Course Roadmap

Rasterization Pipeline

Core Concepts

- Sampling
- Antialiasing
- Transforms

Intro

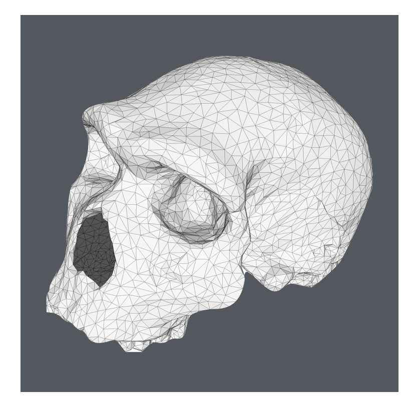
Rasterization

Transforms & Projection

Texture Mapping

Visibility, Shading, Overall Pipeline

Geometric Modeling



Starting today

Lighting & Materials

Cameras & Imaging

Lecture 7:

Introduction to Geometry, Splines & Bézier Curves

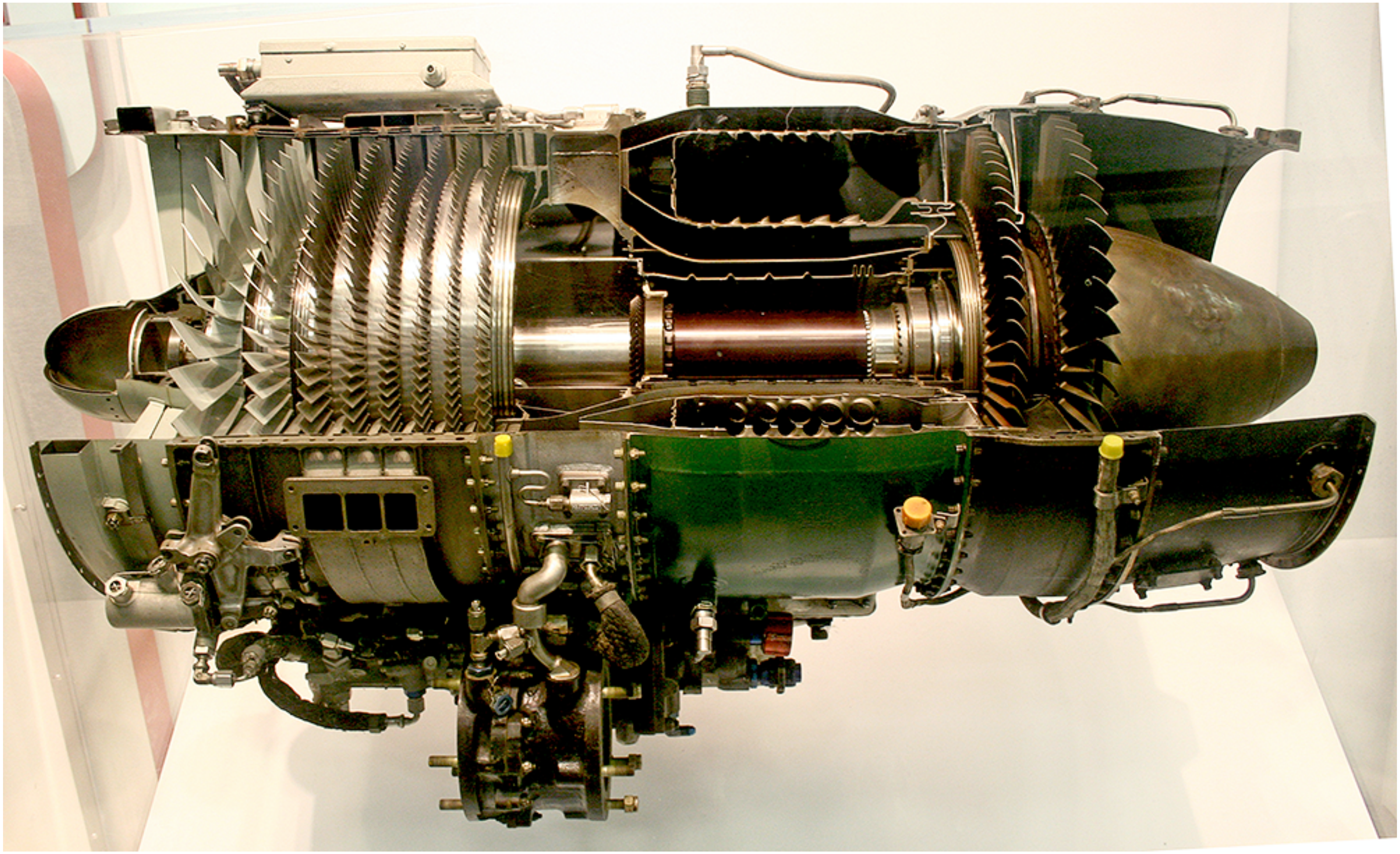
Computer Graphics and Imaging

UC Berkeley CS184/284A

Examples of Geometry

Examples of Geometry

Examples of Geometry



Examples of Geometry

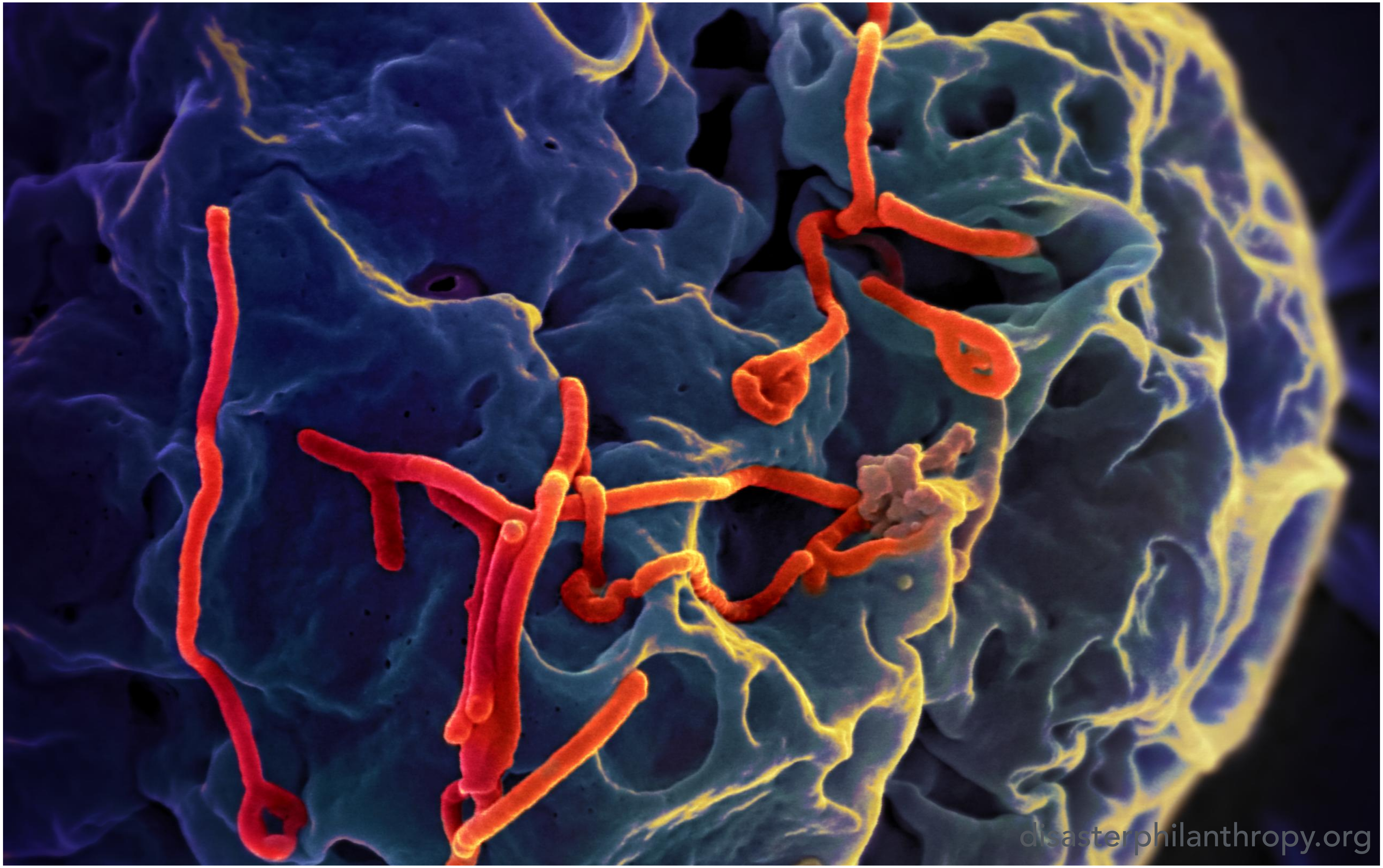
Examples of Geometry

Examples of Geometry

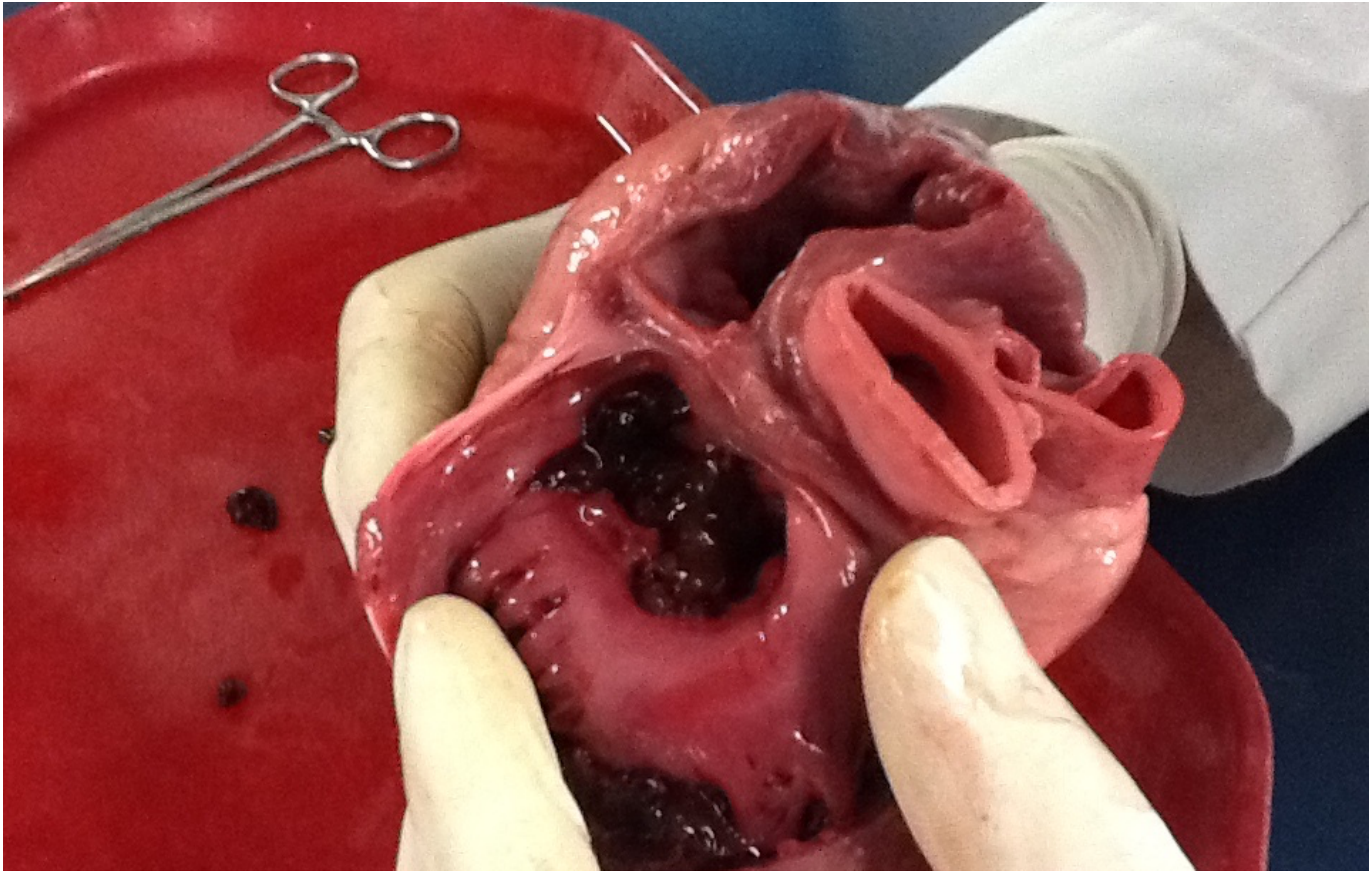
Examples of Geometry

Examples of Geometry

Examples of Geometry



Examples of Geometry



Examples of Geometry

Adriana Franco, National Geographic

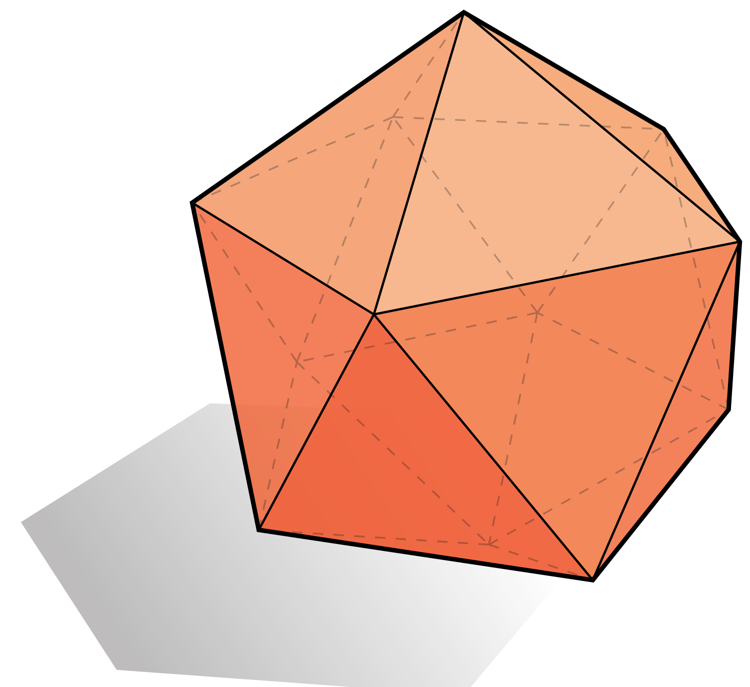
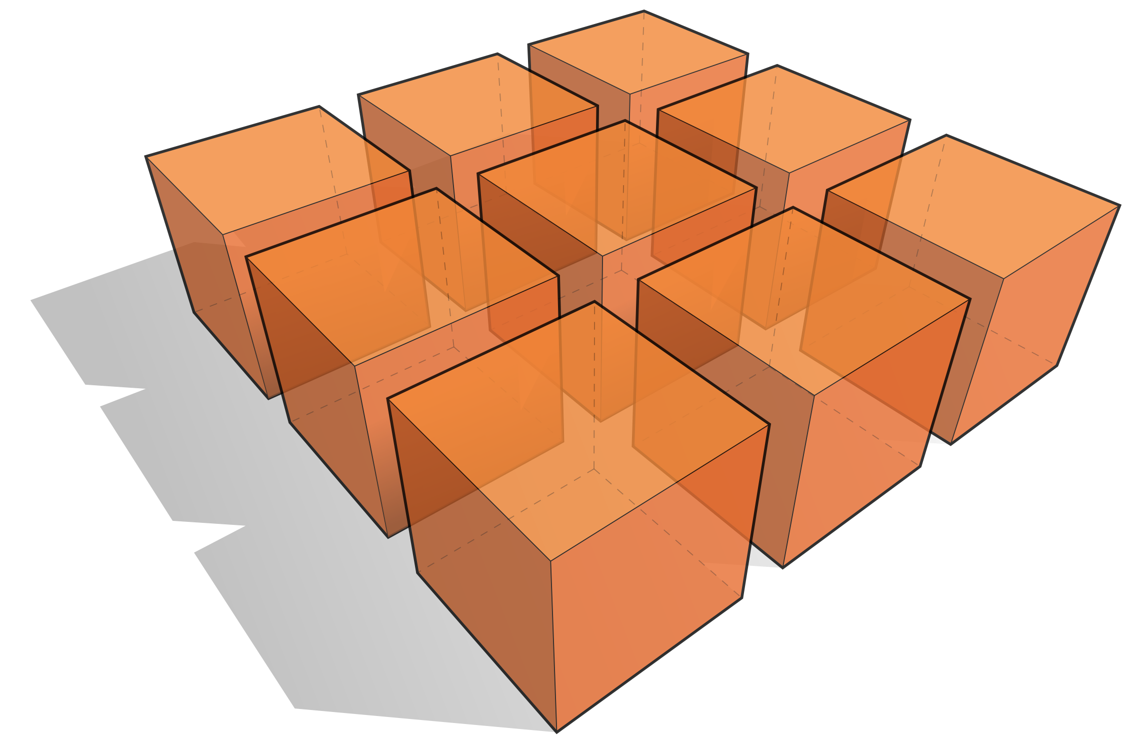
No “Best” Representation – Geometry is Hard!

“I hate meshes.
I cannot believe how hard this is.
Geometry is hard.”

— David Baraff
Senior Research Scientist
Pixar Animation Studios

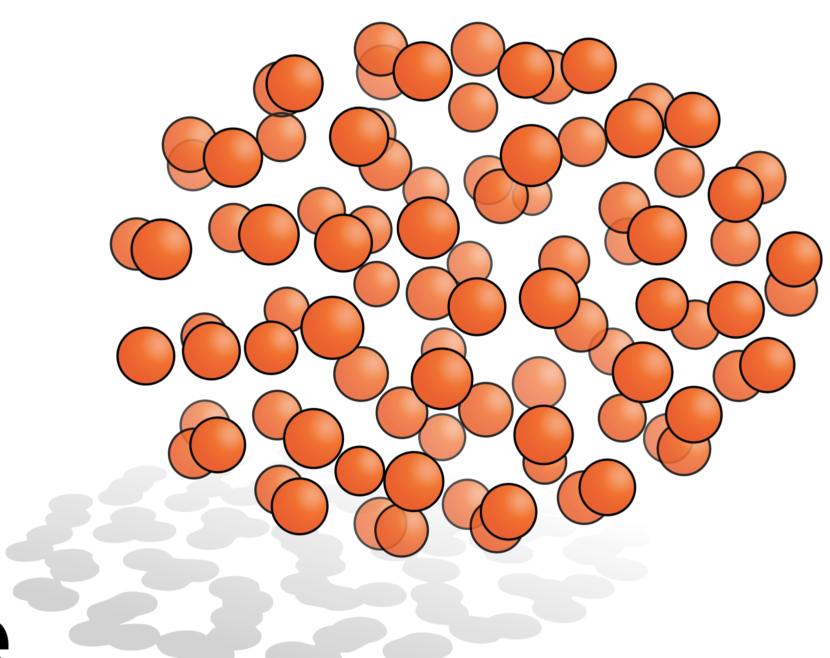
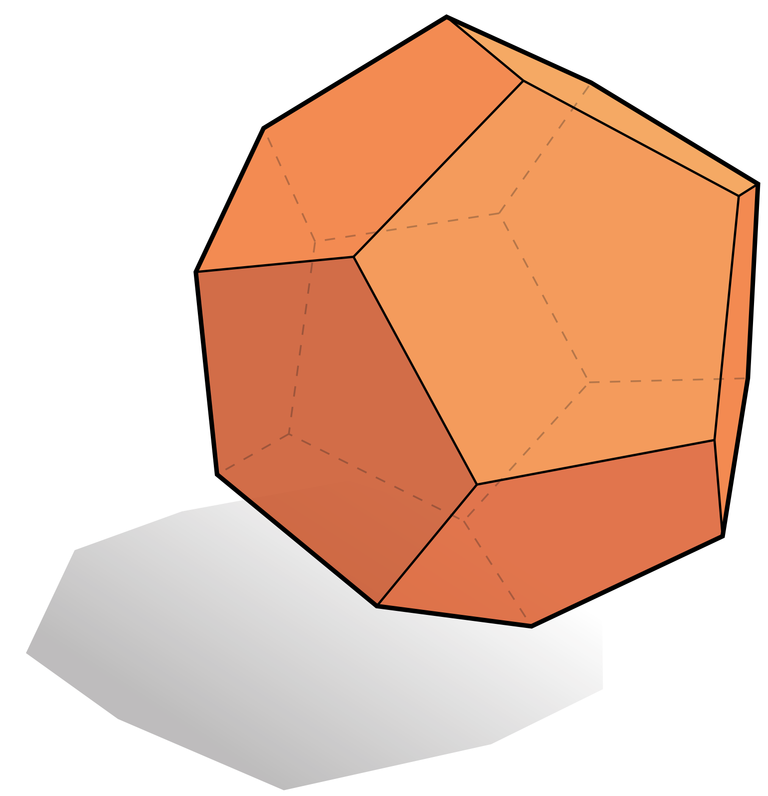
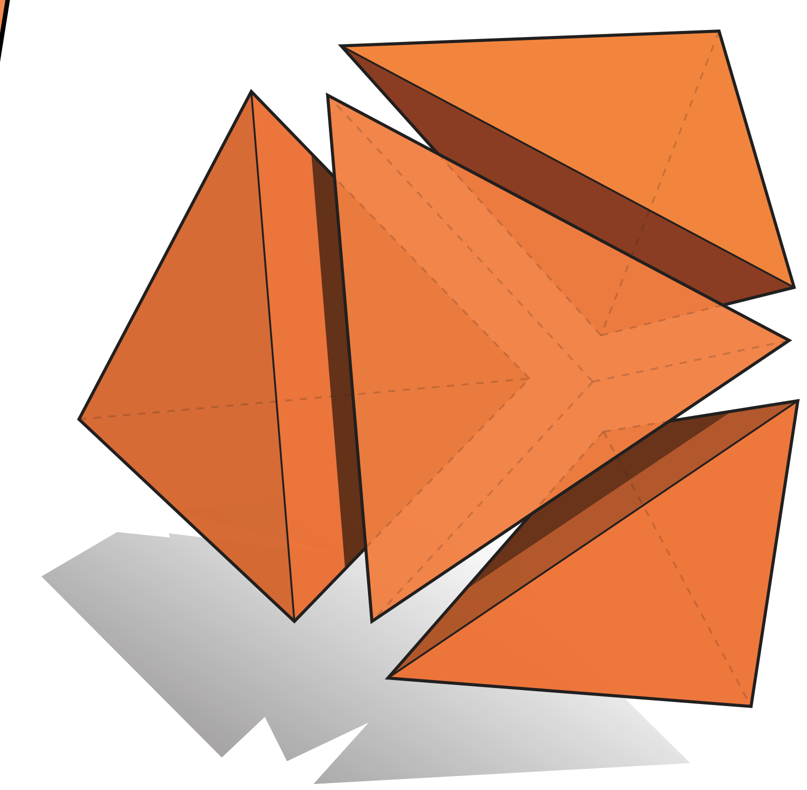
Slide cribbed from Keenan Crane, cribbed from Jeff Erickson.

Many Ways to Represent Geometry

Explicit

- point cloud
- polygon mesh
- subdivision, NURBS
- ...

Implicit

- level sets
- algebraic surface
- distance functions
- ...

Each choice best suited to a different task/type of geometry

Jonathan Ragan-Kelley & Ren Ng

Smooth Curves

Smooth Curves and Surfaces

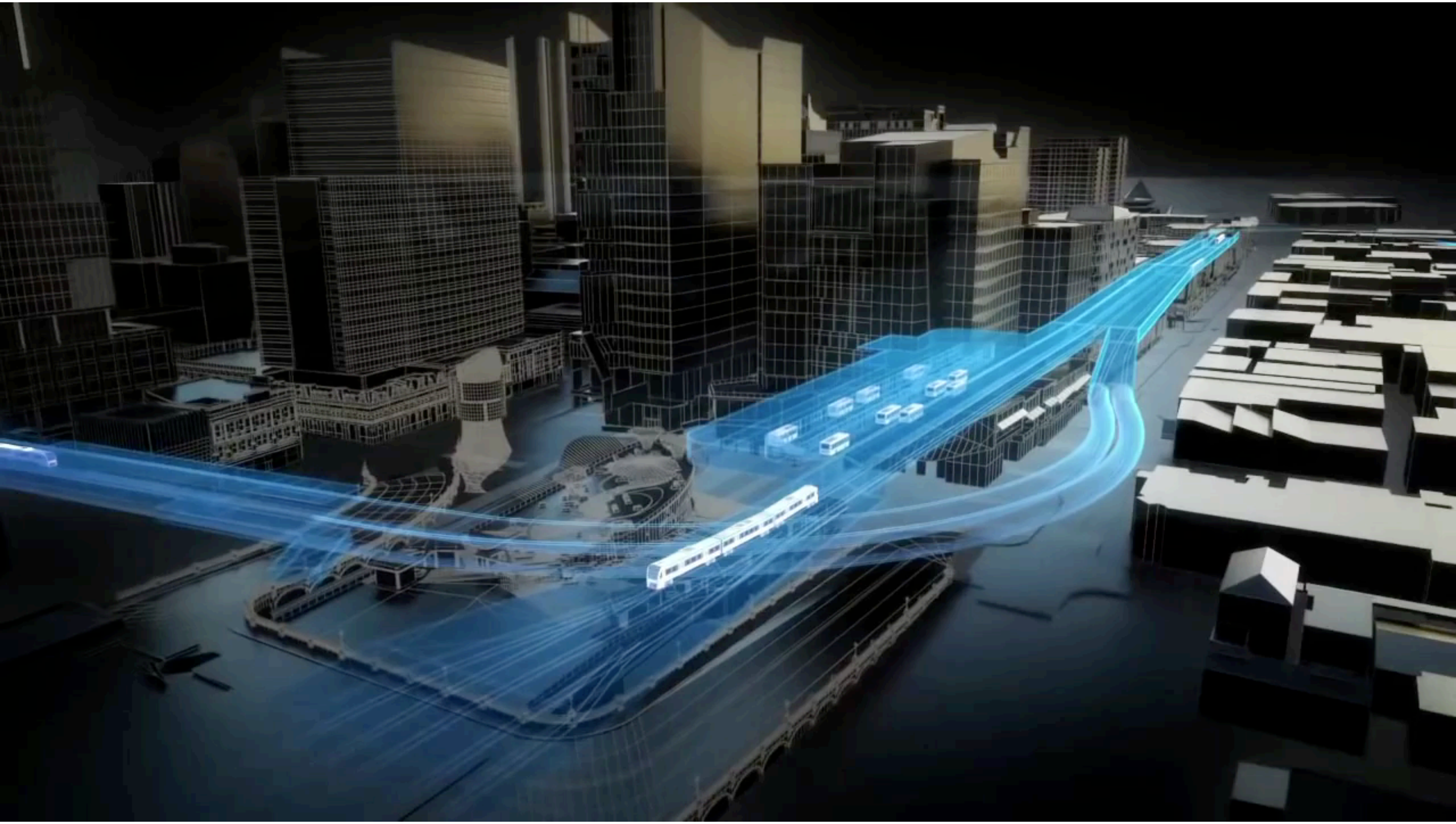
So far we can make:

- Things with corners (lines, triangles, squares, …)

But many applications require designed, smooth shapes

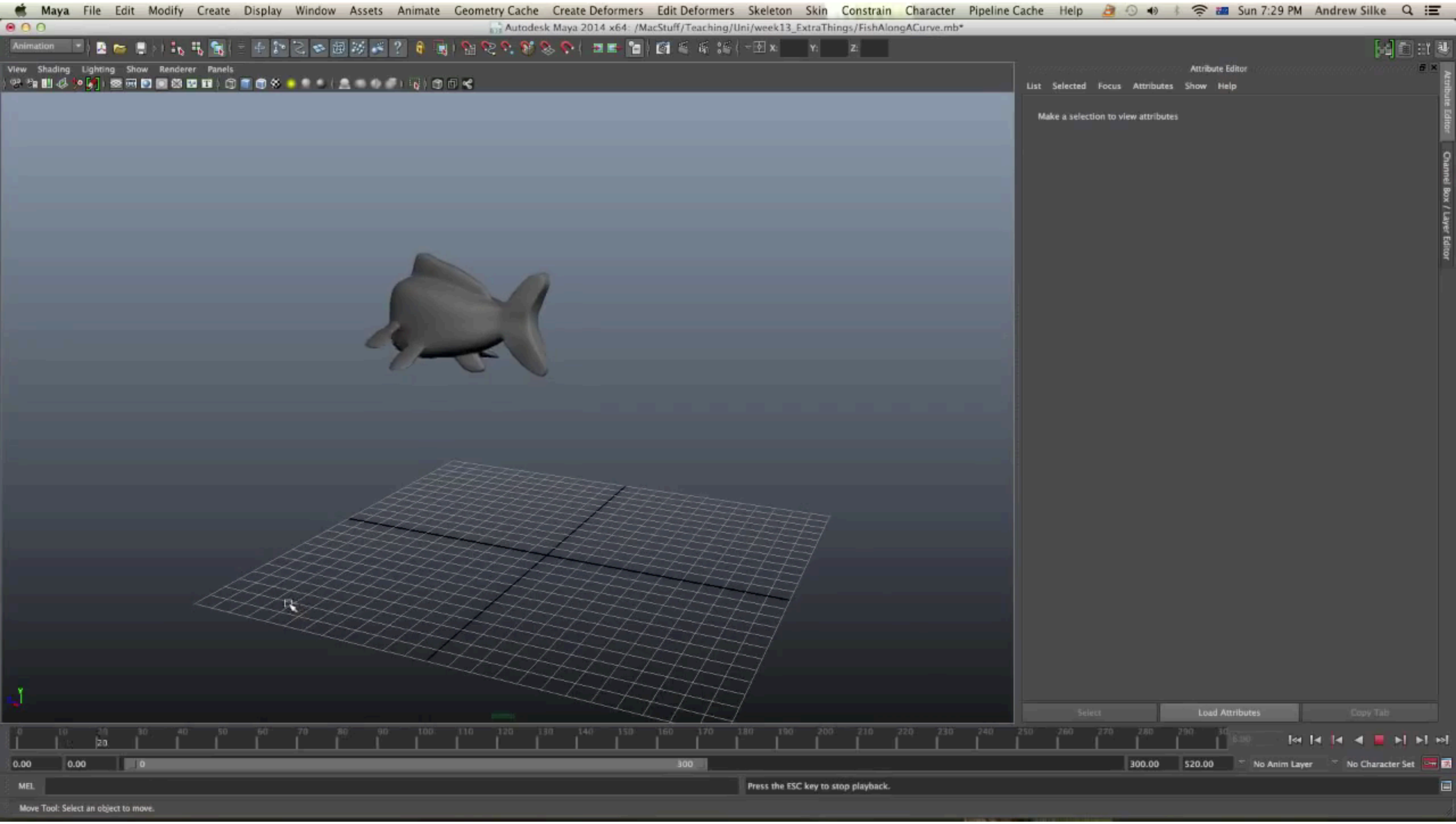
- Camera paths, vector fonts, …
- Resampling filter functions
- CAD design, object modeling, …

Camera Paths



Flythrough of proposed Perth Citylink subway, https://youtu.be/rIJMuQPwr3E

Animation Curves

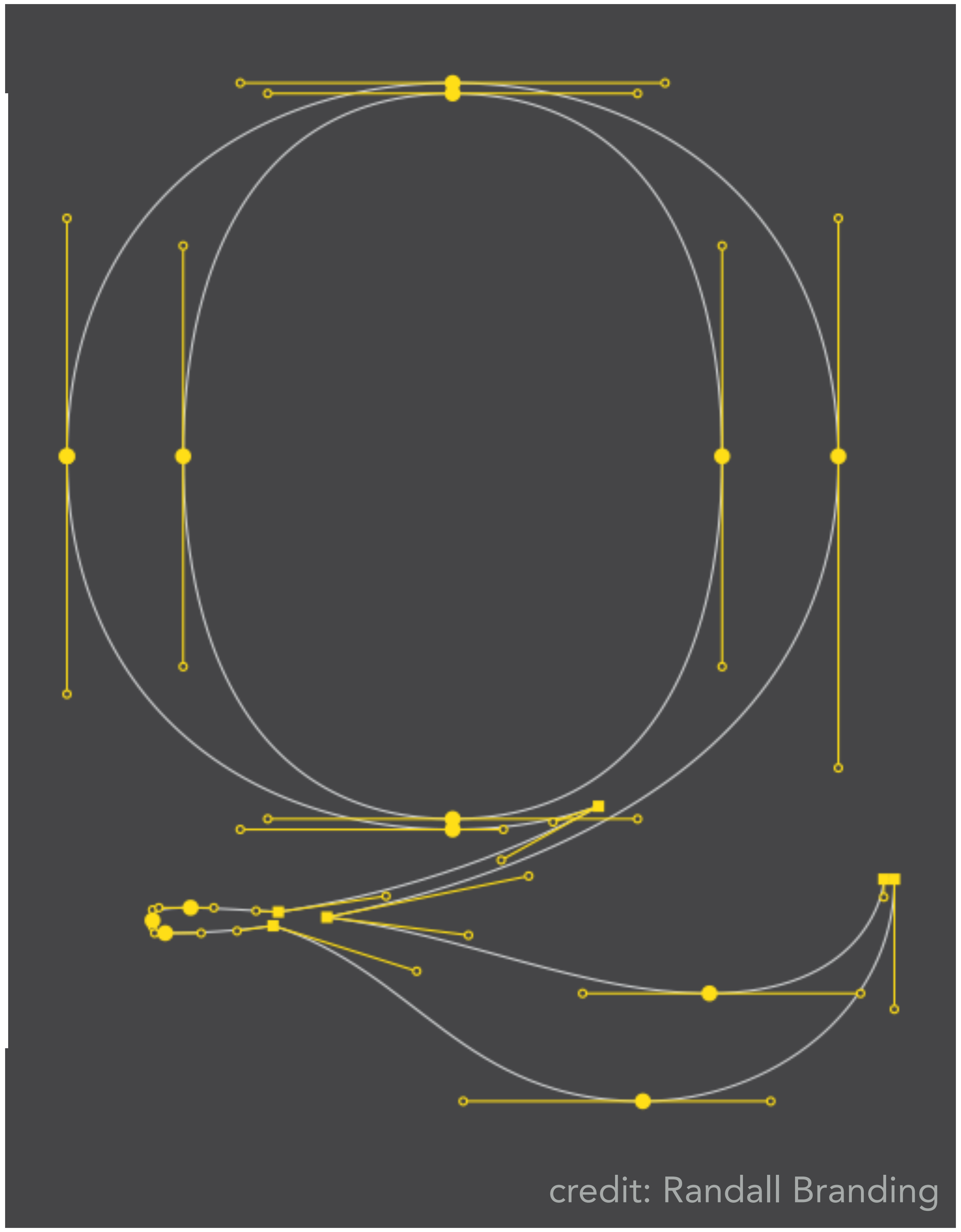


Maya Animation Tutorial: https://youtu.be/b-o5wtZlJPc

Vector Fonts

The Quick Brown Fox Jumps Over The Lazy Dog

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz 0123456789



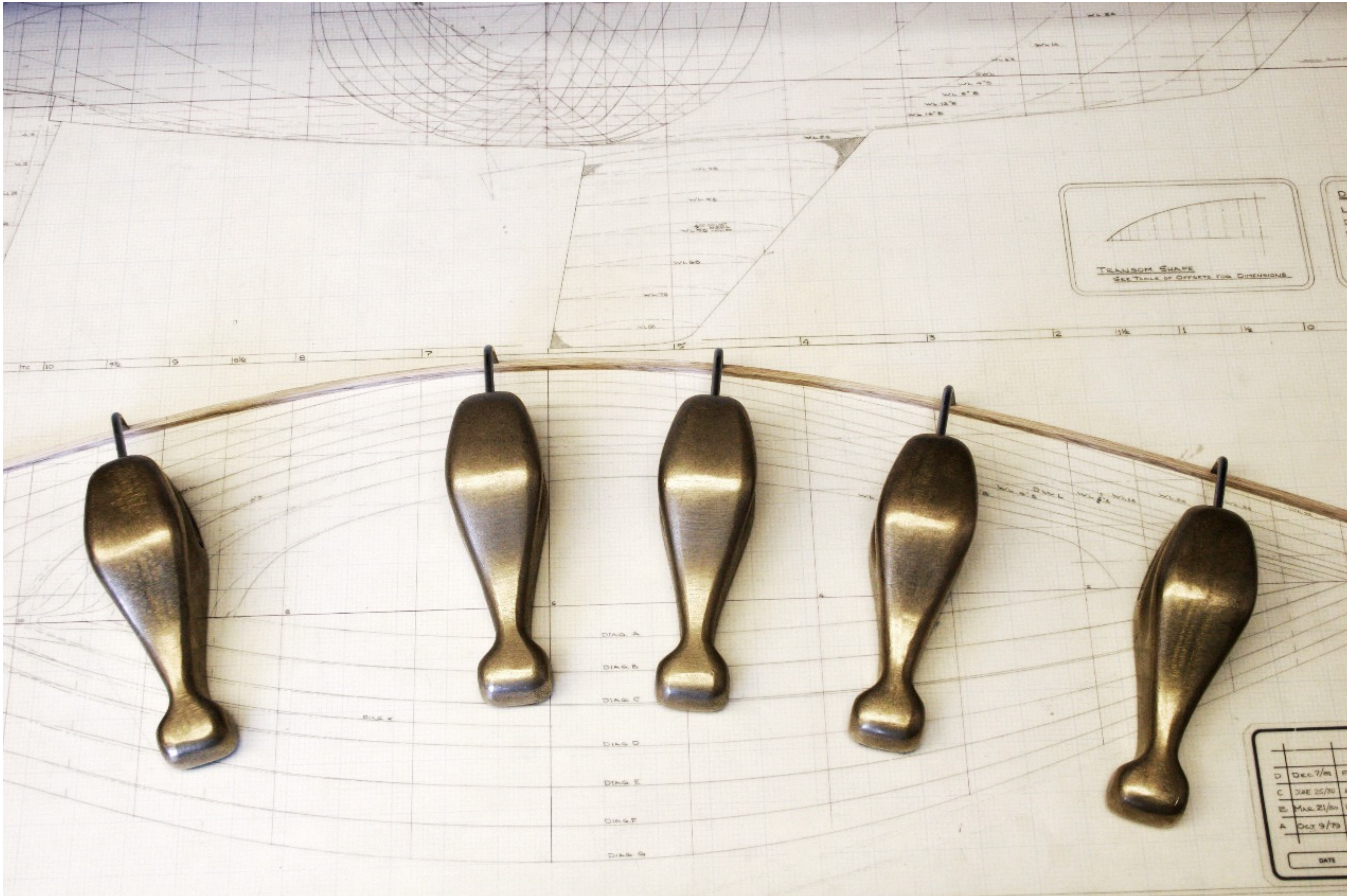
Baskerville font - represented as cubic Bézier splines

CAD Design

3D Car Modeling with Rhinoceros

Splines

A Real Draftsman's Spline



http://www.alatown.com/spline-history-architecture/

Spline Topics

Interpolation

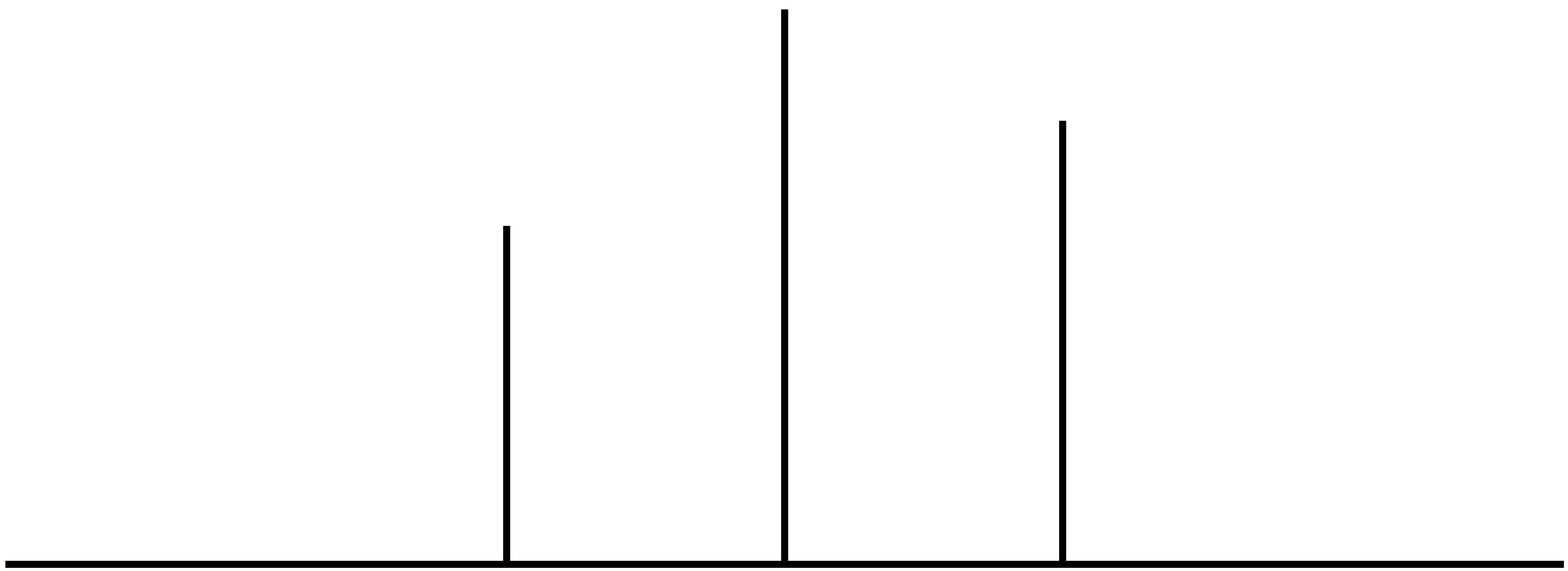
- Cubic Hermite interpolation
- Catmull-Rom interpolation

Bezier curves

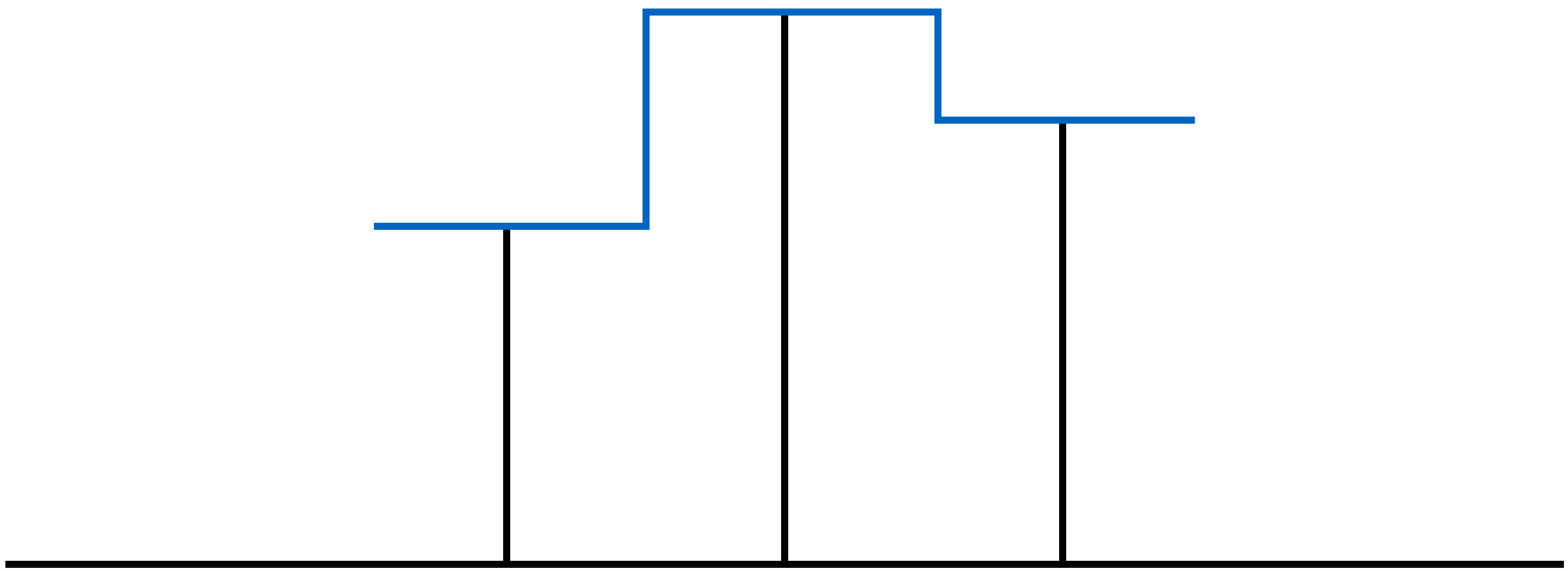
Bezier surfaces

Cubic Hermite Interpolation

Goal: Interpolate Values

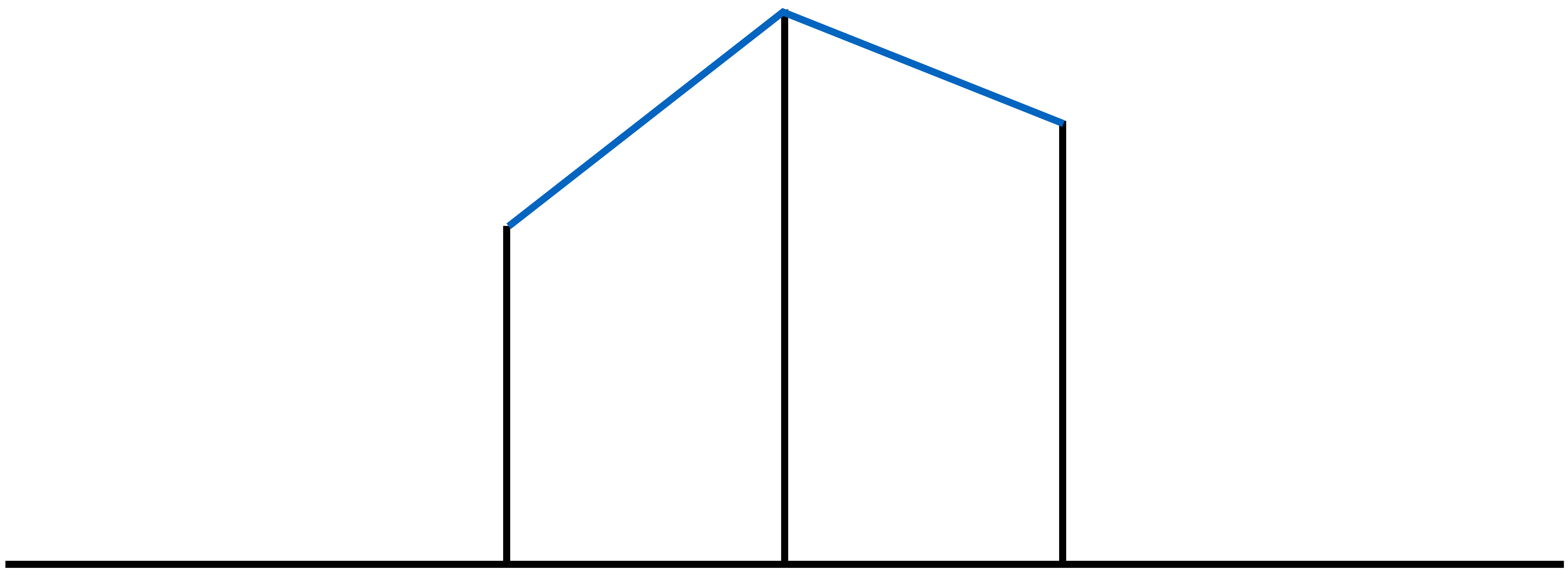


Nearest Neighbor Interpolation



Problem: values not continuous

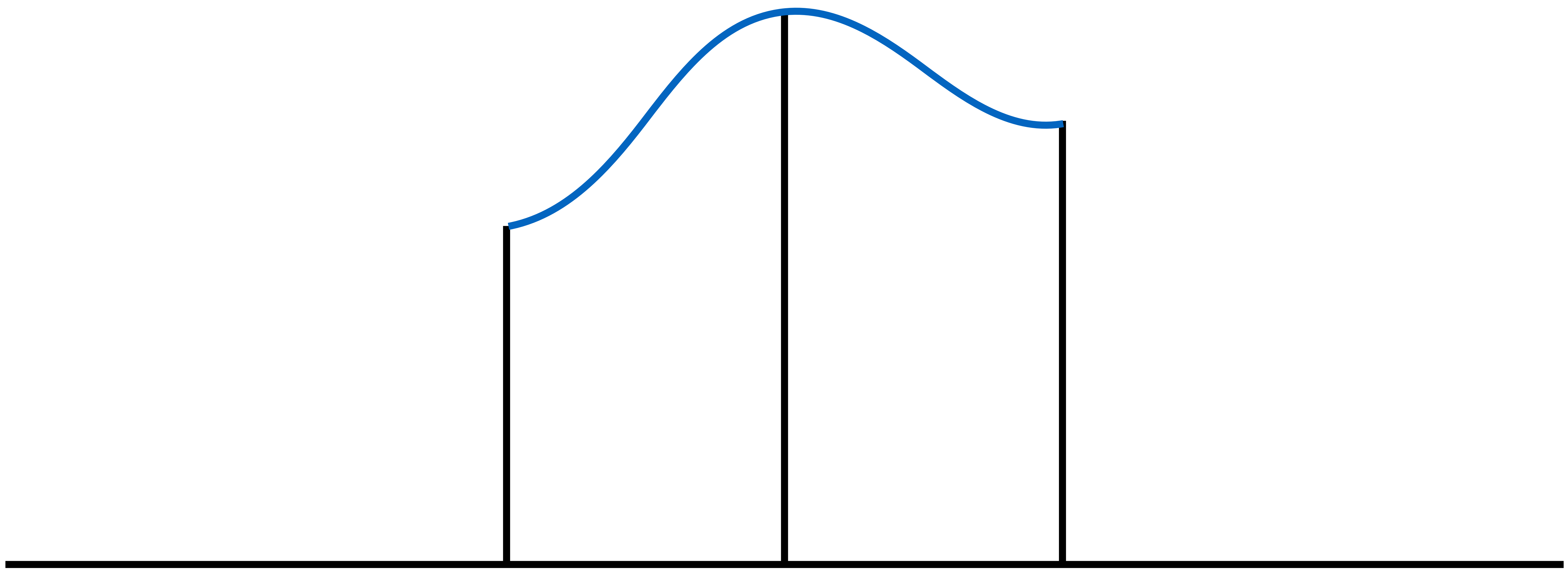
Linear Interpolation



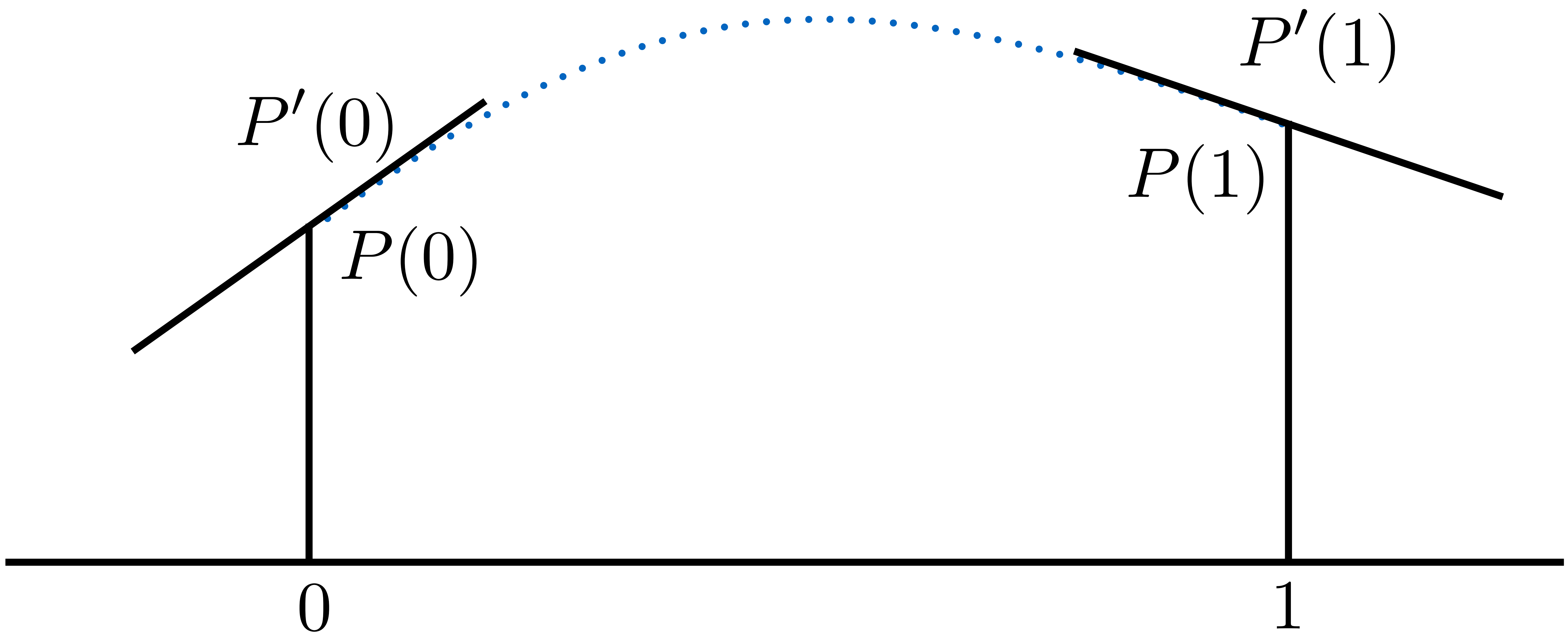
Problem: derivatives not continuous

Jonathan Ragan-Kelley & Ren Ng

Smooth Interpolation?



Cubic Hermite Interpolation



Inputs: values and derivatives at endpoints

Cubic Polynomial Interpolation

Cubic polynomial

$$P(t) = a\, t^3 + b\, t^2 + c\, t + d$$

Why cubic?

4 input constraints – need 4 degrees of freedom

$$P(0) = h_0$$
$$P(1) = h_1$$
$$P'(0) = h_2$$
$$P'(1) = h_3$$

Cubic Polynomial Interpolation

Cubic polynomial

$$P(t) = a\ t^3 + b\ t^2 + c\ t + d$$
$$P'(t) = 3a\ t^2 + 2b\ t + c$$

Set up constraint equations

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

Solve for Polynomial Coefficients

$$h_0 = d$$

$$h_1 = a + b + c + d$$

$$h_2 = c$$

$$h_3 = 3a + 2b + c$$

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

Solve for Polynomial Coefficients

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

$$= \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

(Check that these matrices are inverses)

Matrix Form of Hermite Function

$$P(t) = a\,t^3 + b\,t^2 + c\,t + d$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Interpretation 1: Matrix Rows = Coefficient Formulas

$$P(t) = \boxed{a}\, t^3 + b\, t^2 + c\, t + d$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Interpretation 2:
Matrix Columns = ?

$$P(t) = a\,t^3 + b\,t^2 + c\,t + d$$

$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

$$= \begin{bmatrix} 2t^3 - 3t^2 + 1 \\ -2t^3 + 3t^2 \\ t^3 - 2t^2 + t \\ t^3 - t^2 \end{bmatrix}^T \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

Hermite Basis Functions

$$P(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} H_0(t) & H_1(t) & H_2(t) & H_3(t) \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

$$t^3$$
$$t^2$$
$$t$$
$$1$$

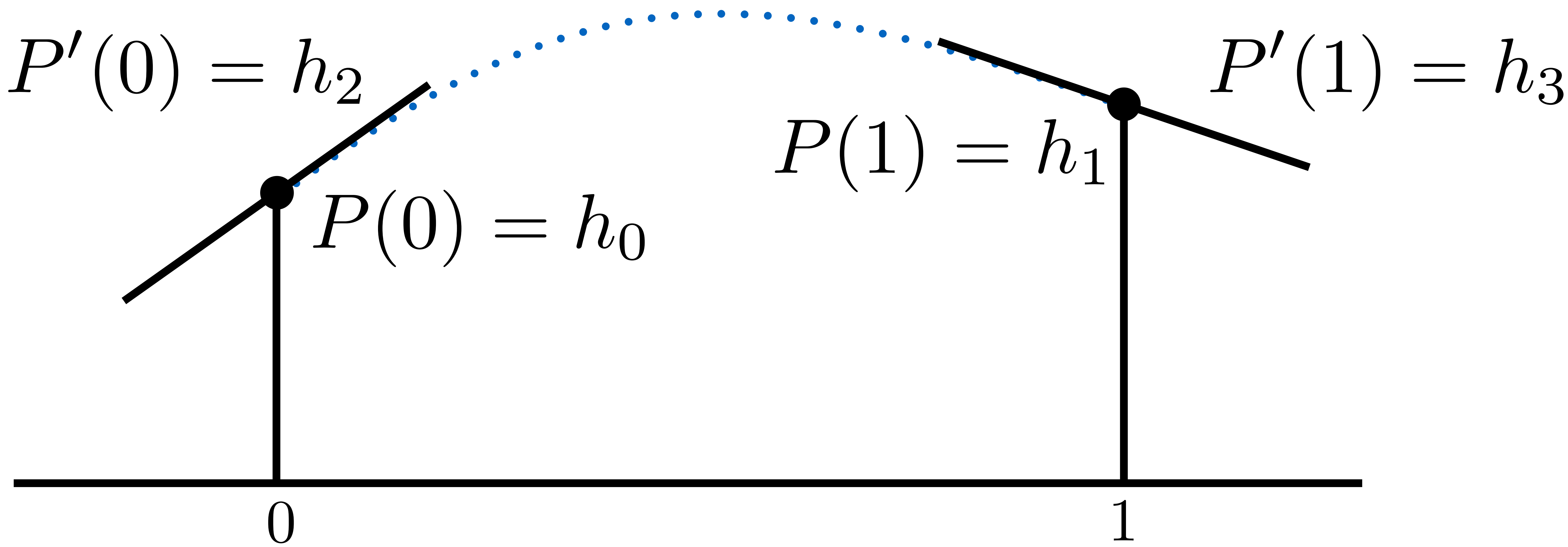
Canonical basis functions for cubic polynomials

$$H_0(t) = 2t^3 - 3t^2 + 1$$
$$H_1(t) = -2t^3 + 3t^2$$
$$H_2(t) = t^3 - 2t^2 + t$$
$$H_3(t) = t^3 - t^2$$

Hermite basis functions for cubic polynomials

Either basis can represent any cubic polynomial through linear combination

Recap: Cubic Hermite Interpolation



Inputs: values and derivatives at endpoints

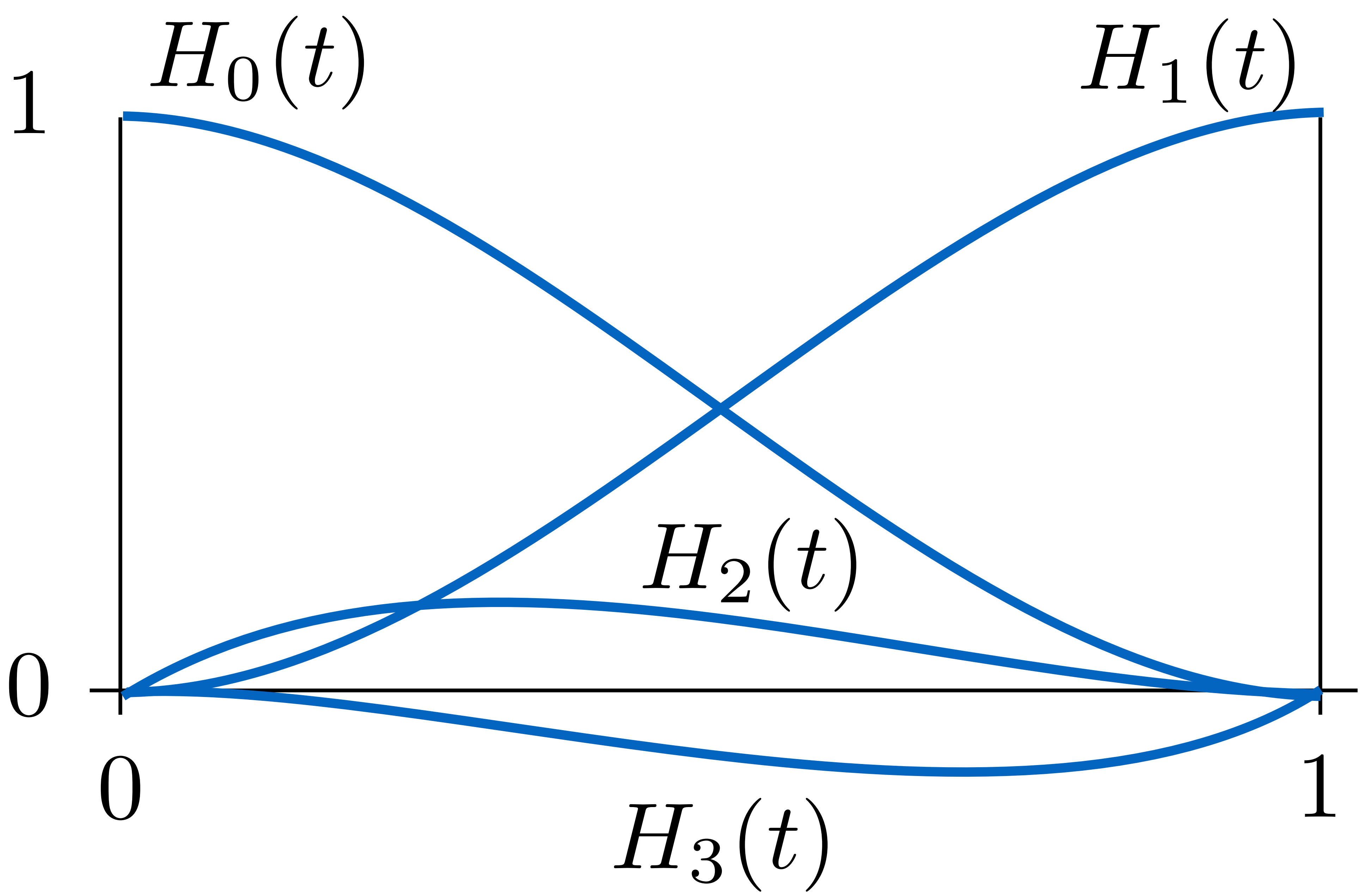
Output: cubic polynomial that interpolates

Solution: weighted sum of Hermite basis functions

$$P(t) = h_0\, H_0(t) + h_1\, H_1(t) + h_2\, H_2(t) + h_3\, H_3(t)$$

Hermite Basis Functions

Hermite basis functions (blending functions)



$$H_0(t) = 2t^3 - 3t^2 + 1$$

$$H_1(t) = -2t^3 + 3t^2$$

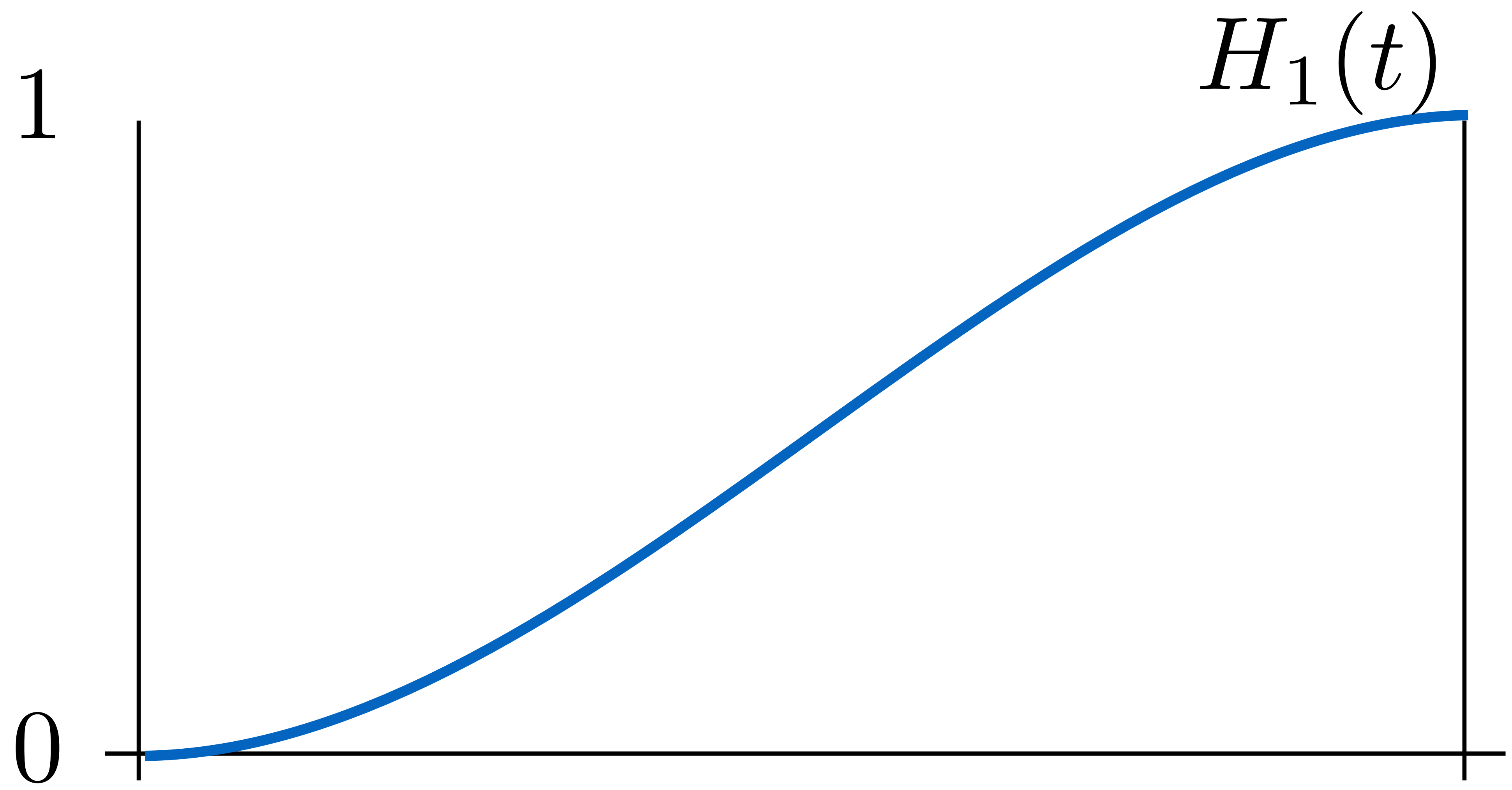
$$H_2(t) = t^3 - 2t^2 + t$$

$$H_3(t) = t^3 - t^2$$

Ease Function

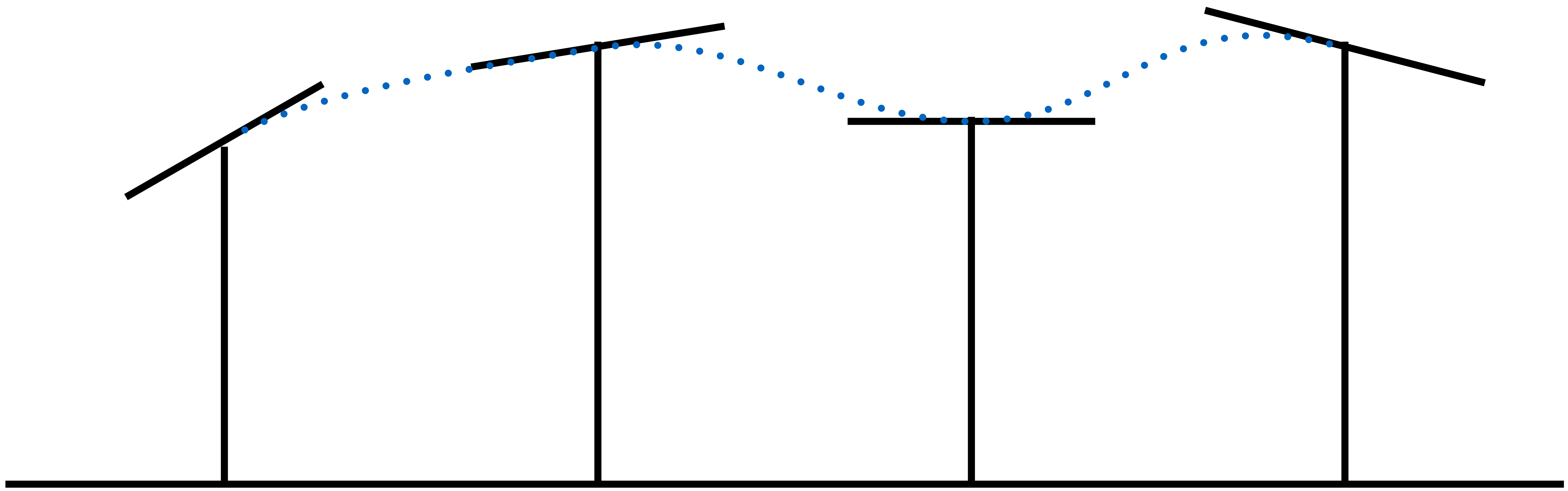
A very useful function

In animation, start and stop gently (zero velocity)



$$H_1(t) = -2t^3 + 3t^2 = t^2(3 - 2t)$$

Hermite Spline Interpolation



Inputs: *sequence* of values and derivatives

Attendance Time

If you are seated in class, go to this form and sign in:

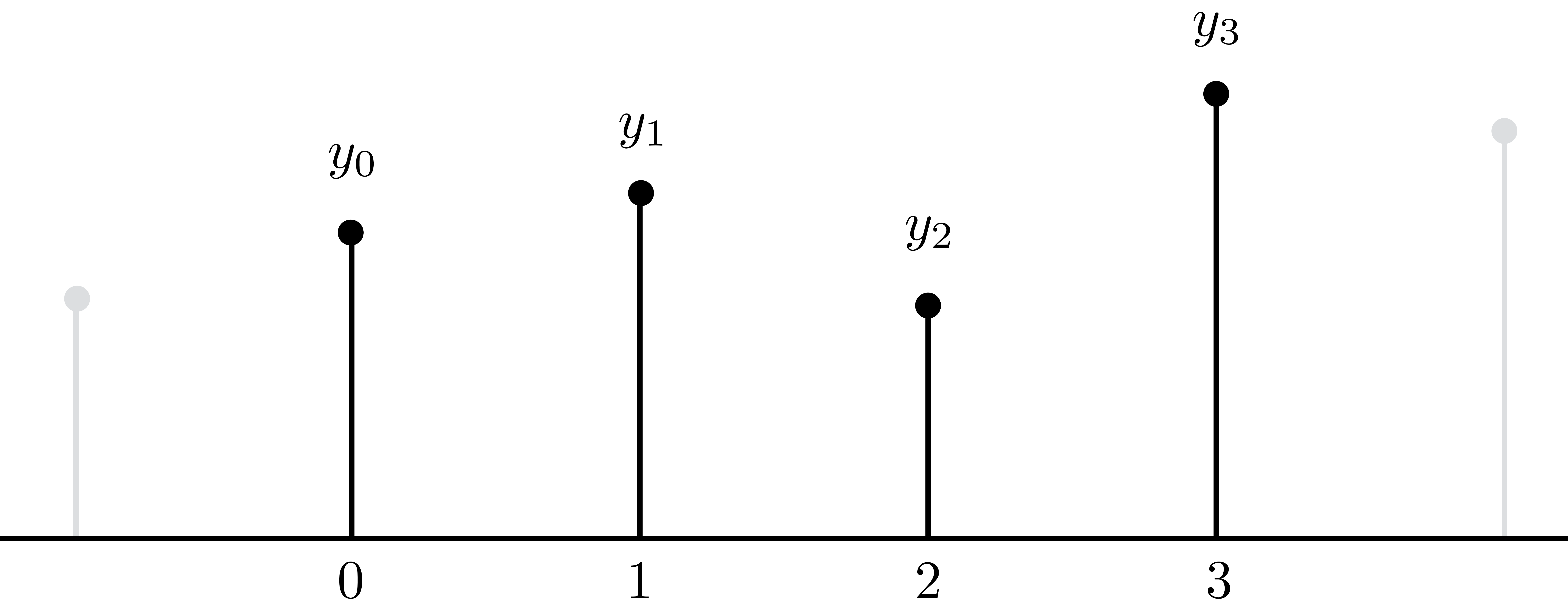
- https://tinyurl.com/184lecture

Notes:

- Time-stamp will be taken when you submit form. Do it now, won't count later.
- Don't tell friends outside class to fill it out now, because we will audit at some point in semester.
- Failing audit will have large negative consequence. You don't need to, because you have an alternative!

Catmull-Rom Interpolation

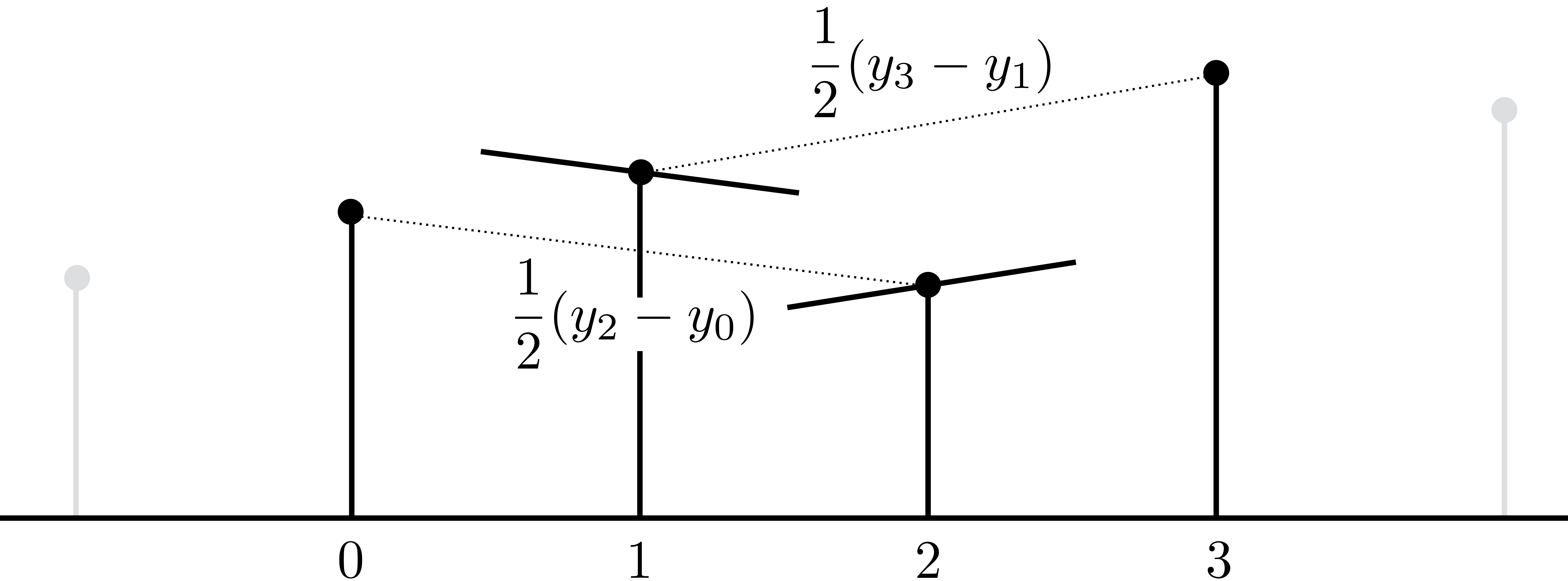
Catmull-Rom Interpolation



Inputs: sequence of values

Jonathan Ragan-Kelley & Ren Ng

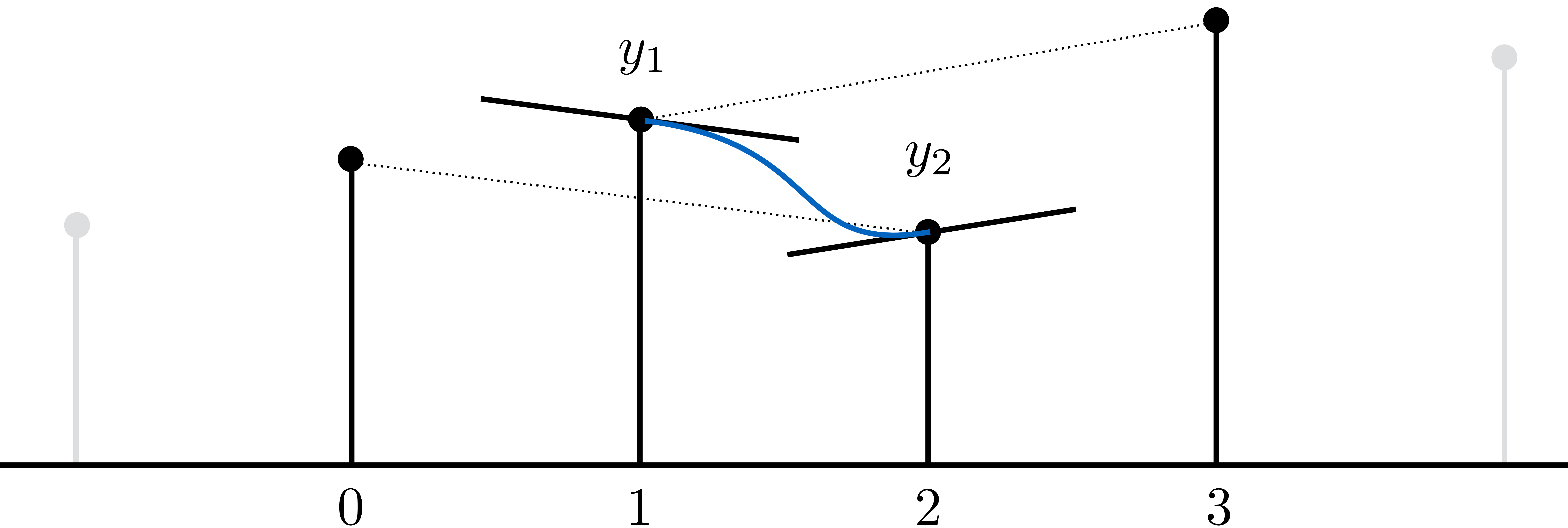
Catmull-Rom Interpolation



Rule for derivatives:

Match slope between previous and next values

Catmull-Rom Interpolation



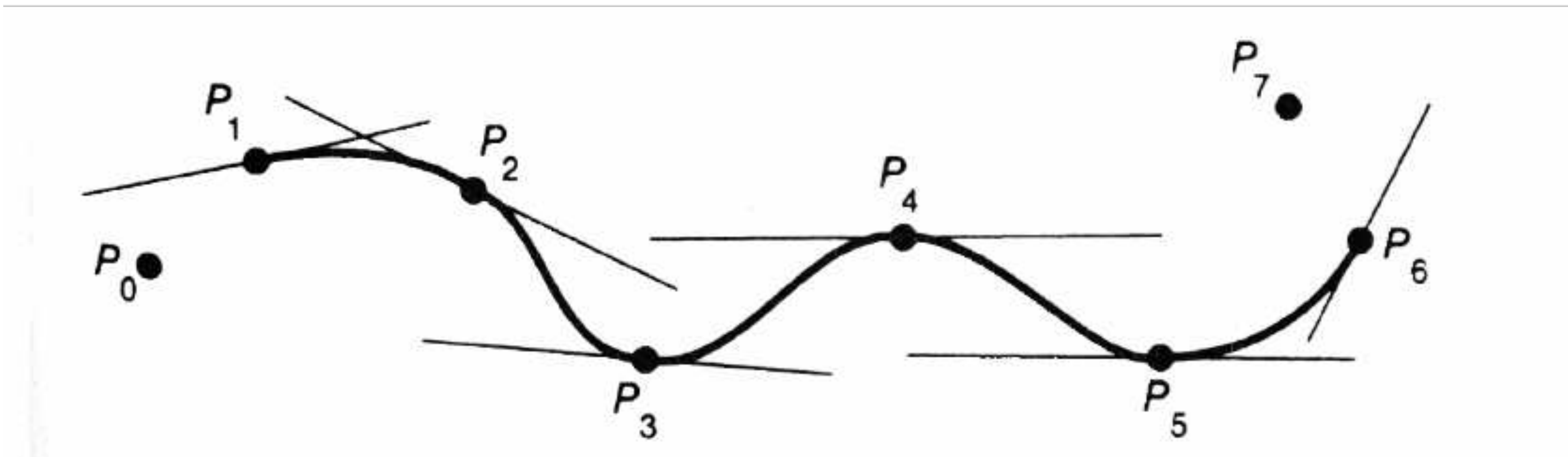
$$\frac{1}{2}(y_2 - y_0) \quad \frac{1}{2}(y_3 - y_1)$$

Then use Hermite interpolation

Catmull-Rom Spline

Input: sequence of points

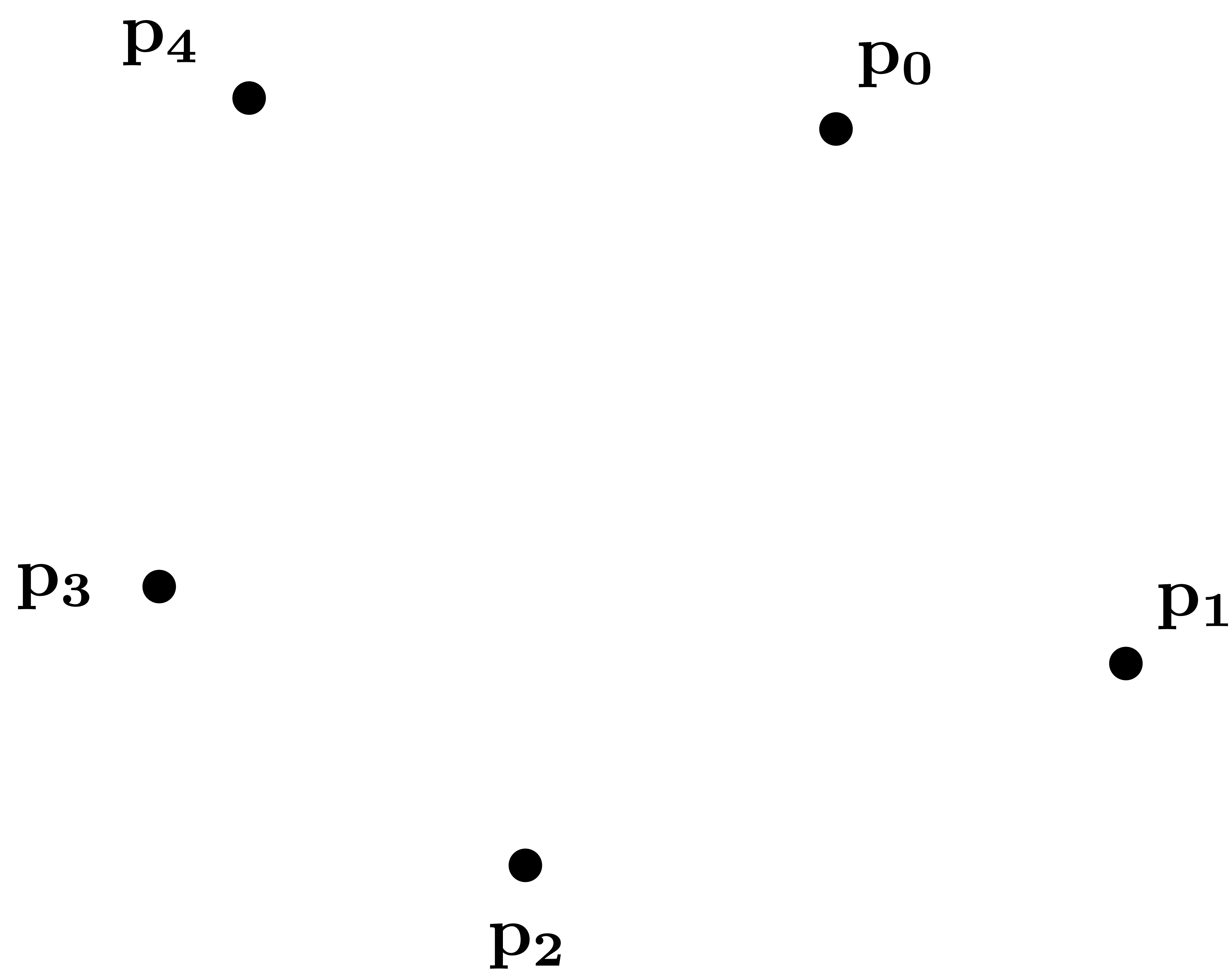
Output: spline that interpolates all points with C1 continuity



Interpolating Points & Vectors

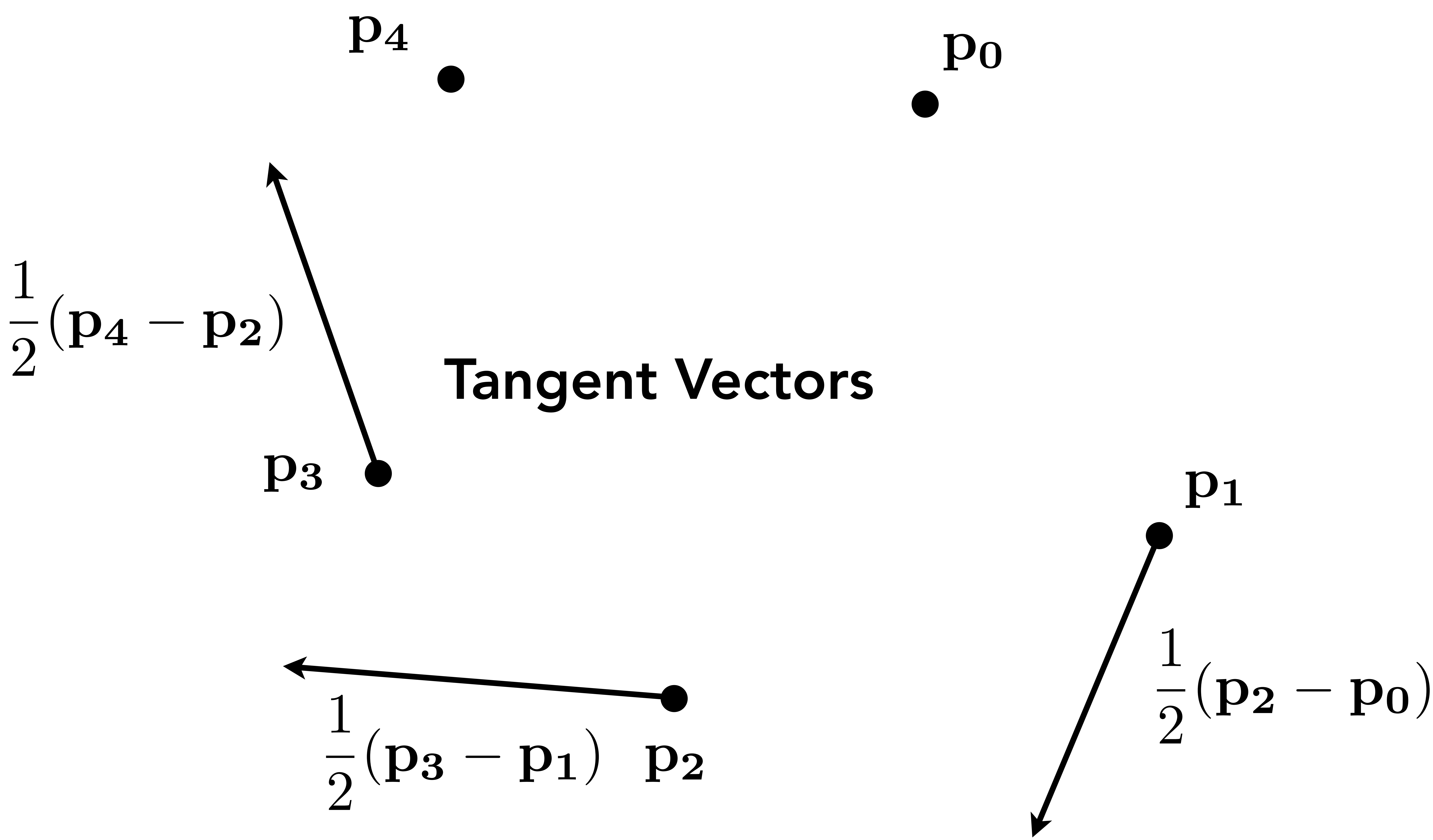
Can Interpolate Points As Easily As Values

E.g. point (0,1,3) in 3D space, or even a general vector in N dimensions



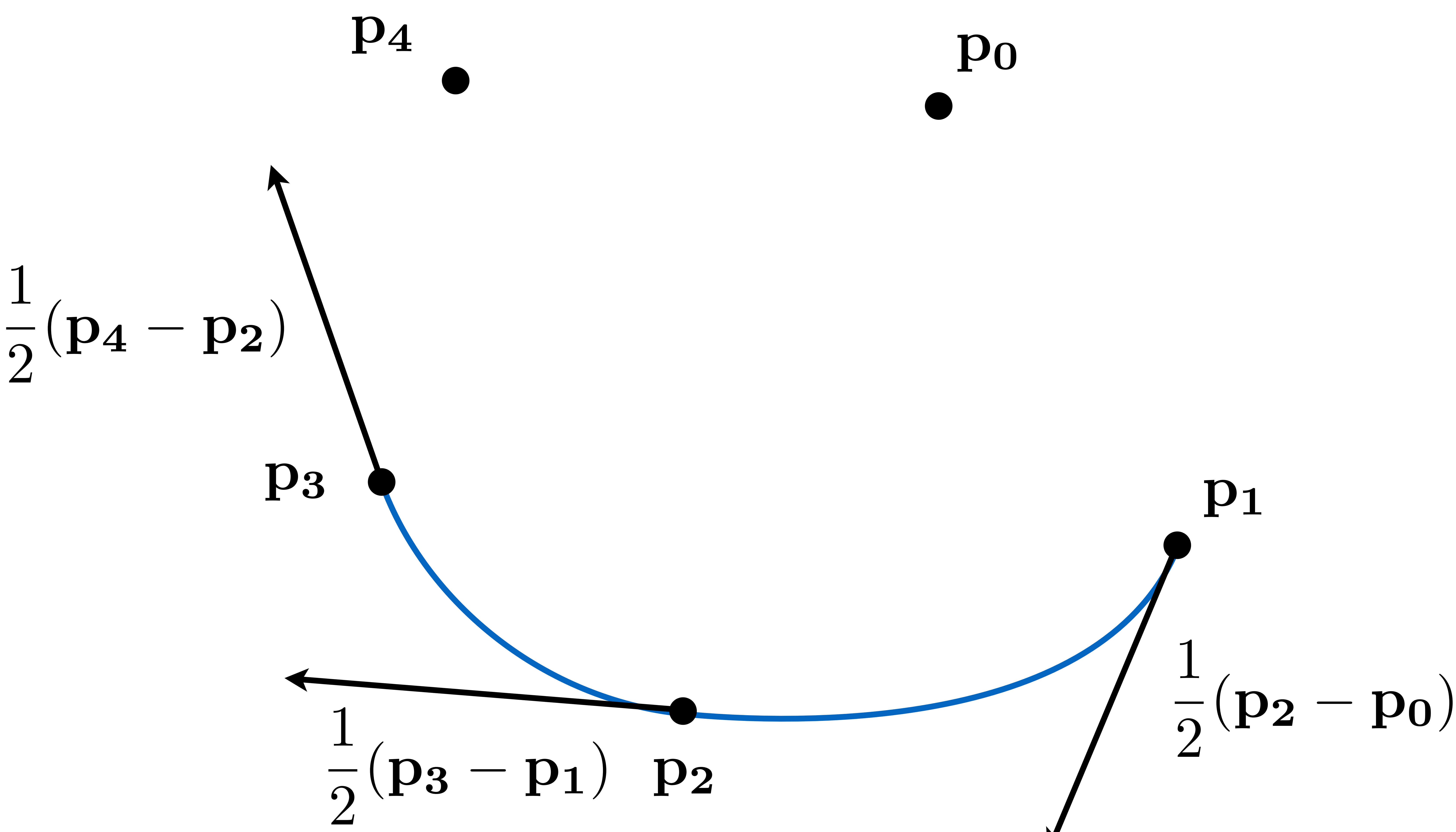
Catmull-Rom 3D spline control points

Can Interpolate Points As Easily As Values



Catmull-Rom 3D tangent vectors

Can Interpolate Points As Easily As Values



Catmull-Rom 3D space curve

Use Basis Functions to Define Curves

General formula for a particular interpolation scheme: $$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{p}_i F_i(t)$$

$$x(t) = \sum_{i=0}^{n} x_i F_i(t) \qquad y(t) = \sum_{i=0}^{n} y_i F_i(t) \qquad z(t) = \sum_{i=0}^{n} z_i F_i(t)$$

Coefficients $\mathbf{p}_i$ can be points & vectors, not just values

$F_i(t)$ are basis functions for the interpolation scheme.

Saw $H_i(t)$ for Hermite interpolation earlier. Will see $C_i(t)$ for Catmull-Rom shortly, and $B_i(t)$ for Bézier scheme later. The basis functions are properties of the interpolation scheme.

Matrix Form of Catmull-Rom Space Curve?

Use Hermite matrix form

- Points & tangents given by Catmull-Rom rules

Hermite points

$$\mathbf{h}_0 = \mathbf{p}_1$$

$$\mathbf{h}_1 = \mathbf{p}_2$$

Hermite tangents

$$\mathbf{h}_2 = \frac{1}{2}(\mathbf{p}_2 - \mathbf{p}_0)$$

$$\mathbf{h}_3 = \frac{1}{2}(\mathbf{p}_3 - \mathbf{p}_1)$$

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

Hermite matrix

Convert Catmull-Rom inputs to Hermite inputs

Matrix Form of Catmull-Rom Space Curve

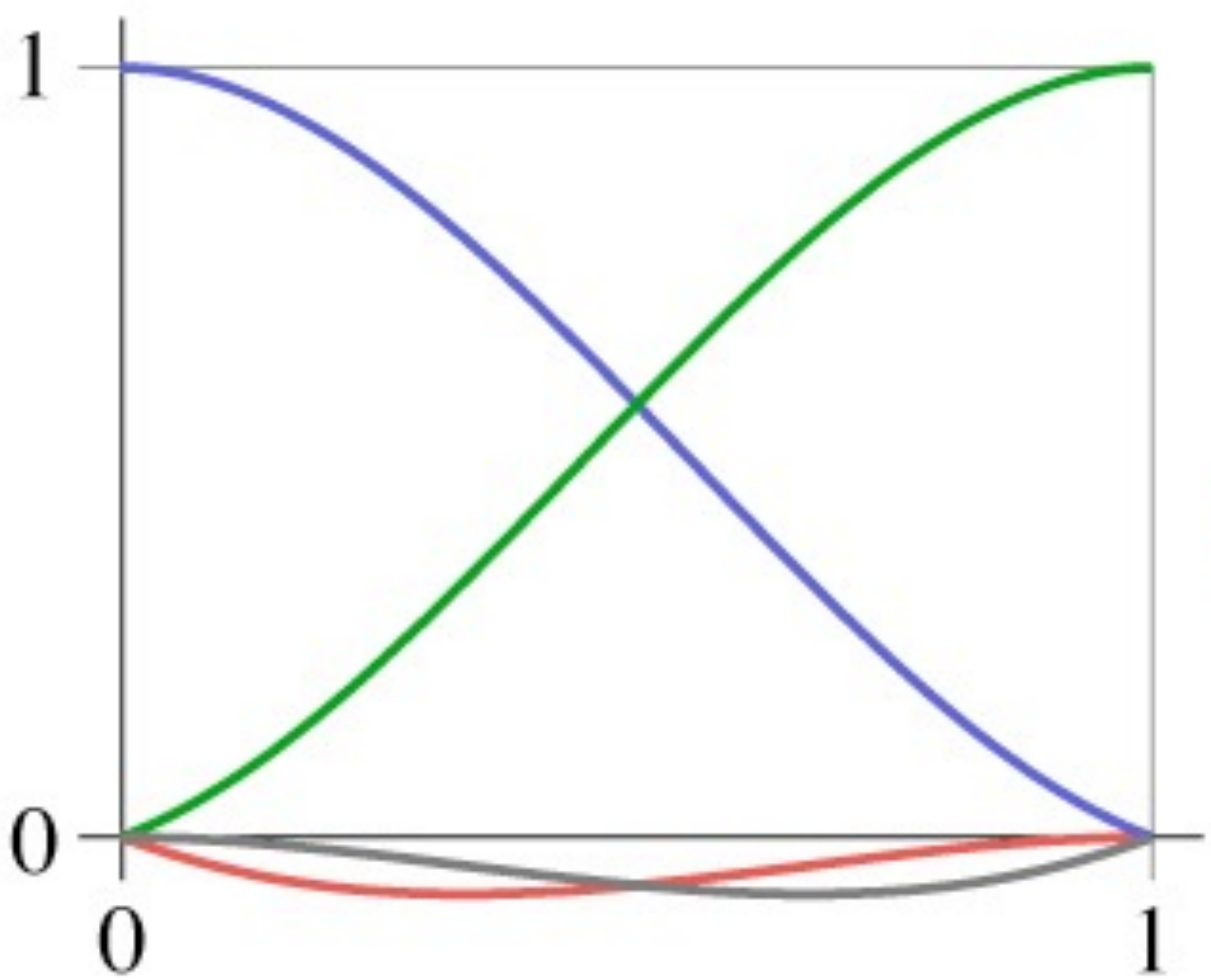
$$P(t) = \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}^T \begin{bmatrix} -\frac{1}{2} & \frac{3}{2} & \frac{3}{2} & \frac{1}{2} \\ 1 & -\frac{5}{2} & 2 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$

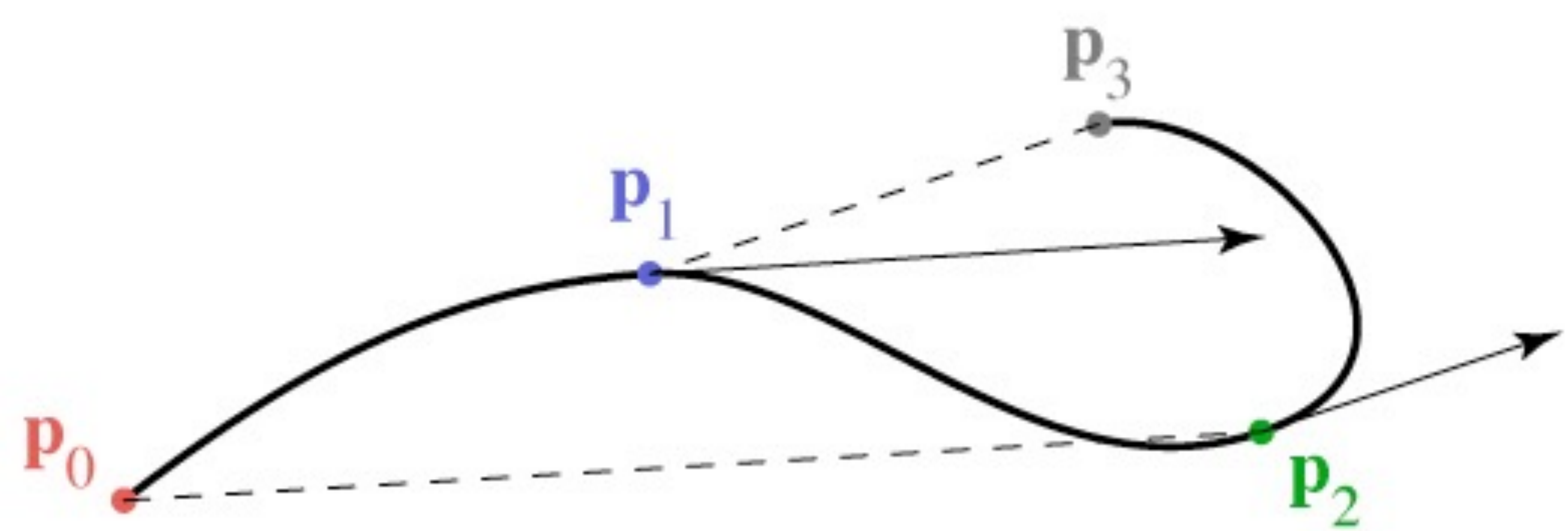
$$= C_0(t)\ \mathbf{p}_0 + C_1(t)\ \mathbf{p}_1 + C_2(t)\ \mathbf{p}_2 + C_3(t)\ \mathbf{p}_3$$

Matrix columns = Catmull-Rom basis functions

Catmull-Rom Basis Functions

Hermite basis functions (blending functions)

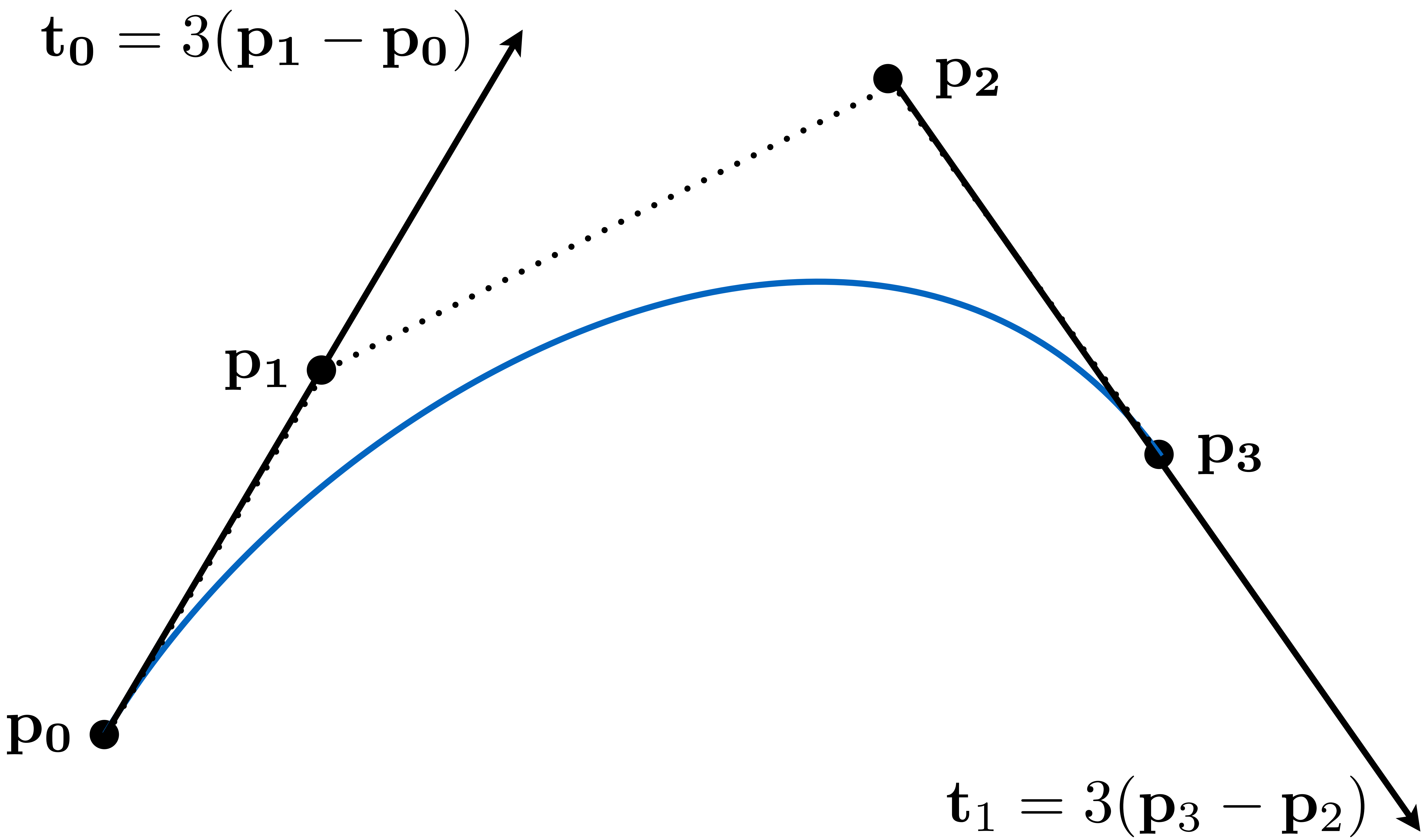




Bézier Curves

Examples of Geometry

Defining Cubic Bézier Curve With Tangents

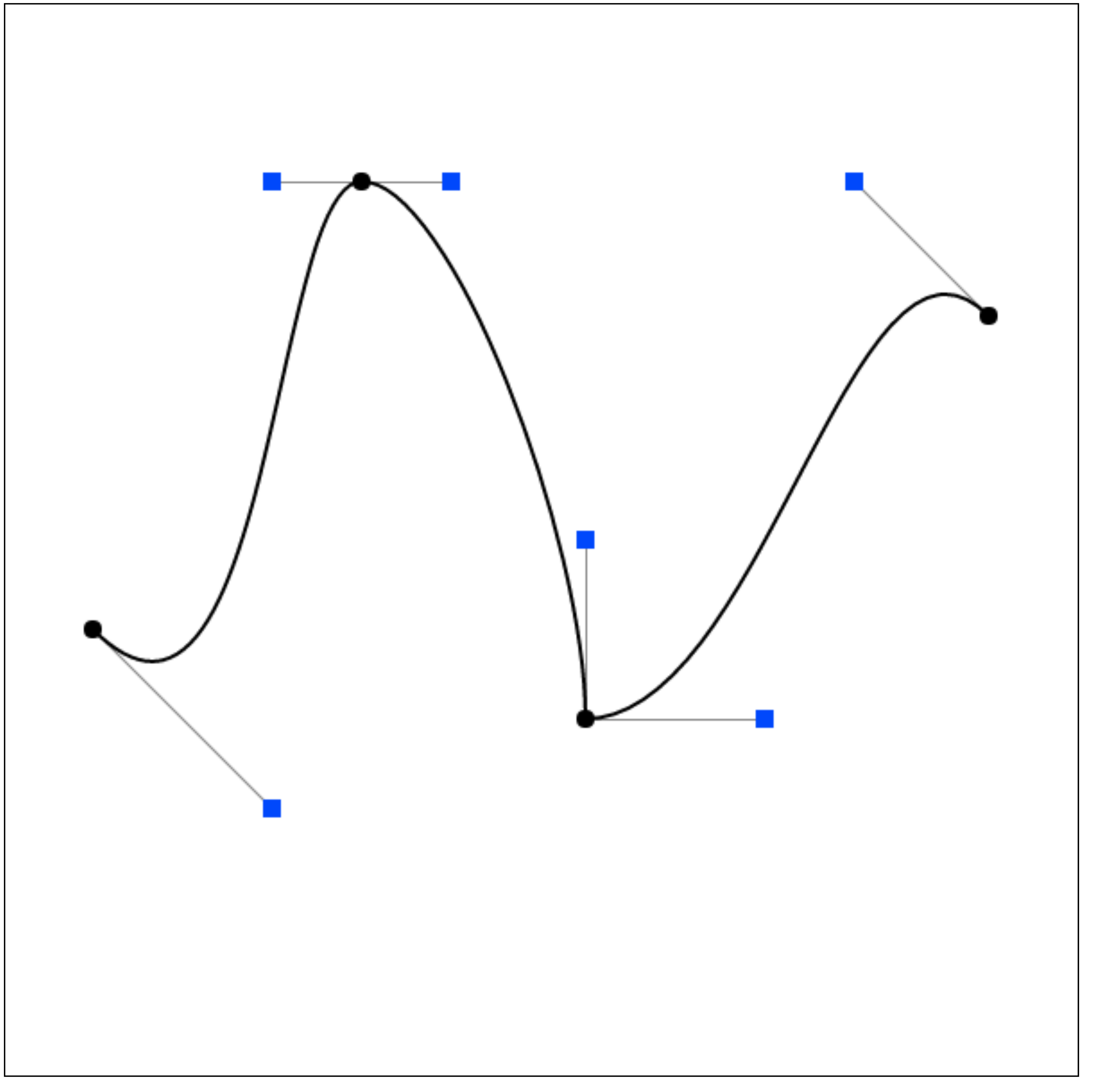


Matrix Form of Cubic Bézier Curve?

$$P(t) = \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}^T \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{bmatrix}$$
$$= B_0^3(t)\ \mathbf{p}_0 + B_1^3(t)\ \mathbf{p}_1 + B_2^3(t)\ \mathbf{p}_2 + B_3^3(t)\ \mathbf{p}_3$$

Good exercise to derive this matrix yourself.
One way: use Hermite matrix equation again.
What are the points and tangents?

Demo – Piecewise Cubic Bézier Curve

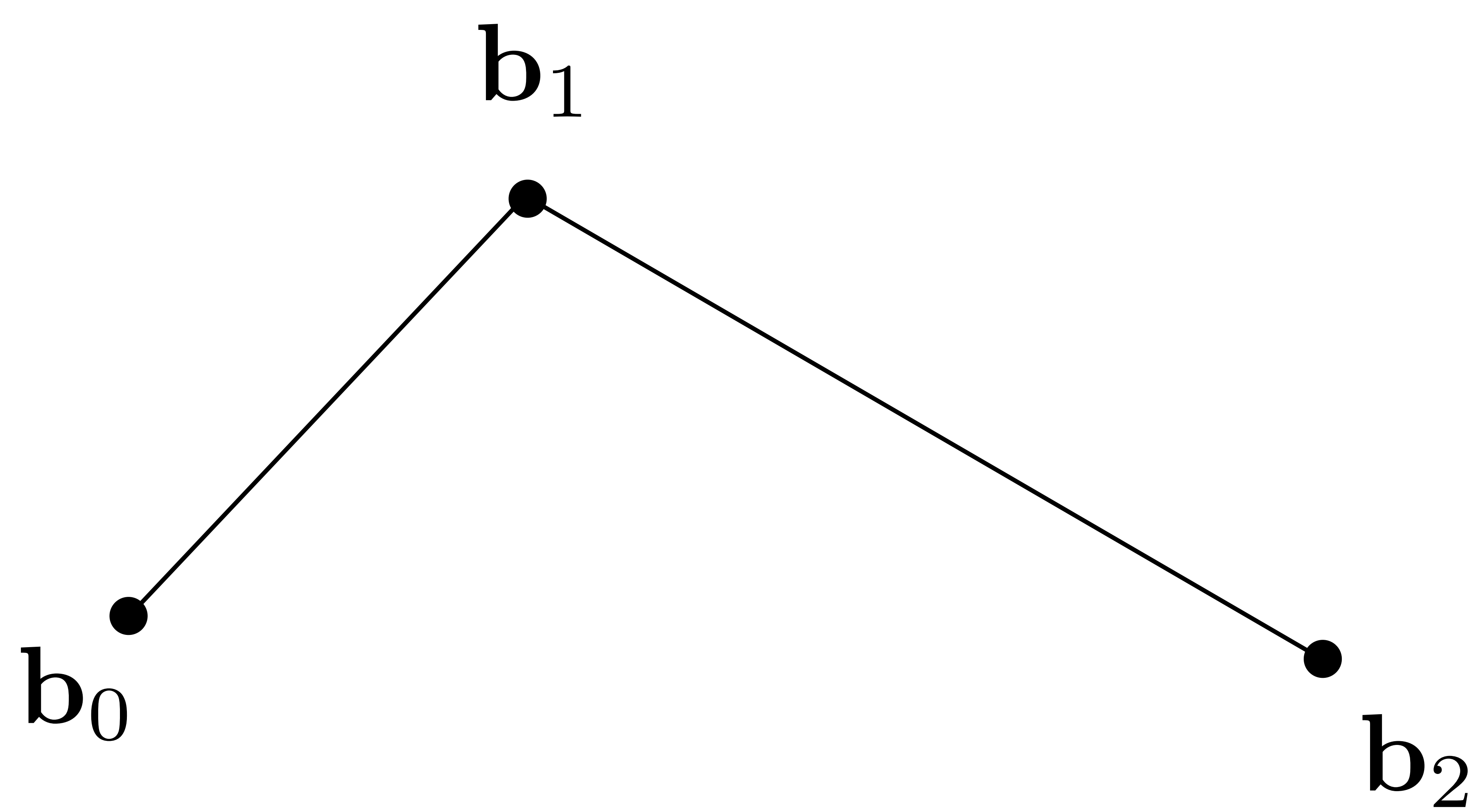


David Eck, http://math.hws.edu/eck/cs424/notes2013/canvas/bezier.html

Evaluating Bézier Curves
De Casteljau Algorithm

Bézier Curves – de Casteljau Algorithm

Consider three points (quadratic Bezier)

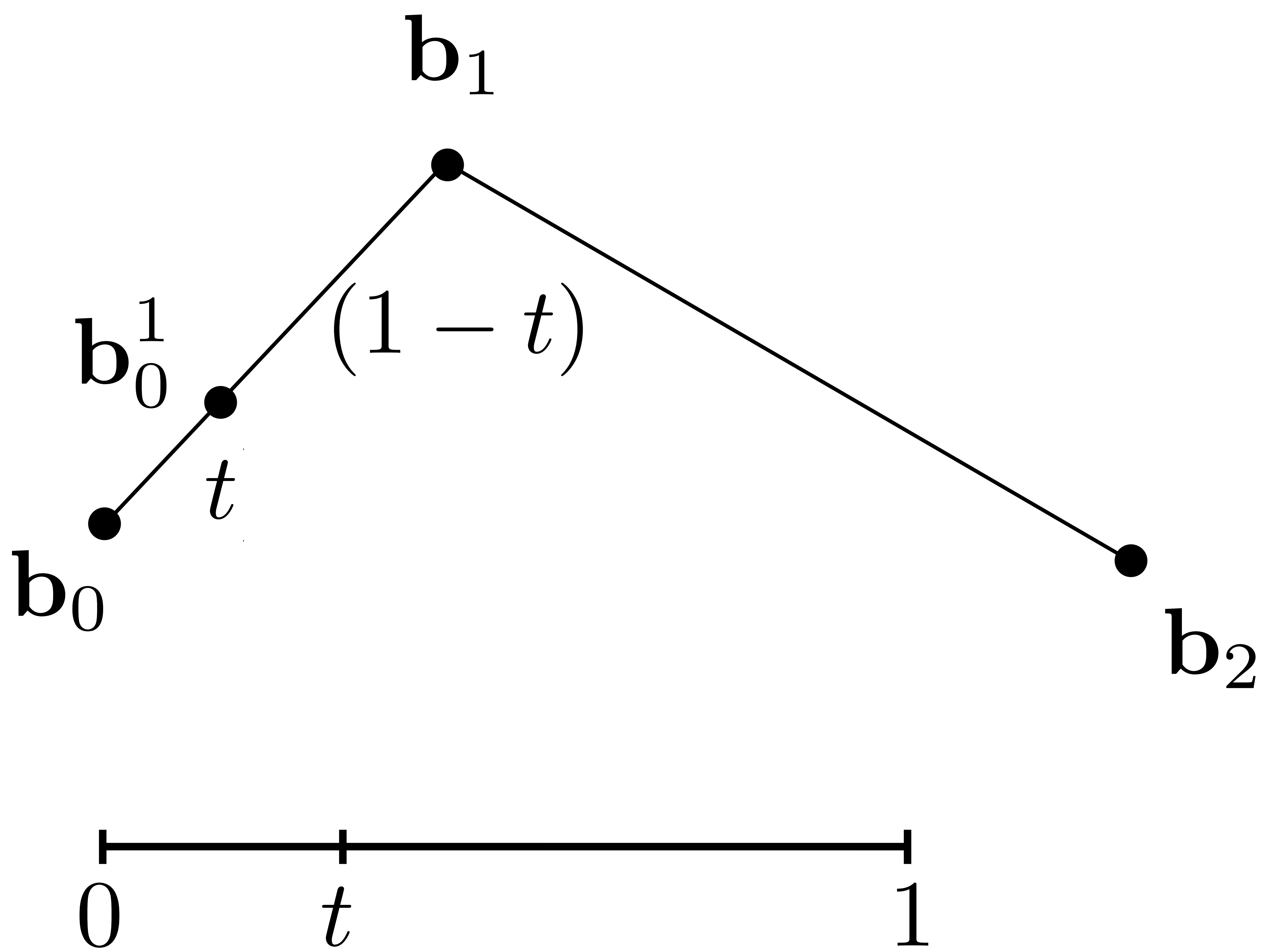


Pierre Bézier
1910 – 1999

Paul de Casteljau
b. 1930

Bézier Curves – de Casteljau Algorithm

Insert a point using linear interpolation

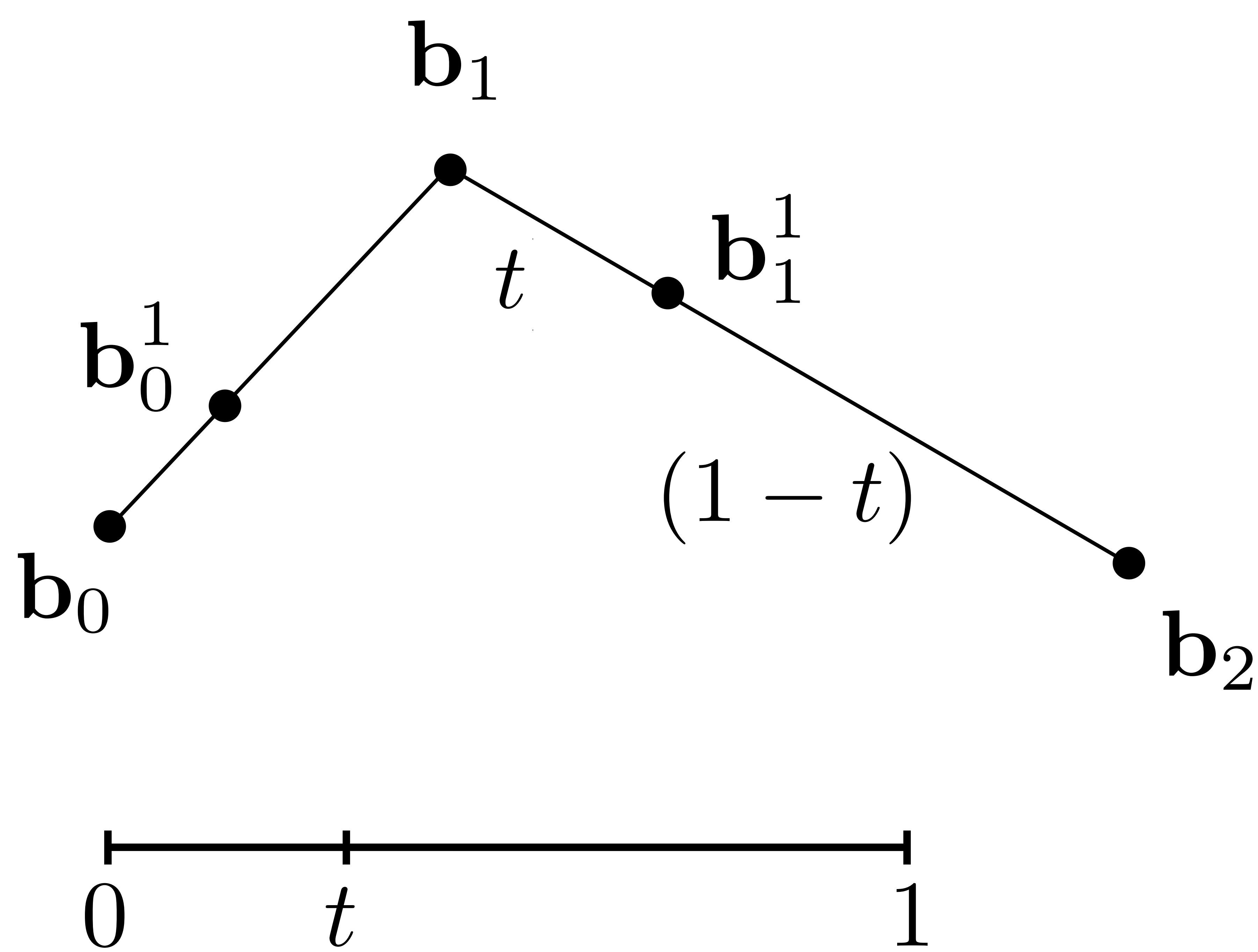


Pierre Bézier
1910 – 1999

Paul de Casteljau
b. 1930

Bézier Curves – de Casteljau Algorithm

Insert on both edges

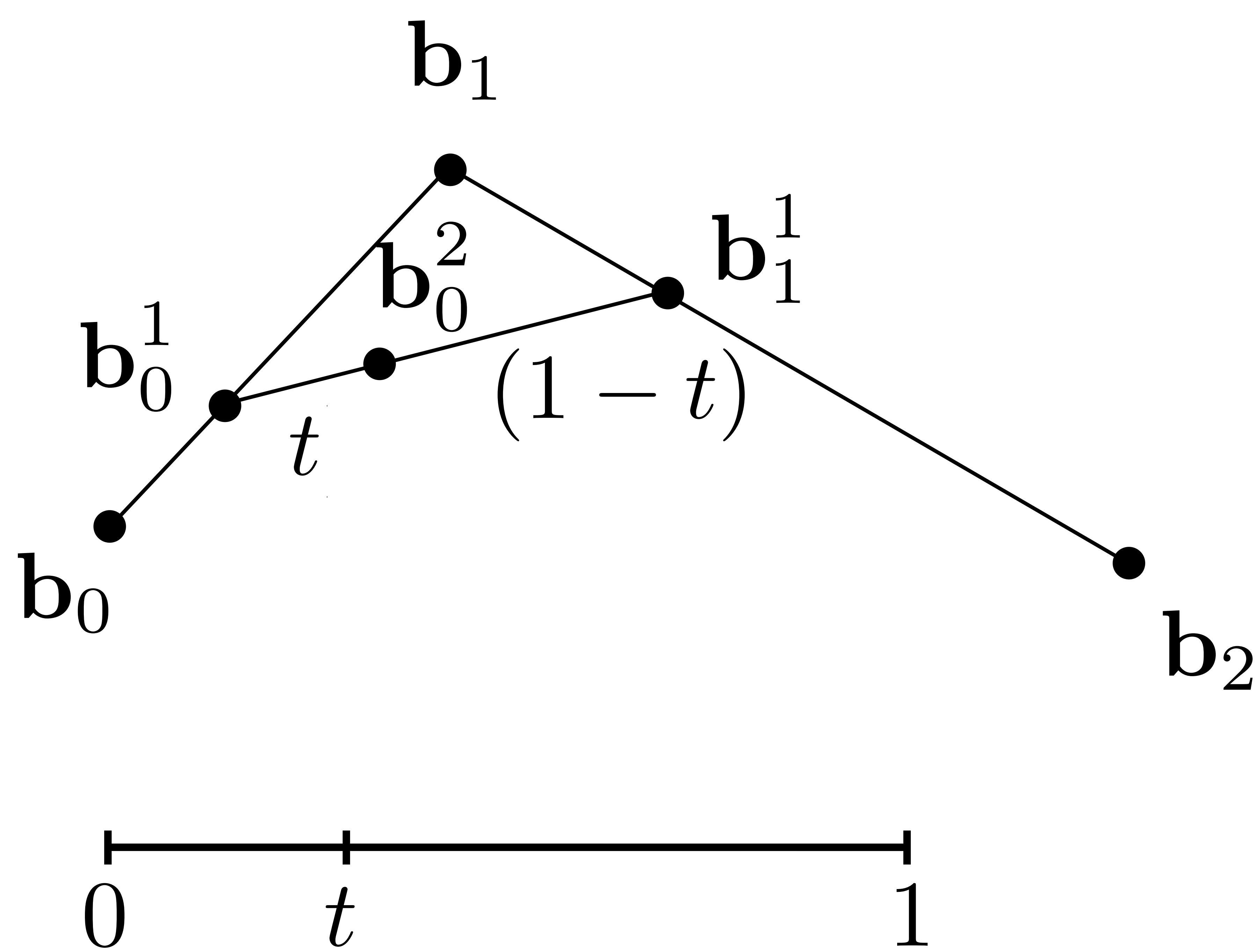


Pierre Bézier
1910 – 1999

Paul de Casteljau
b. 1930

Bézier Curves – de Casteljau Algorithm

Repeat recursively

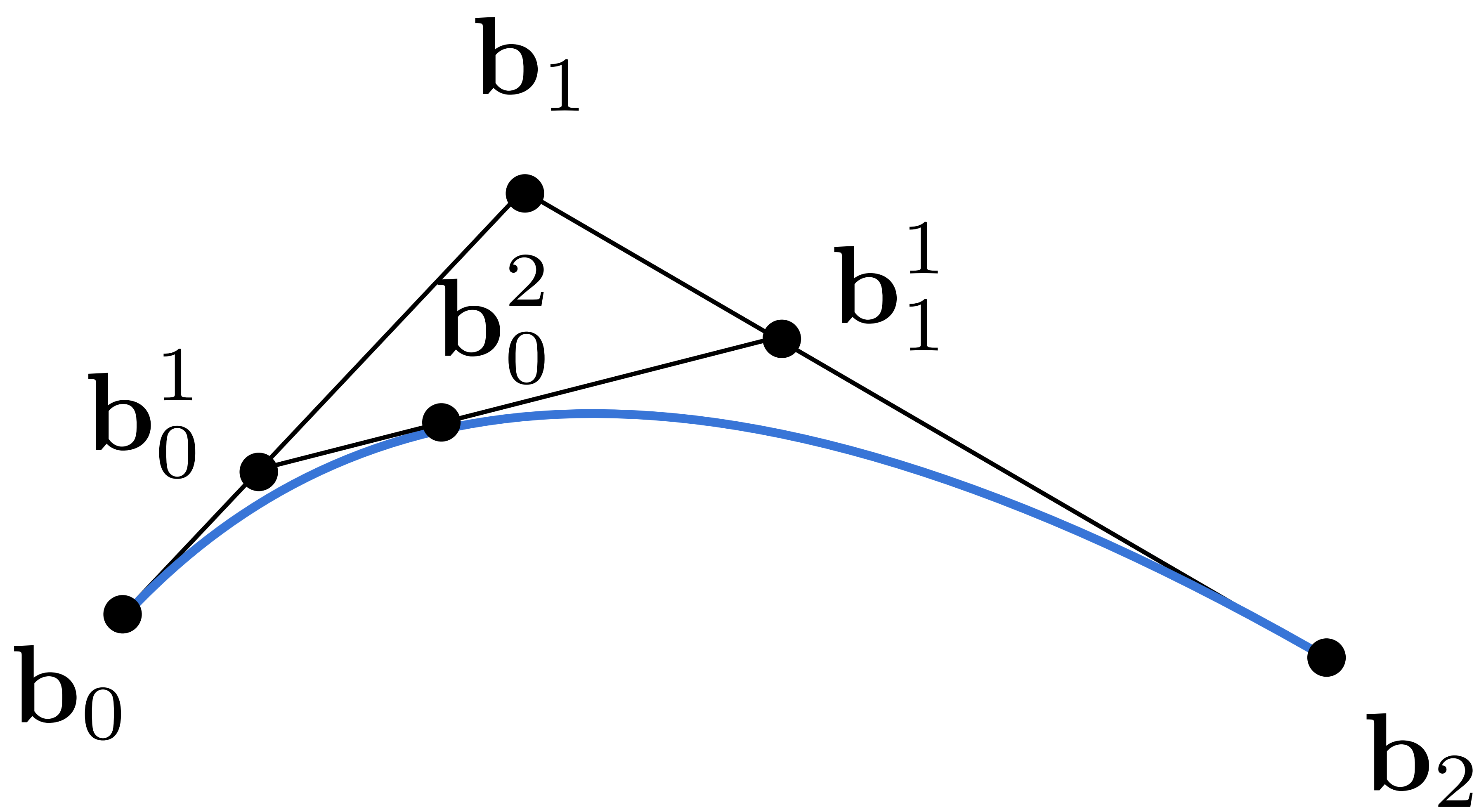


Pierre Bézier
1910 – 1999

Paul de Casteljau
b. 1930

Bézier Curves – de Casteljau Algorithm

Algorithm defines the curve



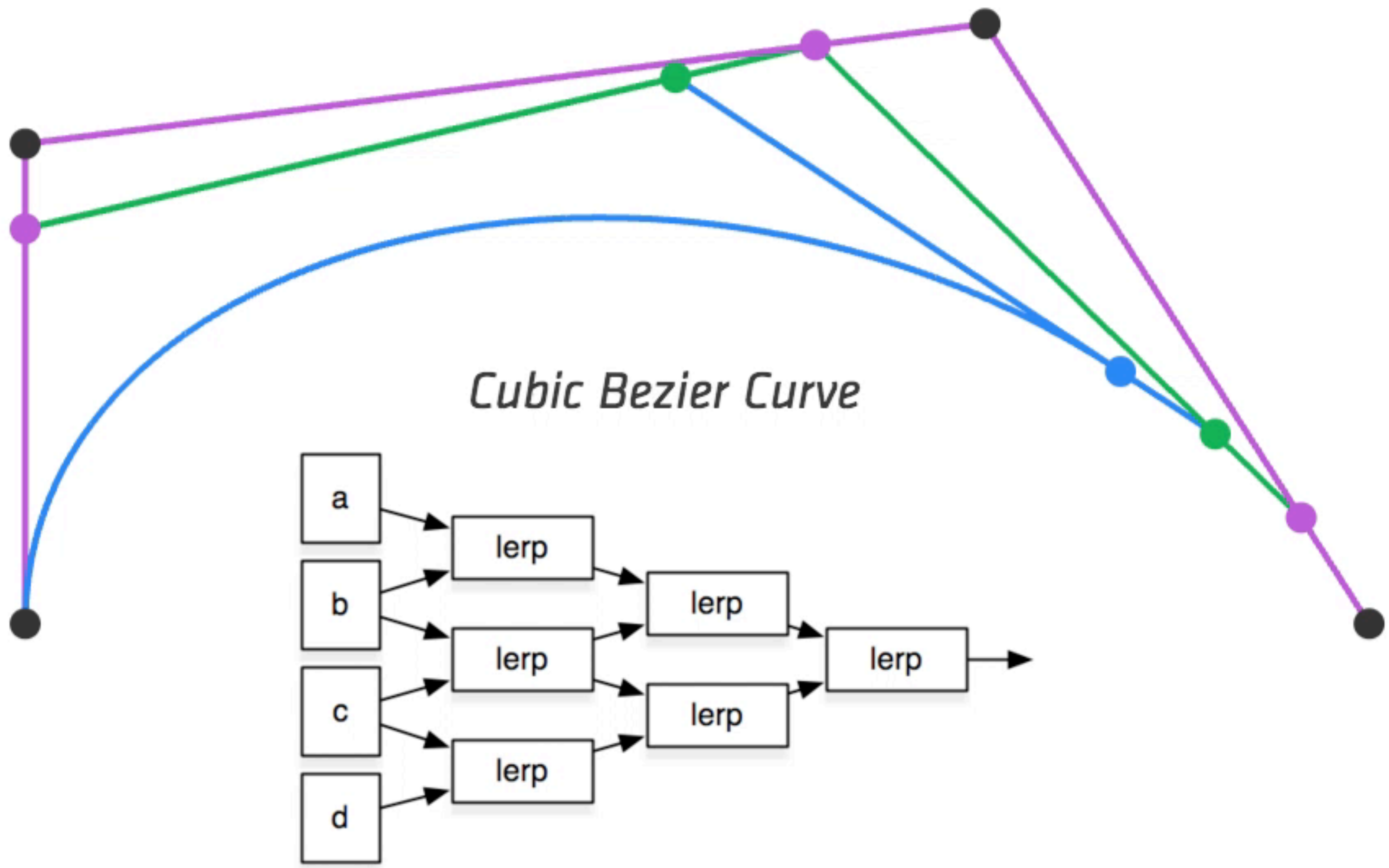
Pierre Bézier
1910 – 1999

Paul de Casteljau
b. 1930

"Corner cutting" recursive subdivision

Successive linear interpolation

Visualizing de Casteljau Algorithm

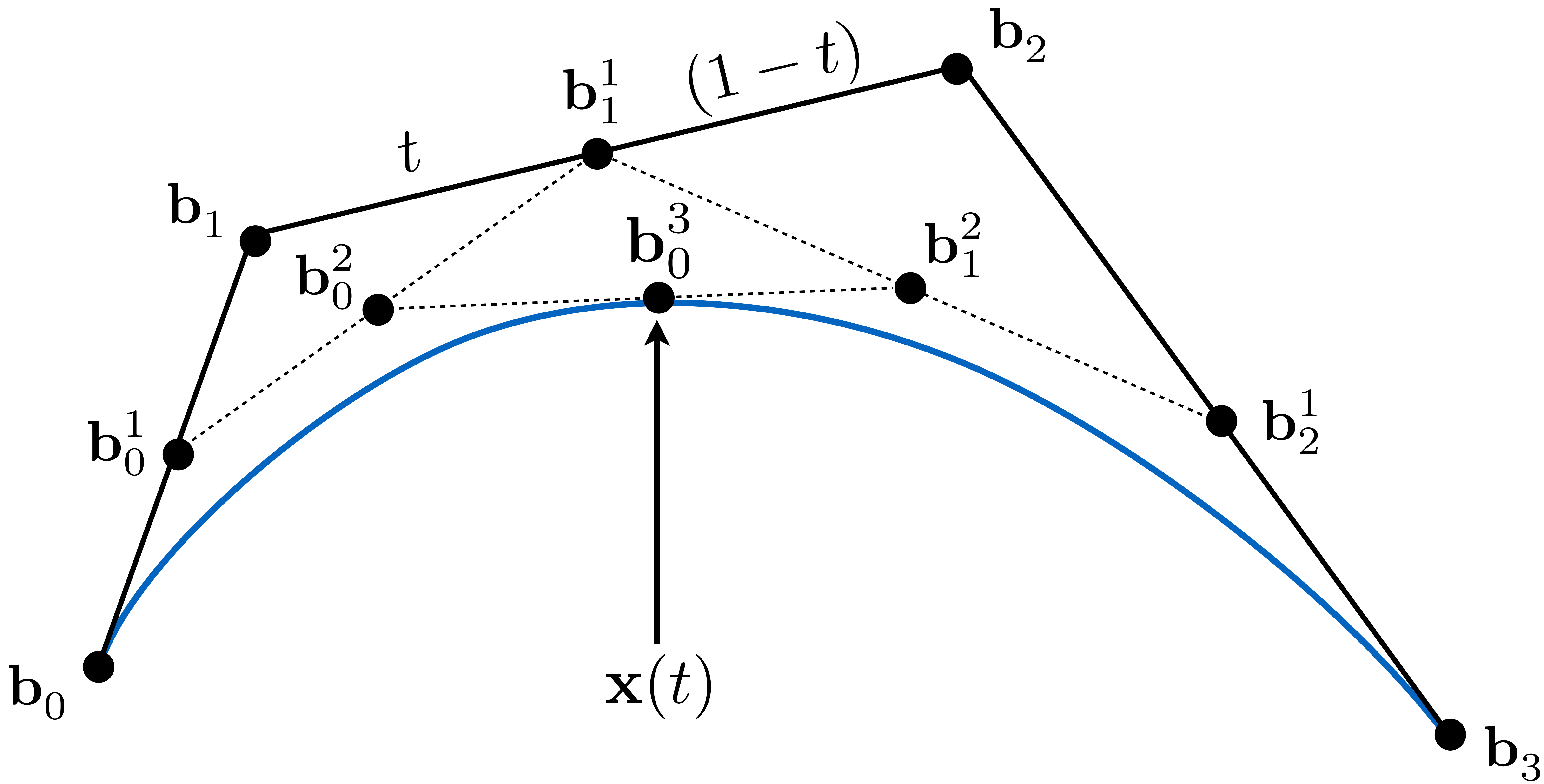


Animation: Steven Wittens, Making Things with Maths, http://acko.net

Cubic Bézier Curve – de Casteljau

Consider four points

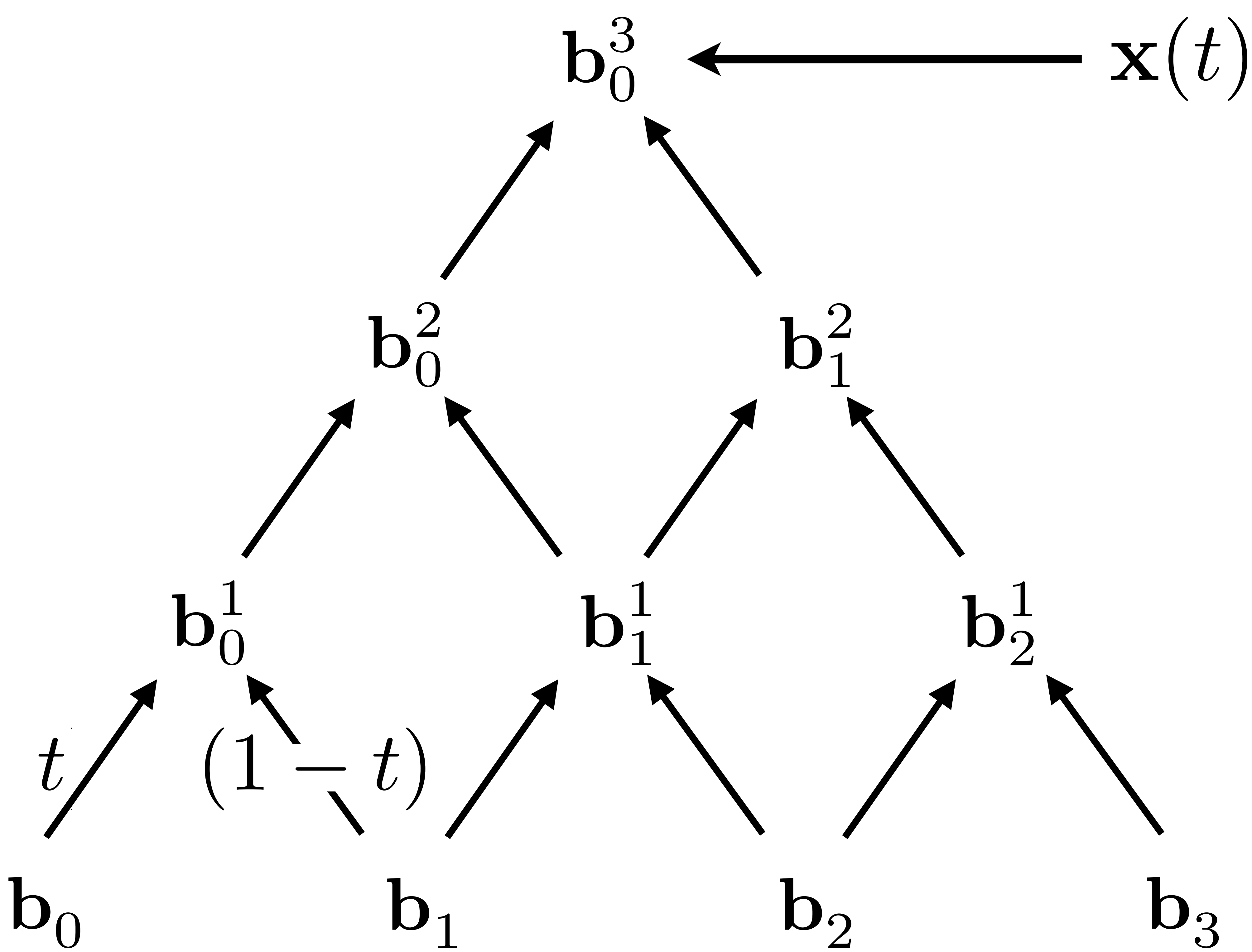
Same recursive linear interpolations

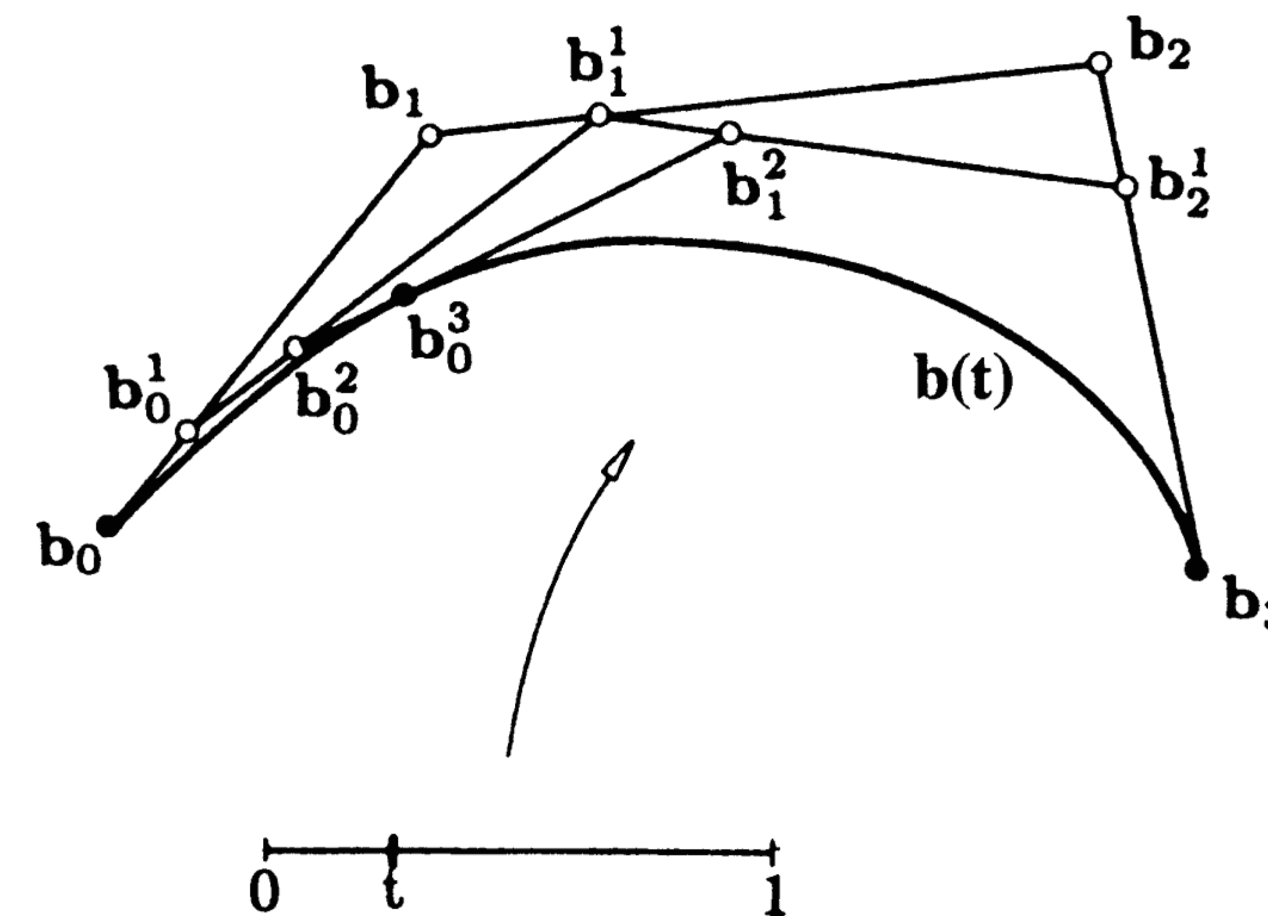


Evaluating Bézier Curves
Algebraic Formula

Bézier Curve – Algebraic Formula

de Casteljau algorithm gives a pyramid of coefficients

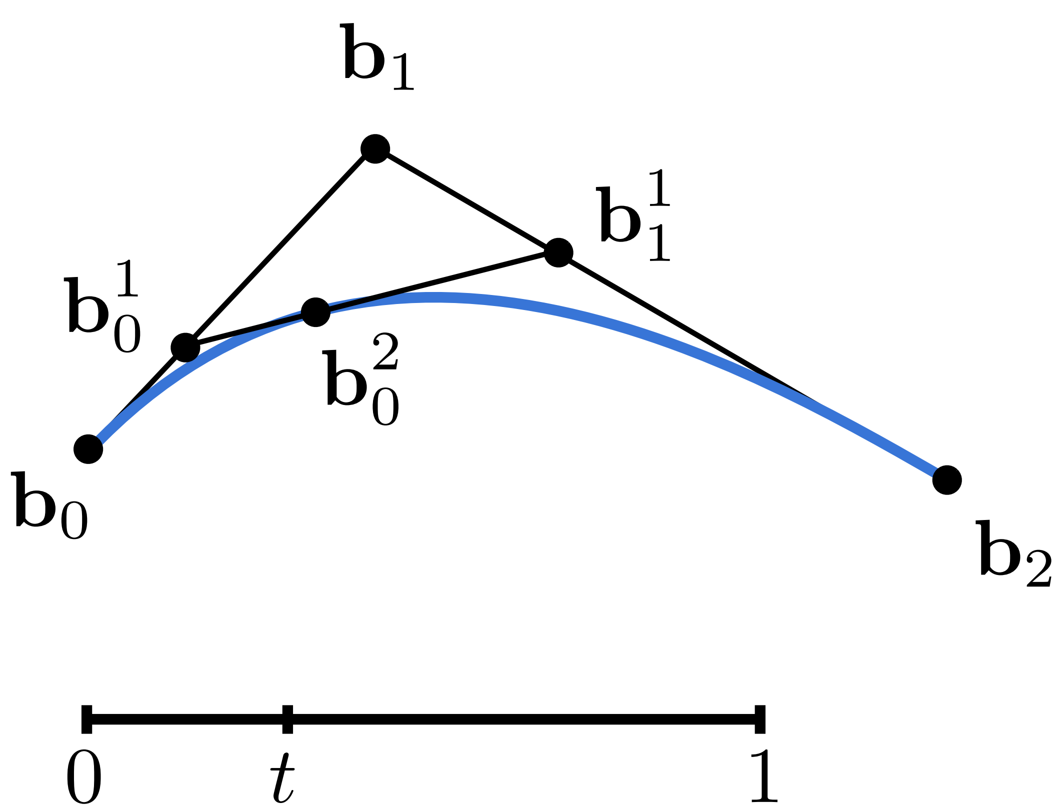




Every rightward arrow is multiplication by t,
Every leftward arrow by (1–t)

Bézier Curve – Algebraic Formula

The de Casteljau algorithm
Example: quadratic Bézier curve from three points



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

$$\mathbf{b}_1^1(t) = (1-t)\mathbf{b}_1 + t\mathbf{b}_2$$

$$\mathbf{b}_0^2(t) = (1-t)\mathbf{b}_0^1 + t\mathbf{b}_1^1$$

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

Bézier Curve – General Algebraic Formula

Bernstein form of a Bézier curve of order n:

$$\mathbf{b}^n(t) = \mathbf{b}_0^n(t) = \sum_{j=0}^{n} \mathbf{b}_j B_j^n(t)$$

Bézier curve order n

(vector polynomial of degree n)

Bernstein polynomial

(scalar polynomial of degree n)

Bézier control points

(vector in R^N)

Bernstein polynomials:

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Bézier Curve – Algebraic Formula: Example

Bernstein form of a Bézier curve of order n:

$$\mathbf{b}^n(t) = \sum_{j=0}^{n} \mathbf{b}_j B_j^n(t)$$

Example: assume n = 3 and we are in R^3

i.e. we could have control points in 3D such as:

$$\mathbf{b}_0 = (0,2,3),\ \mathbf{b}_1 = (2,3,5),\ \mathbf{b}_2 = (6,7,9),\ \mathbf{b}_3 = (3,4,5)$$

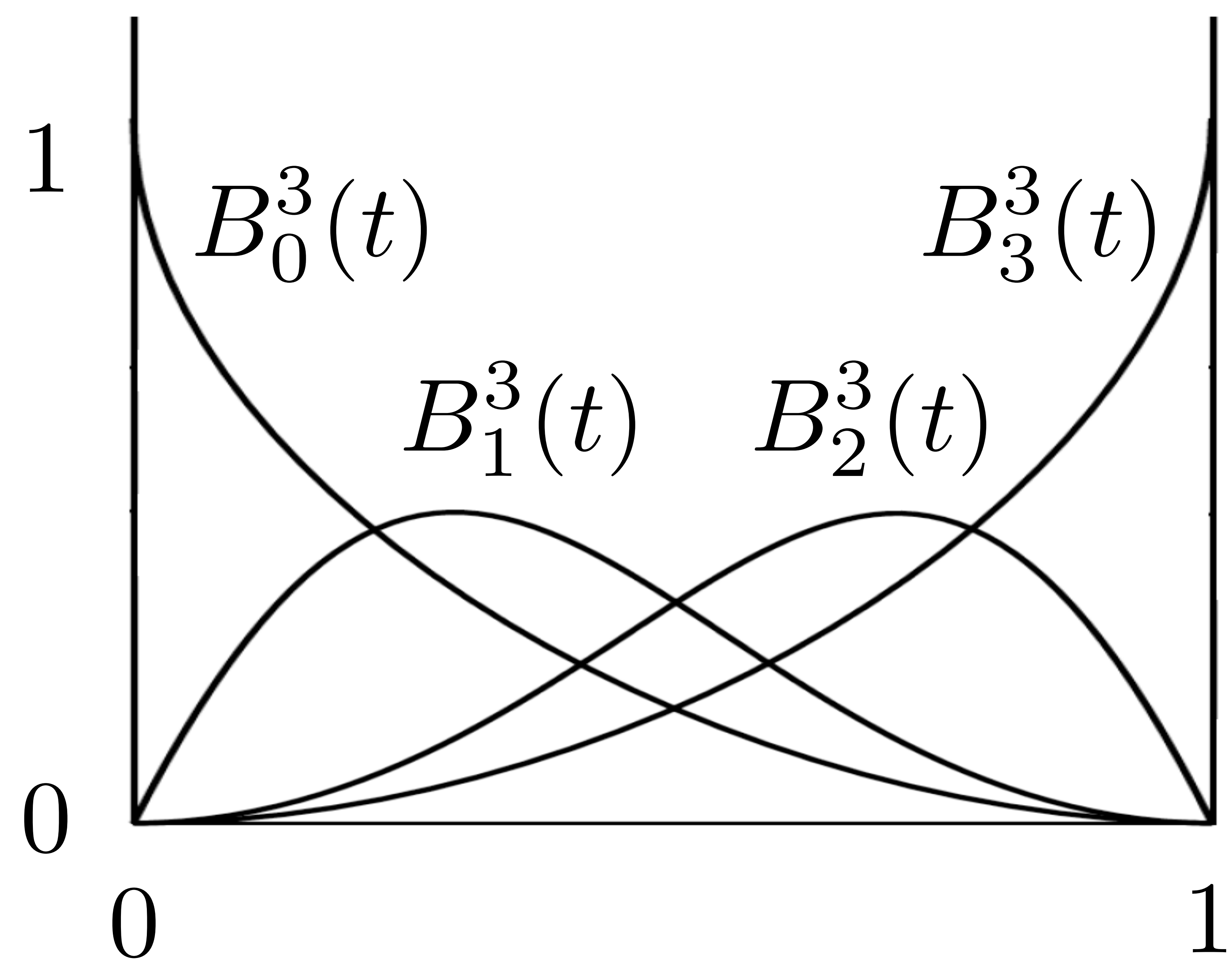
These points define a Bezier curve in 3D that is a cubic polynomial in t:

$$\mathbf{b}^n(t) = \mathbf{b}_0\,(1-t)^3 + \mathbf{b}_1\,3t(1-t)^2 + \mathbf{b}_2\,3t^2(1-t) + \mathbf{b}_3\,t^3$$

Cubic Bézier Basis Functions

Bernstein polynomials:

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

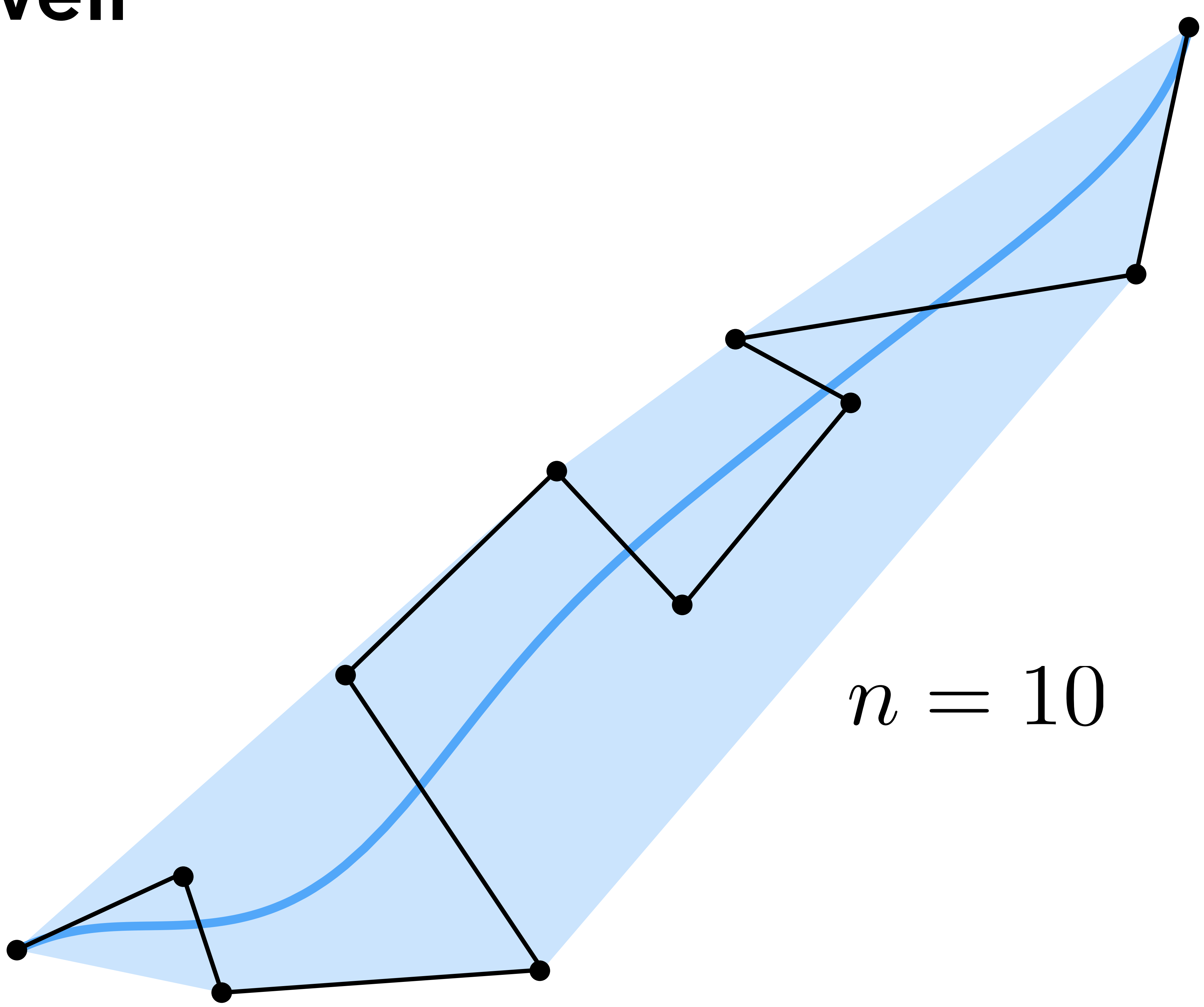


Sergei N. Bernstein
1880 – 1968

Piecewise Bézier Curves
(Bézier Spline)

Higher-Order Bézier Curves?

High-degree Bernstein polynomials don't interpolate well



Very hard to control!

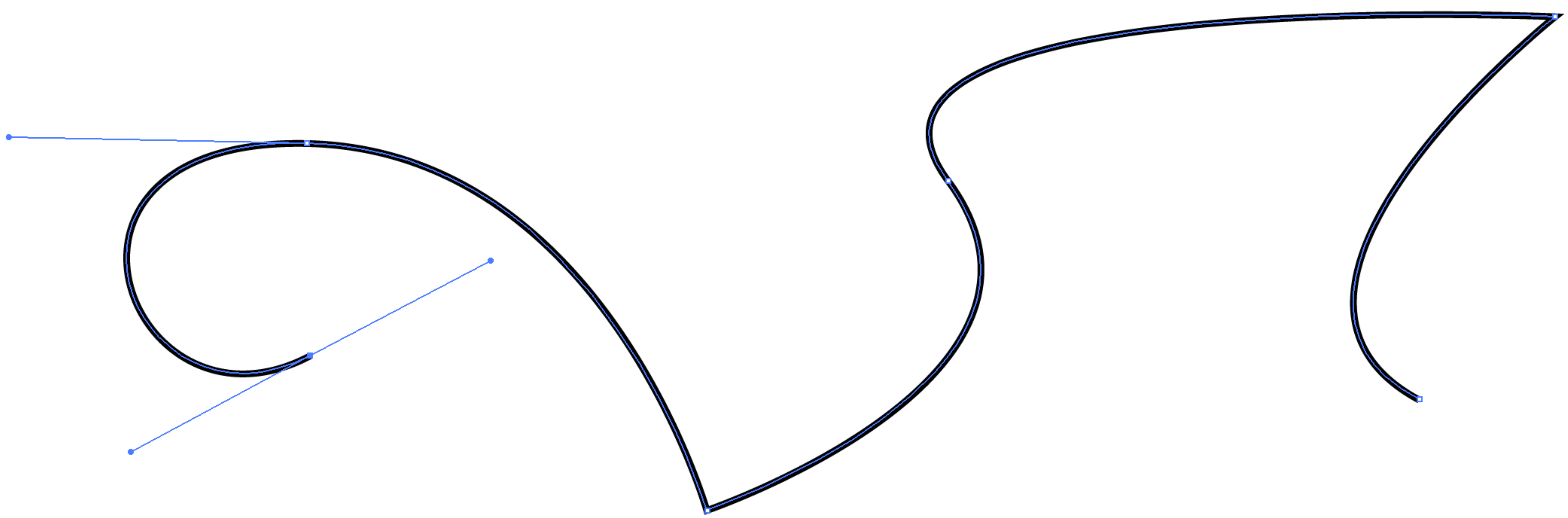
Uncommon

Jonathan Ragan-Kelley & Ren Ng

Piecewise Bézier Curves

Instead, chain many low-order Bézier curve

Piecewise cubic Bézier the most common technique



Widely used (fonts, paths, Illustrator, Keynote, …)

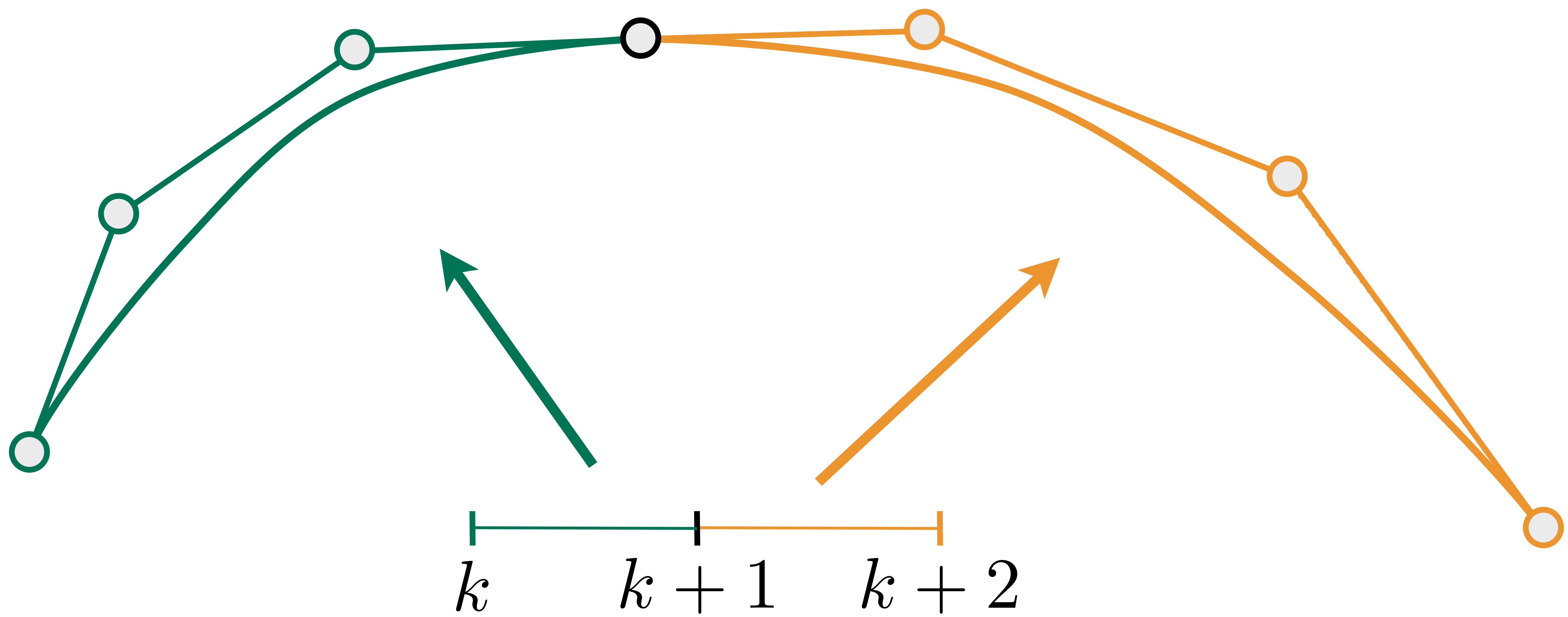
Piecewise Bézier Curve – Continuity

Two Bézier curves

$$\mathbf{a} : [k, k+1] \to \mathbb{R}^N$$

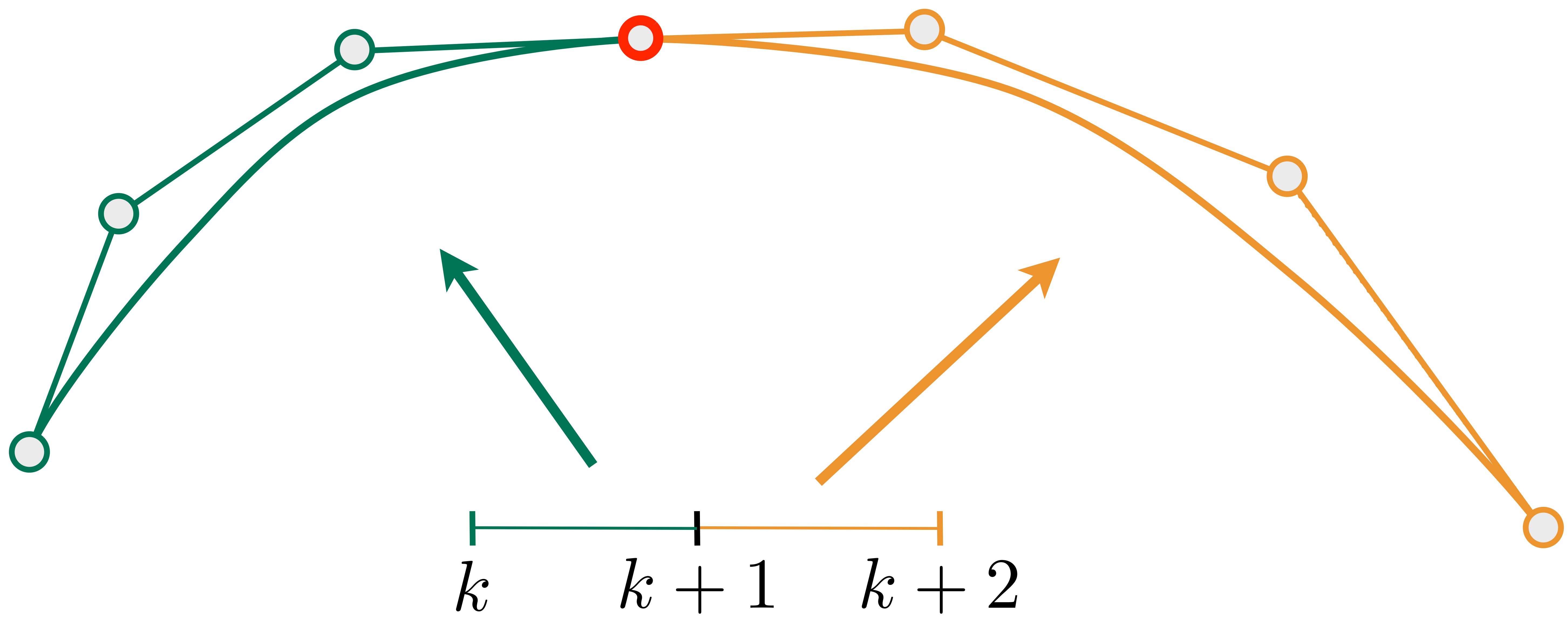
$$\mathbf{b} : [k+1, k+2] \to \mathbb{R}^N$$

Assuming integer partitions here, can generalize



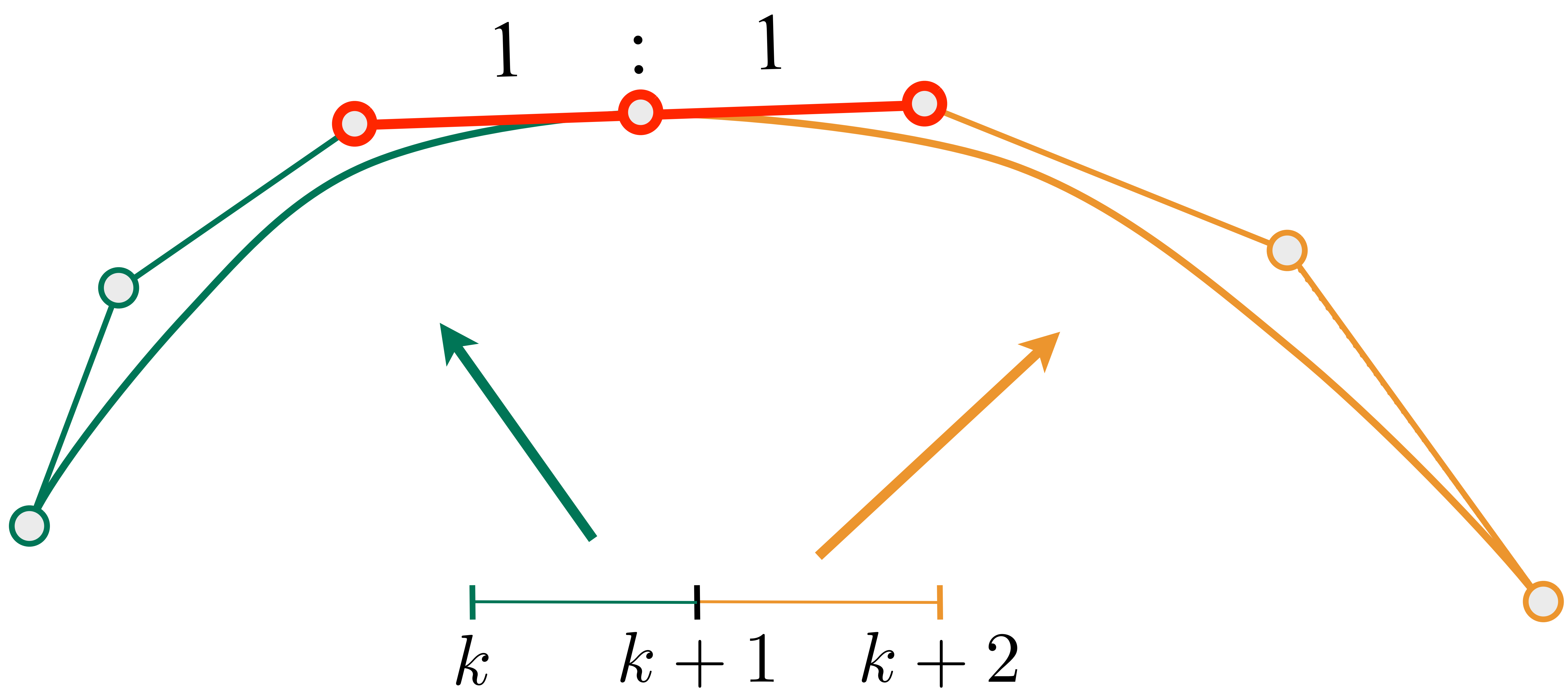
Piecewise Bézier Curve – Continuity

C^0 continuity: $\mathbf{a}_n = \mathbf{b}_0$



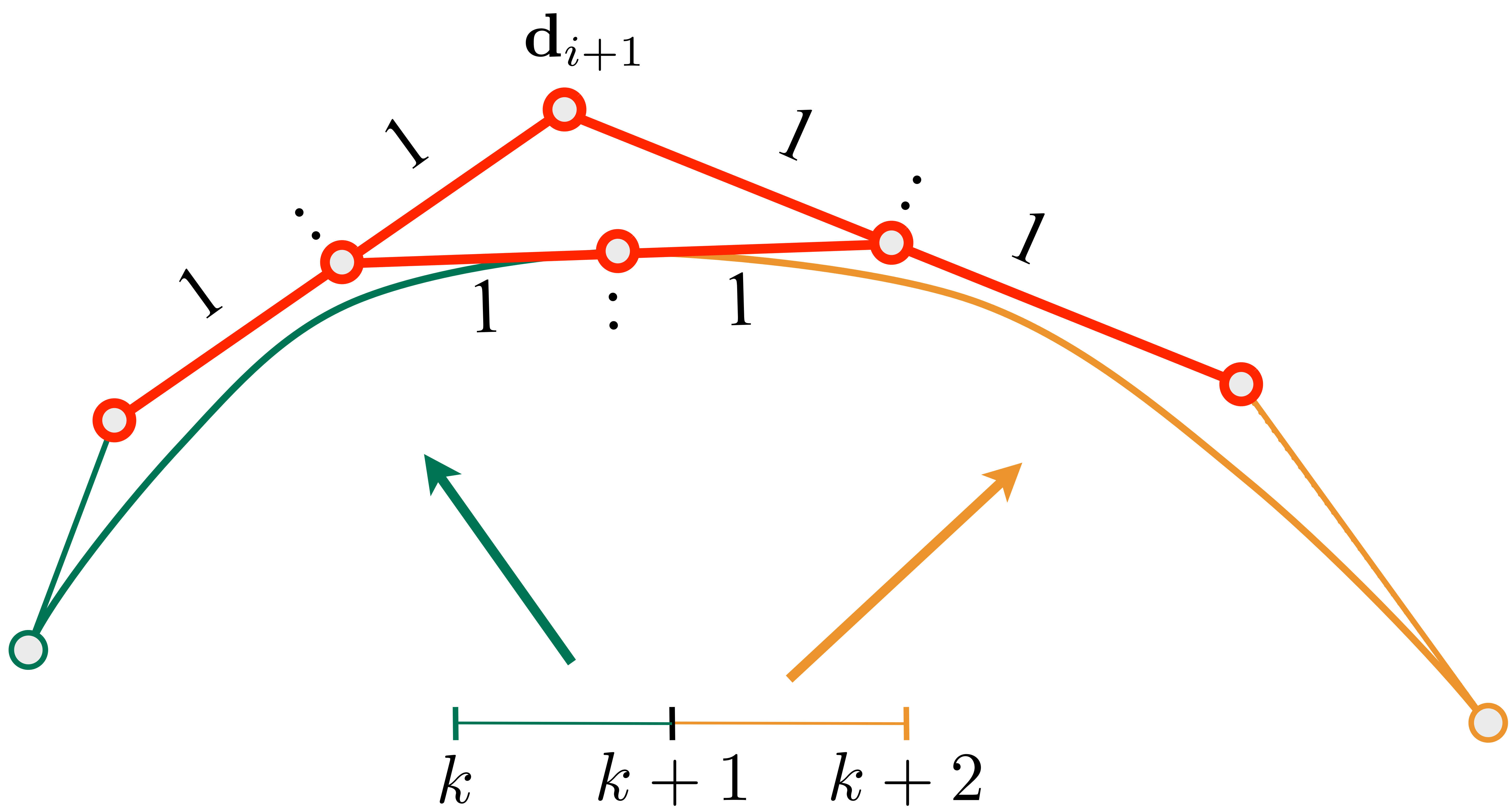
Piecewise Bézier Curve – Continuity

C¹ continuity: $$\mathbf{a}_n = \mathbf{b}_0 = \frac{1}{2}\left(\mathbf{a}_{n-1} + \mathbf{b}_1\right)$$



Piecewise Bézier Curve – Continuity

C^2 continuity: “A-frame” construction



Jonathan Ragan-Kelley & Ren Ng

Properties of Bézier Curves

Interpolates endpoints

- For cubic Bézier: $\mathbf{b}(0) = \mathbf{b}_0; \quad \mathbf{b}(1) = \mathbf{b}_3$

Tangent to end segments

- Cubic case: $\mathbf{b}'(0) = 3(\mathbf{b}_1 - \mathbf{b}_0); \quad \mathbf{b}'(1) = 3(\mathbf{b}_3 - \mathbf{b}_2)$

Affine transformation property

- Transform curve by transforming control points

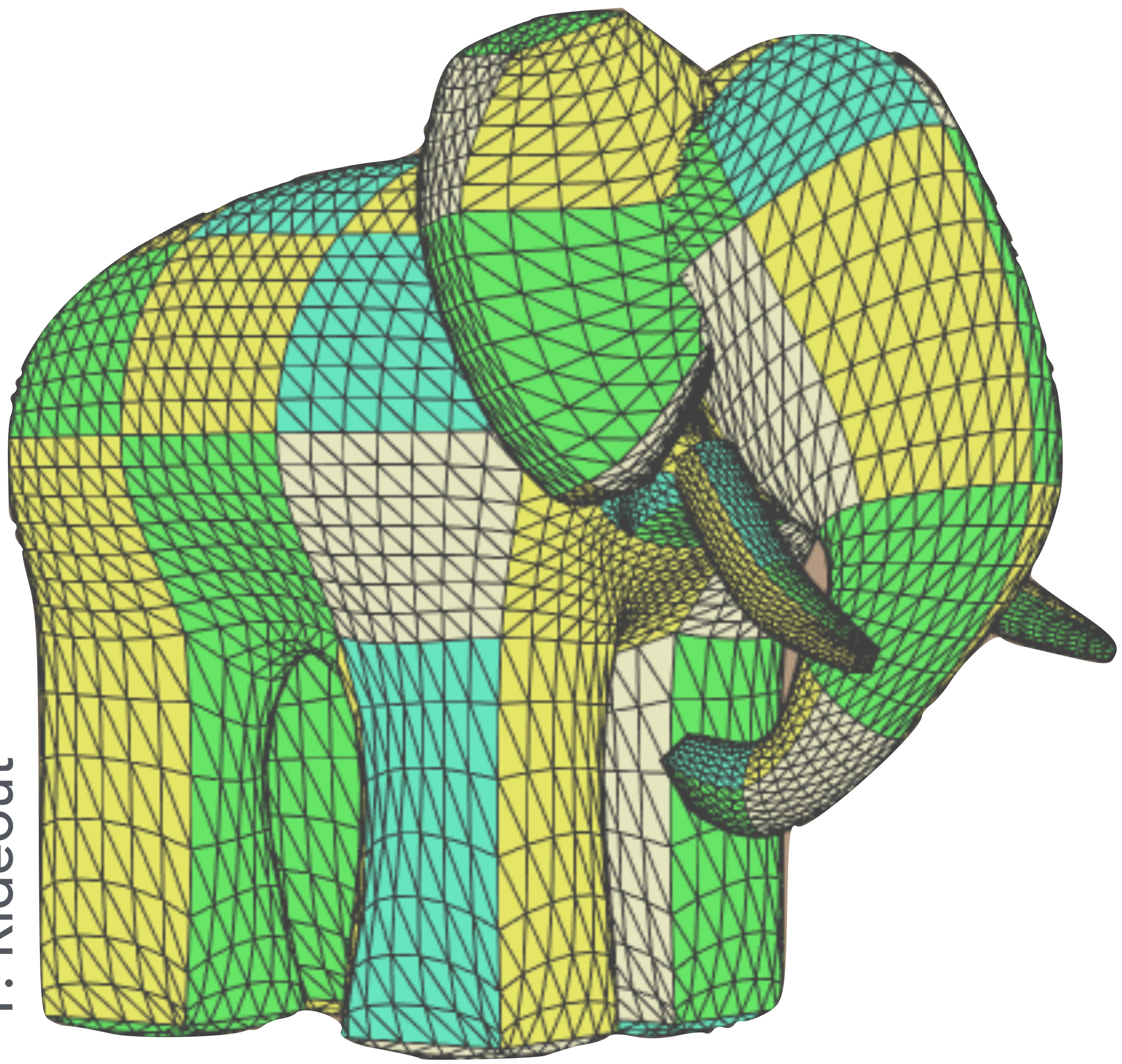
Convex hull property

- Curve is within convex hull of control points

Bézier Surfaces

Bézier Surfaces

Extend Bézier curves to surfaces



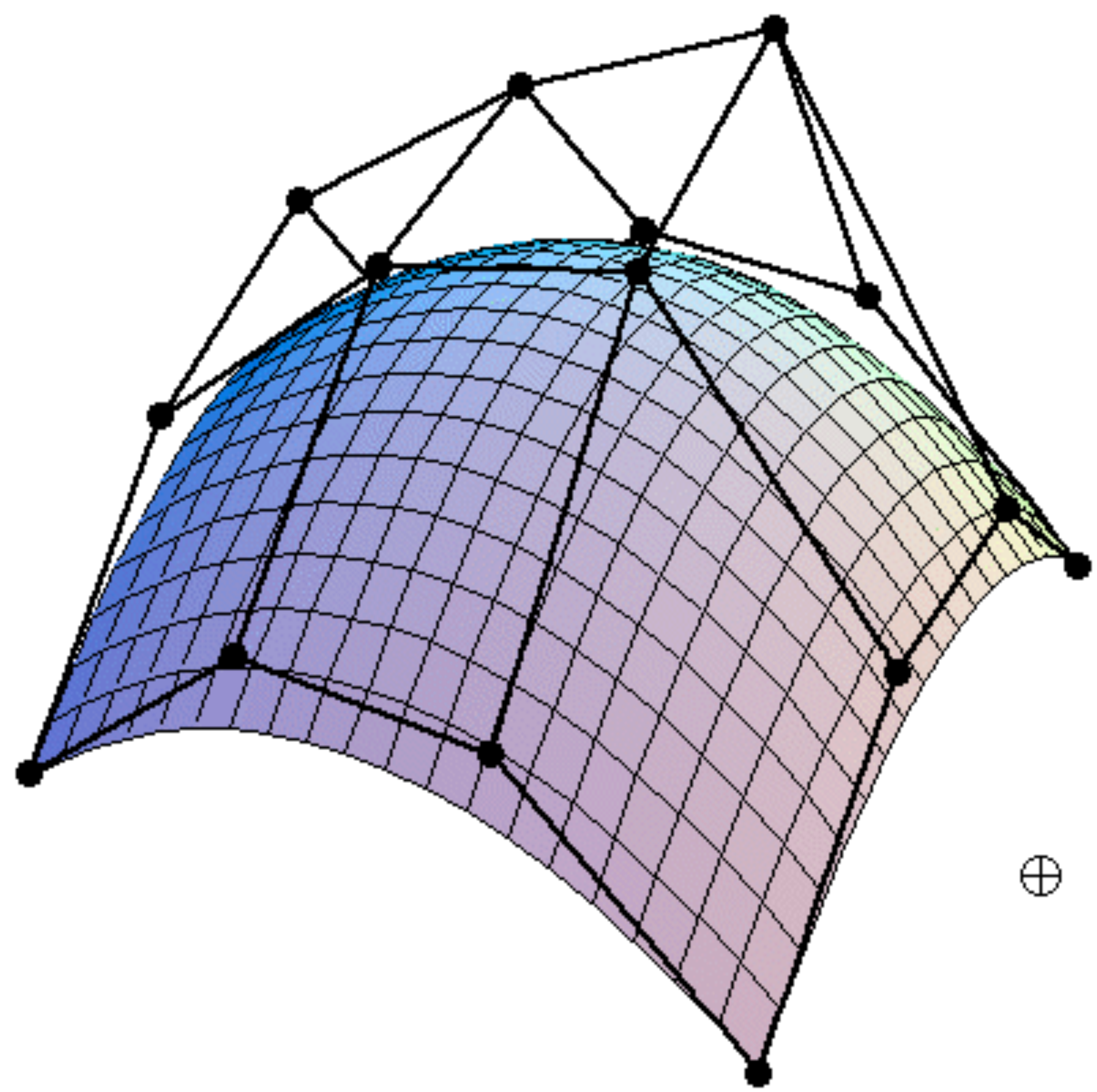
P. Rideout

Ed Catmull's "Gumbo" model

renderspirit.com

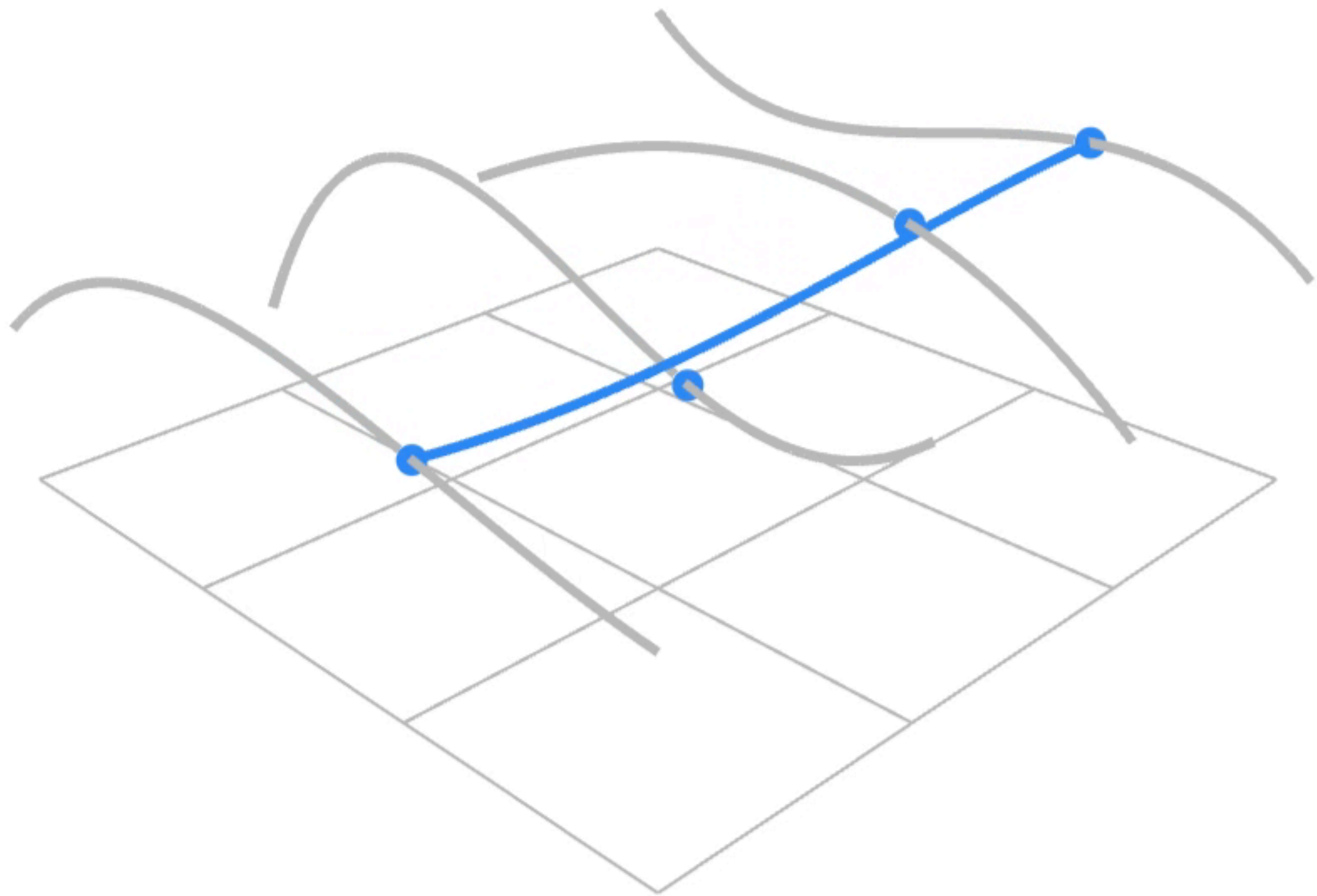
Utah Teapot

Bicubic Bézier Surface Patch



Bezier surface and 4 x 4 array of control points

Visualizing Bicubic Bézier Surface Patch



Animation: Steven Wittens, Making Things with Maths, http://acko.net

Visualizing Bicubic Bézier Surface Patch

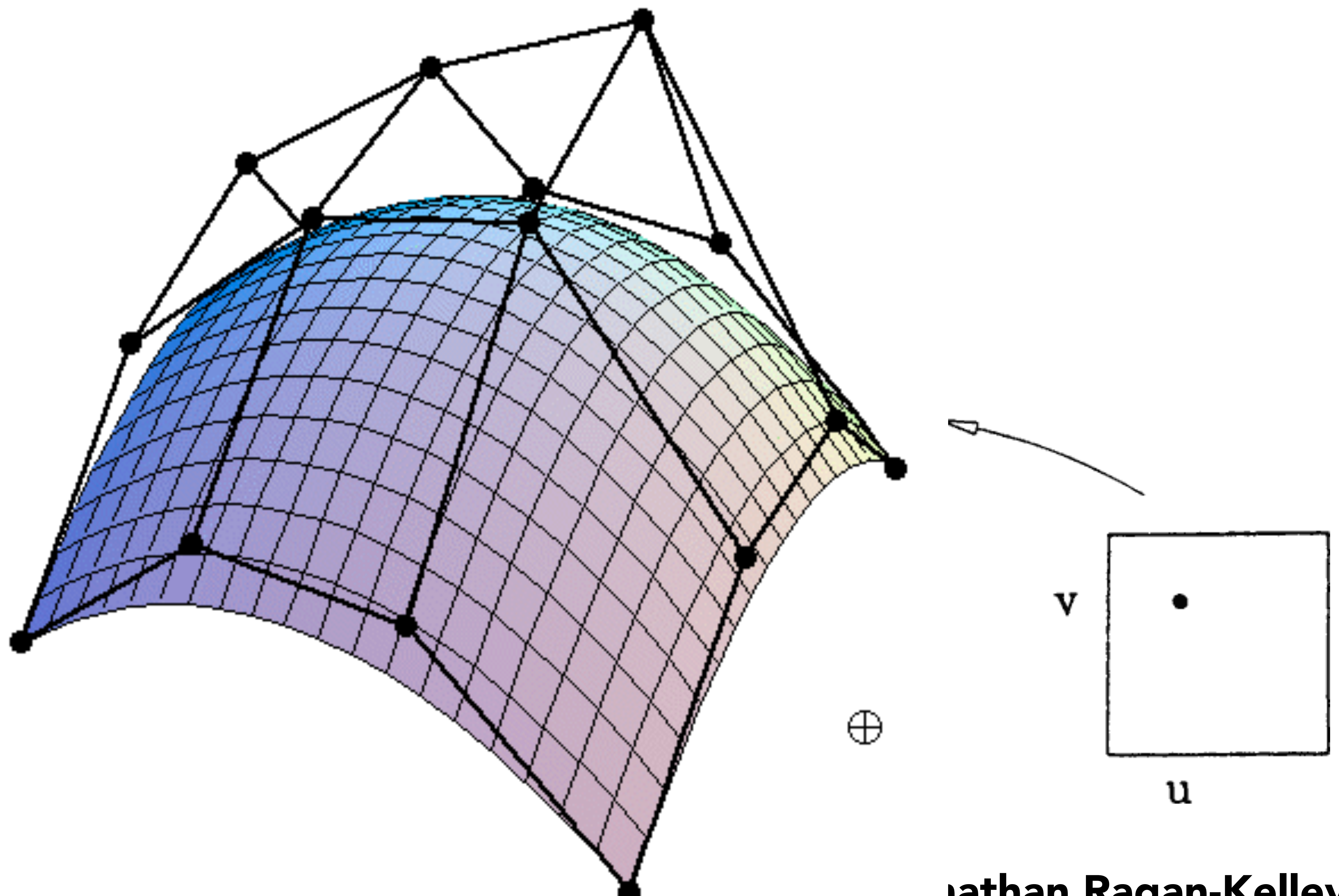
4x4 control points

- Each 4x1 control points in u define a Bezier curve
  - (4 Bezier curves in u)
- Corresponding points on these 4 Bezier curves define 4 control points for a "moving curve" in v
  - This "moving" curve sweeps out the 2D surface

Evaluating Bézier Surfaces

Evaluating Surface Position For Parameters (u,v)

For bi-cubic Bezier surface patch,
- Input: 4x4 control points
- Output is 2D surface parameterized by (u,v) in $[0,1]^2$

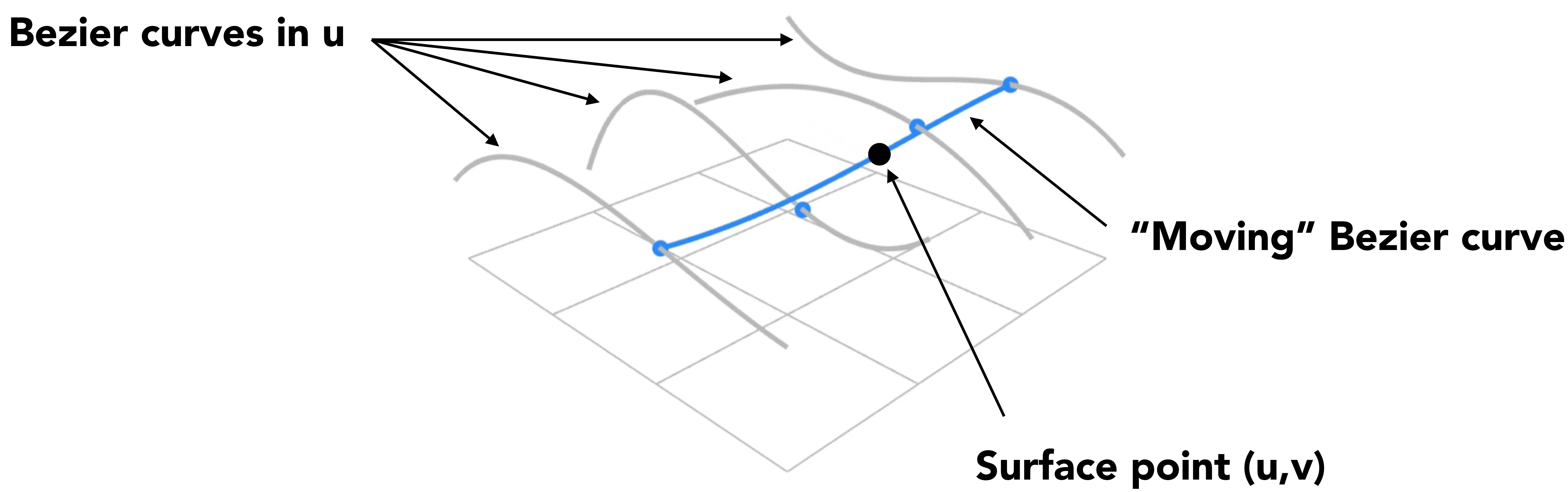


Method 1: Separable 1D de Casteljau Algorithm

Goal: Evaluate surface position corresponding to (u,v)

(u,v)-separable application of de Casteljau algorithm

- Use de Casteljau to evaluate point u on each of the 4 Bezier curves in u. This gives 4 control points for the “moving” Bezier curve
- Use 1D de Casteljau to evaluate point v on the “moving” curve



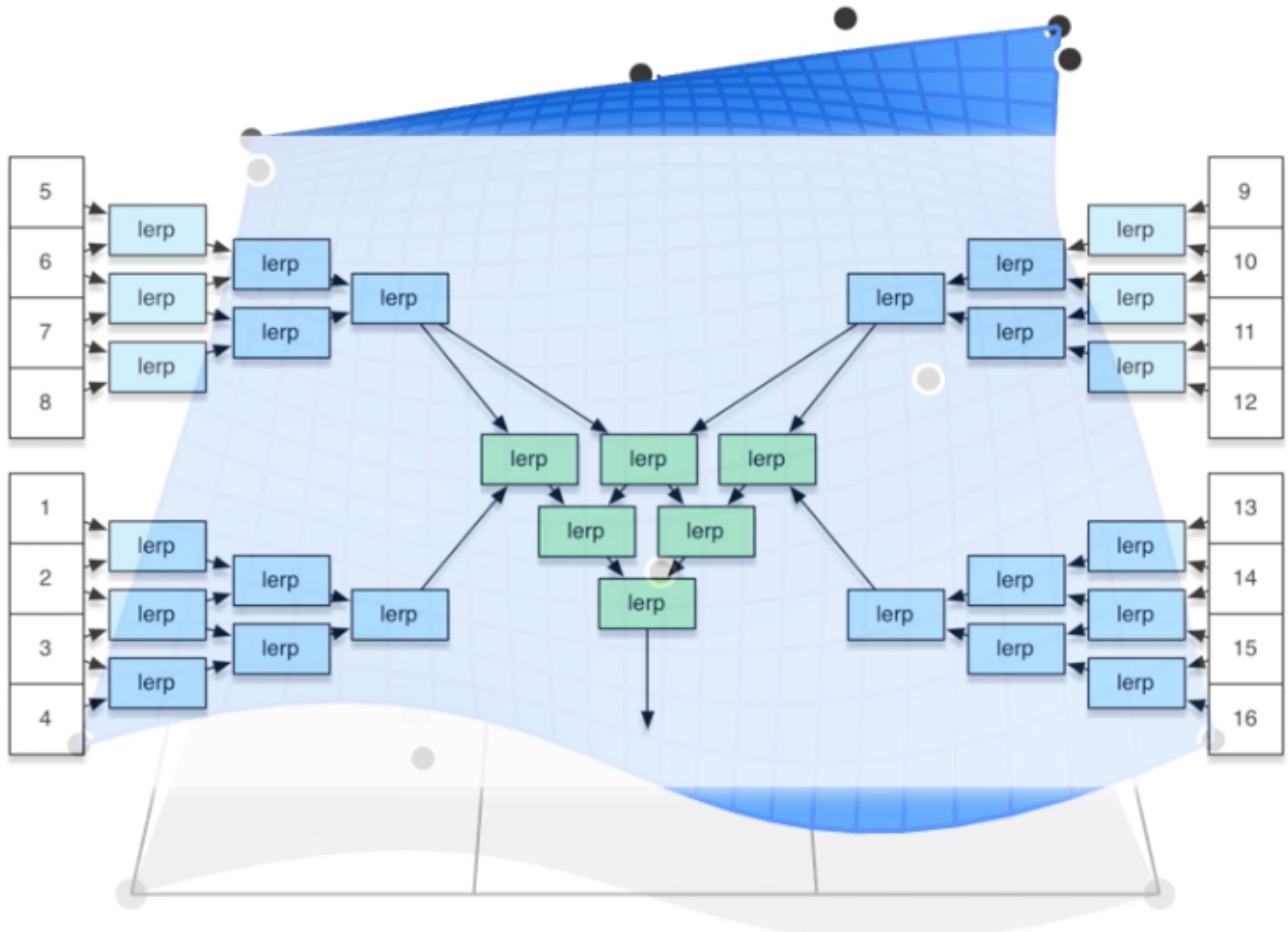
Method 1: Separable 1D de Casteljau Algorithm

Ev

Me
Al

- …ne
…or

-



Method 2: Algebraic Evaluation

Let the moving curve be a degree m Bézier curve

$$\mathbf{b}^m(u) = \sum_{i=0}^{m} \mathbf{b}_i B_i^m(u)$$

(remember, Bernstein polynomials)

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Let each control point $\mathbf{b}_i$ be moving along a Bézier curve of degree n

$$\mathbf{b}_i = \mathbf{b}_i(v) = \sum_{j=0}^{n} \mathbf{b}_{i,j} B_j^n(v)$$

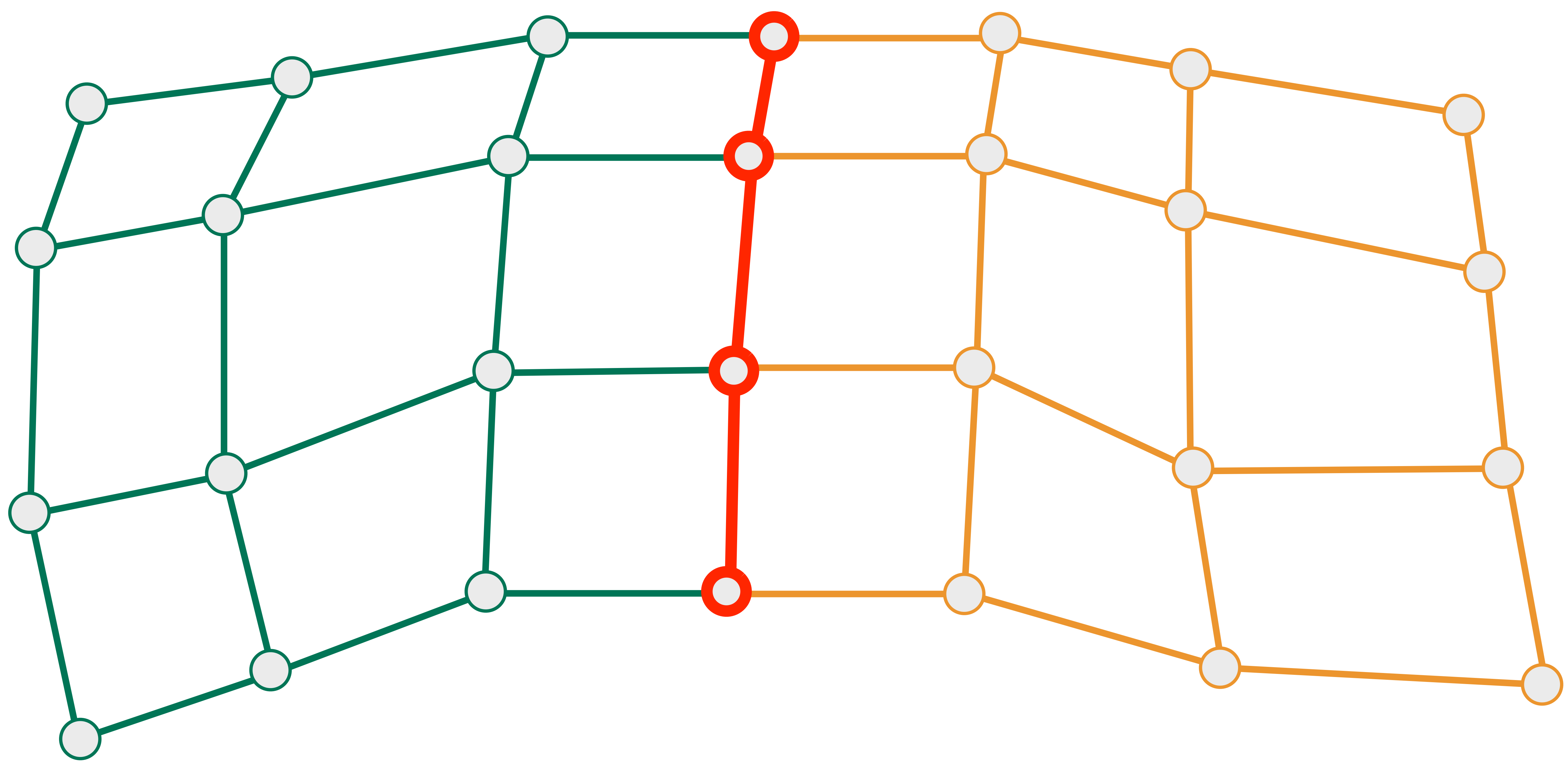
Tensor product Bézier patch

$$\mathbf{b}^{m,n}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{b}_{i,j} B_i^m(u) B_j^n(v)$$

Bézier Surface Continuity

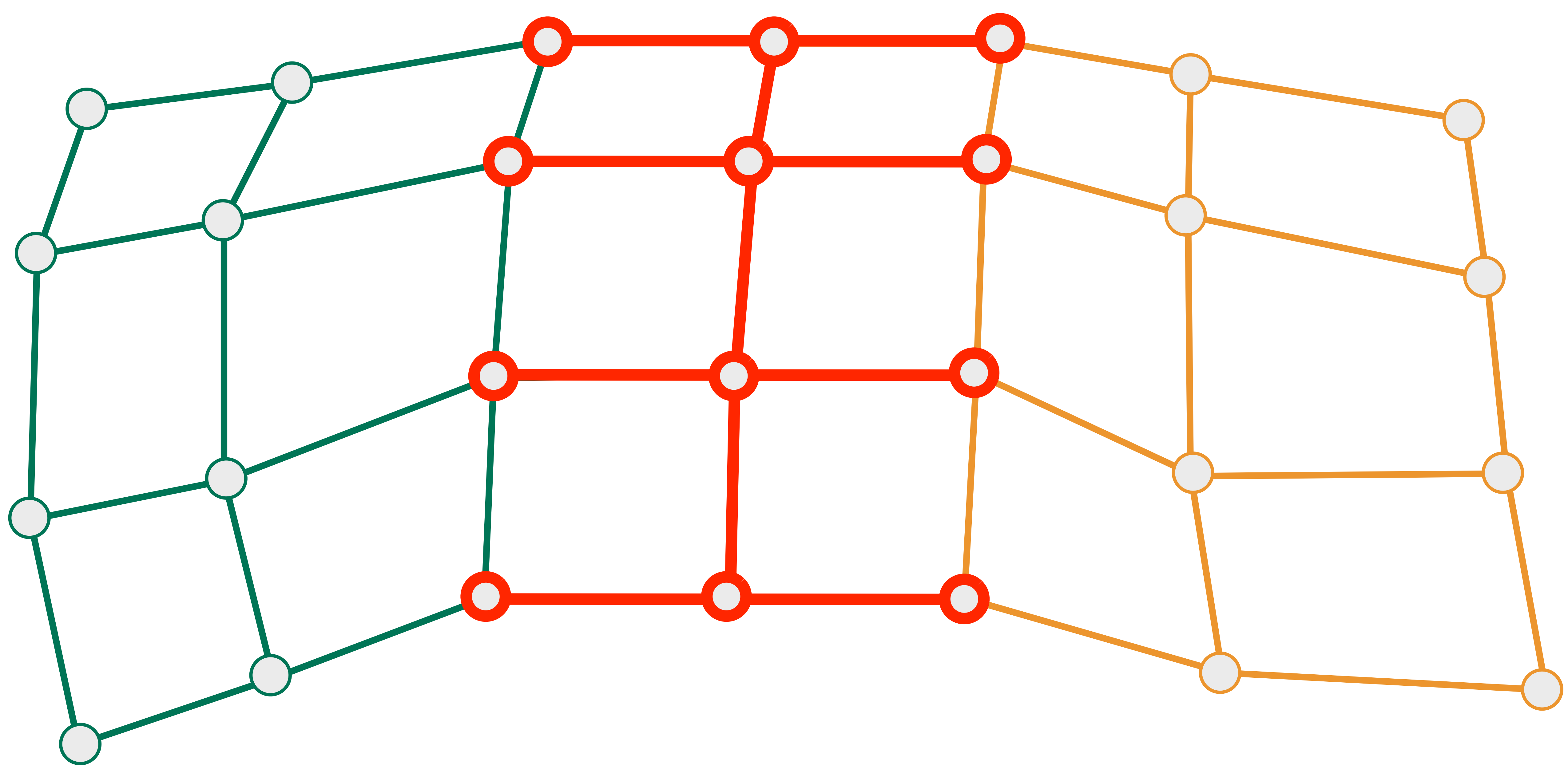
Piecewise Bézier Surfaces

C^0 continuity: Boundary curves



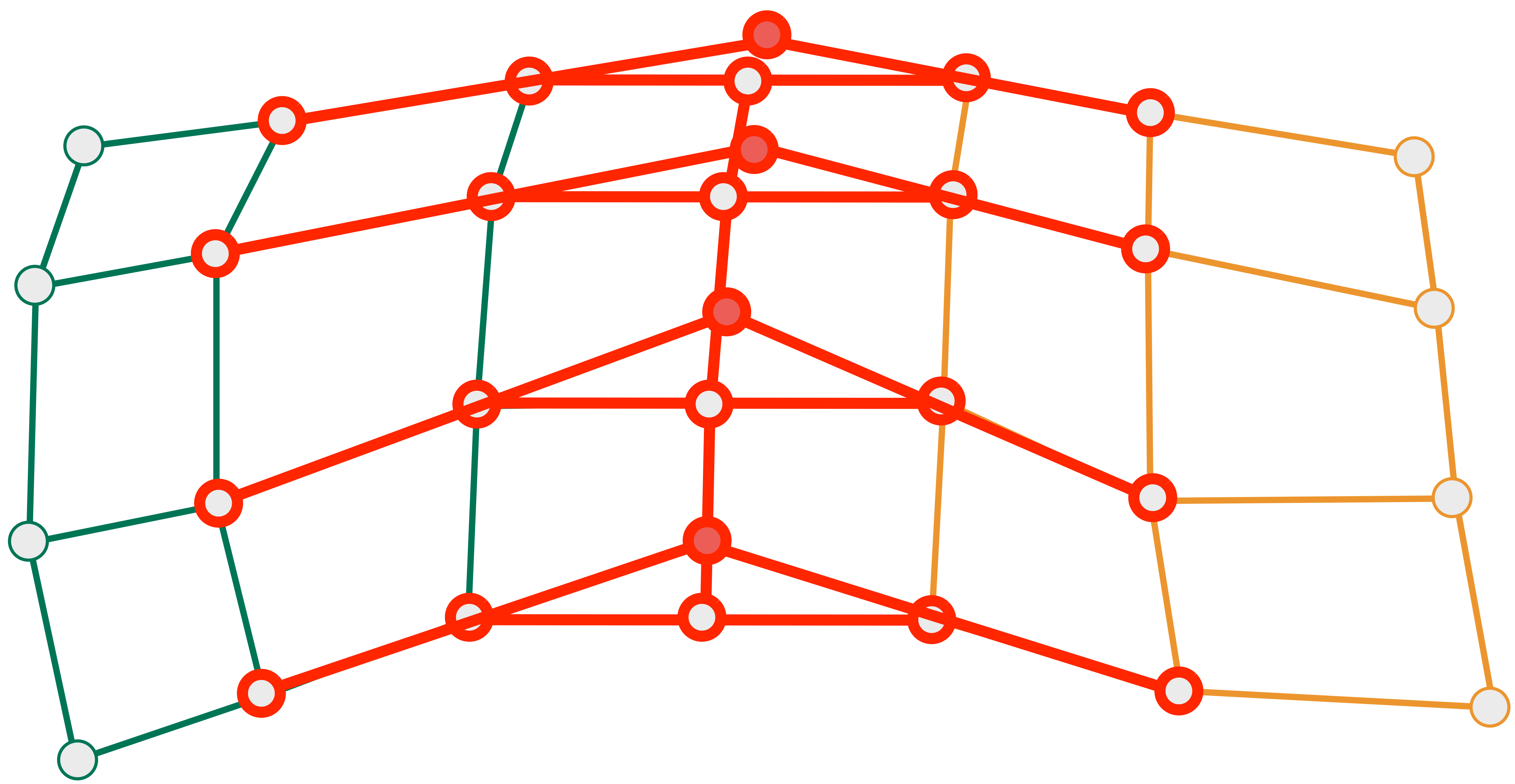
Piecewise Bézier Surfaces

C^1 continuity: Collinearity



Piecewise Bézier Surfaces

C^2 continuity: A-frames



Things to Remember

Splines

- Cubic Hermite and Catmull-Rom interpolation
- Matrix representation of cubic polynomials

Bézier curves

- Easy-to-control spline
- Recursive linear interpolation – de Casteljau algorithm
- Properties of Bézier curves
- Piecewise Bézier curve – continuity types and how to achieve

Bézier surfaces

- Bicubic Bézier patches – tensor product surface
- 2D de Casteljau algorithm

Acknowledgments

Thanks to Pat Hanrahan, Mark Pauly and Steve Marschner for presentation resources.