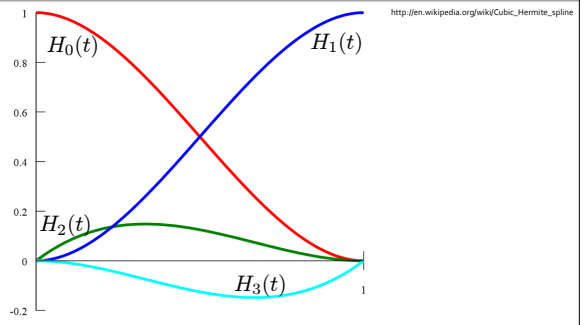
Back to the Curve

$$\begin{aligned} P(t) =& at^3 + bt^2 + ct + d \\ =& (2h_0 - 2h_1 + h_2 + h_3)t^3 + \\ & (-3h_0 + 3h_1 - 2h_2 - h_3)t^2 + \\ & h_2 t + h_0 \end{aligned}$$

Back to the Curve

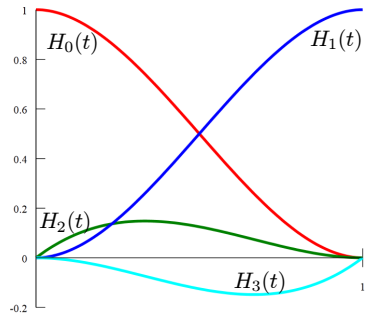
$$\begin{aligned} P(t) =& at^3 + bt^2 + ct + d \\ =& (2h_0 - 2h_1 + h_2 + h_3)t^3 + \\ & (-3h_0 + 3h_1 - 2h_2 - h_3)t^2 + \\ & h_2 t + h_0 \\ =& h_0(2t^3 - 3t^2 + 1) + h_1(-2t^3 + 3t^2) + \\ & h_2(t^3 - 2t^2 + t) + h_3(t^3 - t^2) \end{aligned}$$

Hermite Basis



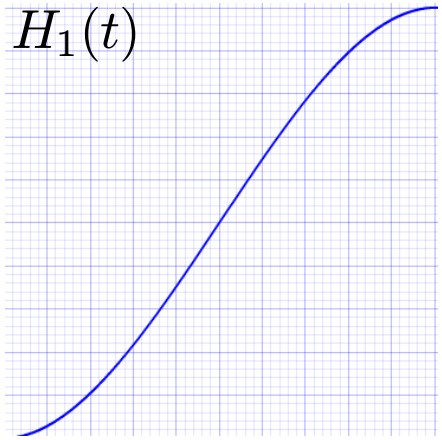
$$=h_0(2t^3-3t^2+1)+h_1(-2t^3+3t^2)+$$
$$h_2(t^3-2t^2+t)+h_3(t^3-t^2)$$

How to Compute Cubic Curve



$$P(t) = P(0)H_0(t) + P(1)H_1(t)$$
$$+ P'(0)H_2(t) + P'(1)H_1(t)$$

"Ease" Function



Ends slow

Starts slow

Continuity vs. Smoothness

Why Cubic?

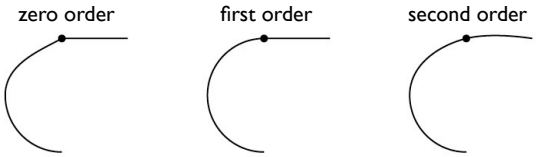
- Typically, a nice compromise between visually smoothness and not too “wiggly”
- Can we define how “smooth” a curve is?
  - Yes! standard form uses how *continuous* a curve is.

Continuity

- Given a parametric curve f(t), we know the curve is continuous if there are no “breaks”
- Standard mathematical definition of continuous function:
  - $\lim_{x \to z} f(x) = f(z)$
  - Must approach f(z) from both sides

Generalizing 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



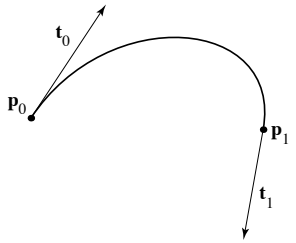
Why Cubic?

- Typically, a nice compromise between visually smoothness and not too “wiggly”
- Higher order polynomials offer more derivatives, but often will overfit as a result without much added benefit (depends on the application though)

Bézier Curves

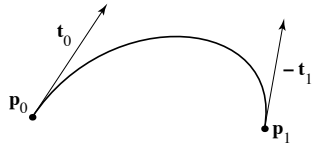
Hermite to Bézier

- Mixture of points and vectors is awkward
- Specify tangents as differences of points



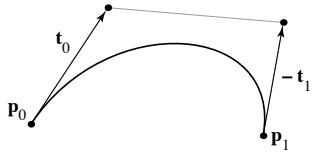
Hermite to Bézier

- Mixture of points and vectors is awkward
- Specify tangents as differences of points



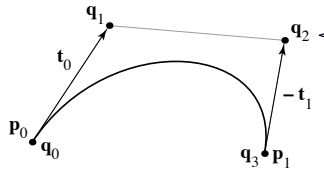
Hermite to Bézier

- Mixture of points and vectors is awkward
- Specify tangents as differences of points



Hermite to Bézier

- Mixture of points and vectors is awkward
- Specify tangents as differences of points

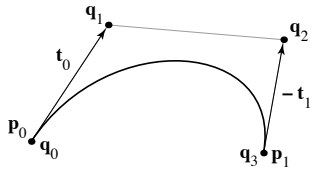


I'm calling these points **q** just for this slide and the next one.

– note derivative is defined as 3 times offset

Hermite to Bézier

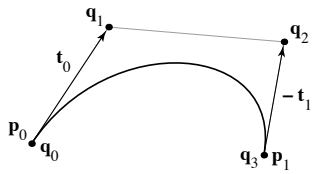
$$\mathbf{p}_0 = \mathbf{q}_0$$
$$\mathbf{p}_1 = \mathbf{q}_3$$
$$\mathbf{t}_0 = 3(\mathbf{q}_1 - \mathbf{q}_0)$$
$$\mathbf{t}_1 = 3(\mathbf{q}_3 - \mathbf{q}_2)$$



$$\begin{bmatrix}\mathbf{p}_0\\ \mathbf{p}_1\\ \mathbf{v}_0\\ \mathbf{v}_1\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ -3 & 3 & 0 & 0\\ 0 & 0 & -3 & 3\end{bmatrix} \begin{bmatrix}\mathbf{q}_0\\ \mathbf{q}_1\\ \mathbf{q}_2\\ \mathbf{q}_3\end{bmatrix}$$

Hermite to Bézier

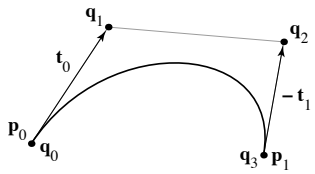
$$\mathbf{p}_0 = \mathbf{q}_0$$
$$\mathbf{p}_1 = \mathbf{q}_3$$
$$\mathbf{t}_0 = 3(\mathbf{q}_1 - \mathbf{q}_0)$$
$$\mathbf{t}_1 = 3(\mathbf{q}_3 - \mathbf{q}_2)$$



$$\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\\ \mathbf{c}\\ \mathbf{d}\end{bmatrix} = \begin{bmatrix}2 & -2 & 1 & 1\\ -3 & 3 & -2 & -1\\ 0 & 0 & 1 & 0\\ 1 & 0 & 0 & 0\end{bmatrix} \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ -3 & 3 & 0 & 0\\ 0 & 0 & -3 & 3\end{bmatrix} \begin{bmatrix}\mathbf{q}_0\\ \mathbf{q}_1\\ \mathbf{q}_2\\ \mathbf{q}_3\end{bmatrix}$$

Hermite to Bézier

$$\mathbf{p}_0 = \mathbf{q}_0$$
$$\mathbf{p}_1 = \mathbf{q}_3$$
$$\mathbf{t}_0 = 3(\mathbf{q}_1 - \mathbf{q}_0)$$
$$\mathbf{t}_1 = 3(\mathbf{q}_3 - \mathbf{q}_2)$$



$$\begin{bmatrix}\mathbf{a}\\ \mathbf{b}\\ \mathbf{c}\\ \mathbf{d}\end{bmatrix} = \begin{bmatrix}-1 & 3 & -3 & 1\\ 3 & -6 & 3 & 0\\ -3 & 3 & 0 & 0\\ 1 & 0 & 0 & 0\end{bmatrix} \begin{bmatrix}\mathbf{q}_0\\ \mathbf{q}_1\\ \mathbf{q}_2\\ \mathbf{q}_3\end{bmatrix}$$

Bézier matrix

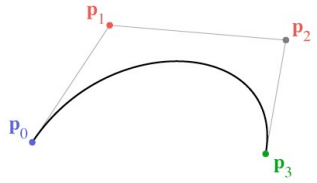
$$\mathbf{f}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

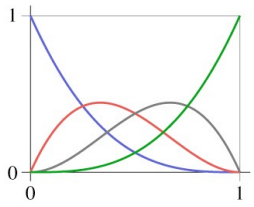
– note that these are the Bernstein polynomials

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

and that defines Bézier curves for any degree

Bézier basis



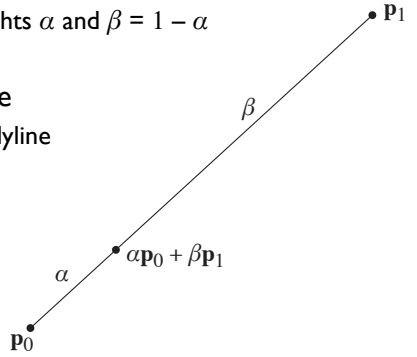


Another way to Bézier segments

- A really boring spline segment: f(t) = p0
  - it only has one control point
  - the curve stays at that point for the whole time
- Only good for building a *piecewise constant* spline
  - a.k.a. a set of points

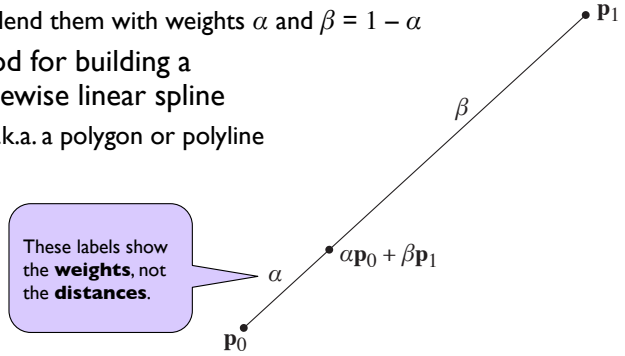
Another way to Bézier segments

- A piecewise linear spline segment
  - two control points per segment
  - blend them with weights α and $\beta = 1 - \alpha$
- Good for building a piecewise linear spline
  - a.k.a. a polygon or polyline



Another way to Bézier segments

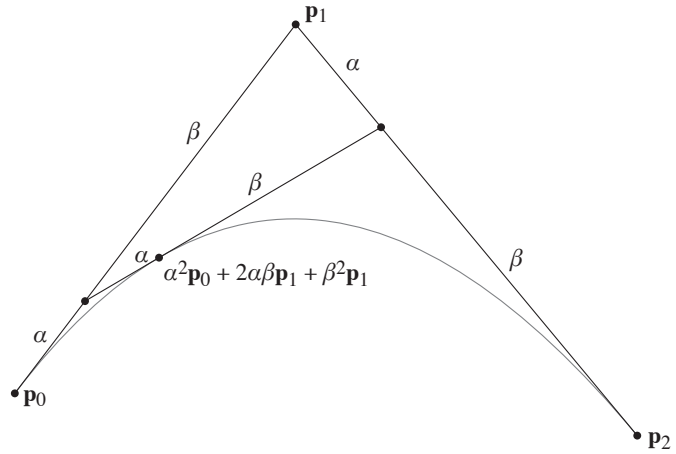
- A piecewise linear spline segment
  - two control points per segment
  - blend them with weights α and $\beta = 1 - \alpha$
- Good for building a piecewise linear spline
  - a.k.a. a polygon or polyline



Another way to Bézier segments

- A linear blend of two piecewise linear segments
  - three control points now
  - interpolate on both segments using α and β
  - blend the results with the same weights
- makes a quadratic spline segment
  - finally, a curve!

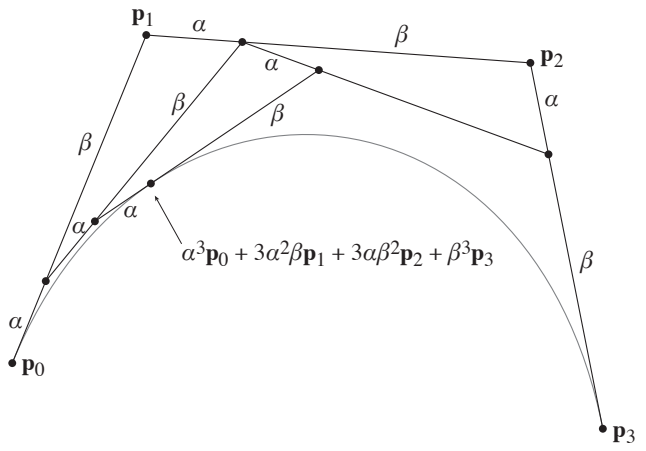
$$\begin{aligned}
\mathbf{p}_{1,0} &= \alpha\mathbf{p}_0 + \beta\mathbf{p}_1 \\
\mathbf{p}_{1,1} &= \alpha\mathbf{p}_1 + \beta\mathbf{p}_2 \\
\mathbf{p}_{2,0} &= \alpha\mathbf{p}_{1,0} + \beta\mathbf{p}_{1,1} \\
&= \alpha\alpha\mathbf{p}_0 + \alpha\beta\mathbf{p}_1 + \beta\alpha\mathbf{p}_1 + \beta\beta\mathbf{p}_2 \\
&= \alpha^2\mathbf{p}_0 + 2\alpha\beta\mathbf{p}_1 + \beta^2\mathbf{p}_2
\end{aligned}$$



Another way to Bézier segments

- Cubic segment: blend of two quadratic segments
  - four control points now (overlapping sets of 3)
  - interpolate on each quadratic using α and β
  - blend the results with the same weights
- makes a cubic spline segment
  - this is the familiar one for graphics—but you can keep going

$$\begin{aligned}
\mathbf{p}_{3,0} =& \alpha\mathbf{p}_{2,0} + \beta\mathbf{p}_{2,1} \\
=& \alpha\alpha\alpha\mathbf{p}_0 + \alpha\alpha\beta\mathbf{p}_1 + \alpha\beta\alpha\mathbf{p}_1 + \alpha\beta\beta\mathbf{p}_2 \\
& \beta\alpha\alpha\mathbf{p}_1 + \beta\alpha\beta\mathbf{p}_2 + \beta\beta\alpha\mathbf{p}_2 + \beta\beta\beta\mathbf{p}_3 \\
=& \alpha^3\mathbf{p}_0 + 3\alpha^2\beta\mathbf{p}_1 + 3\alpha\beta^2\mathbf{p}_2 + \beta^3\mathbf{p}_3
\end{aligned}$$

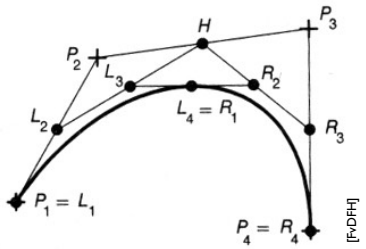


de Casteljau's algorithm

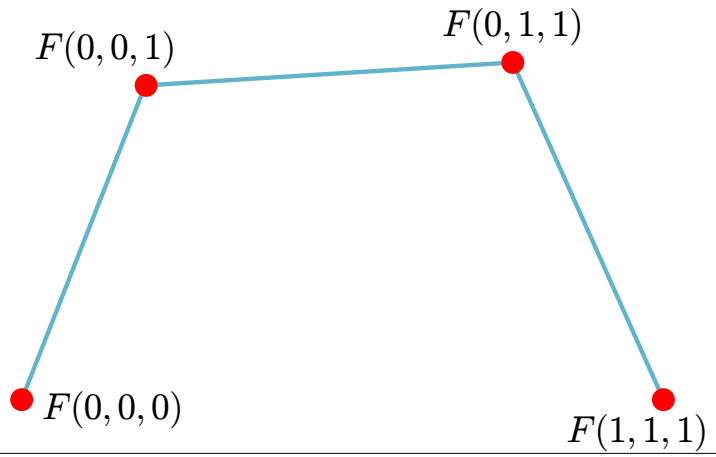
- A recurrence for computing points on Bézier spline segments:

$$\mathbf{p}_{0,i} = \mathbf{p}_i$$
$$\mathbf{p}_{n,i} = \alpha\mathbf{p}_{n-1,i} + \beta\mathbf{p}_{n-1,i+1}$$

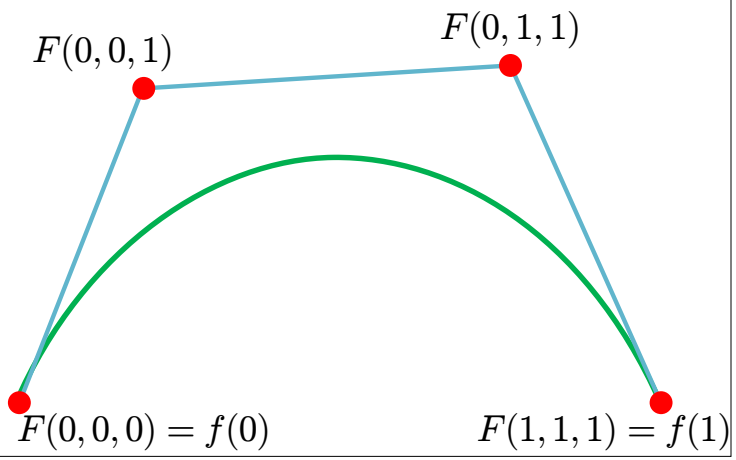
- Cool additional feature: also subdivides the segment into two shorter ones



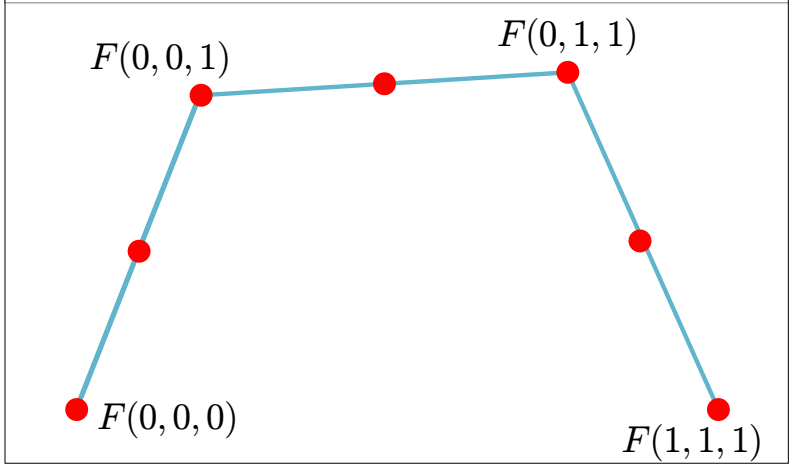
Cubic Control Polygon



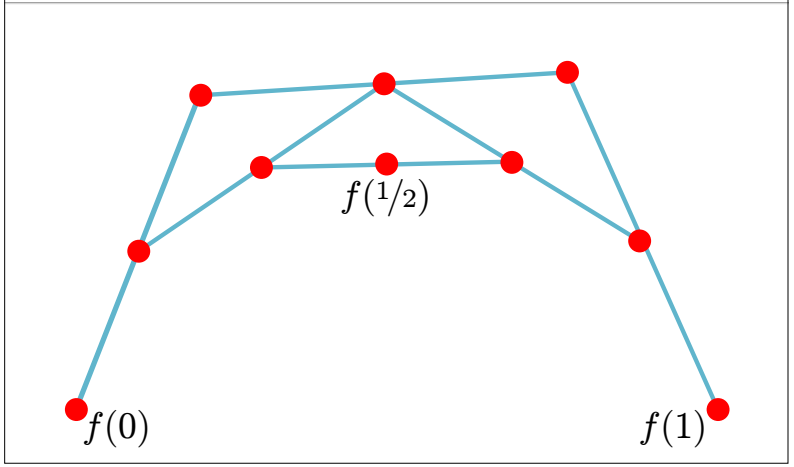
Cubic Control Polygon



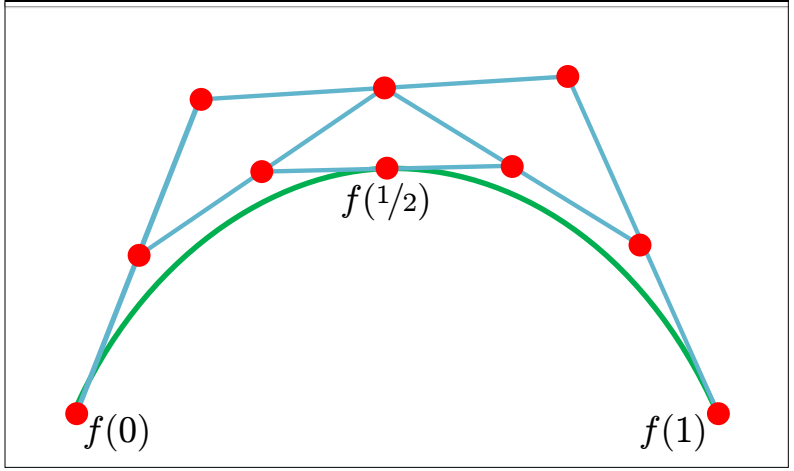
Cubic Control Polygon



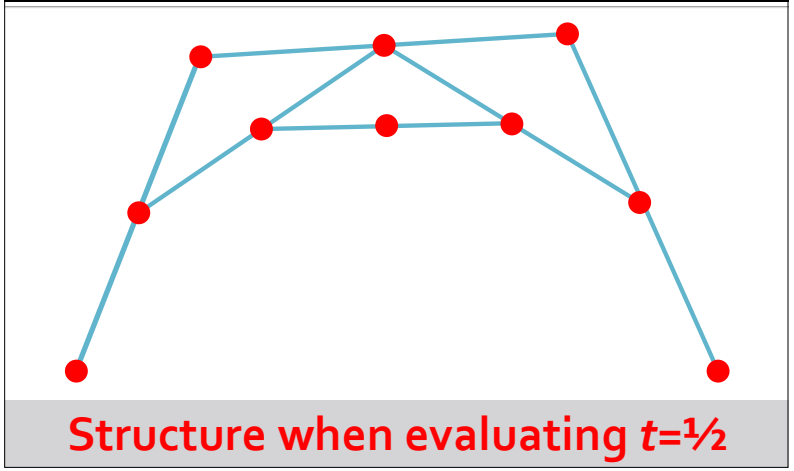
Cubic Control Polygon



Cubic Control Polygon

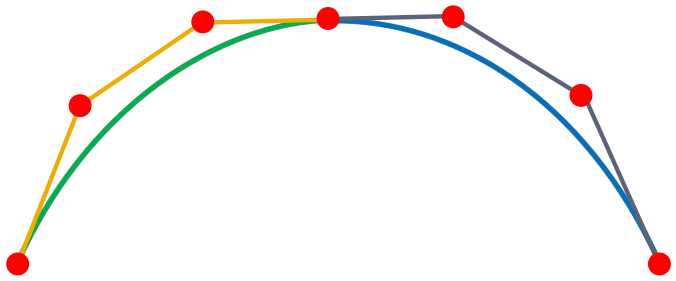


Extra Observation



Structure when evaluating t=½

Extra Observation



Two new control polygons!

Cubic Bézier splines

- Very widely used type, especially in 2D
  - e.g. it is a primitive in PostScript/PDF
- Can represent smooth curves with corners
- Nice de Casteljau recurrence for evaluation
- Can easily add points at any position
- Illustrator demo

Lec27 Required Reading

- FOCG, Ch. 15 (all)

Reminder: Assignment 07

Assigned: Wednesday, Nov. 27
Written Due: Monday, Dec. 9, 4:59:59 pm
Program Due: Wednesday, Dec. 11, 4:59:59 pm