Announcements

- Assignment 2
  - Programming is graded (**Mean: 80%**)
- Assignment 3
  - **Programming** was due
  - **Theory** is due **Friday** (Nov 21st)
- Assignment 4
  - You can start planning

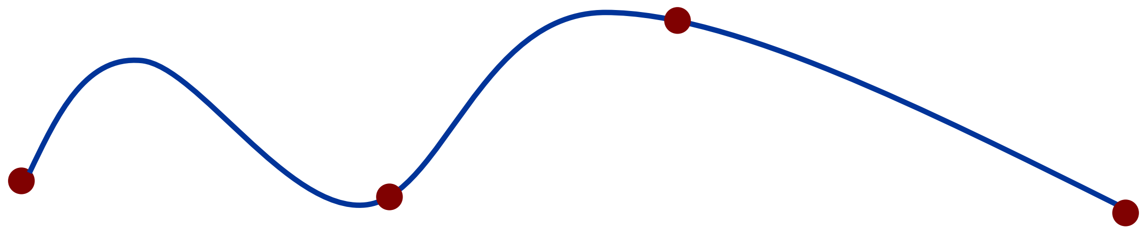
Interpolation, Parametric Curves and Surfaces

Computer Graphics, CSCD18

Fall 2008

Instructor: Leonid Sigal

What is interpolation?



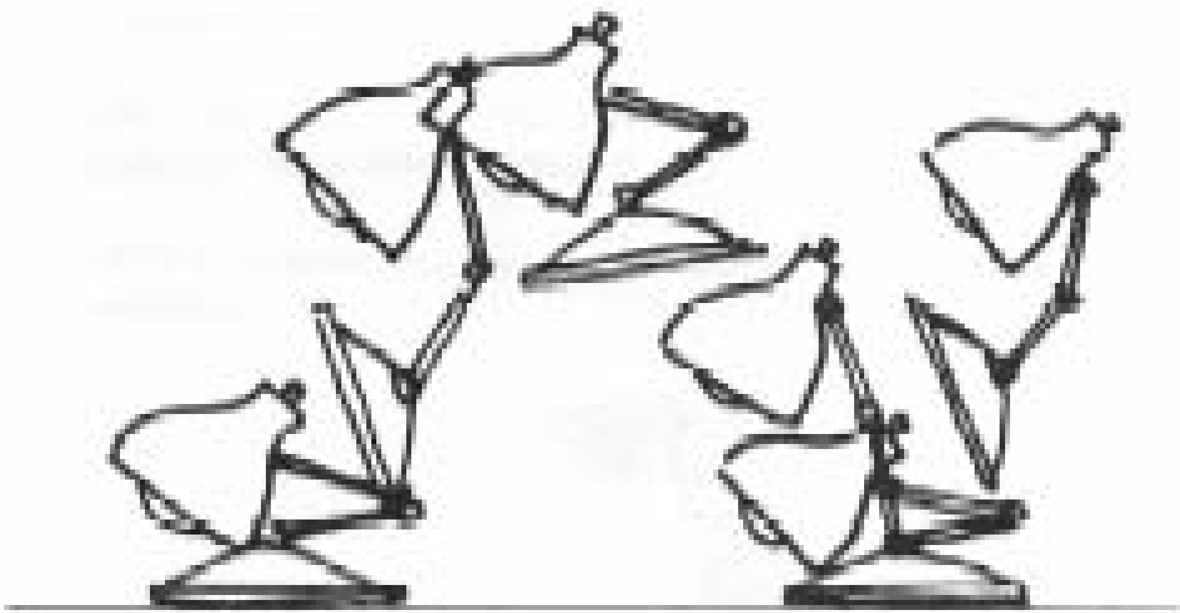
Why do we need interpolation?

- Animation
- Curved surface

Keyframe Animation

- **Idea:** specify variables that describe **keyframes** and interpolate them over the sequence

(e.g. Assignment 1 & 2)



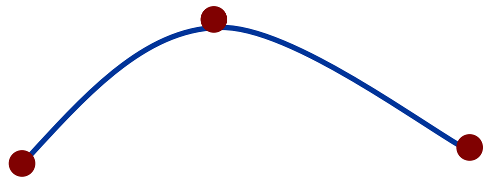
Interpolation Basics

- **Goal:** develop vocabulary of modeling primitives, that can extend meshes or global analytic shapes

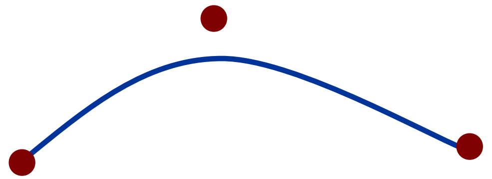
- We would like to define curves that meet the following criteria:
  - Interaction should be natural and intuitive
  - Smoothness should be controllable
  - Analytic derivatives should exist and be easy to compute
  - Adjustable resolution (easy to zoom in and out)
  - Representation should be compact

Curves Basics

- **Interpolation**
  - Curve goes through "control points"



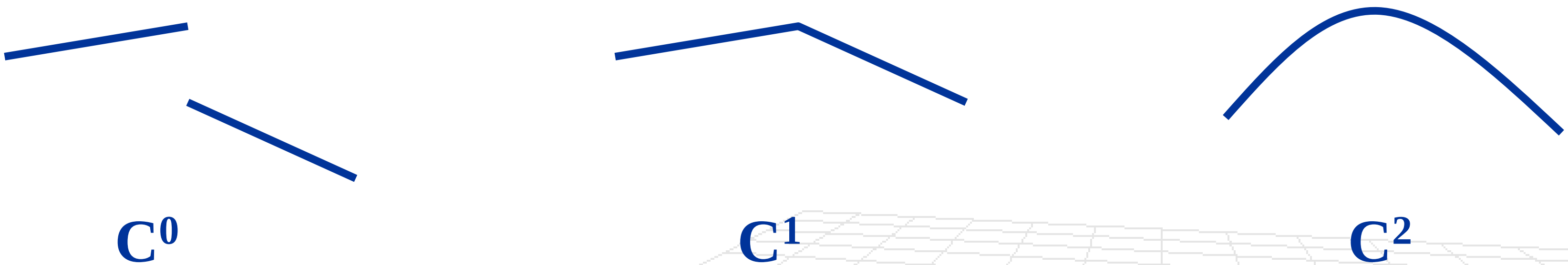
- **Approximation**
  - Curve approximates but does not go through "control points"



- **Extrapolation**
  - Extending curve beyond domain of control points

Continuity

- $\mathbf{C^n}$ continuous function implies that $\mathbf{n}$-th order derivatives exist

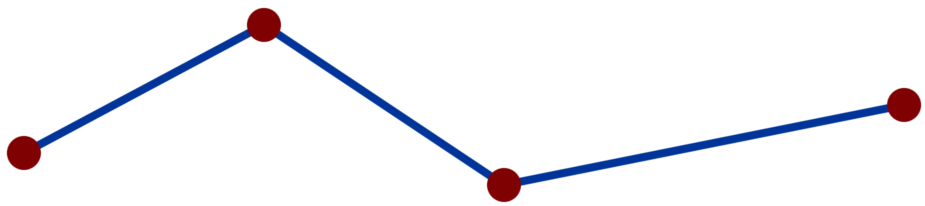


For animation purposes, C^2 continuous functions typically are sufficient

What is the continuity of the n-th order polynomial?

Linear Interpolation

- Simplest possible interpolation technique
  - Peace wise linear curve



- **Pros:**
  - Really simple to implement
  - Local (interpolation only depends on the closest two control points)
- **Cons:**
  - Only $\mathbf{C}^1$ continuous (typically **bad for animation**)

Cubic Interpolation

- Consider a 2D cubic interplant (a curve in 2D)

$$\mathbf{c}(\mathbf{t}) = [\mathbf{x}(\mathbf{t})\ \mathbf{y}(\mathbf{t})]$$

where
$$\mathbf{x}(\mathbf{t}) = \mathbf{a}_0 + \mathbf{a}_1\mathbf{t} + \mathbf{a}_2\mathbf{t}^2 + \mathbf{a}_3\mathbf{t}^3$$
$$\mathbf{y}(\mathbf{t}) = \mathbf{b}_0 + \mathbf{b}_1\mathbf{t} + \mathbf{b}_2\mathbf{t}^2 + \mathbf{b}_3\mathbf{t}^3$$

Alternatively,

$$\mathbf{c}(\mathbf{t}) = \underbrace{[1\quad \mathbf{t}\quad \mathbf{t}^2\quad \mathbf{t}^3]}_{\text{bases}} \underbrace{\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}}_{\text{coefficients}}$$

Cubic Interpolation

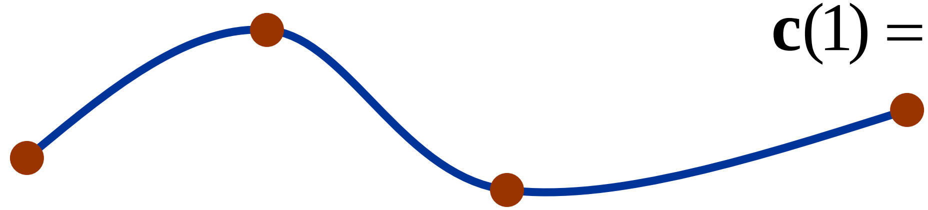
We have **8 unknowns** (coefficients) how many 2D points do we need to constrain the curve?

$$\mathbf{c}(\mathbf{t}) = \underbrace{\begin{bmatrix} 1 & \mathbf{t} & \mathbf{t}^2 & \mathbf{t}^3 \end{bmatrix}}_{\text{bases}} \underbrace{\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}}_{\text{coefficients}}$$

Cubic Interpolation

$$\mathbf{c}\left(\frac{1}{3}\right) = [\mathbf{x}_2, \mathbf{y}_2]$$

$$\mathbf{c}(1) = [\mathbf{x}_4, \mathbf{y}_4]$$



$$\mathbf{c}(0) = [\mathbf{x}_1, \mathbf{y}_1]$$

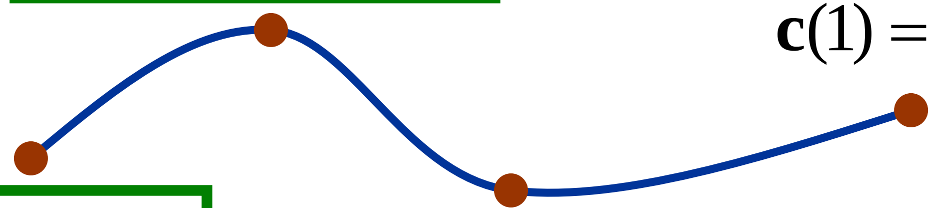
$$\mathbf{c}\left(\frac{2}{3}\right) = [\mathbf{x}_3, \mathbf{y}_3]$$

$$\mathbf{c}(\mathbf{t}) = \underbrace{[1 \quad \mathbf{t} \quad \mathbf{t}^2 \quad \mathbf{t}^3]}_{\text{bases}} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\} \text{coefficients}$$

Cubic Interpolation

$$\mathbf{c}\left(\frac{1}{3}\right) = [\mathbf{x}_2, \mathbf{y}_2]$$

$$\mathbf{c}(1) = [\mathbf{x}_4, \mathbf{y}_4]$$



$$\mathbf{c}(0) = [\mathbf{x}_1, \mathbf{y}_1]$$

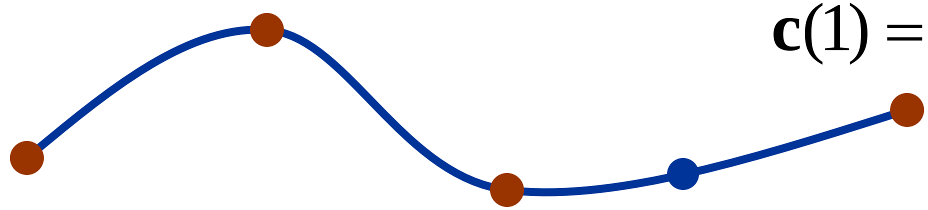
$$\mathbf{c}\left(\frac{2}{3}\right) = [\mathbf{x}_3, \mathbf{y}_3]$$

$$\begin{bmatrix} \mathbf{x}_1 & \mathbf{y}_1 \\ \mathbf{x}_2 & \mathbf{y}_2 \\ \mathbf{x}_3 & \mathbf{y}_3 \\ \mathbf{x}_4 & \mathbf{y}_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right)^2 & \left(\frac{1}{3}\right)^3 \\ 1 & \left(\frac{2}{3}\right) & \left(\frac{2}{3}\right)^2 & \left(\frac{2}{3}\right)^3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\} \text{coefficients}$$

Cubic Interpolation

$$\mathbf{c}\left(\frac{1}{3}\right) = [\mathbf{x}_2, \mathbf{y}_2]$$

$$\mathbf{c}(1) = [\mathbf{x}_4, \mathbf{y}_4]$$



$$\mathbf{c}(0) = [\mathbf{x}_1, \mathbf{y}_1]$$

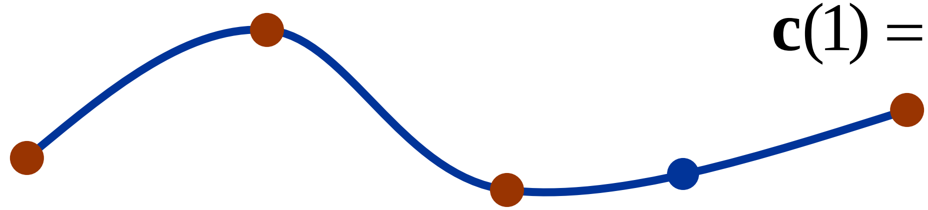
$$\mathbf{c}\left(\frac{2}{3}\right) = [\mathbf{x}_3, \mathbf{y}_3]$$

$$\begin{bmatrix} \mathbf{x}_1 & \mathbf{y}_1 \\ \mathbf{x}_2 & \mathbf{y}_2 \\ \mathbf{x}_3 & \mathbf{y}_3 \\ \mathbf{x}_4 & \mathbf{y}_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right)^2 & \left(\frac{1}{3}\right)^3 \\ 1 & \left(\frac{2}{3}\right) & \left(\frac{2}{3}\right)^2 & \left(\frac{2}{3}\right)^3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\} \text{coefficients}$$

Cubic Interpolation

$$\mathbf{c}\left(\frac{1}{3}\right) = [\mathbf{x}_2, \mathbf{y}_2]$$

$$\mathbf{c}(1) = [\mathbf{x}_4, \mathbf{y}_4]$$



$$\mathbf{c}(0) = [\mathbf{x}_1, \mathbf{y}_1]$$

$$\mathbf{c}\left(\frac{2}{3}\right) = [\mathbf{x}_3, \mathbf{y}_3]$$

$$\mathbf{c}(\mathbf{t}) = \underbrace{[1 \quad \mathbf{t} \quad \mathbf{t}^2 \quad \mathbf{t}^3]}_{\textbf{bases}} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\} \textbf{coefficients}$$

Cubic Interpolation

- Consider a 2D cubic interplant (a curve in 2D)

$$\mathbf{c}(\mathbf{t}) = [\mathbf{x}(\mathbf{t})\ \mathbf{y}(\mathbf{t})]$$

where

$$\mathbf{x}(\mathbf{t}) = \mathbf{a}_0 + \mathbf{a}_1\mathbf{t} + \mathbf{a}_2\mathbf{t}^2 + \mathbf{a}_3\mathbf{t}^3$$

$$\mathbf{y}(\mathbf{t}) = \mathbf{b}_0 + \mathbf{b}_1\mathbf{t} + \mathbf{b}_2\mathbf{t}^2 + \mathbf{b}_3\mathbf{t}^3$$

Alternatively we can place derivative constrains

$$\tau(\mathbf{t}) = \frac{\mathbf{d}\,\mathbf{c}(\mathbf{t})}{\mathbf{dt}} = \underbrace{\frac{\mathbf{d}}{\mathbf{dt}}[1 \quad \mathbf{t} \quad \mathbf{t}^2 \quad \mathbf{t}^3]}_{\text{bases}} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\} \text{coefficients}$$

Cubic Interpolation

- Consider a 2D cubic interplant (a curve in 2D)

$$\mathbf{c}(\mathbf{t}) = [\mathbf{x}(\mathbf{t})\ \mathbf{y}(\mathbf{t})]$$

where
$$\mathbf{x}(\mathbf{t}) = \mathbf{a}_0 + \mathbf{a}_1\mathbf{t} + \mathbf{a}_2\mathbf{t}^2 + \mathbf{a}_3\mathbf{t}^3$$
$$\mathbf{y}(\mathbf{t}) = \mathbf{b}_0 + \mathbf{b}_1\mathbf{t} + \mathbf{b}_2\mathbf{t}^2 + \mathbf{b}_3\mathbf{t}^3$$

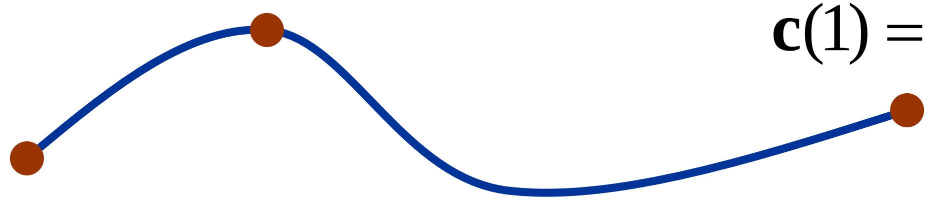
Alternatively we can place derivative constrains

$$\tau(\mathbf{t}) = \frac{\mathbf{d}\,\mathbf{c}(\mathbf{t})}{\mathbf{dt}} = \underbrace{[0 \quad 1 \quad 2\mathbf{t} \quad 3\mathbf{t}^2]}_{\text{different bases}} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\}\text{same coefficients}$$

Cubic Interpolation

$$\mathbf{c}\left(\frac{1}{3}\right)=[\mathbf{x}_2,\mathbf{y}_2] \qquad \frac{\mathbf{dc}}{\mathbf{dt}}\left(\frac{1}{3}\right)=[\mathbf{x}'_2,\mathbf{y}'_2]$$

$$\mathbf{c}(1)=[\mathbf{x}_4,\mathbf{y}_4]$$



$$\mathbf{c}(0)=[\mathbf{x}_1,\mathbf{y}_1]$$

$$\begin{bmatrix} \mathbf{x}_1 & \mathbf{y}_1 \\ \mathbf{x}_2 & \mathbf{y}_2 \\ \mathbf{x}_3 & \mathbf{y}_3 \\ \mathbf{x}'_2 & \mathbf{y}'_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \left(\frac{1}{3}\right) & \left(\frac{1}{3}\right)^2 & \left(\frac{1}{3}\right)^3 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2\left(\frac{1}{3}\right) & 3\left(\frac{1}{3}\right)^2 \end{bmatrix} \left.\begin{bmatrix} \mathbf{a}_0 & \mathbf{b}_0 \\ \mathbf{a}_1 & \mathbf{b}_1 \\ \mathbf{a}_2 & \mathbf{b}_2 \\ \mathbf{a}_3 & \mathbf{b}_3 \end{bmatrix}\right\} \textbf{coefficients}$$

Cubic Interpolation

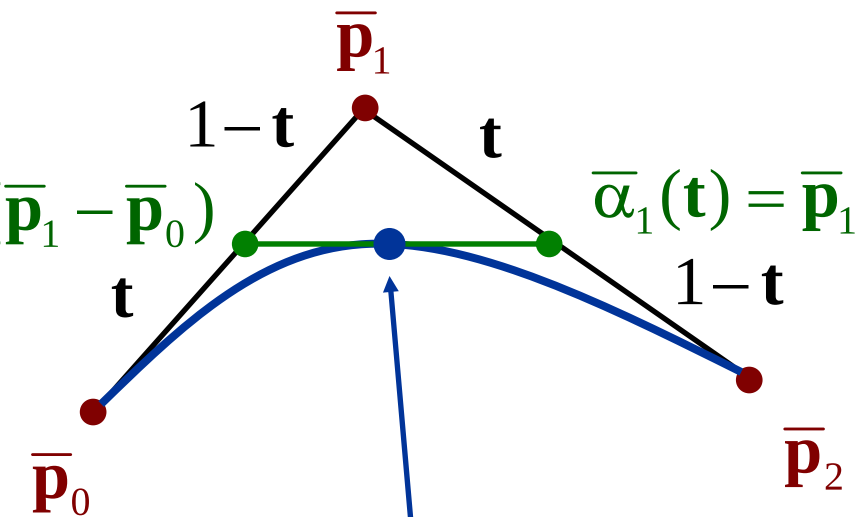
- What happens if there are more then 4 points?
  - There may **not be a solution** that goes through all the control points (or any of the control points)
  - Interpolation may not result in intuitive results

- Cubic interpolation is **global**
  - Changing one control point changes the interpolation for all points

- In general (at least for animation) local control is better

Bezier Curves

- **Idea:** cascade of linear interpolations



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

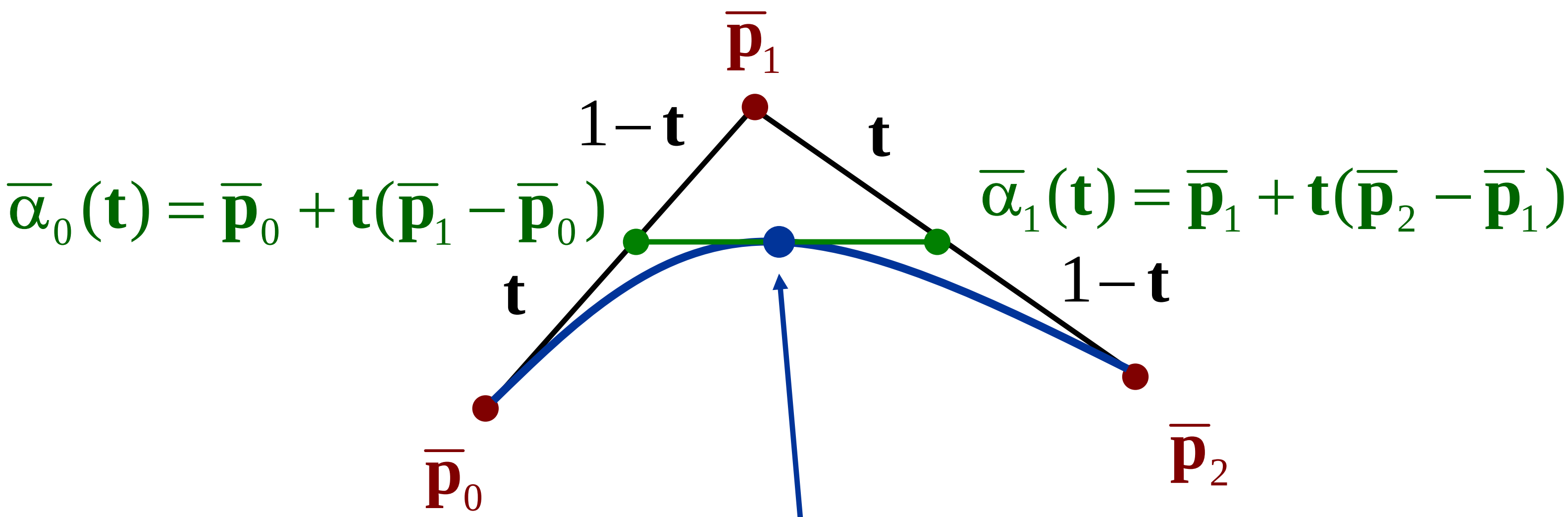
$$\bar{\alpha}_1(\mathbf{t}) = \bar{\mathbf{p}}_1 + \mathbf{t}(\bar{\mathbf{p}}_2 - \bar{\mathbf{p}}_1)$$

$$\mathbf{c}(\mathbf{t}) = \bar{\alpha}_0(\mathbf{t}) + \mathbf{t}\left(\bar{\alpha}_1(\mathbf{t}) - \bar{\alpha}_0(\mathbf{t})\right)$$

If we plug in all the expressions into $\mathbf{c}(\mathbf{t})$ we get a polynomial in terms of control points

Bezier Curves

- **Idea:** cascade of linear interpolations



$$\mathbf{c}(\mathbf{t}) = \overline{\mathbf{p}}_0(1-\mathbf{t})^2 + 2\overline{\mathbf{p}}_1(1-\mathbf{t})\mathbf{t} + \overline{\mathbf{p}}_2\mathbf{t}^2$$

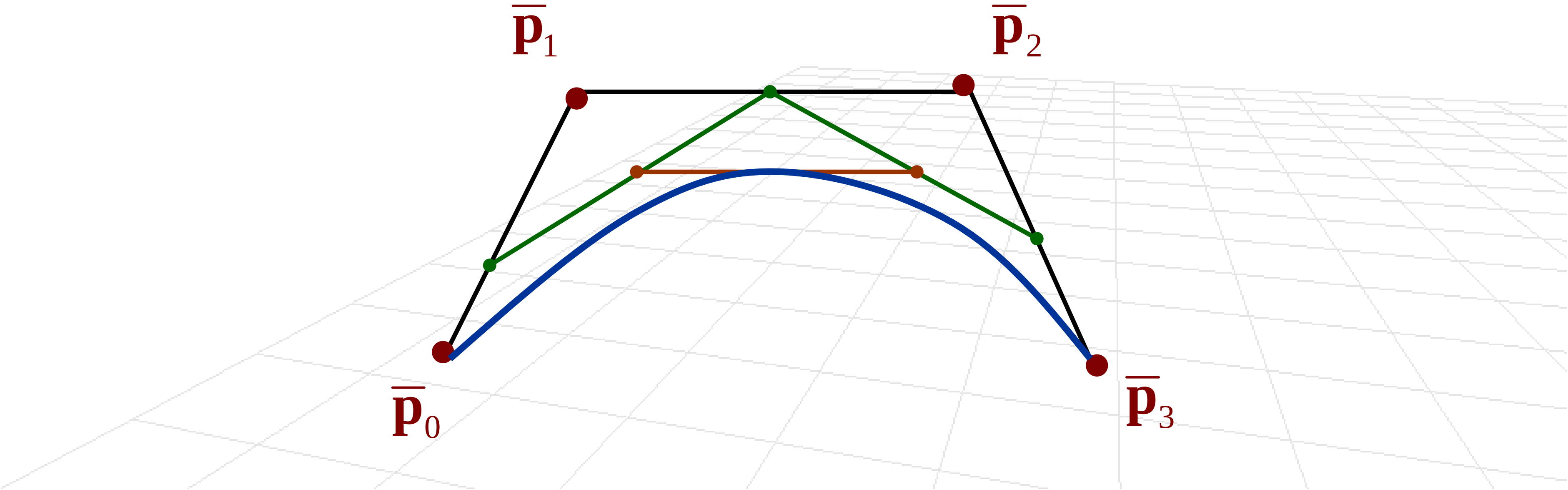
$$= \sum_{\mathbf{i}=0}^{2} \overline{\mathbf{p}}_{\mathbf{i}} \mathbf{B}_{\mathbf{i}}^{2}(\mathbf{t})$$

i-th **Bernstein polynomial** of degree 2

Bezier Curves Generalization

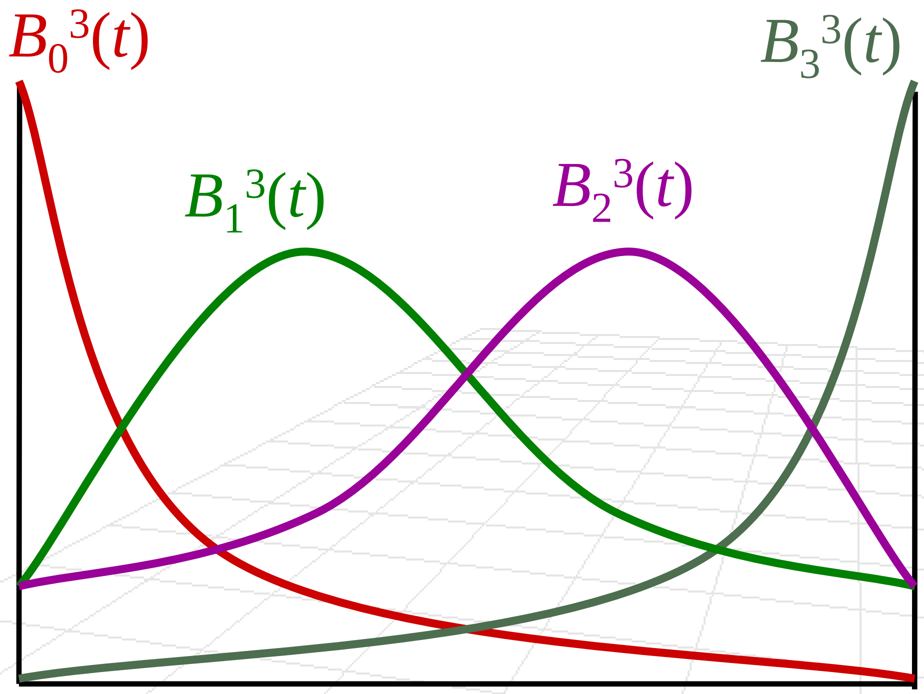
Generalization to N+1 points $\mathbf{c}(\mathbf{t}) = \sum_{\mathbf{i}=0}^{\mathbf{N}} \overline{\mathbf{p}}_{\mathbf{i}} \mathbf{B}_{\mathbf{i}}^{\mathbf{N}}(\mathbf{t})$

$$\mathbf{B}_{\mathbf{i}}^{\mathbf{N}}(\mathbf{t}) = \binom{\mathbf{N}}{\mathbf{i}} (1-\mathbf{t})^{\mathbf{N}-\mathbf{i}} \mathbf{t}^{\mathbf{i}} = \frac{\mathbf{N}!}{(\mathbf{N}-\mathbf{i})!\mathbf{i}!} (1-\mathbf{t})^{\mathbf{N}-\mathbf{i}} \mathbf{t}^{\mathbf{i}}$$



Bernstein Polynomials of Degree 3

- **Note:** Bezier curve with 4 points will be a combination of these curves.



Bezier Curves Properties

- Bezier curve interpolates between the first and the last point, but not the intermediate points

- Bezier curves have **nice properties** that make them useful in graphics
  - **Affine invariance**: affine transformation of the curve implies transformation of control points (nothing else)
  - **Convex hall property**: any point on a curve is by definition a convex combination of the control points, hence the curve must be inside the (convex) polygon defined by those points
  - **Linear precision:** as convex polygon approximates the line, so will the curve
  - **Variation Diminishing:** No line has more intersections with the curve than with control points (no accessive fluctuations)

Derivatives of Bezier Curves

$$\mathbf{c}(\mathbf{t}) = \sum_{\mathbf{i}=0}^{\mathbf{N}} \overline{\mathbf{p}}_{\mathbf{i}} \mathbf{B}_{\mathbf{i}}^{\mathbf{N}}(\mathbf{t}) \qquad \mathbf{B}_{\mathbf{i}}^{\mathbf{N}}(\mathbf{t}) = \binom{\mathbf{N}}{\mathbf{i}} (1-\mathbf{t})^{\mathbf{N}-\mathbf{i}} \mathbf{t}^{\mathbf{i}} = \frac{\mathbf{N}!}{(\mathbf{N}-\mathbf{i})!\,\mathbf{i}!} (1-\mathbf{t})^{\mathbf{N}-\mathbf{i}} \mathbf{t}^{\mathbf{i}}$$

convex sum of points, hence is a point

- We want to differentiate with respect to **t**

$$\tau(\mathbf{t}) = \frac{\mathbf{d}}{\mathbf{dt}} \mathbf{c}(\mathbf{t}) = \frac{\mathbf{d}}{\mathbf{dt}} \sum_{\mathbf{i}=0}^{\mathbf{N}} \overline{\mathbf{p}}_{\mathbf{i}} \mathbf{B}_{\mathbf{i}}^{\mathbf{N}}(\mathbf{t})$$

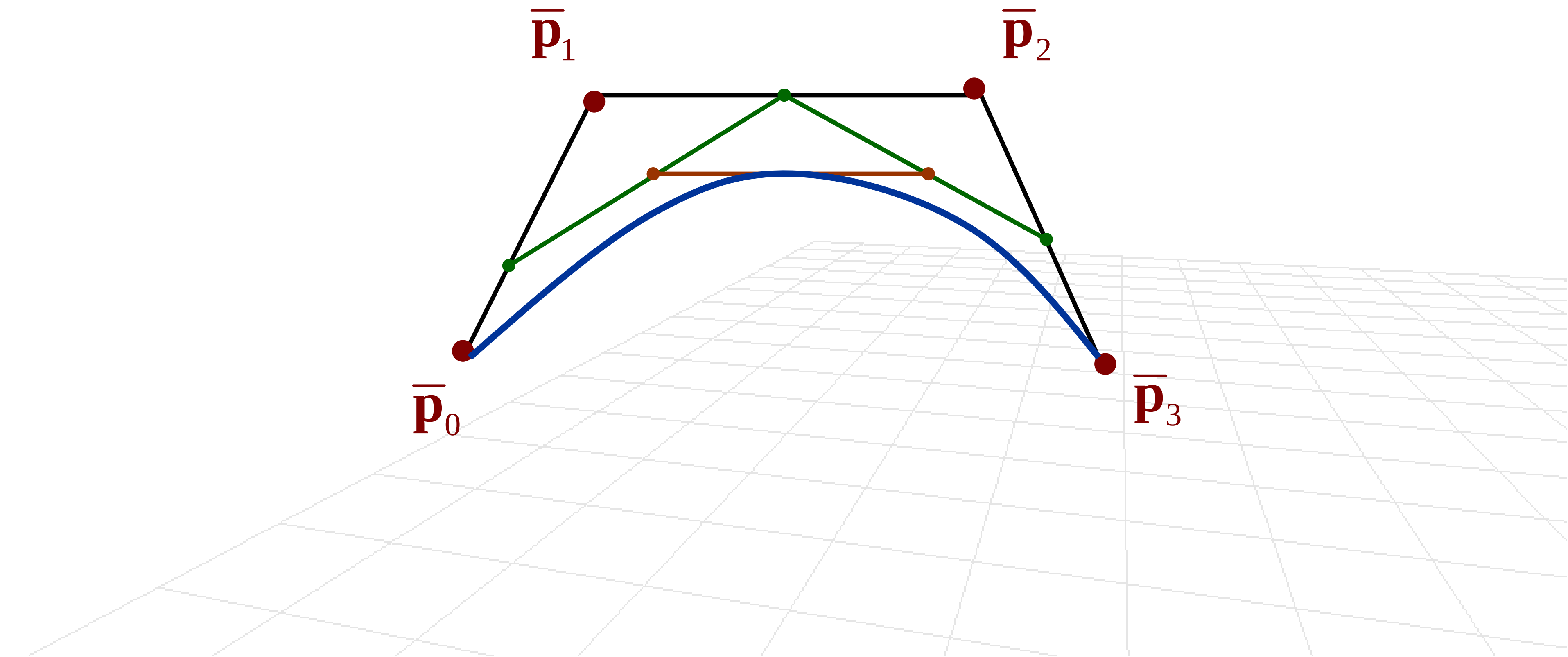
with some work

$$= \sum_{\mathbf{i}=0}^{\mathbf{N}-1} (\overline{\mathbf{p}}_{\mathbf{i}+1} - \overline{\mathbf{p}}_{\mathbf{i}}) \mathbf{B}_{\mathbf{i}}^{\mathbf{N}-1}(\mathbf{t})$$

convex sum of vectors, hence is a vector

Derivatives of Bezier Curves

Property: tangents at the end points of a Bezier curve are always parallel to vector from the end point to the adjacent point



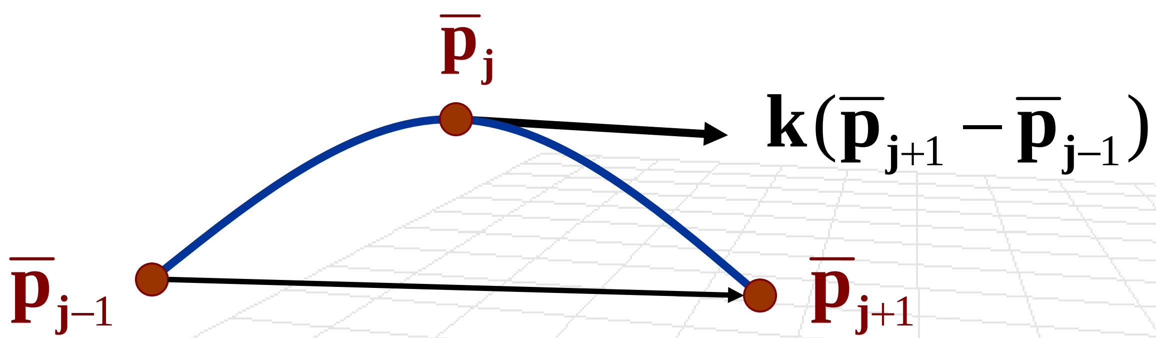
Final word on Bezier curves

- **Pros:**
  - Has nice properties (e.g. affine invariance)
  - Derivatives are easy to compute
- **Cons:**
  - Tough to control a high-order polynomial
  - Global (curve is a function of all control points)

Catmull-Rom Splines

- **Idea:** piecewise cubic curves of degree-3 with $\mathbf{C}^1$ continuity

- A user specifies points and the tangent at each point is set to be parallel to the vector between adjacent points



- $\mathbf{k}$ is the set by the user parameter, that determines the "tension" of the curve

Catmull-Rom Splines

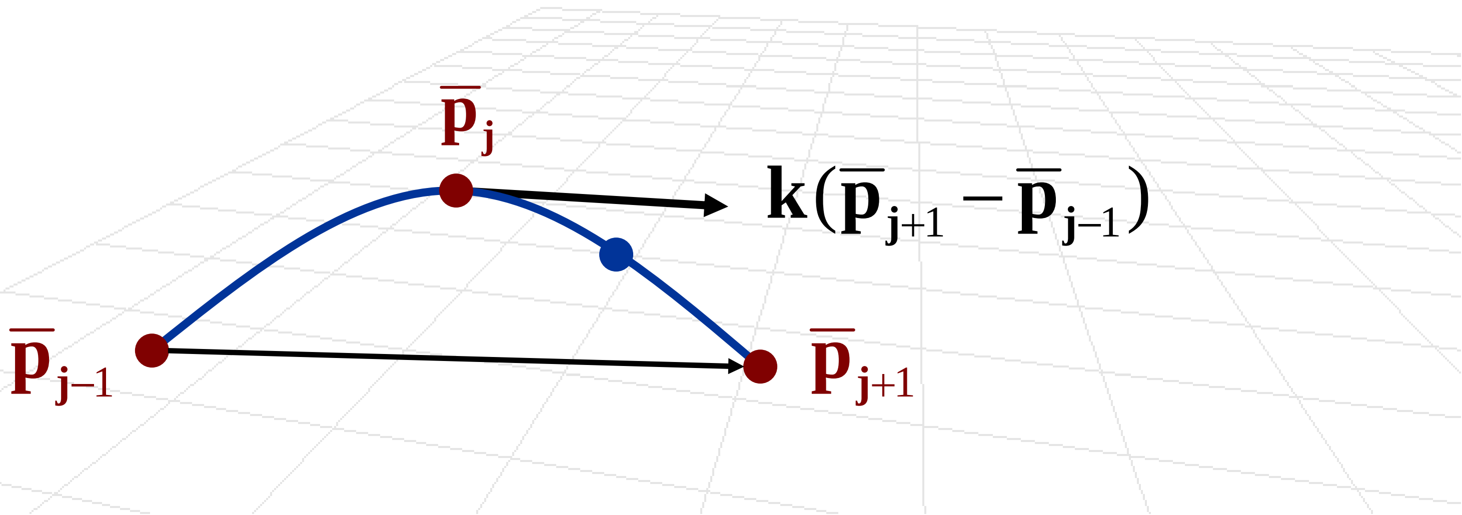
- To interpolate a value for the point between $\mathbf{p_j}$ and $\mathbf{p_{j+1}}$ one needs to consider 4 bits of information

$$\overline{\mathbf{p}}_{\mathbf{j}}$$

$$\overline{\mathbf{p}}_{\mathbf{j}+1}$$

$$\mathbf{k}(\overline{\mathbf{p}}_{\mathbf{j}+1}-\overline{\mathbf{p}}_{\mathbf{j}-1})$$

$$\mathbf{k}(\overline{\mathbf{p}}_{\mathbf{j}+2}-\overline{\mathbf{p}}_{\mathbf{j}})$$



Catmull-Rom Splines

- To interpolate a value for the point between $\mathbf{p_j}$ and $\mathbf{p_{j+1}}$ one needs to consider 4 bits of information

$$\overline{\mathbf{p}}_j$$

$$\overline{\mathbf{p}}_{j+1}$$

$$\mathbf{k}(\overline{\mathbf{p}}_{j+1} - \overline{\mathbf{p}}_{j-1})$$

$$\mathbf{k}(\overline{\mathbf{p}}_{j+2} - \overline{\mathbf{p}}_j)$$

4 points lead to cubic interplant (see lecture notes for details)