CS130 : Computer Graphics

Curves (cont.)

Tamar Shinar
Computer Science & Engineering
UC Riverside

Blending Functions

Blending functions are more convenient basis than monomial basis

- "canonical form" (monomial basis)

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

- "geometric form" (blending functions)

$$\mathbf{f}(u) = b_0(u)\mathbf{p}_0 + b_1(u)\mathbf{p}_1 + b_2(u)\mathbf{p}_2 + b_3(u)\mathbf{p}_3$$

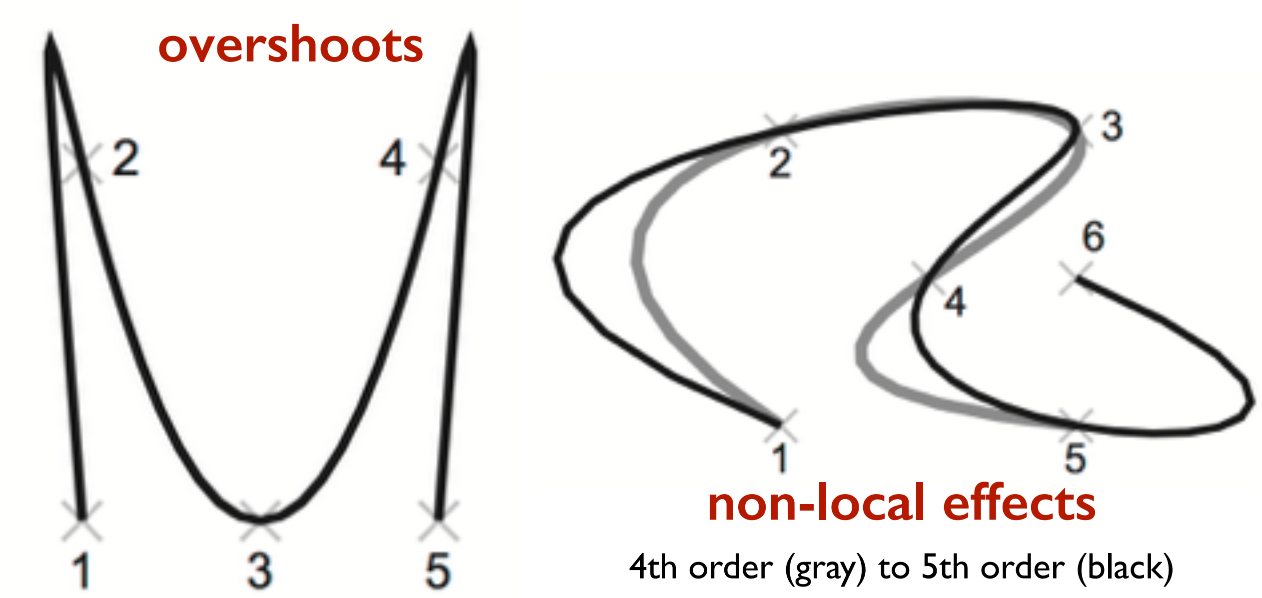
- geometric form (bottom) is more intuitive because it combines control points with blending functions
[see Shirley Section 15.3]

Interpolating Polynomials

Interpolating polynomials

- Given n+1 data points, can find a unique interpolating polynomial of degree n
- Different methods:
  - Vandermonde matrix
  - Lagrange interpolation
  - Newton interpolation

higher order interpolating polynomials are rarely used

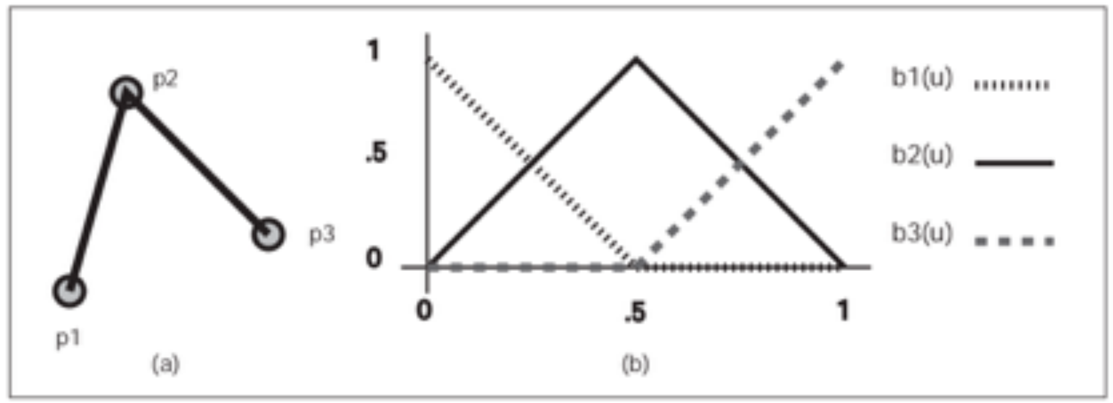


These images demonstrate problems with using higher order polynomials:
- overshoots
- non-local effects (in going from the 4th order polynomial in grey to the 5th order polynomial in black)

Piecewise Polynomial Curves

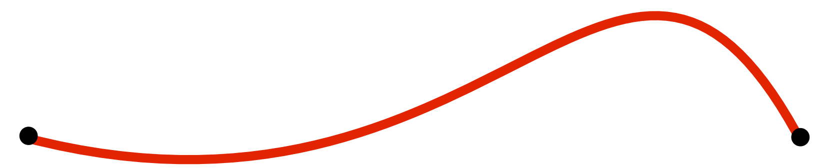
Example: blending functions for two line segments

$$\mathbf{f}(u) = \begin{cases} \mathbf{f}_1(2u) & u \leq 0.5 \\ \mathbf{f}_2(2u-1) & u > 0.5 \end{cases}$$



b1(u) = 1-2u, 0 <= u <= .5
0 .5 <= u <= 1
b2(u) = 2u, 0 <= u <= .5
2(1-u), .5 <= u <= 1
b3(u) = 0 , 0 <= u <= .5
2u-1, .5 <= u <= 1

Cubics



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

- Allow up to C^2 continuity at knots
- need 4 control points
  - may be 4 points on the curve, combination of points and derivatives, ...
- good smoothness and computational properties

need 4 control points: might be 4 points on the curve, combination of points and derivatives, ...

We can get any 3 of 4 properties

1. piecewise cubic
2. curve interpolates control points
3. curve has local control
4. curves has C2 continuity at knots

Cubics

- Natural cubics
  - C2 continuity
  - n points -> n-1 cubic segments
- control is non-local :(
- ill-conditioned x(

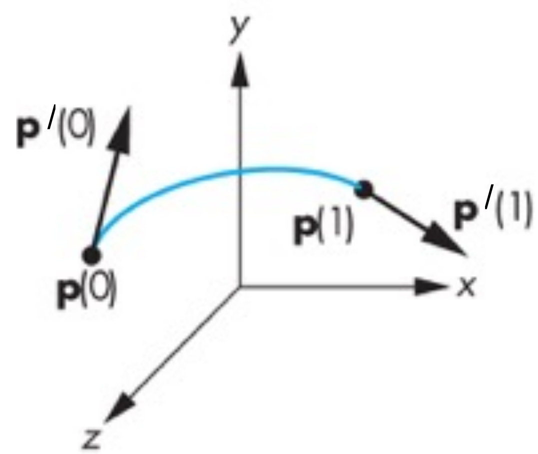
Cubic Hermite Curves

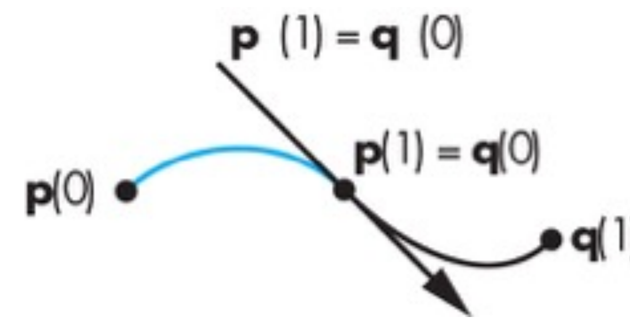
- C1 continuity
- specify both positions and derivatives

Cubic Hermite Curves

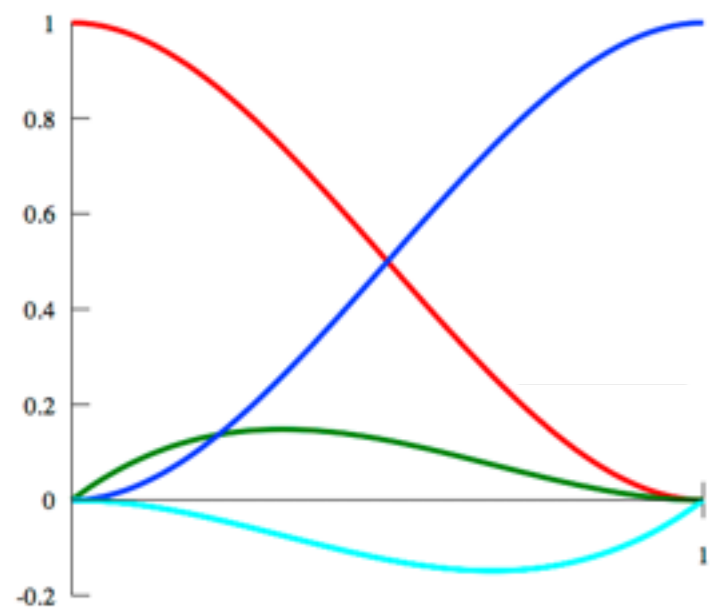
Specify endpoints and derivatives

construct curve with C^1 continuity





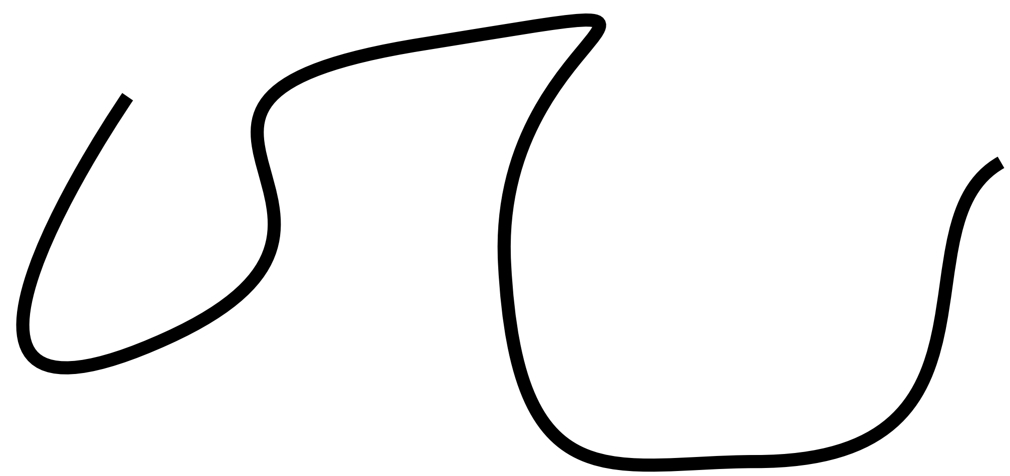
Hermite blending functions



[Wikimedia Commons]

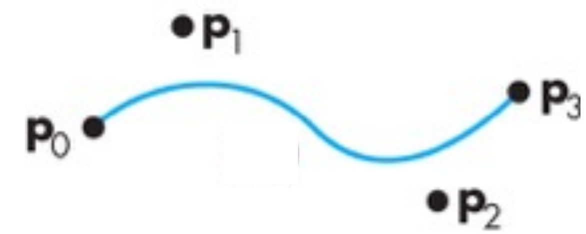
$$b_0(u) = 2u^3 - 3u^2 + 1$$
$$b_1(u) = -2u^3 + 3u^2$$
$$b_2(u) = u^3 - 2u^2 + u$$
$$b_3(u) = u^3 - u^2$$

Example: keynote curve tool



Interpolating vs. Approximating Curves

Interpolating

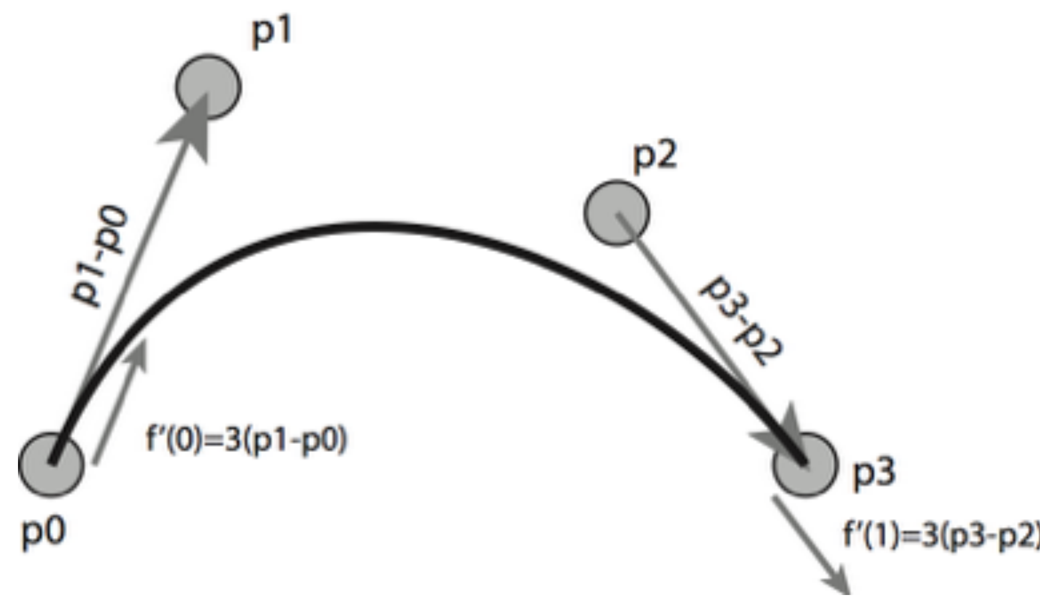


Approximating
(non-interpolating)

approximating

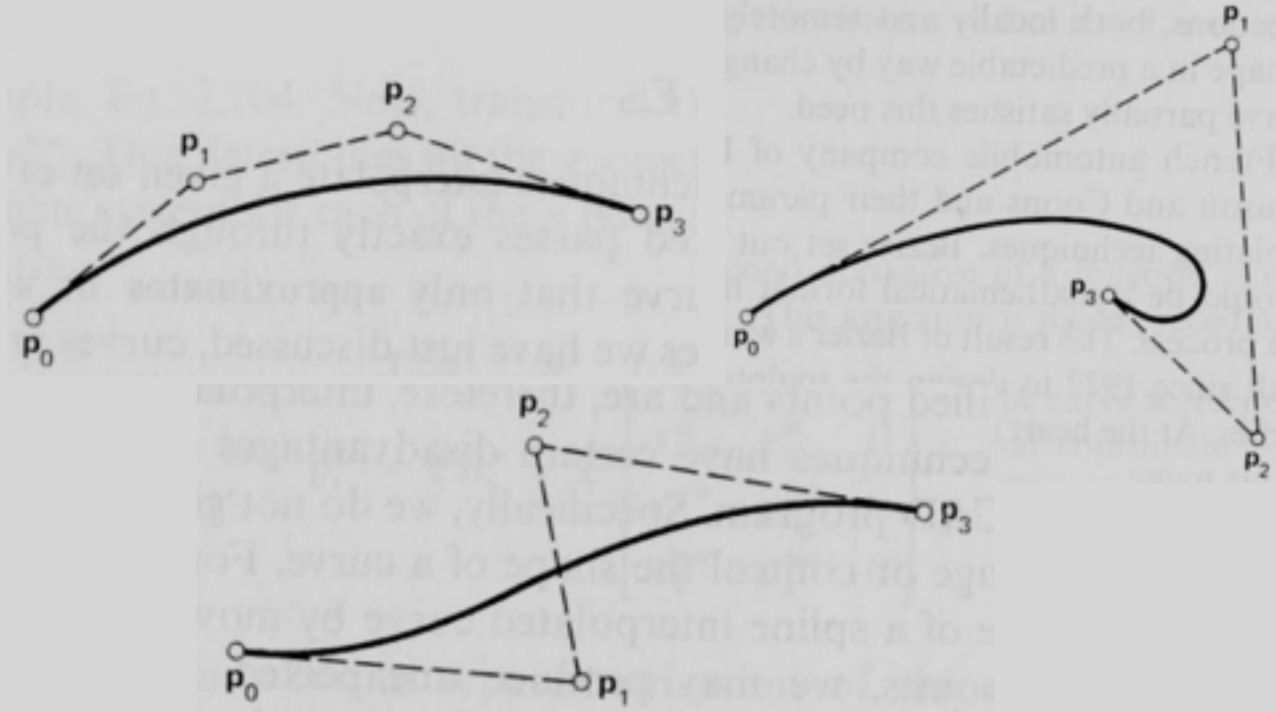
Cubic Bezier Curves

Cubic Bezier Curves

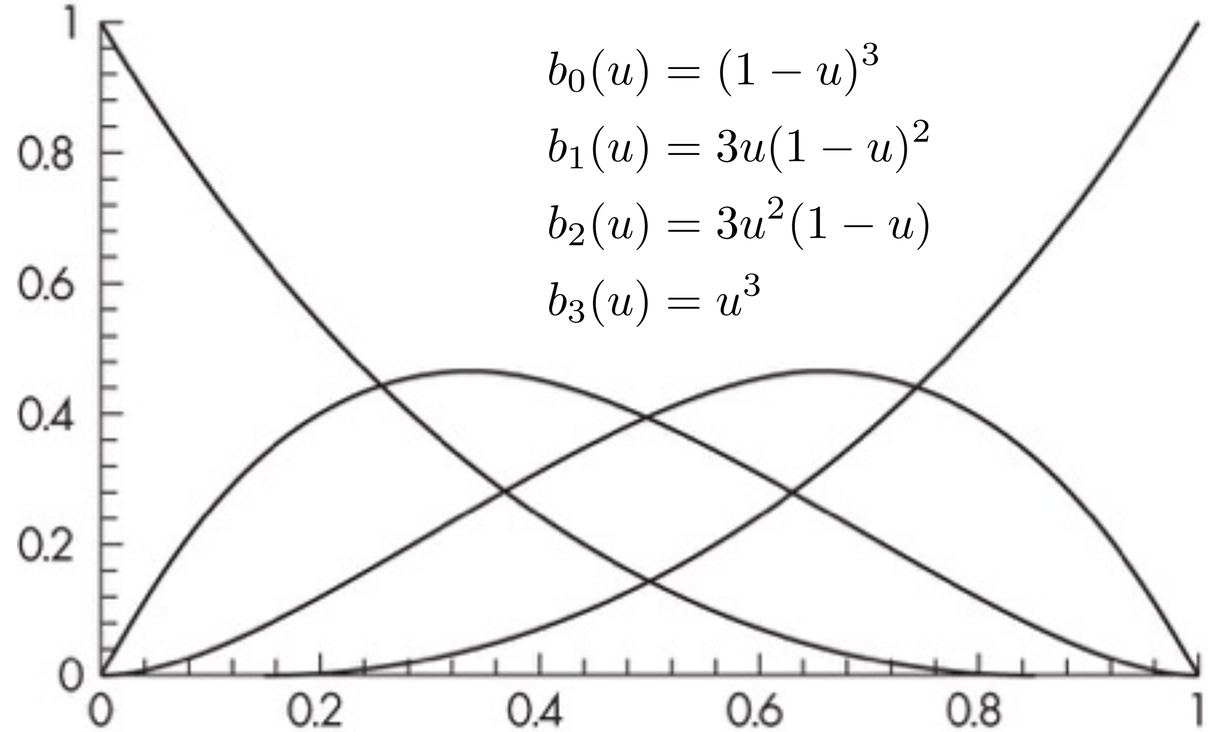


-The curve interpolates its first (u=0) and last (u = 1) control points
- first derivative at the beginning is the vector from first to second point, scaled by degree

Cubic Bezier Curve Examples

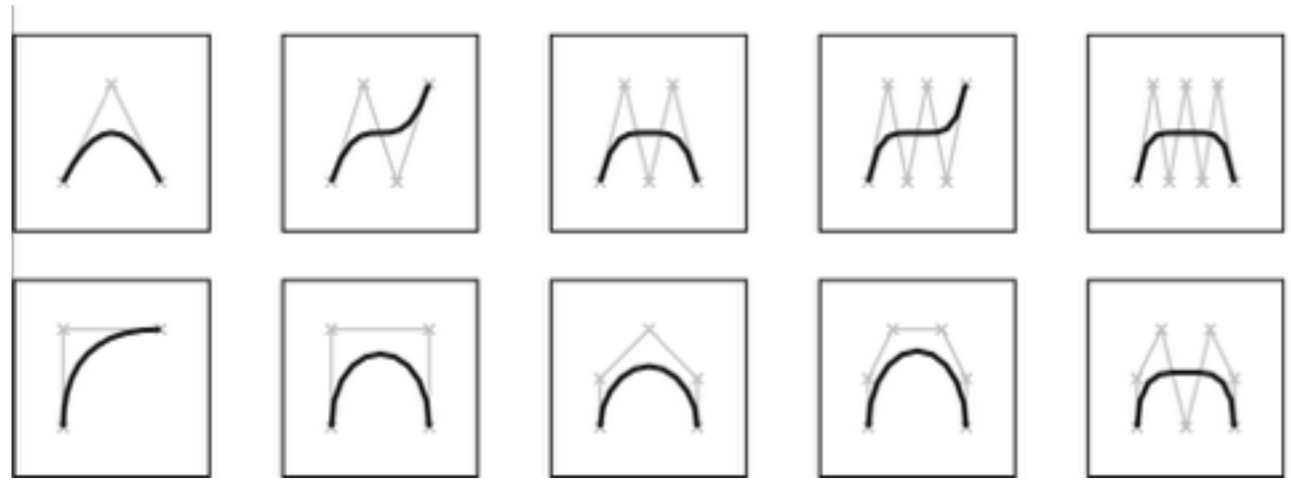


Cubic Bezier blending functions



$$b_0(u) = (1-u)^3$$
$$b_1(u) = 3u(1-u)^2$$
$$b_2(u) = 3u^2(1-u)$$
$$b_3(u) = u^3$$

Bezier Curves Degrees 2-6

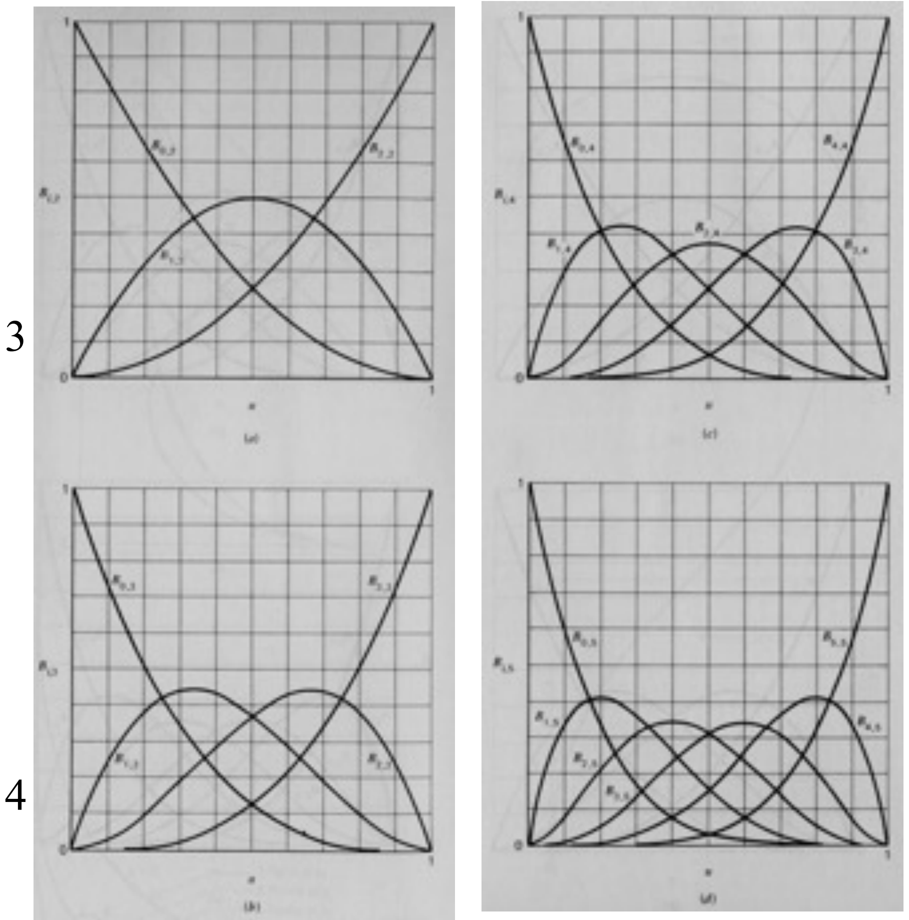


Bernstein Polynomials

- The blending functions are a special case of the Bernstein polynomials

$$b_{\mathrm{kd}}(u) = \frac{d!}{k!(d-k)!} u^k (1-u)^{d-k}$$

- These polynomials give the blending polynomials for any degree Bezier form
  - All roots at 0 and 1
  - For any degree they all sum to 1
  - They are all between 0 and 1 inside (0,1)



n = 3

n = 5

n = 4

n = 6

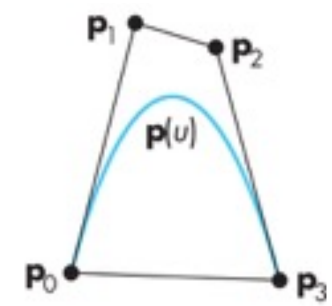
Bezier Curve Properties

- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision

Shirley Section 15.6.1 (p. 368)

Bezier Curve Properties

- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision



Shirley Section 15.6.1 (p. 368)

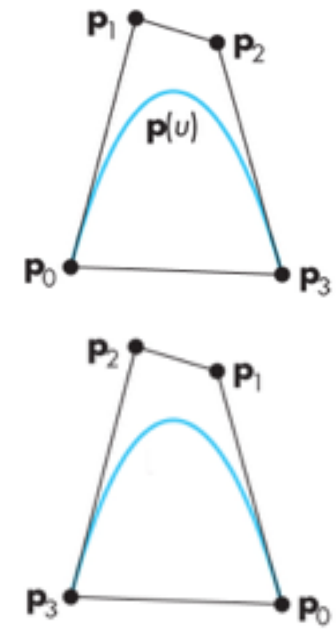
Bezier Curve Properties

- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision

Shirley Section 15.6.1 (p. 368)

Bezier Curve Properties

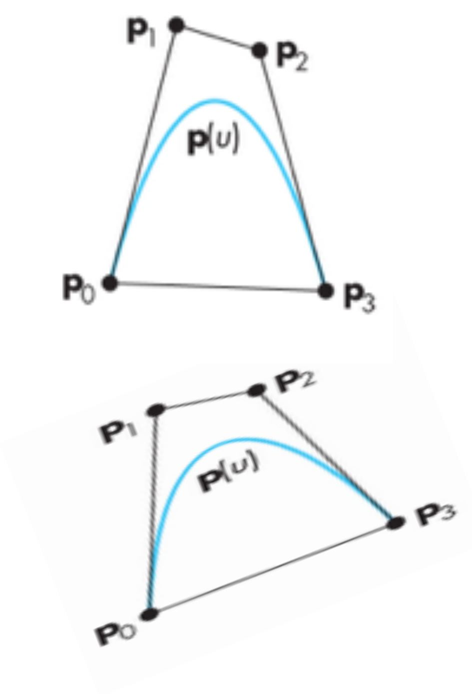
- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision



Shirley Section 15.6.1 (p. 368)

Bezier Curve Properties

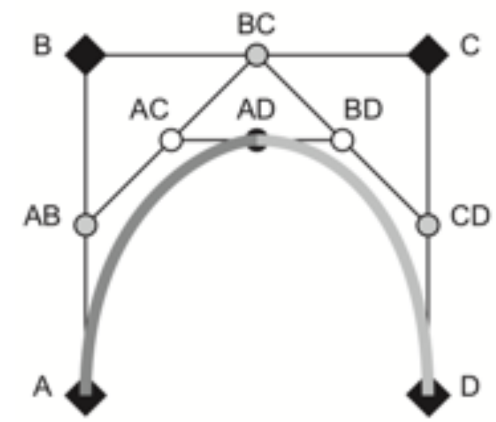
- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision



Shirley Section 15.6.1 (p. 368)

Bezier Curve Properties

- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision



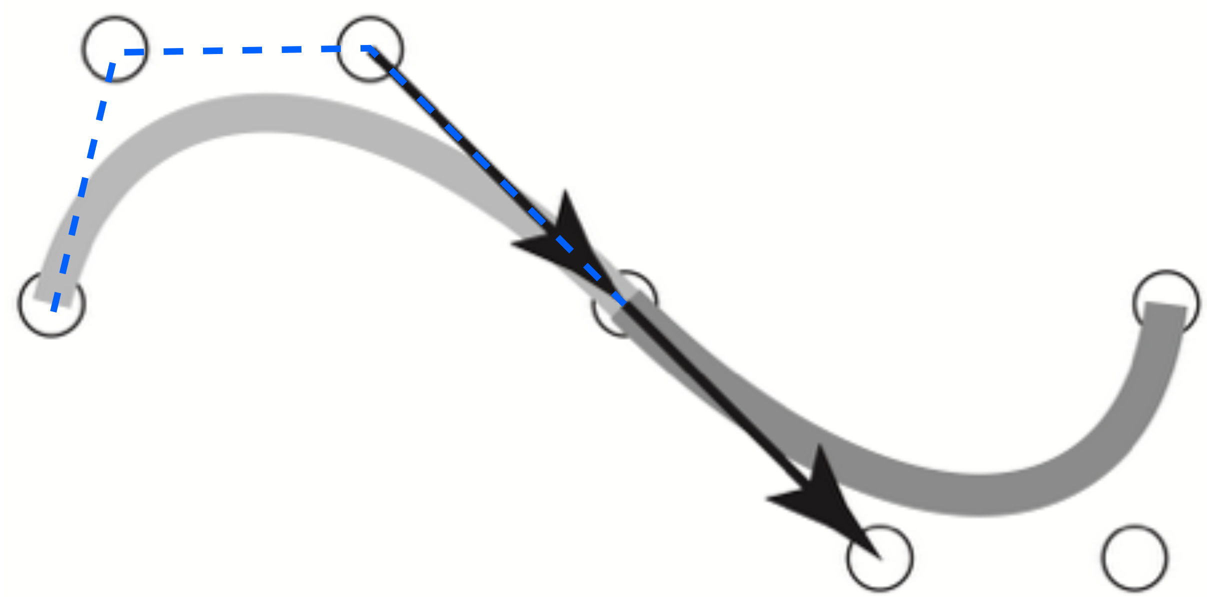
Shirley Section 15.6.1 (p. 368)

Bezier Curve Properties

- curve lies in the convex hull of the data
- variation diminishing
- symmetry
- affine invariant
- efficient evaluation and subdivision

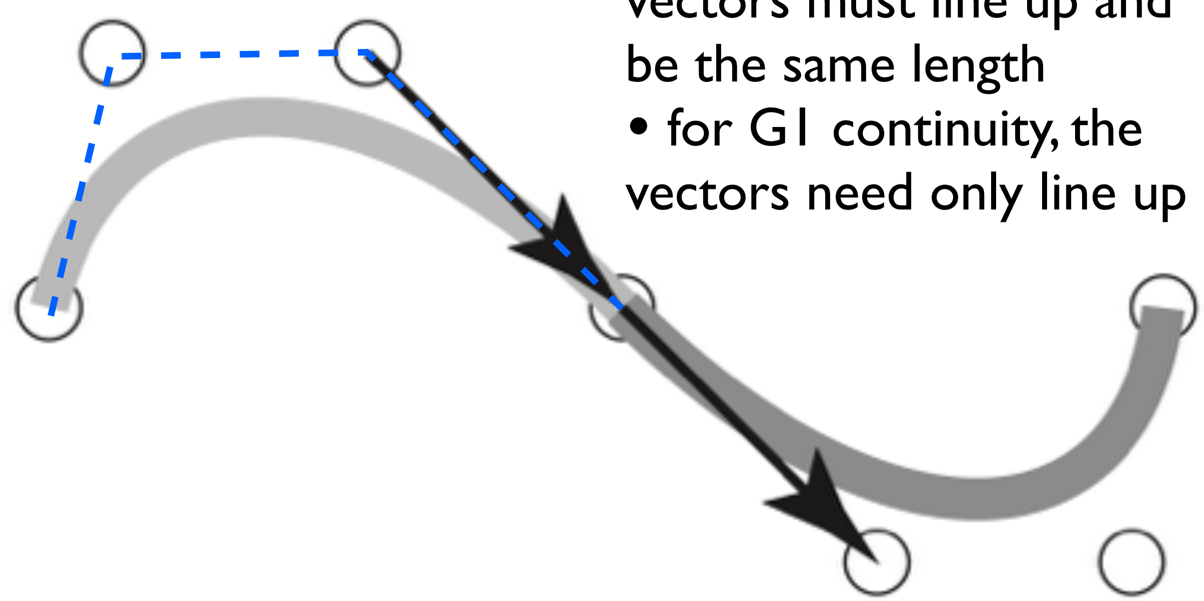
Shirley Section 15.6.1 (p. 368)

Joining Cubic Bezier Curves



for C1 continuity, the vectors must line up and be the same length
for G1 continuity, the vectors need only line up

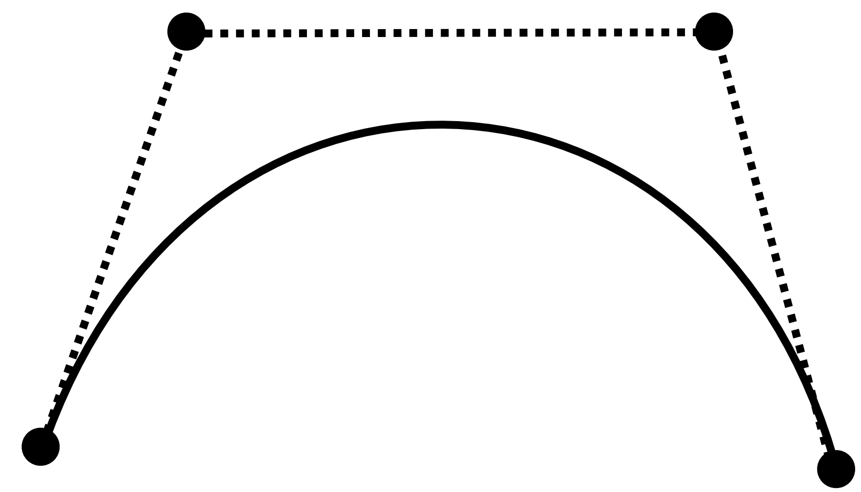
Joining Cubic Bezier Curves



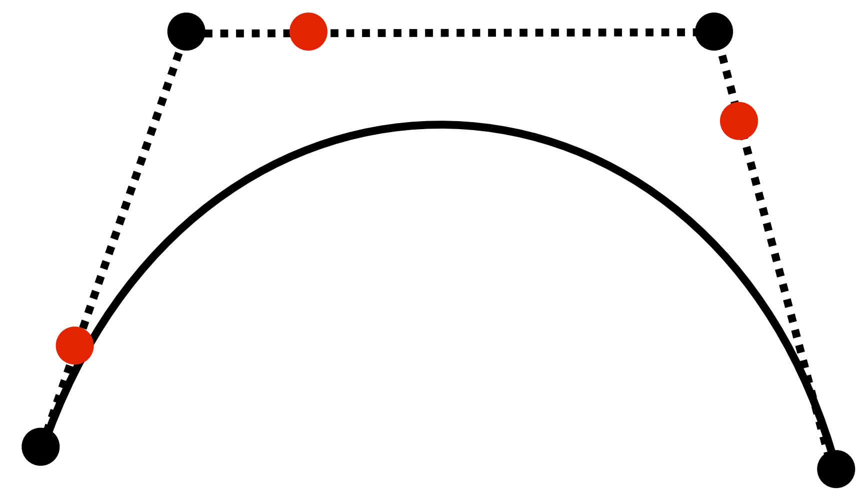
- for C1 continuity, the vectors must line up and be the same length
- for G1 continuity, the vectors need only line up

for C1 continuity, the vectors must line up and be the same length
for G1 continuity, the vectors need only line up

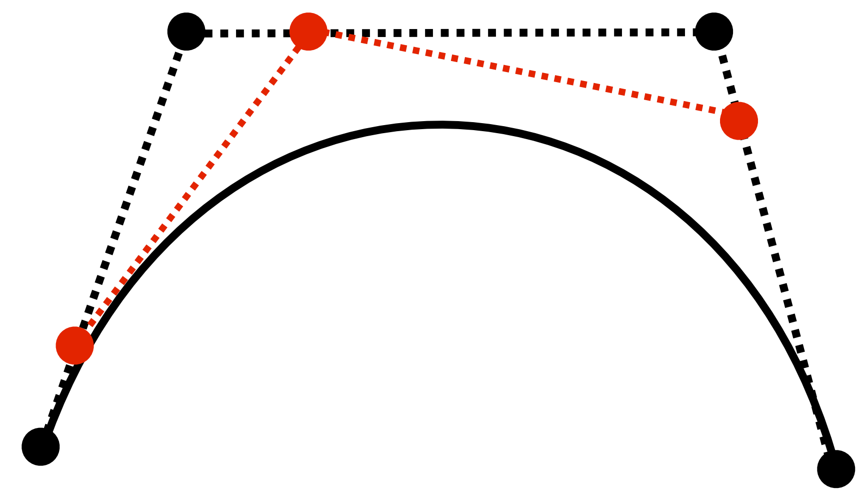
Evaluating p(u) geometrically



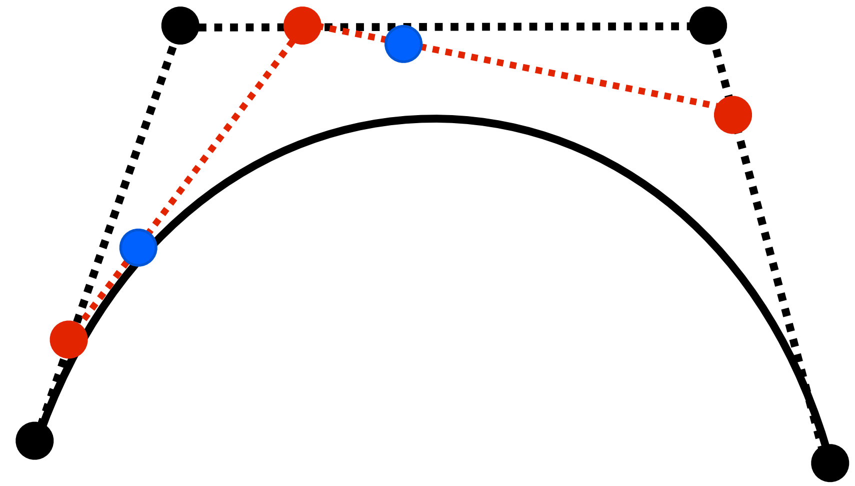
Evaluating p(u) geometrically



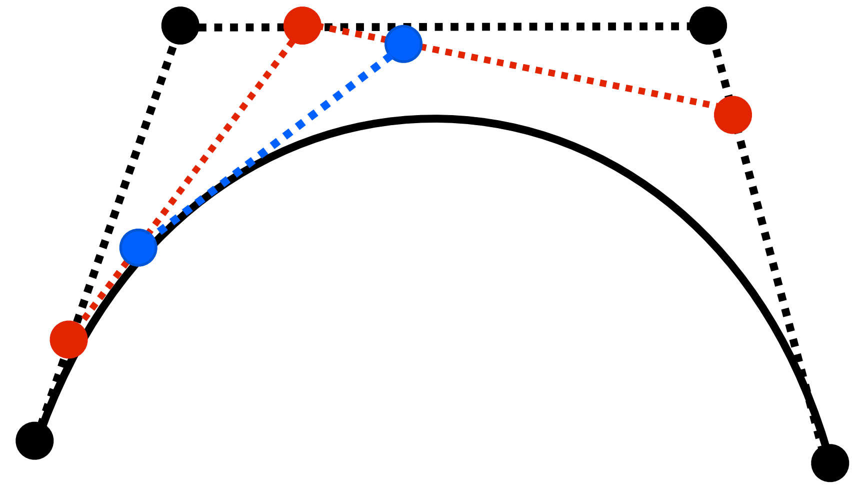
Evaluating p(u) geometrically



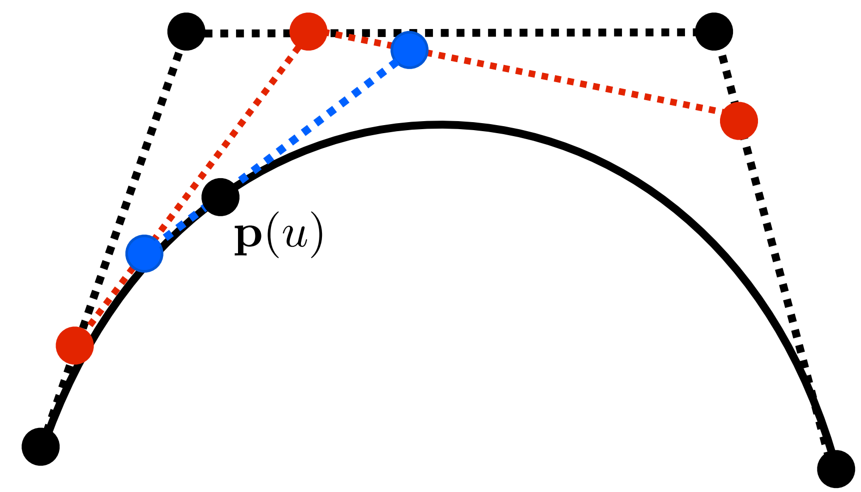
Evaluating p(u) geometrically



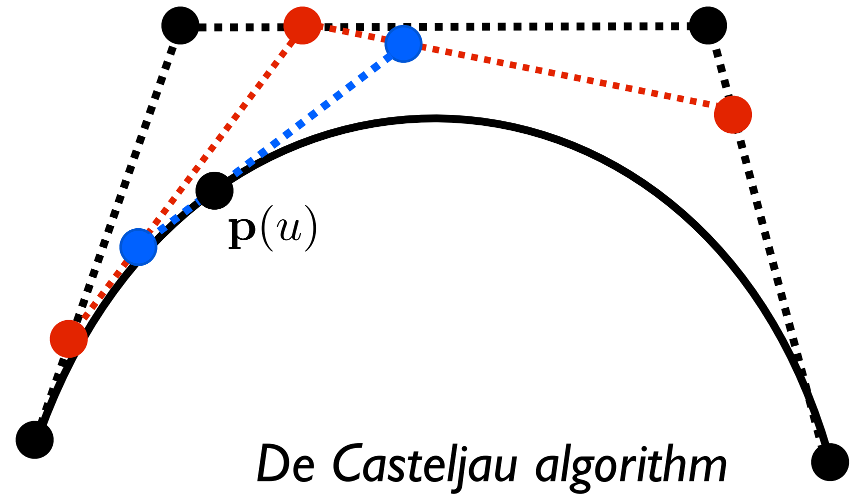
Evaluating p(u) geometrically



Evaluating p(u) geometrically

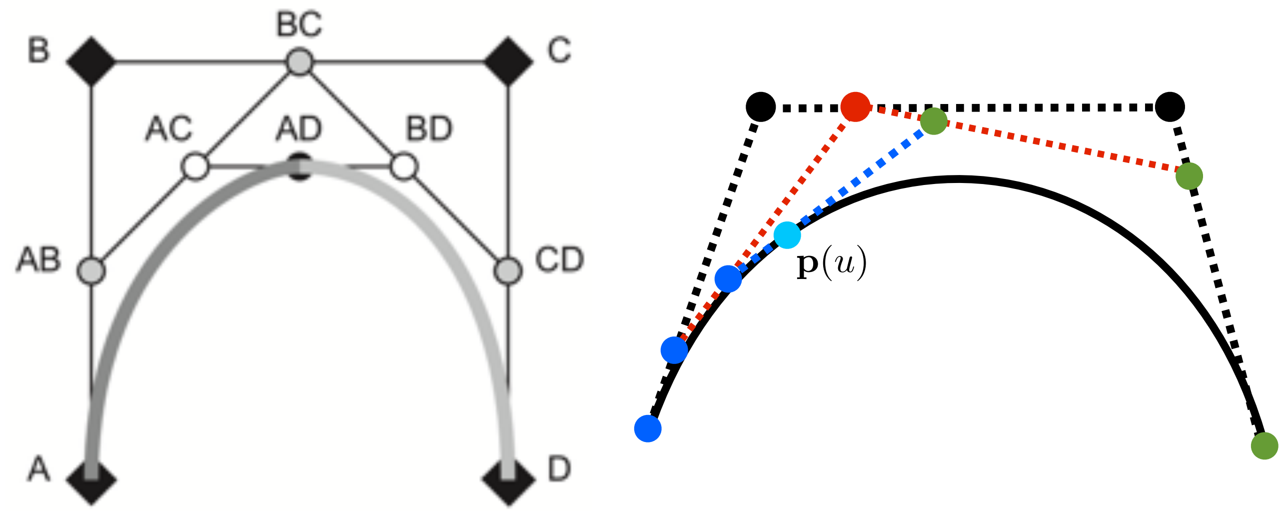


Evaluating p(u) geometrically



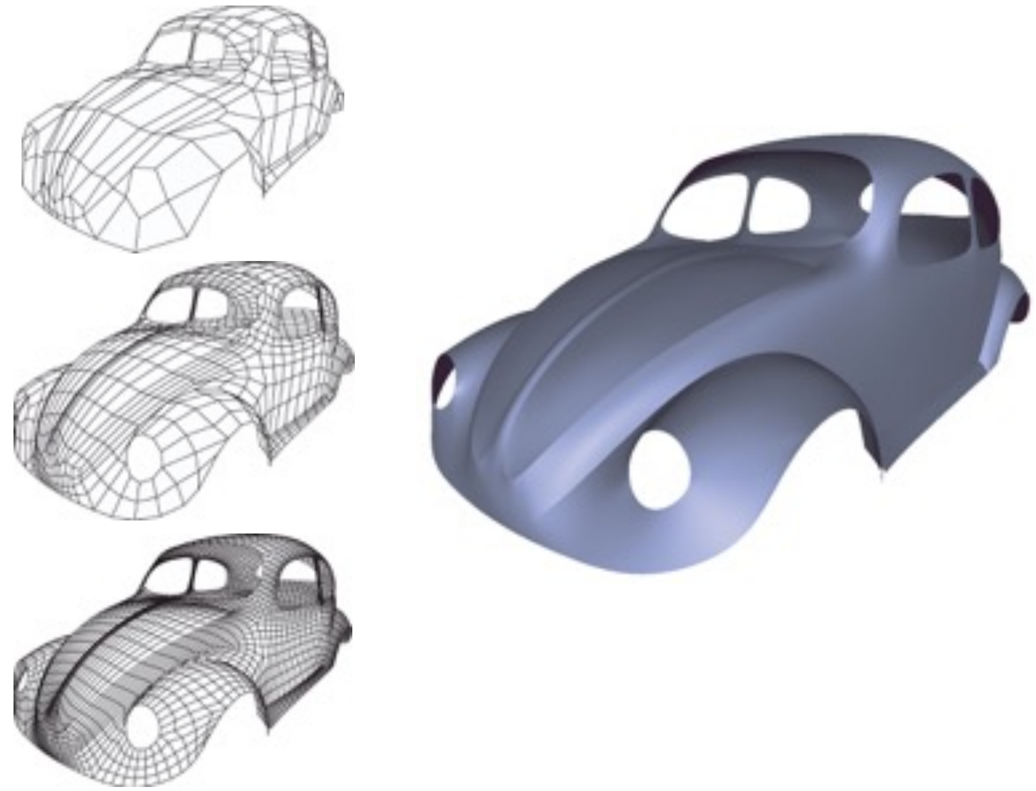
Kas-tell-joh

Bezier subdivision



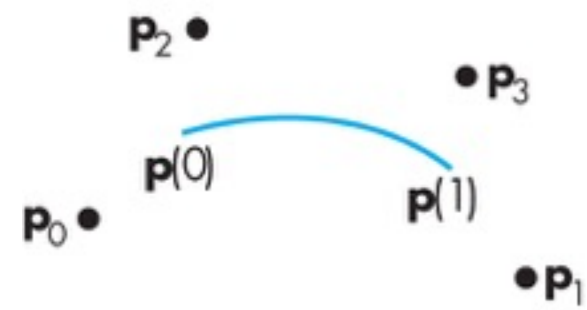
de Casteljau algorithm
Left: Subdivide the curve at the point u=.5
Right: Subdivide the curve at some other point u

Recursive Subdivision for Rendering



Cubic B-Splines

Cubic B-Splines



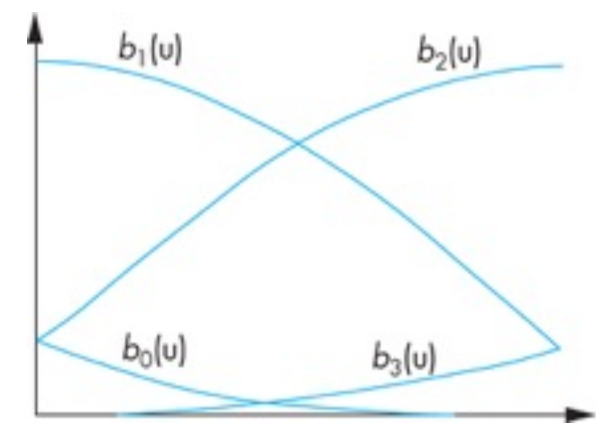
Spline blending functions

$$b_0(u) = \frac{1}{6}(1-u)^3$$

$$b_1(u) = \frac{1}{6}(4 - 6u^2 + 3u^3)$$

$$b_2(u) = \frac{1}{6}(1 + 3u + 3u^2 - 3u^3)$$

$$b_3(u) = \frac{1}{6}u^3$$

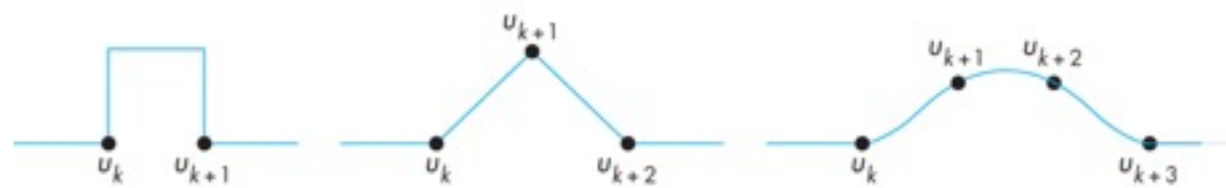


General Splines

- Defined recursively by *Cox-de Boor recursion formula*

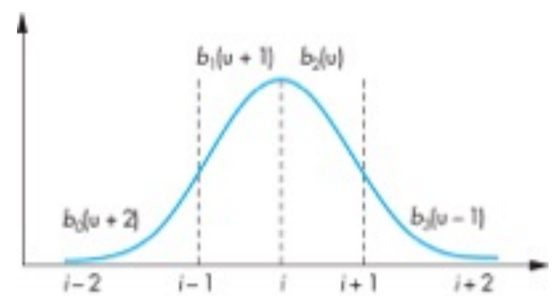
$$b_{j,0}(t) = \begin{cases} 1 & \text{if} \quad t_j \leq t \\ 0 & \text{otherwise} \end{cases}$$

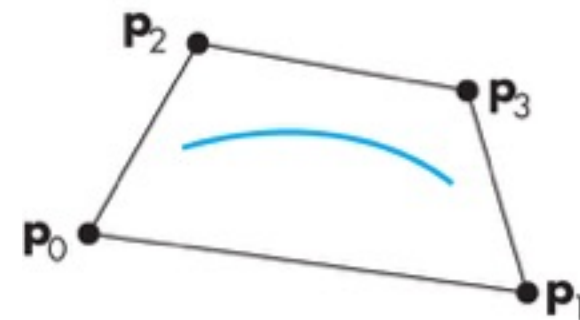
$$b_{j,n}(t) := \frac{t - t_j}{t_{j+n} - t_j} b_{j,n-1}(t) + \frac{t_{j+n+1} - t}{t_{j+n+1} - t_{j+1}} b_{j+1,n-1}(t)$$



Spline properties

Basis functions



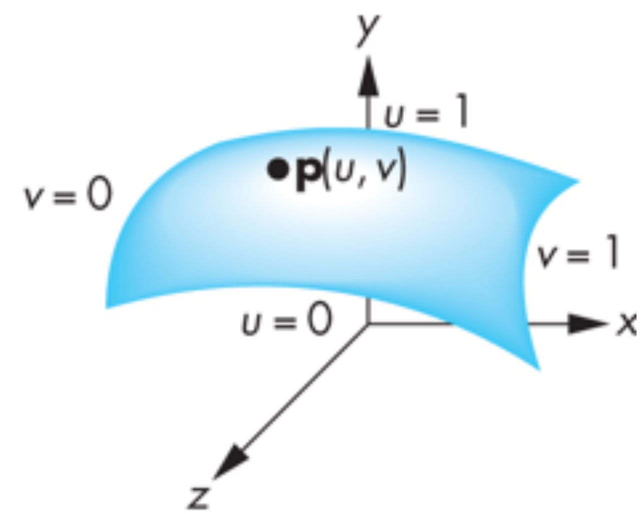


convexity

Surfaces

Parametric Surface

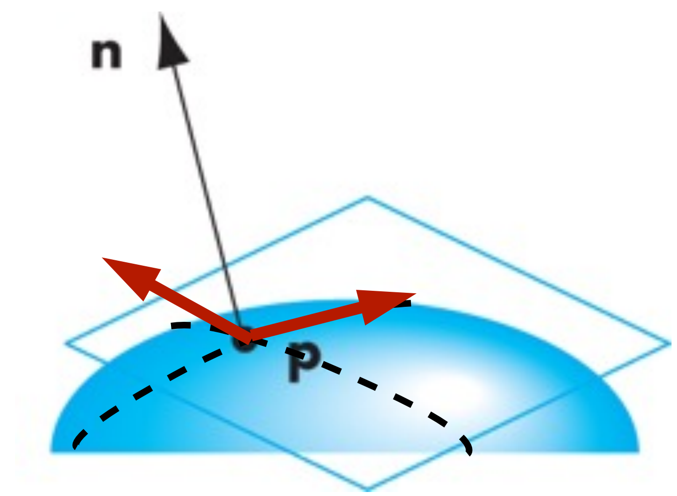
$$x = x(u, v)$$
$$y = y(u, v)$$
$$z = z(u, v)$$



Parametric Surface - tangent plane

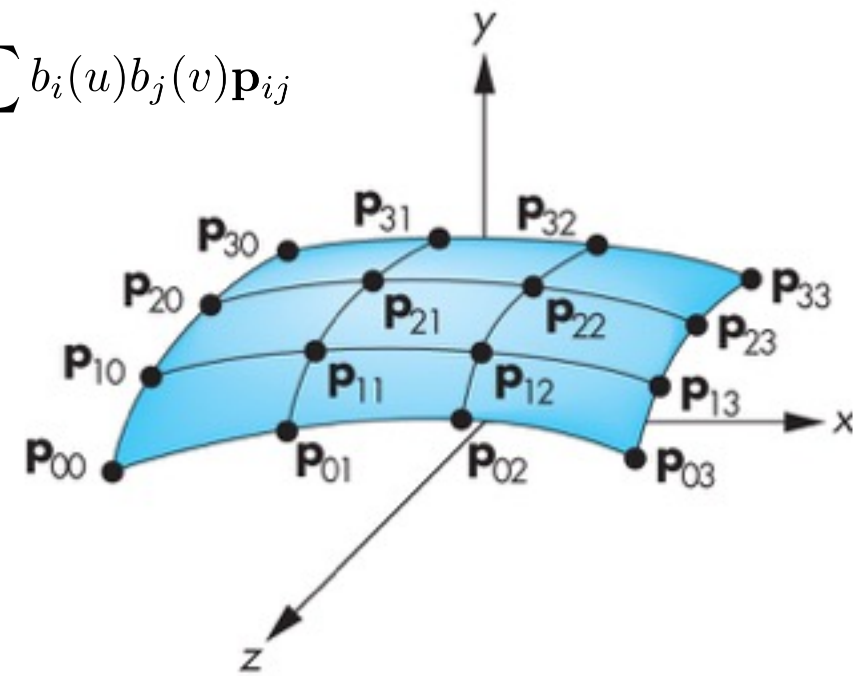
$$\mathbf{t}_u = \begin{pmatrix} \frac{\partial x}{\partial u} \\ \frac{\partial y}{\partial u} \\ \frac{\partial z}{\partial u} \end{pmatrix}$$

$$\mathbf{t}_v = \begin{pmatrix} \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial v} \\ \frac{\partial z}{\partial v} \end{pmatrix}$$



Bicubic Surface Patch

$$\mathbf{f}(u,v) = \sum_i \sum_j b_i(u) b_j(v) \mathbf{p}_{ij}$$



Bezier Surface Patch

$$\mathbf{f}(u,v) = \sum_i \sum_j b_i(u) b_j(v) \mathbf{p}_{ij}$$

Patch lies in
convex hull