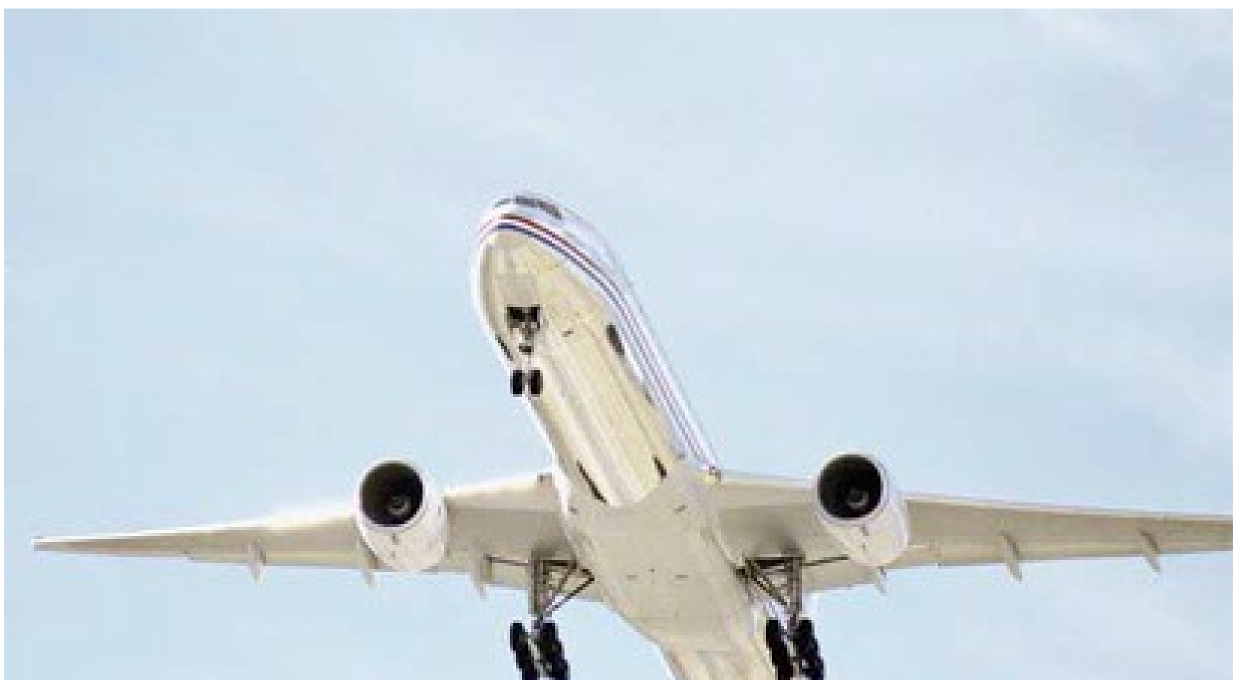
CS100433
Curves and Surface

Junqiao Zhao 赵君峤
Department of Computer Science and Technology
College of Electronics and Information Engineering
Tongji University

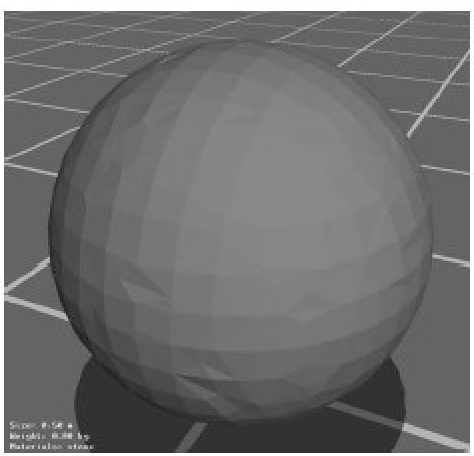
Motivation: smoothness

- In many applications we need smooth shapes
  - without discontinuities
- So far we can make things with corners



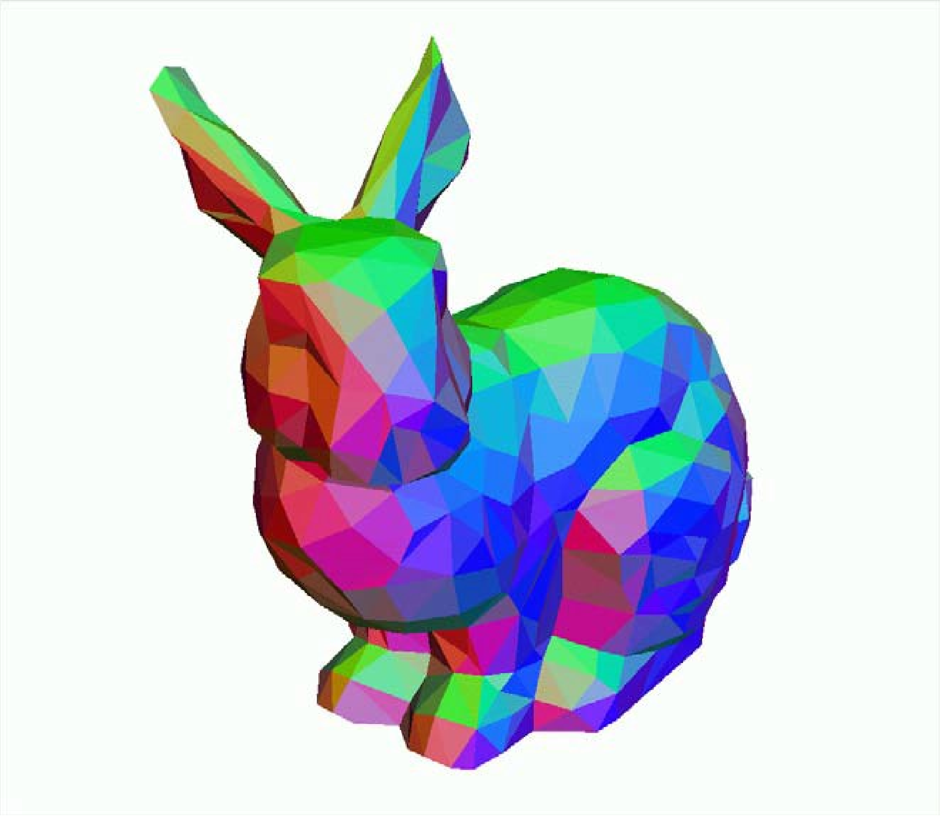
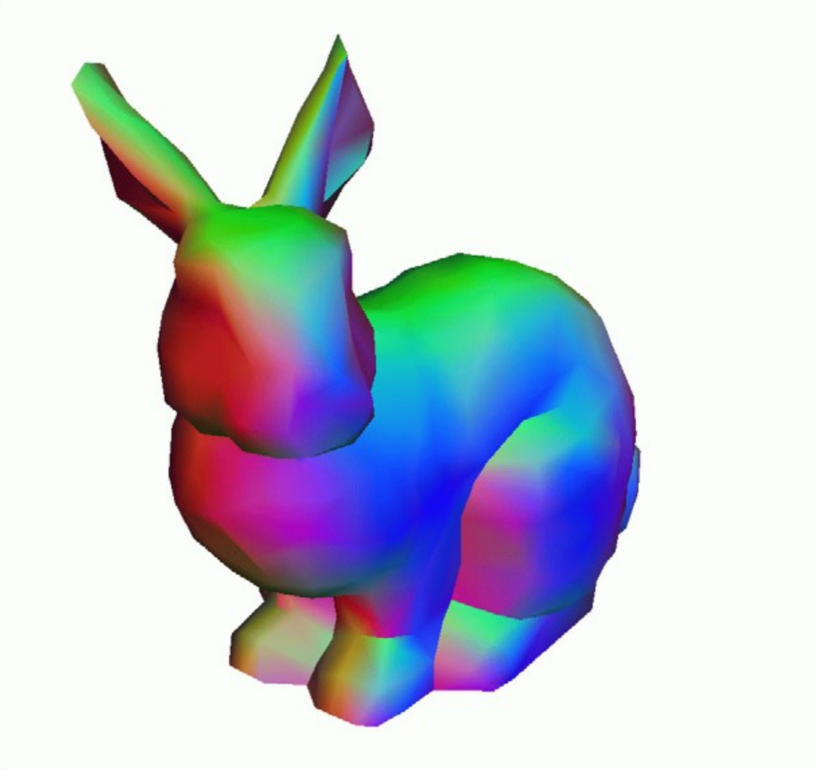
Limitations of Polygonal Meshes

- Planar facets
- Fixed resolution
- Deformation is difficult
- Hard for parameterization



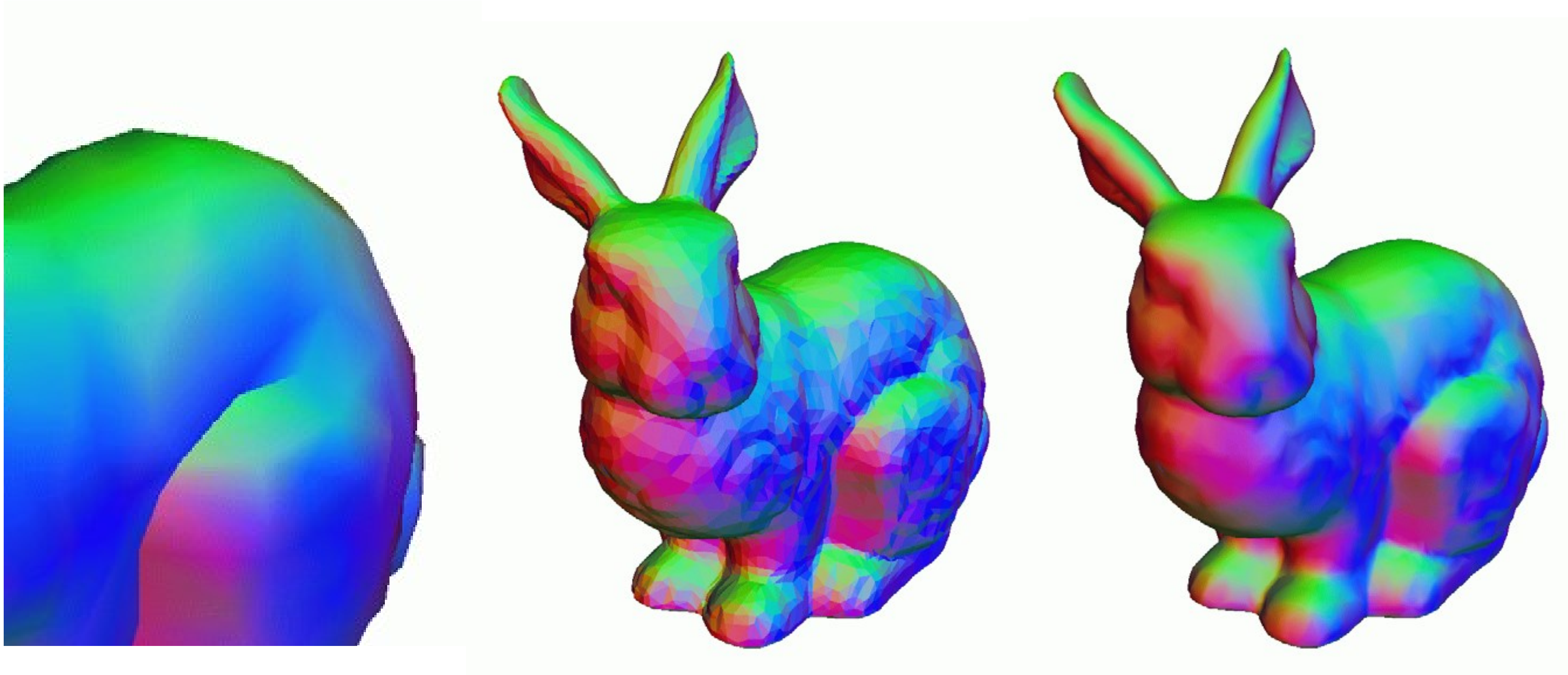
Stanford Bunny (1k triangles)

- Normal Smooth

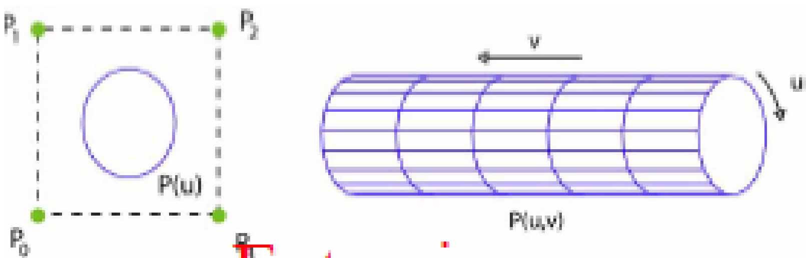



Problem

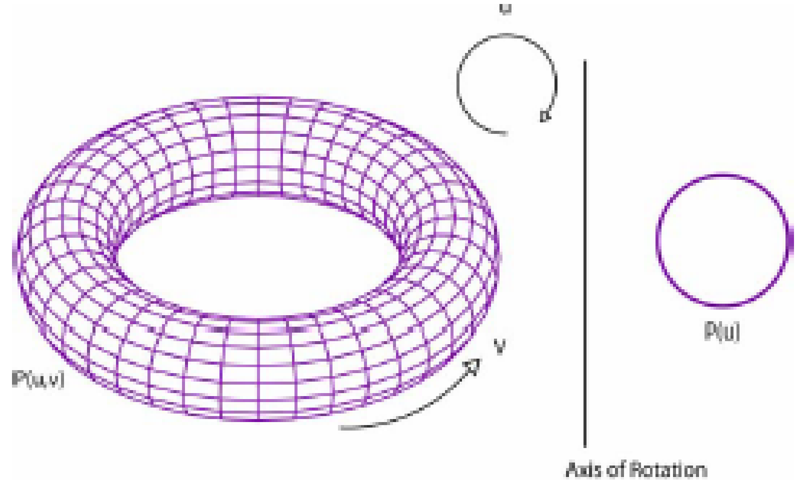
- Still low resolution especially at silhouettes
- So using more triangles
- 10K triangles or more? Not always good enough



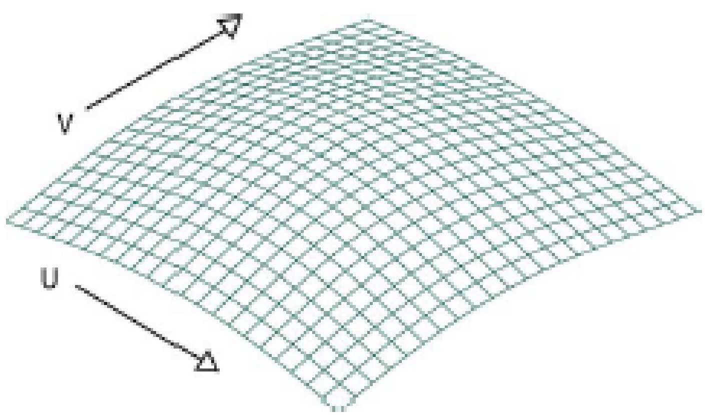
Non-Polygonal Modeling



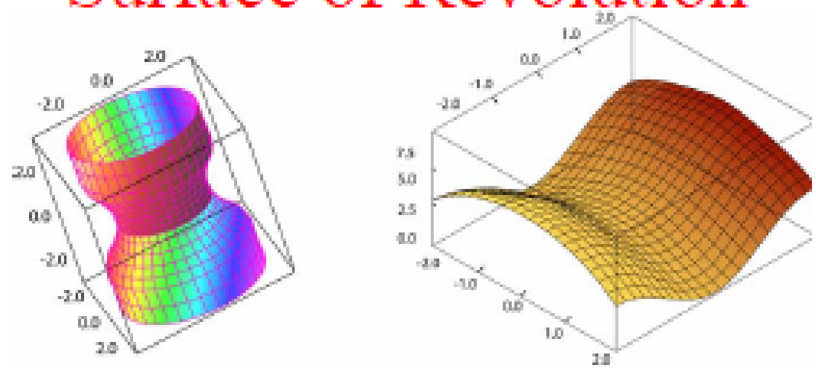
Extrusion



Surface of Revolution



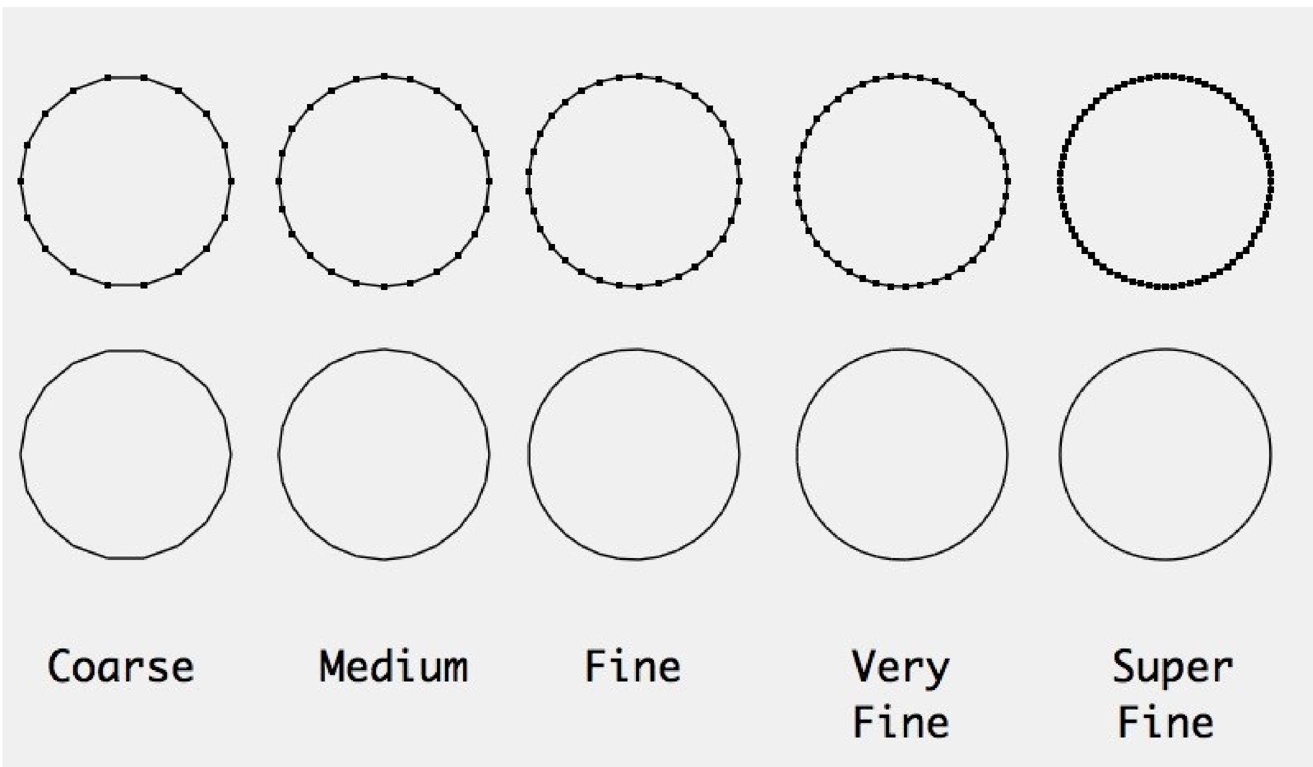
Spline Surfaces/Patches



Quadrics and other implicit polynomials

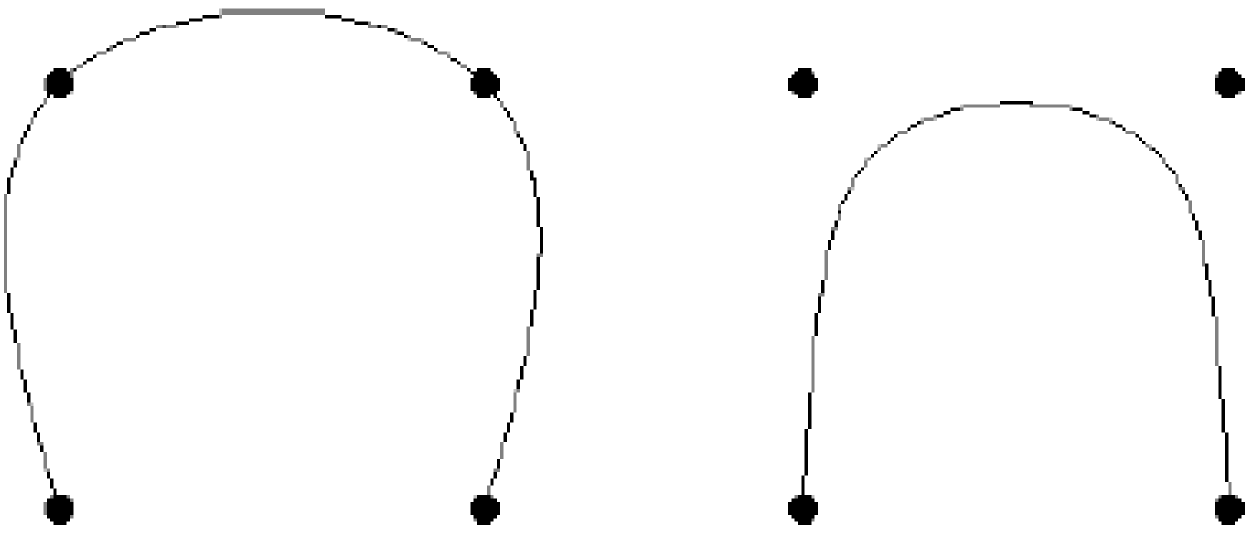
Curves

- Draw using discretization
- Can it be modeled as line segments?



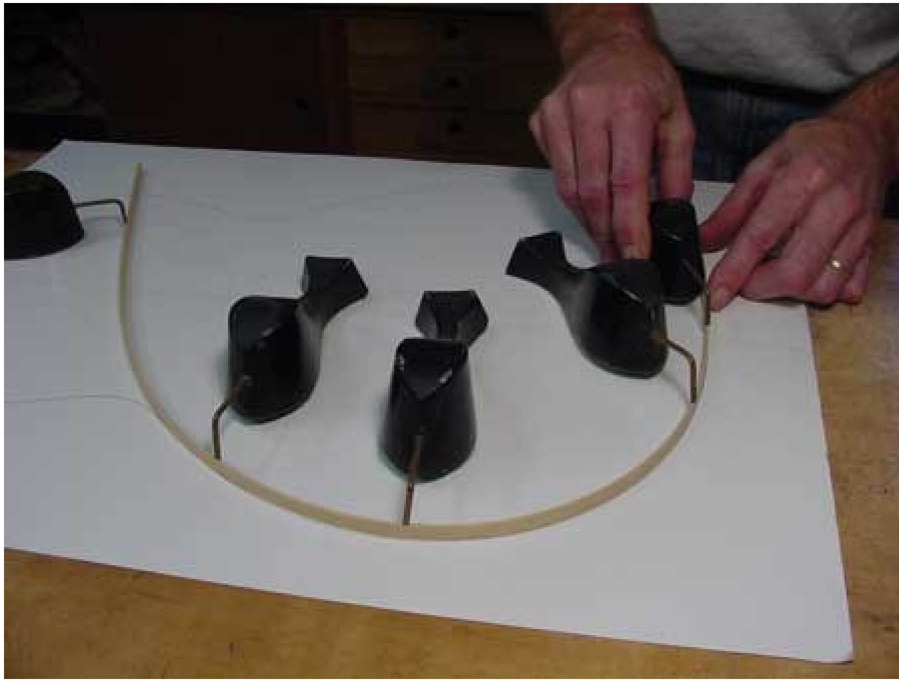
Spline curve

- Smooth curve defined by some control points
- Moving the control points changes the curve
- **Interpolation vs approximation**



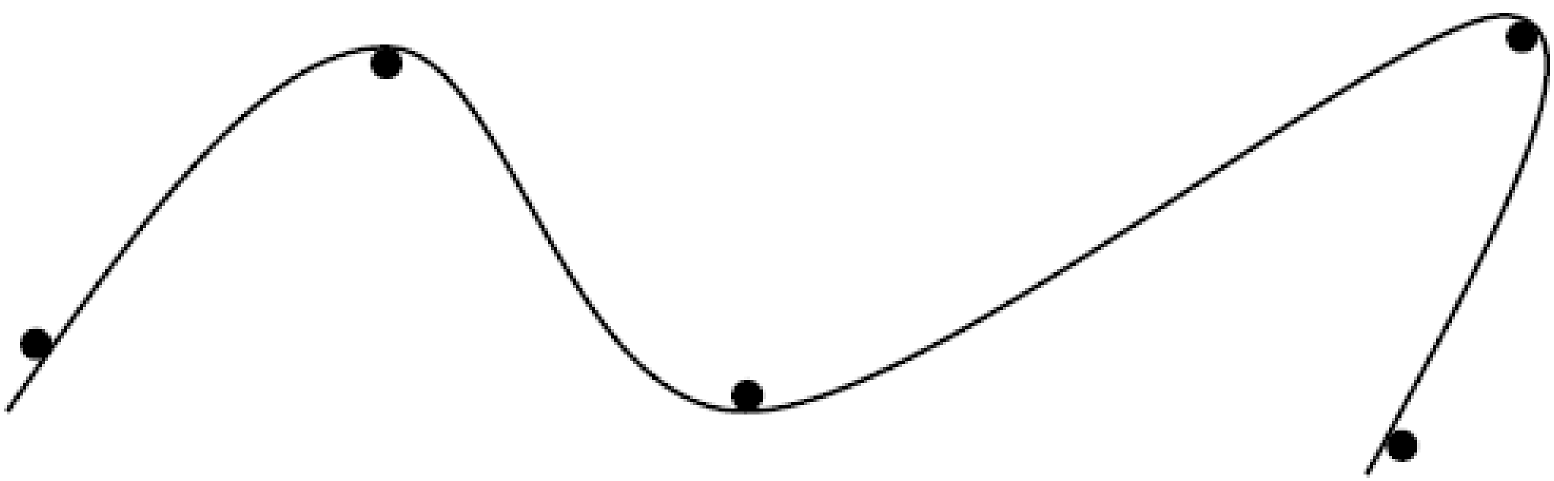
Interpolation Splines

- Imaging an elastic bar made of wood, bamboo or metal



Smoothness and Control

- Physically, curvature minimization based on fixed pins
- Mathematically, choosing low-order polynomials as smooth functions and passing through control points



Interpolation Curves

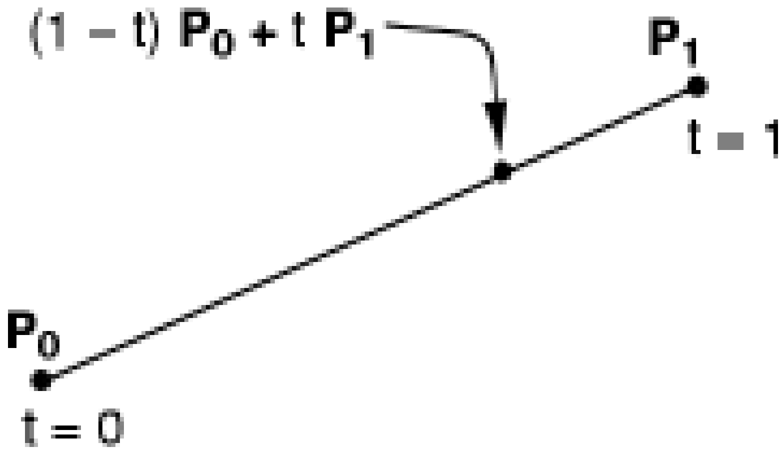
- Curve is constrained to pass through all control points
- Given points P_0, P_1, ... , P_n, find lowest degree polynomial which passes through the points

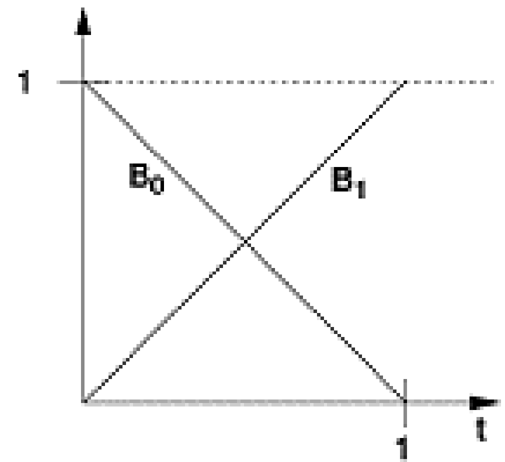
$$x(t) = a_{n-1}t^{n-1} + \ldots + a_2t^2 + a_1t + a_0$$
$$y(t) = b_{n-1}t^{n-1} + \ldots + b_2t^2 + b_1t + b_0$$

$$Q(t) = GBT(t) = Geometry\ G \cdot Spline\ Basis\ B \cdot Power\ Basis\ T(t)$$

Linear Interpolation

- Simplest "curve" between two points



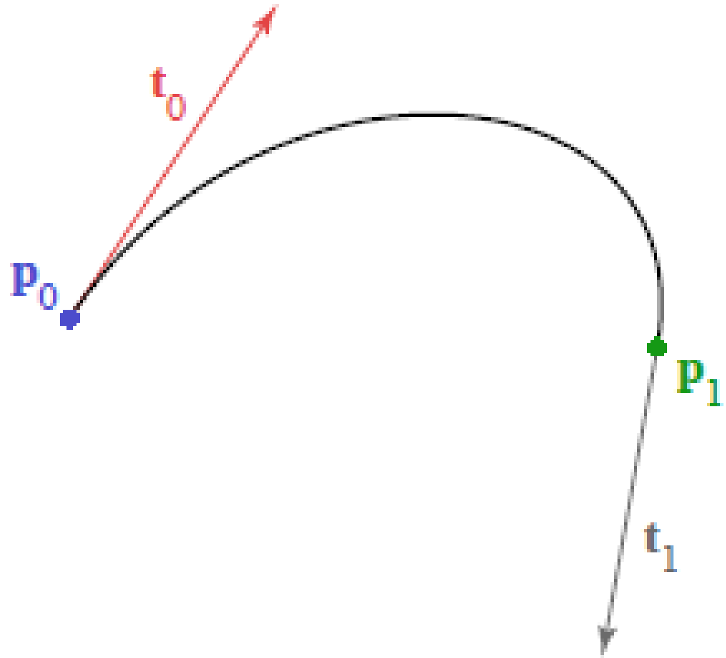


$$Q(t) = \begin{pmatrix} Q_x(t) \\ Q_y(t) \\ Q_z(t) \end{pmatrix} = \begin{pmatrix} (P_0) & (P_1) \end{pmatrix} \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} t \\ 1 \end{pmatrix}$$

$$Q(t) = GBT(t) = Geometry\ G \cdot Spline\ Basis\ B \cdot Power\ Basis\ T(t)$$

Hermite spline curve

- Piecewise cubic
- Constraints: endpoints and tangents



Hermite spline curve

- Solve constraints to find coefficients

$$x(t) = at^3 + bt^2 + ct + d$$
$$x'(t) = 3at^2 + 2bt + c$$
$$x(0) = x_0 = d$$
$$x(1) = x_1 = a + b + c + d$$
$$x'(0) = x'_0 = c$$
$$x'(1) = x'_1 = 3a + 2b + c$$

$$d = x_0$$
$$c = x'_0$$
$$a = 2x_0 - 2x_1 + x'_0 + x'_1$$
$$b = -3x_0 + 3x_1 - 2x'_0 - x'_1$$

Hermite spline curve

$$f(t) = at^3 + bt^2 + ct + d$$

$$[p_0\ p_1\ p_2\ p_3]\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}\begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$$

$$Q(t) = GBT(t) = Geometry\ G \cdot Spline\ Basis\ B \cdot Power\ Basis\ T(t)$$

Hermite spline curve

$$\mathrm{f}(t) = \mathrm{a}t^3 + \mathrm{b}t^2 + \mathrm{c}t + \mathrm{d}$$

$$[p_0\ p_1\ t_0\ t_1]\begin{bmatrix} 2 & -3 & 0 & 1 \\ -2 & 3 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$$

$$f(t) = b_0(t)p_0 + b_1(t)p_1 + b_2(t)p_2 + b_3(t)p_3$$

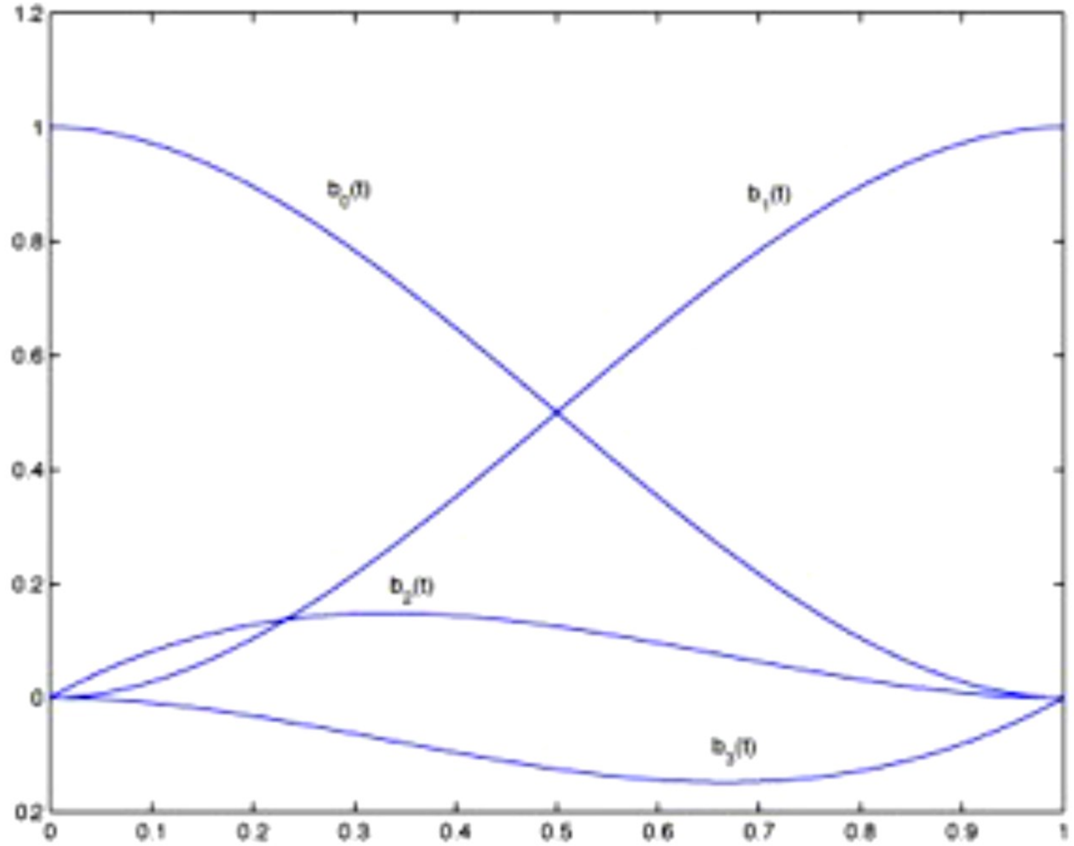
Hermite spline curve

$$\mathbf{f}(t) = \mathbf{a}t^3 + \mathbf{b}t^2 + \mathbf{c}t + \mathbf{d}$$

$$[p_0\ p_1\ p_2\ p_3]\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}\begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$$

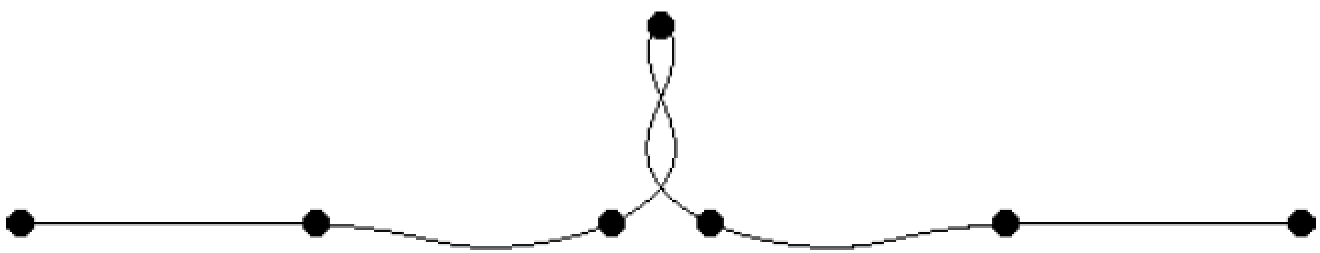
$$f(t) = b_0(t)p_0 + b_1(t)p_1 + b_2(t)p_2 + b_3(t)p_3$$

The hermite basis functions

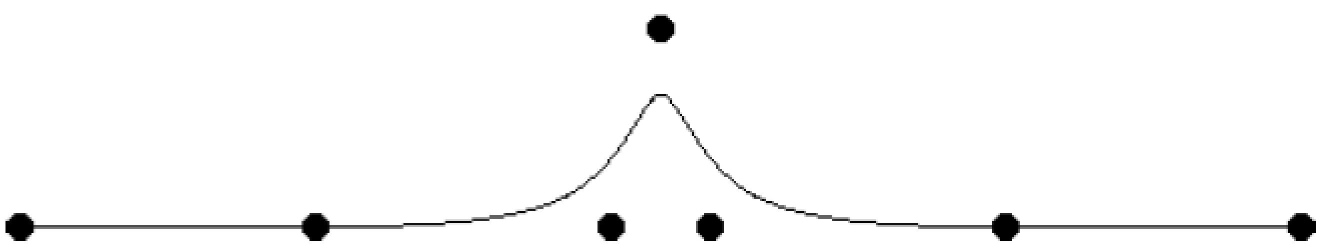


Interpolation vs Approximation

- Interpolation Curve –over constrained →lots of (undesirable?) oscillations

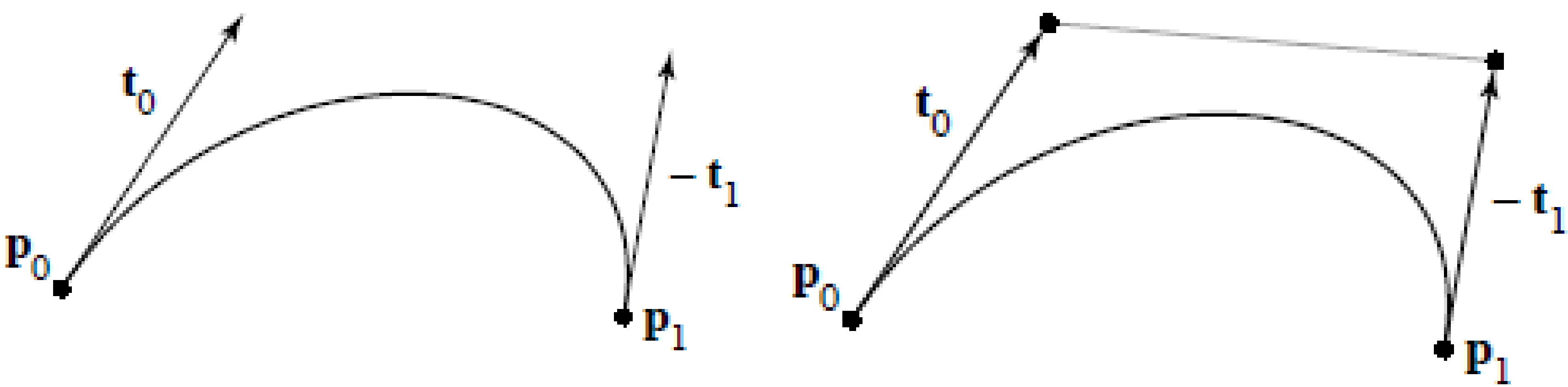


- Approximation Curve –more reasonable?



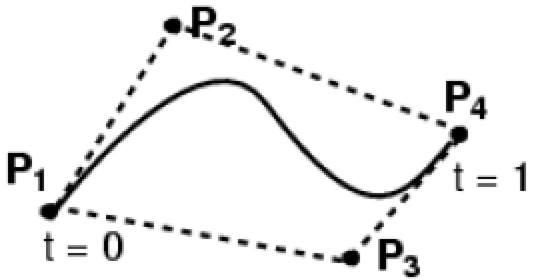
Hermite to Bézier

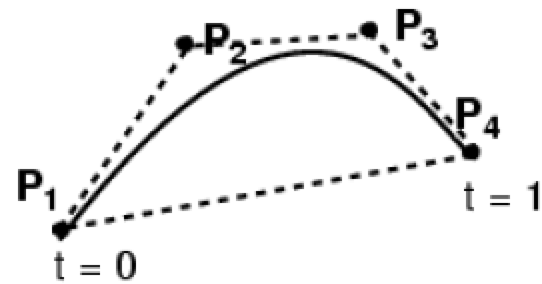
- Mixture of points and vectors is awkward
- Specify tangents as differences of points

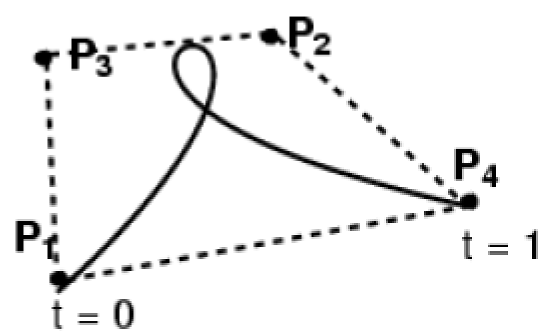


Cubic Bézier Curve

- Control points
- Curve passes through first & last control point
- Curve is tangent at $\mathbf{P_1}$ to $(\mathbf{P_1}-\mathbf{P_2})$ and at $\mathbf{P_4}$ to $(\mathbf{P_4}-\mathbf{P_3})$

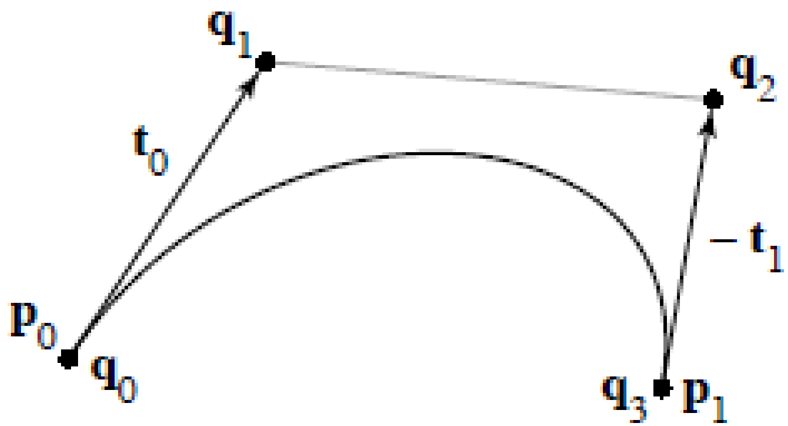






Hermite to Bézier

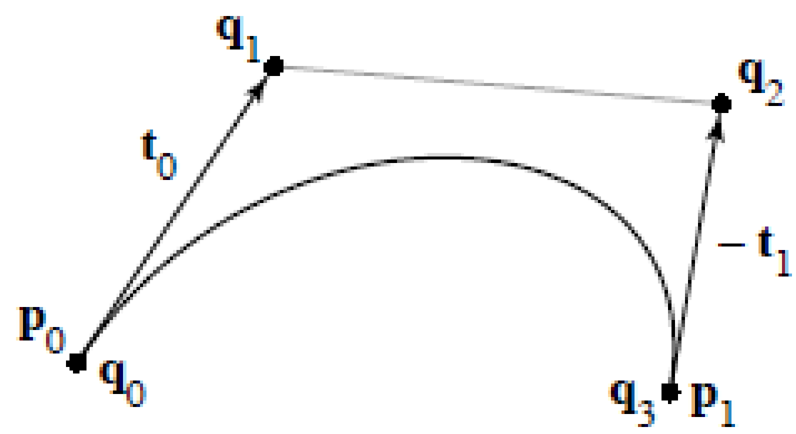
$$\mathbf{p}_0 = \mathbf{q}_0$$
$$\mathbf{p}_1 = \mathbf{q}_3$$
$$\mathbf{t}_0 = 3(\mathbf{q}_1 - \mathbf{q}_0)$$
$$\mathbf{t}_1 = 3(\mathbf{q}_3 - \mathbf{q}_2)$$



- $[p_0\ p_1\ t_0\ t_1] = [q_0\ q_1 q_2\ q_3]\begin{bmatrix} 1 & 0 & -3 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & -3 \\ 0 & 1 & 0 & 3 \end{bmatrix}$

Hermite to Bézier

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



- $[q_0\ q_1 q_2\ q_3]\begin{bmatrix}1 & 0 & -3 & 0\\ 0 & 0 & 3 & 0\\ 0 & 0 & 0 & -3\\ 0 & 1 & 0 & 3\end{bmatrix}\begin{bmatrix}2 & -3 & 0 & 1\\ -2 & 3 & 0 & 0\\ 1 & -2 & 1 & 0\\ 1 & -1 & 0 & 0\end{bmatrix}\begin{bmatrix}t^3\\ t^2\\ t\\ 1\end{bmatrix}$

Bézier matrix

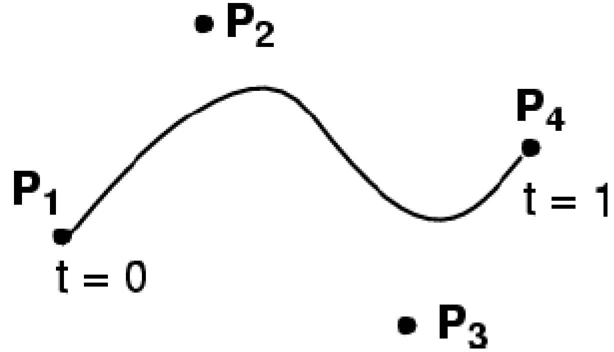
- $Q(t) = [q_0\ q_1\ q_2\ q_3]\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$

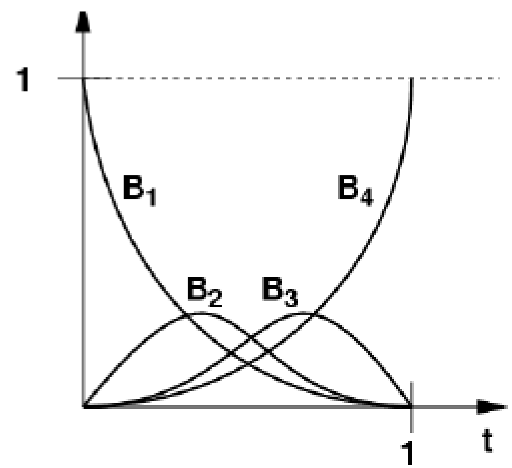
- These are defined as Bernstein polynomials

$$B_i^n(t) = \frac{n!}{i!\,(n-i)!} t^i (1-t)^{n-i}, \qquad 0 \le i \le n$$

$$Q(t) = GBT(t) = Geometry\ G \cdot Spline\ Basis\ B \cdot Power\ Basis\ T(t)$$

Cubic Bézier Curves





$$Q(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4$$

$$Q(t) = \mathbf{GBT(t)} \qquad B_{Bezier} = \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$B_1(t) = (1-t)^3; B_2(t) = 3t(1-t)^2; B_3(t) = 3t^2(1-t); B_4(t) = t^3$$

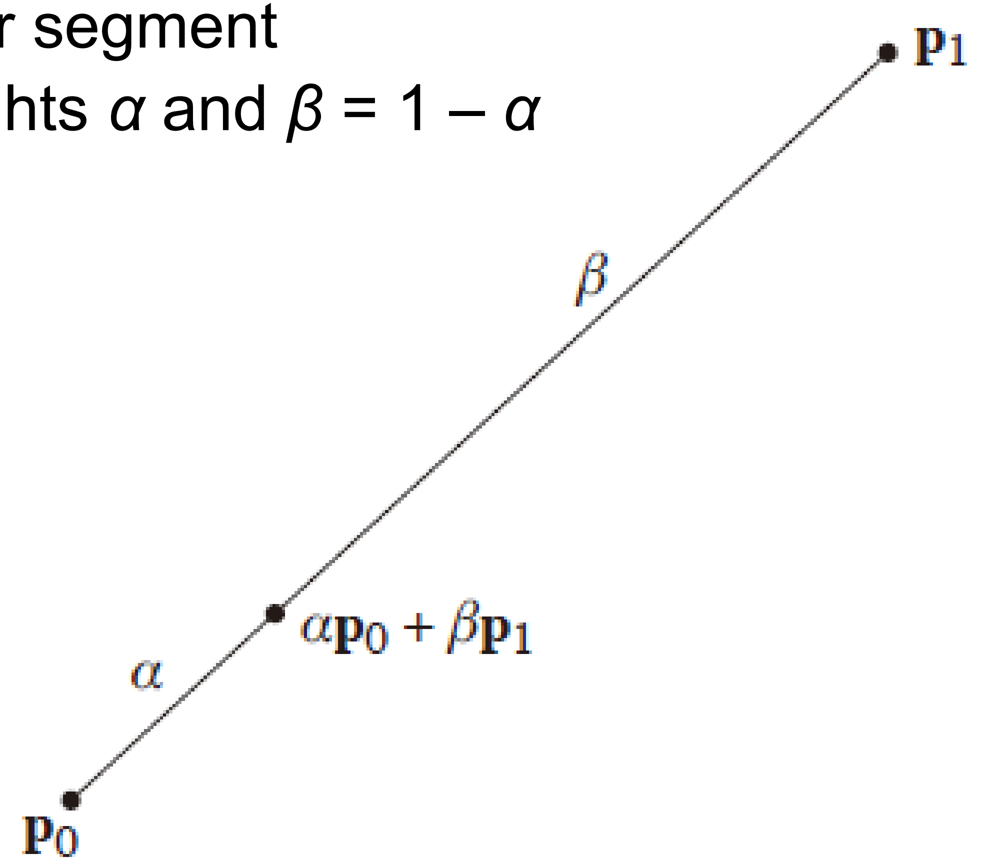
- Questions?

Another way to Bézier segments

- Start from $Q(t) = p_0$

Another way to Bézier segments

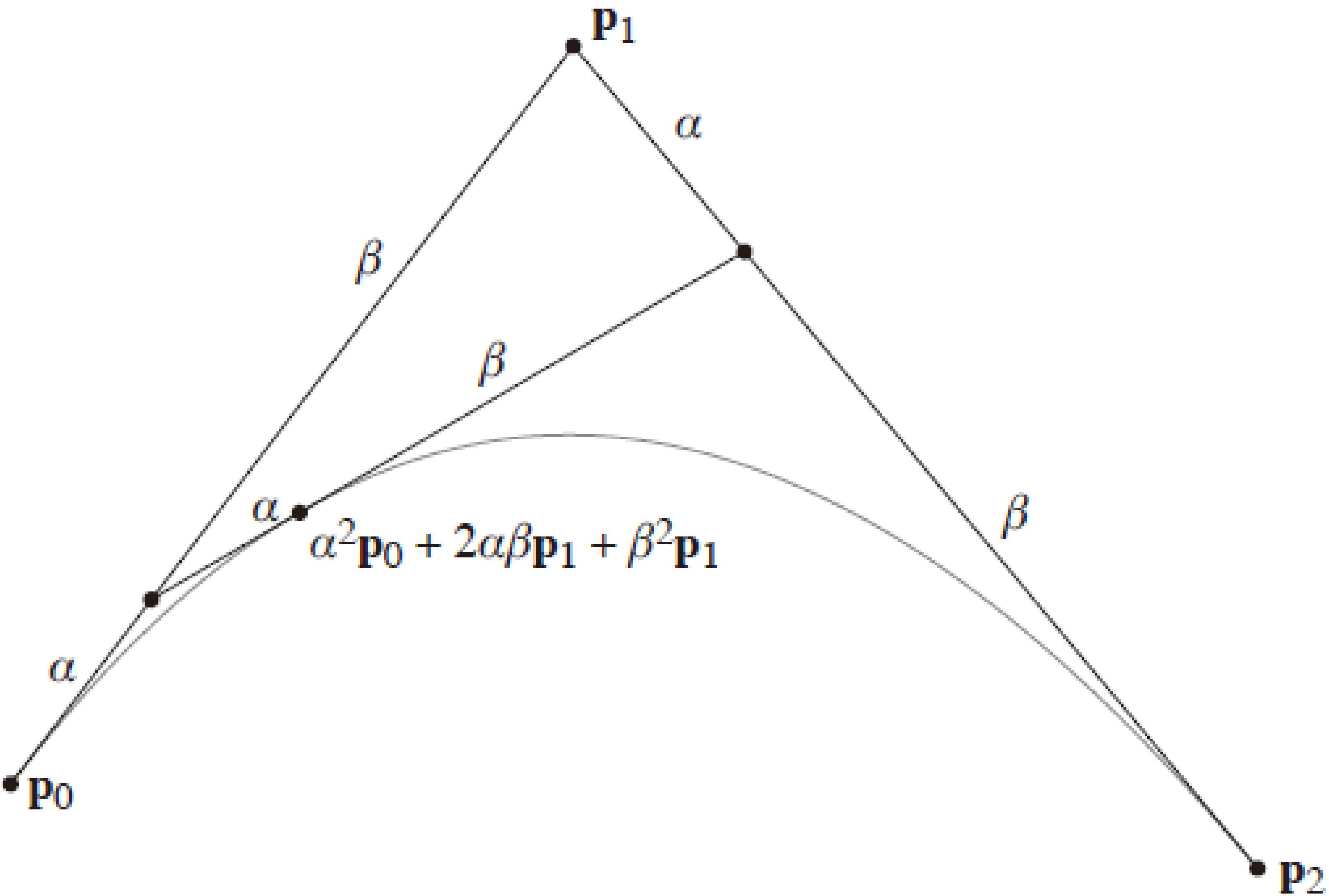
- A piecewise linear spline segment
  - two control points per segment
  - blend them with weights α and $\beta = 1 - \alpha$



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!

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

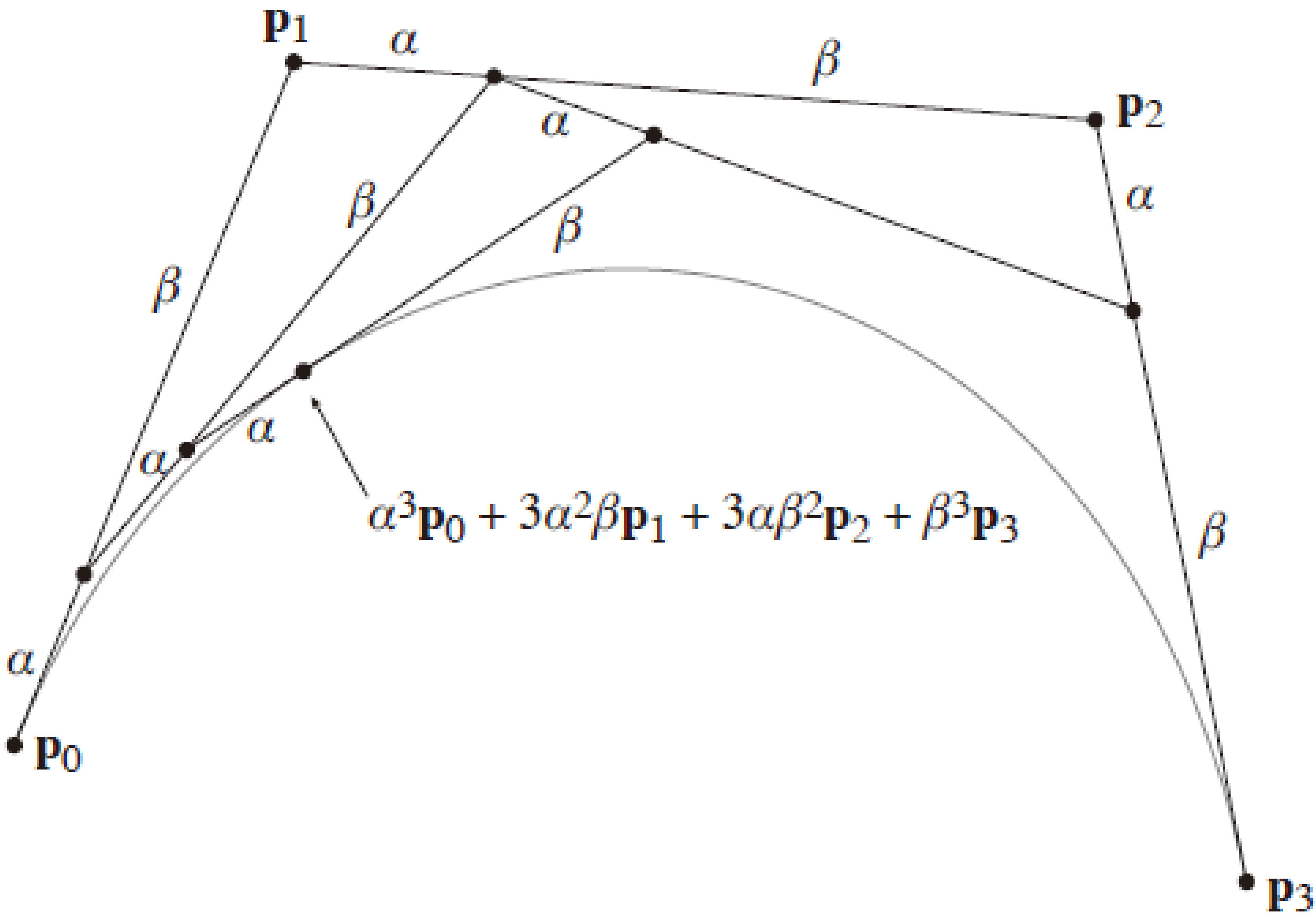


$\mathbf{p}_1$
α
β
β
α
$\alpha^2\mathbf{p}_0 + 2\alpha\beta\mathbf{p}_1 + \beta^2\mathbf{p}_1$
β
α
$\mathbf{p}_0$
$\mathbf{p}_2$

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

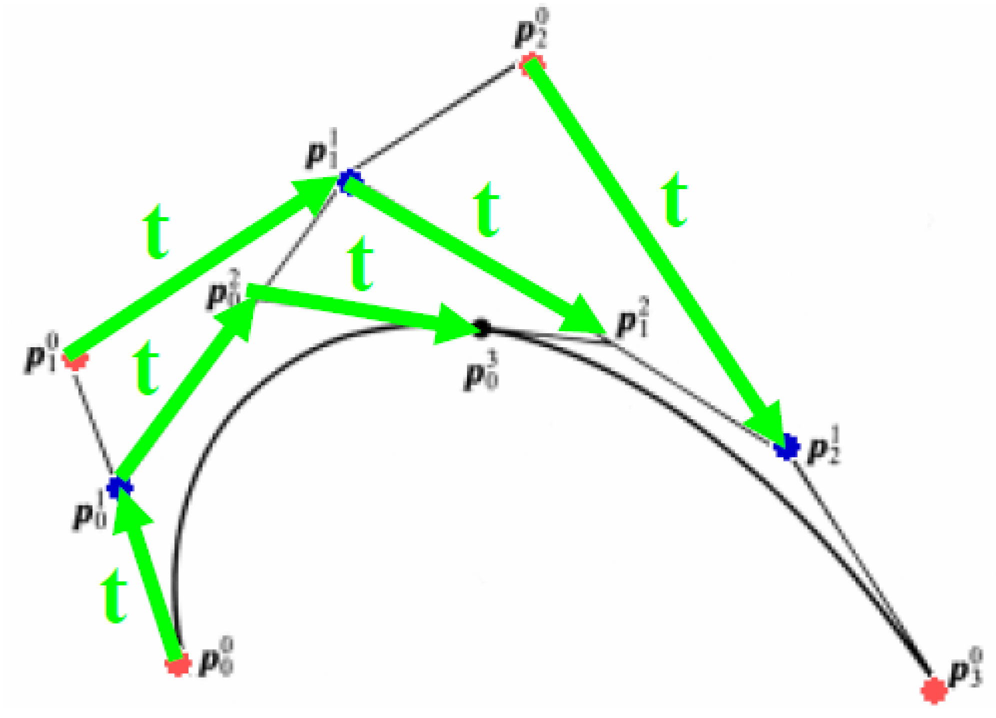
$$\begin{aligned}\mathbf{p}_{3,0} =& \alpha\mathbf{p}_{2,0} + \beta\mathbf{p}_{2,1} \\ =& \alpha\alpha\alpha\mathbf{p}_0 + \alpha\alpha\beta\mathbf{p}_1 + \alpha\beta\alpha\mathbf{p}_1 + \alpha\beta\beta\mathbf{p}_2 \\ & \beta\alpha\alpha\mathbf{p}_1 + \beta\alpha\beta\mathbf{p}_2 + \beta\beta\alpha\mathbf{p}_2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

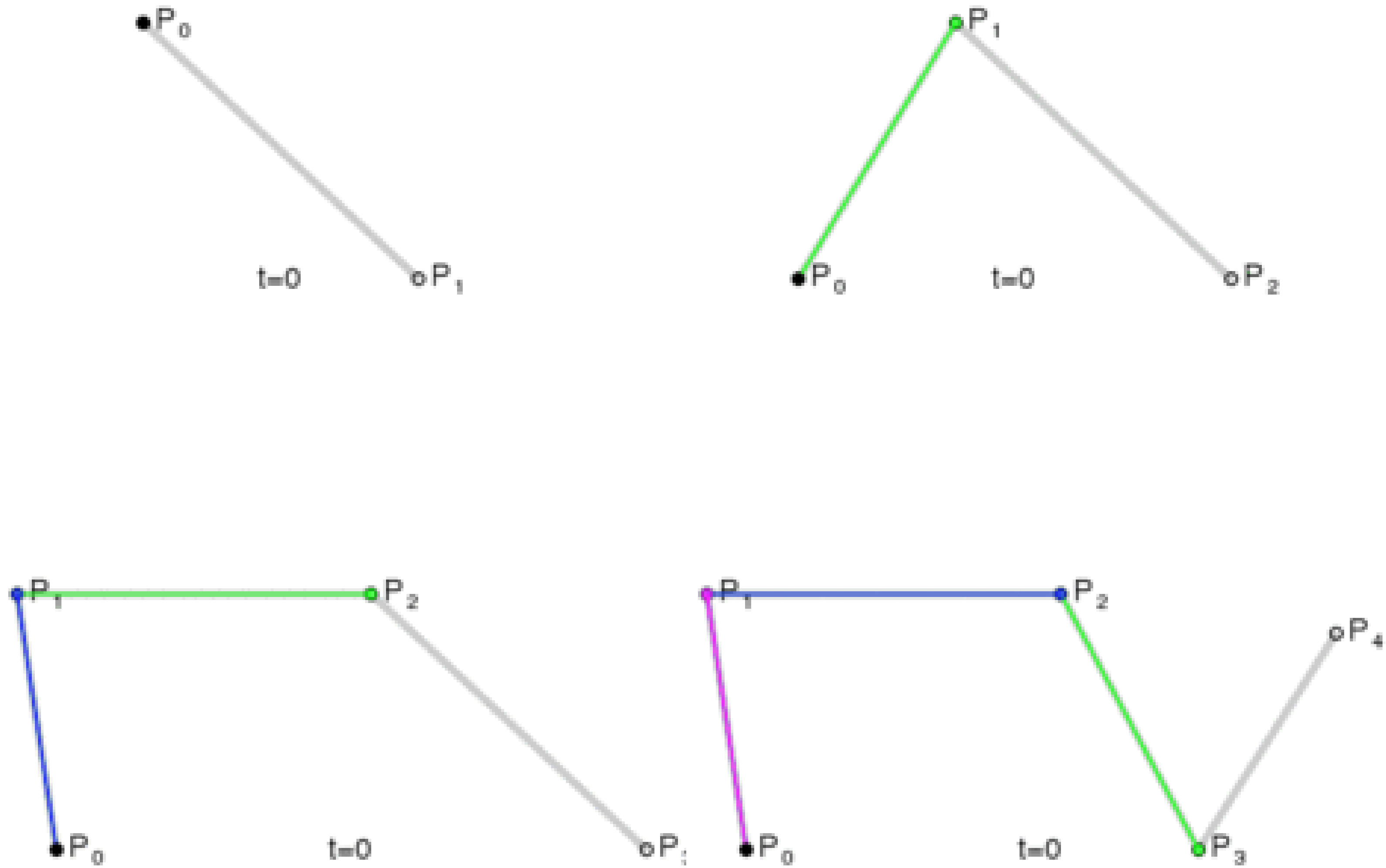


$\mathbf{p}_1$
α
β
$\mathbf{p}_2$
α
β
α
β
β
α
α
$\alpha^3\mathbf{p}_0 + 3\alpha^2\beta\mathbf{p}_1 + 3\alpha\beta^2\mathbf{p}_2 + \beta^3\mathbf{p}_3$
β
α
$\mathbf{p}_0$
$\mathbf{p}_3$

de Casteljau's algorithm

- A recurrence for computing points on Bézier spline segments



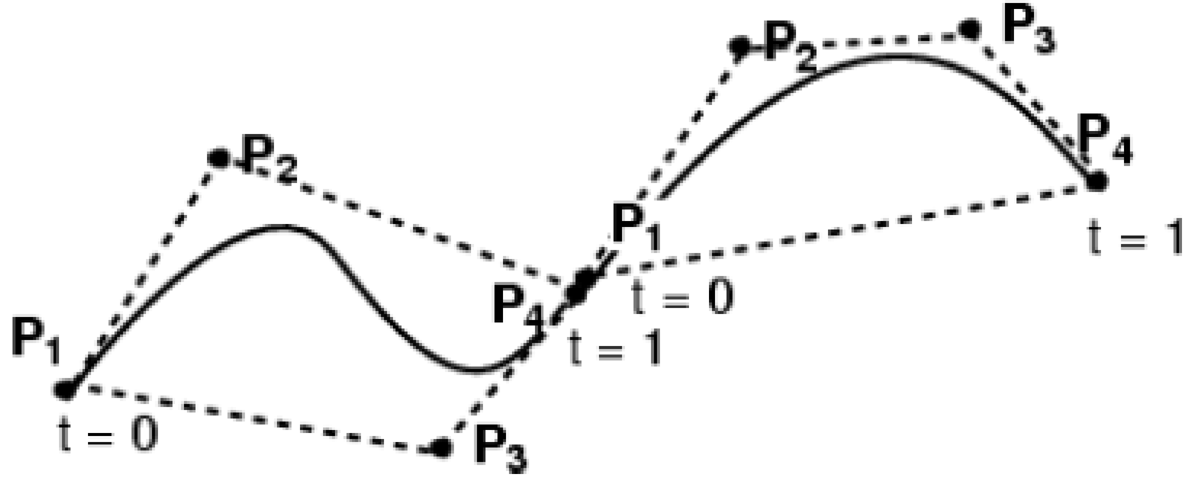

P_0
P_1
t=0
P_1
P_0
t=0
P_2
P_1
P_2
P_0
t=0
P_1
P_2
P_4
P_0
t=0
P_3

Continuity

- C^0 continuous
  - curve/surface has no breaks/gaps/holes
  - "watertight“
- C^1 continuous
  - curve/surface derivative is continuous
  - "looks smooth, no facets“
- C^2 continuous
  - curve/surface 2nd derivative is continuous
  - Actually important for shading

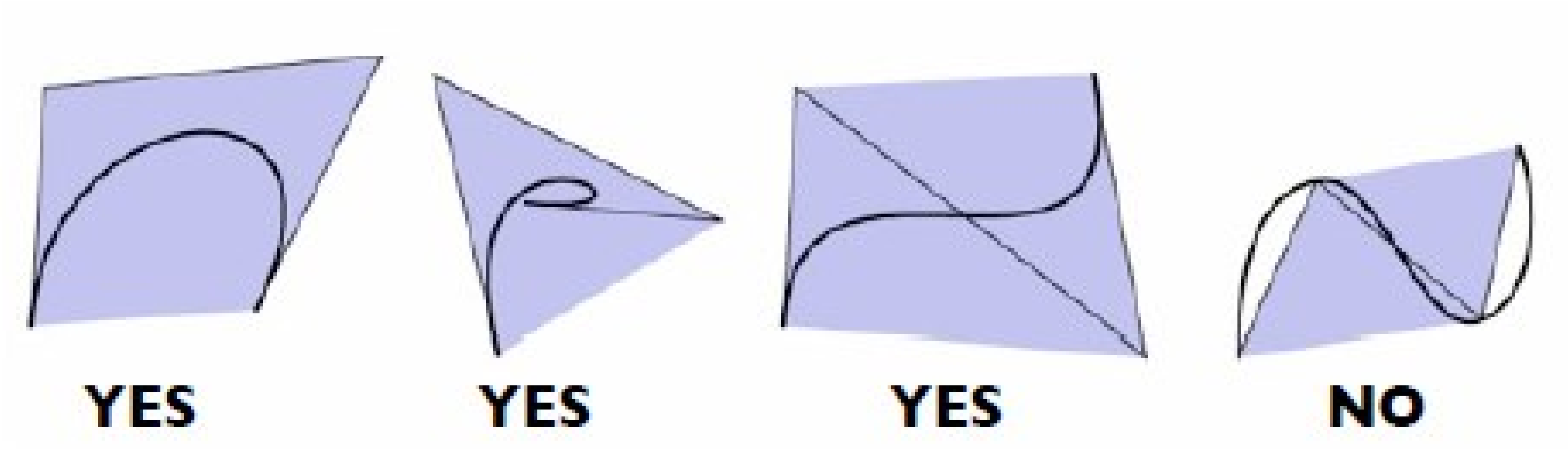
Connecting Cubic Bézier Curves

- How can we guarantee C0 continuity (no gaps)?
- How can we guarantee C1 continuity (tangent vectors match)?
- Asymmetric: Curve goes through some control points but misses others



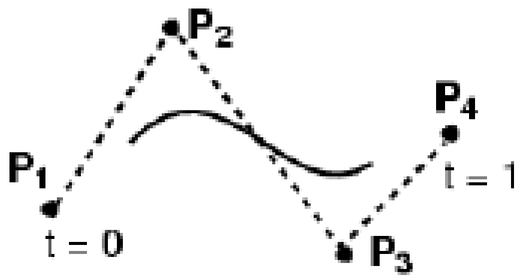
Properties of Bézier Splines

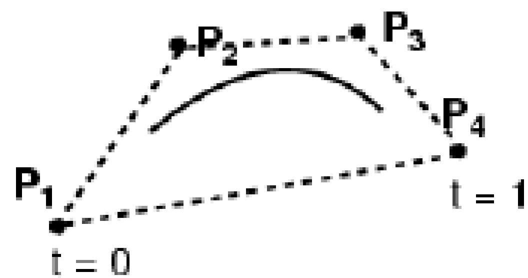
- Convex hull property
- Continuity
- Affine invariance

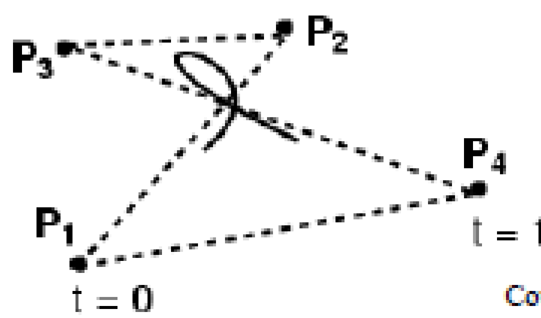


BSpline

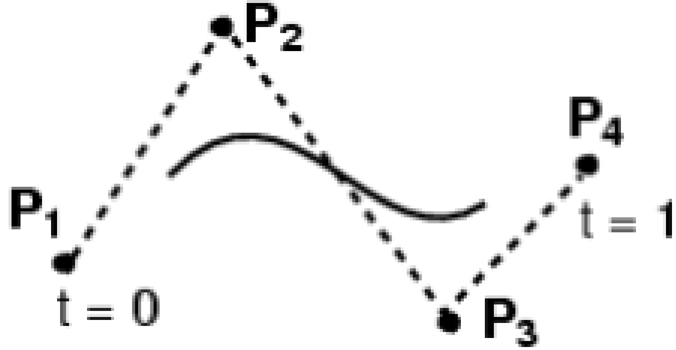
- ≥4 control points, Knot points
- Locally cubic (Bézier), Low order in general
- Curve is not constrained to pass through any control points

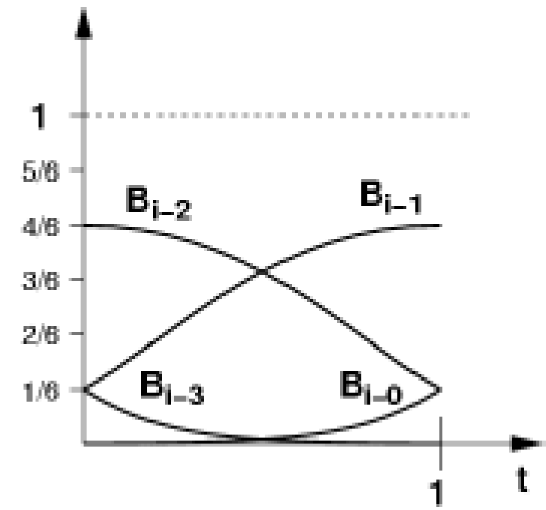






BSplines



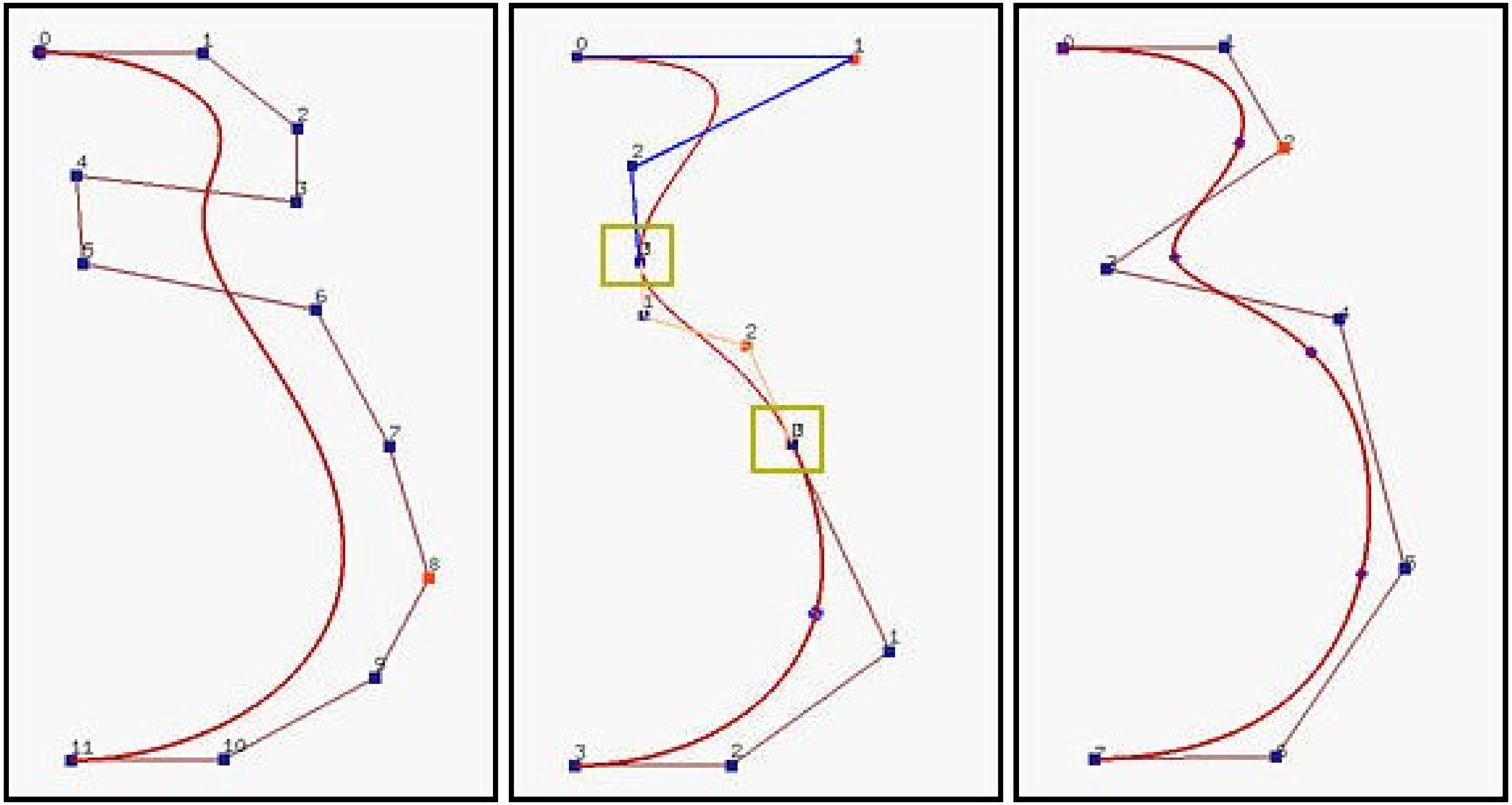


$$Q(t) = \frac{(1-t)^3}{6} P_{i-3} + \frac{3t^3 - 6t^2 + 4}{6} P_{i-2} + \frac{-3t^3 + 3t^2 + 3t + 1}{6} P_{i-1} + \frac{t^3}{6} P_i$$

$$Q(t) = \mathbf{GBT(t)}$$

$$B_{B-Spline} = \frac{1}{6} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}$$

BSpline vs Bézier

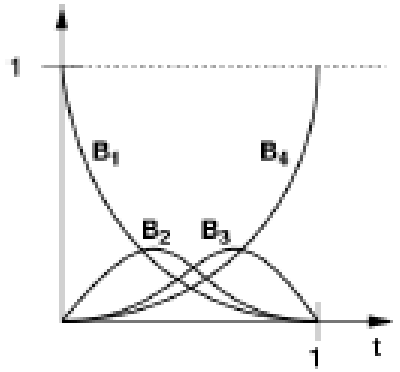


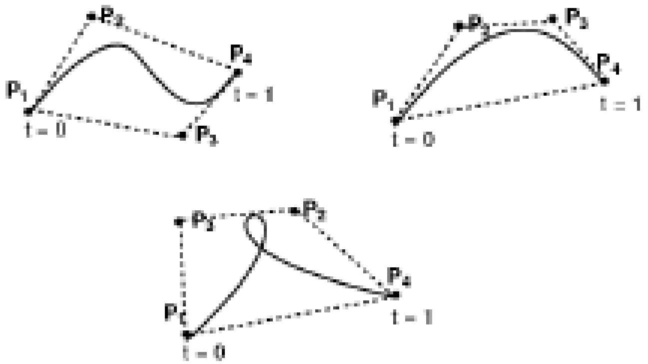
https://pages.mtu.edu/~shene/COURSES/cs3621/NOTES/

Bézier vs BSpline

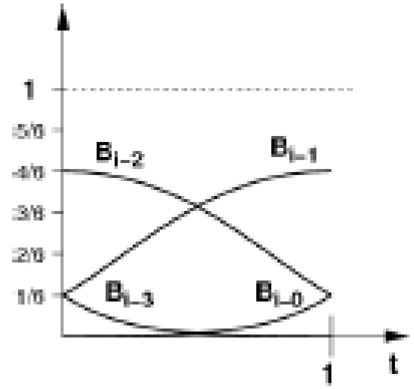
- Relationship to the control points is different

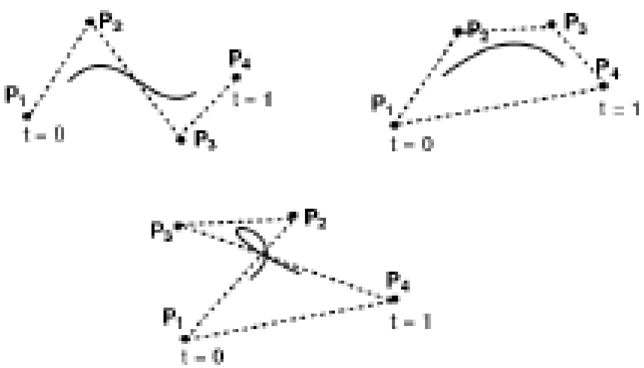
Bézier





BSpline





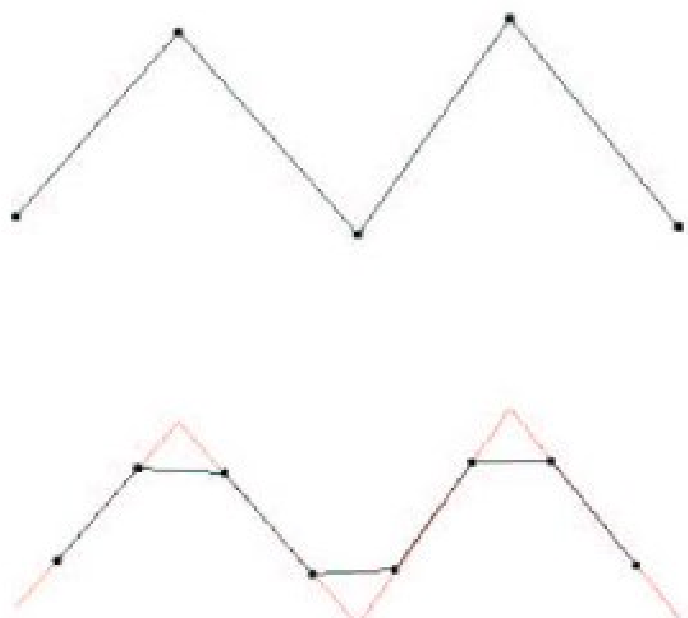
NURBS

- BSpline: uniform cubic Bspline
- Restriction of Polynomials
- NURBS: Non-Uniform Rational BSpline
  - **non-uniform** = different spacing between the blending functions,
  - **rational** = quotients of polynomials

- Questions?

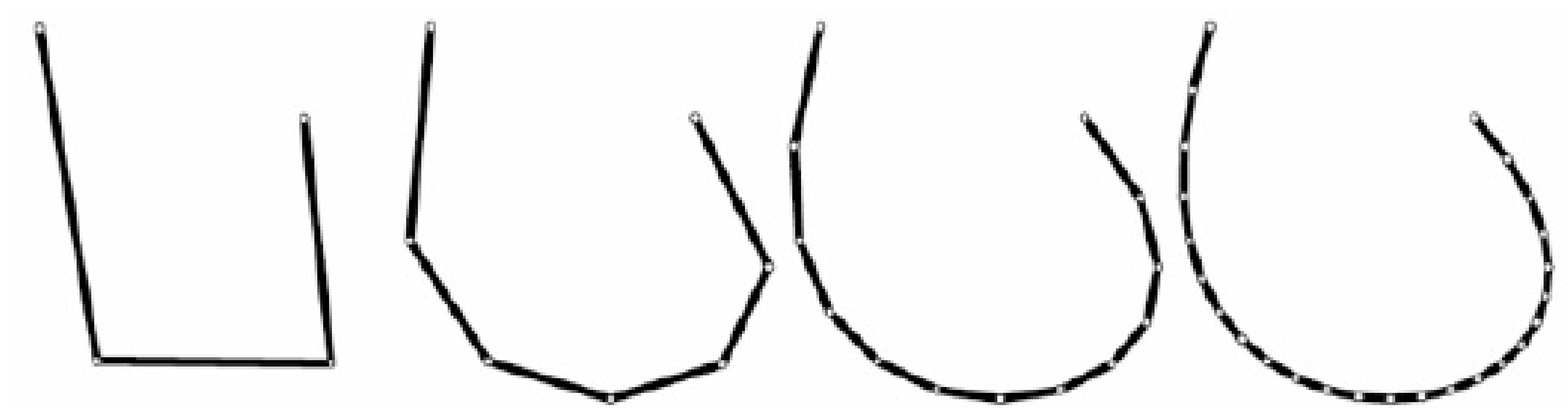
Subdivision Curve

- Chaikin's algorithm
  - 1 quarter 3 quarter algorithm



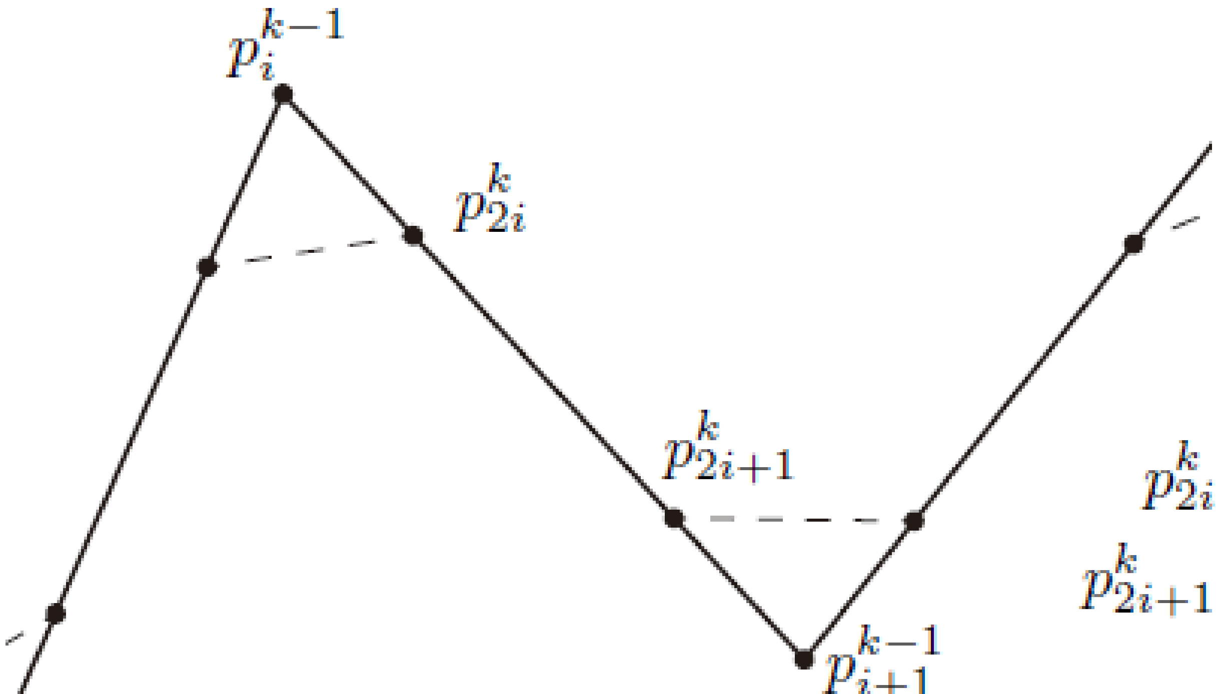
Subdivision curves

- Curve is defined as the *limit of a refinement process*
  - properties of curve depend on the rules
  - – some rules make polynomial curves, some don't
  - – complexity shifts from implementations to proofs



Corner cutting in equations

- New points are linear combinations of old ones
- Different treatment for odd-numbered and even numbered points.

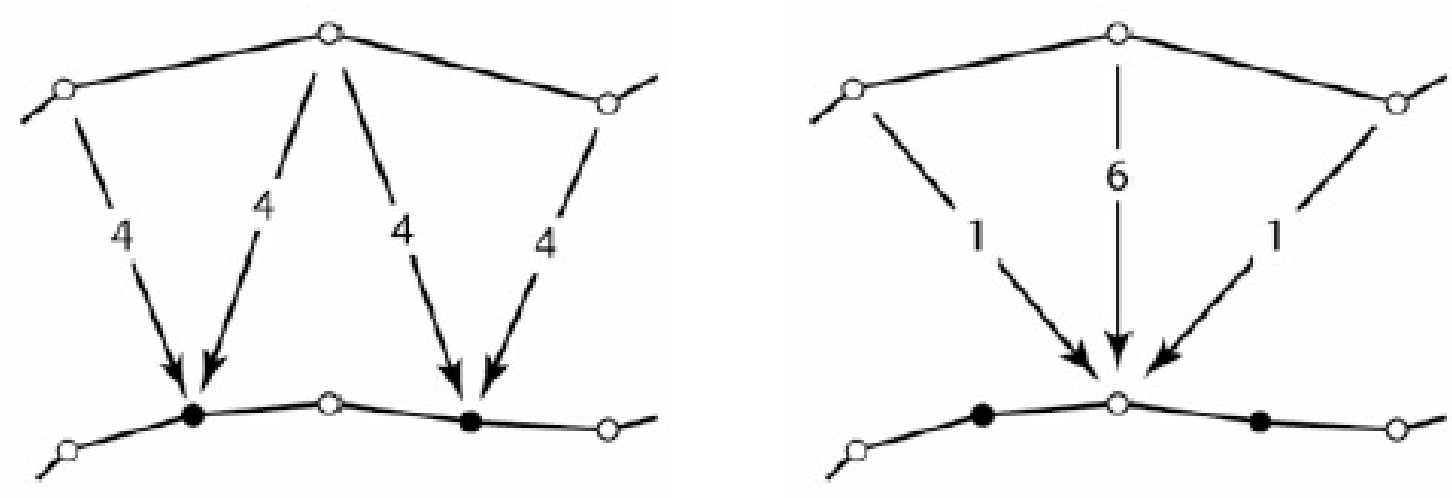


$$p_{2i}^k = (3p_i^{k-1} + p_{i+1}^{k-1})/4$$

$$p_{2i+1}^k = (p_i^{k-1} + 3p_{i+1}^{k-1})/4$$

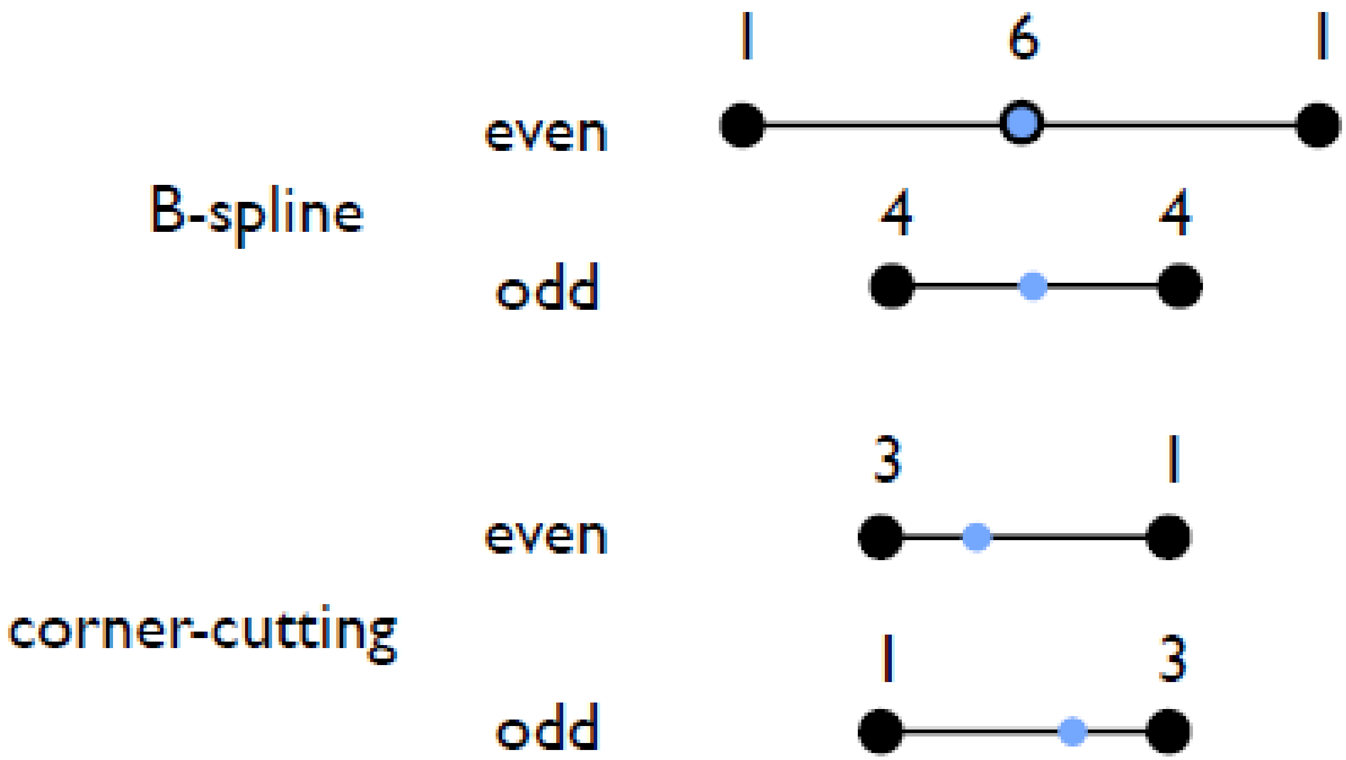
Subdivision for B-splines

- Control vertices of refined spline are linear combinations of the coarse spline

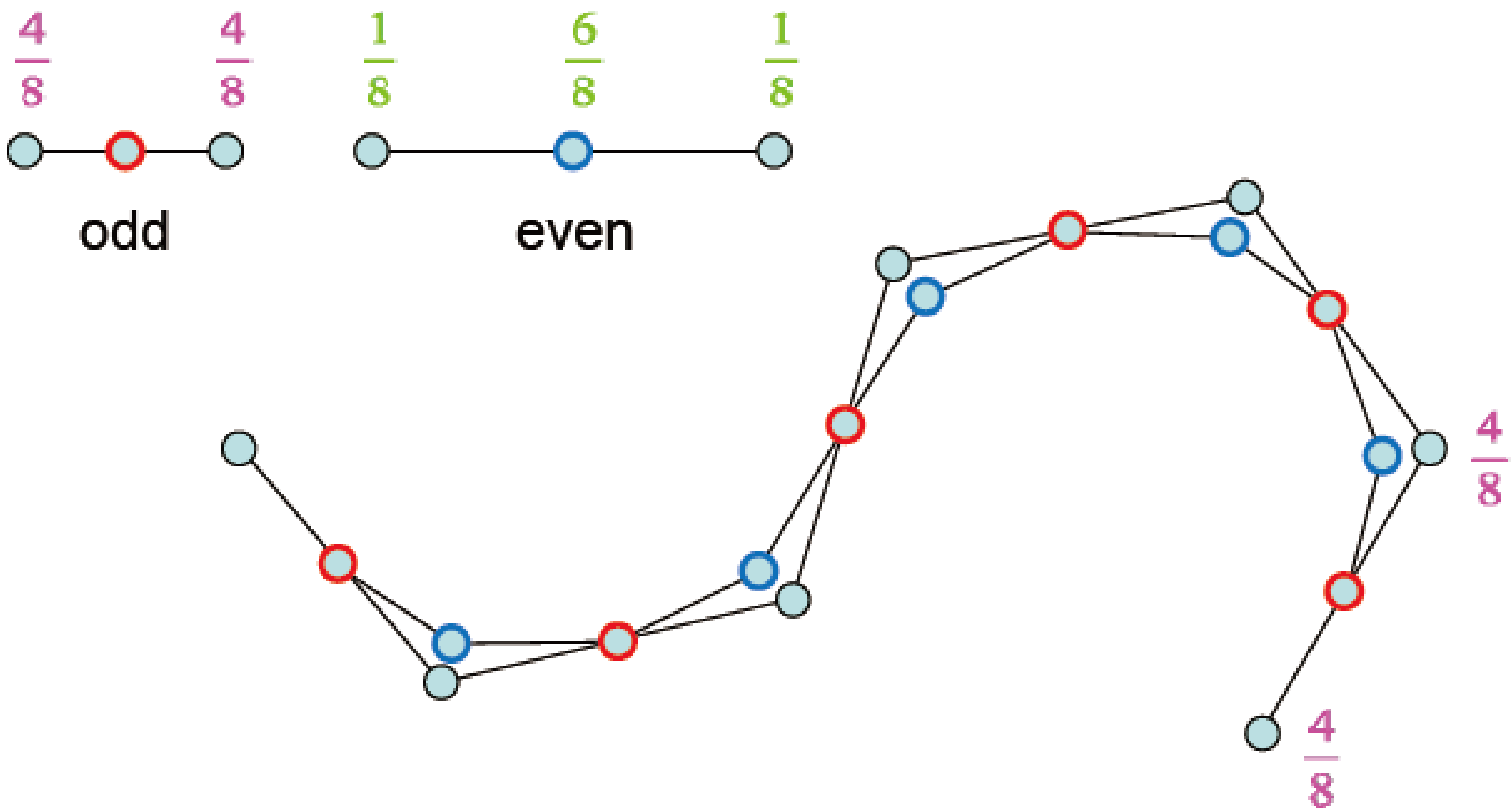


ODD **EVEN**

Subdivision rules as a mask



Cubic B-Spline



[Stanford CS468 Fall 2010 slides]

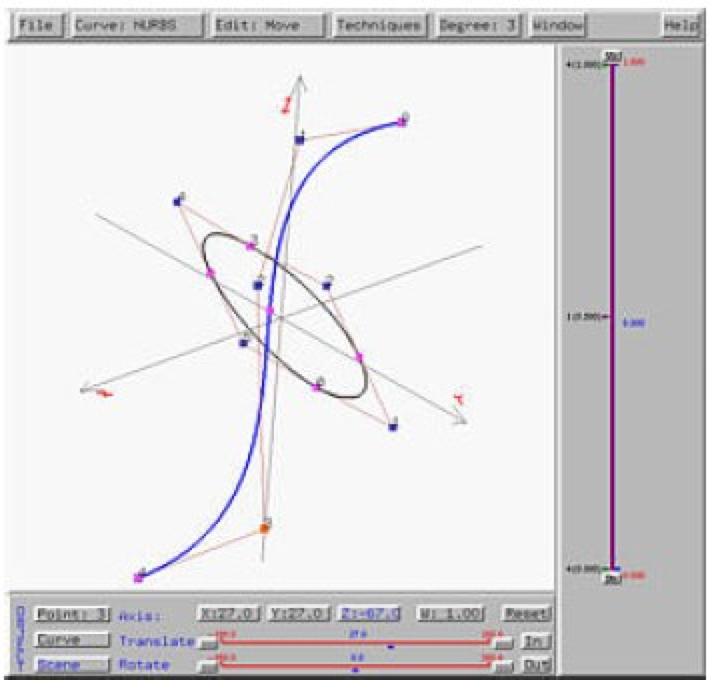
- Questions?

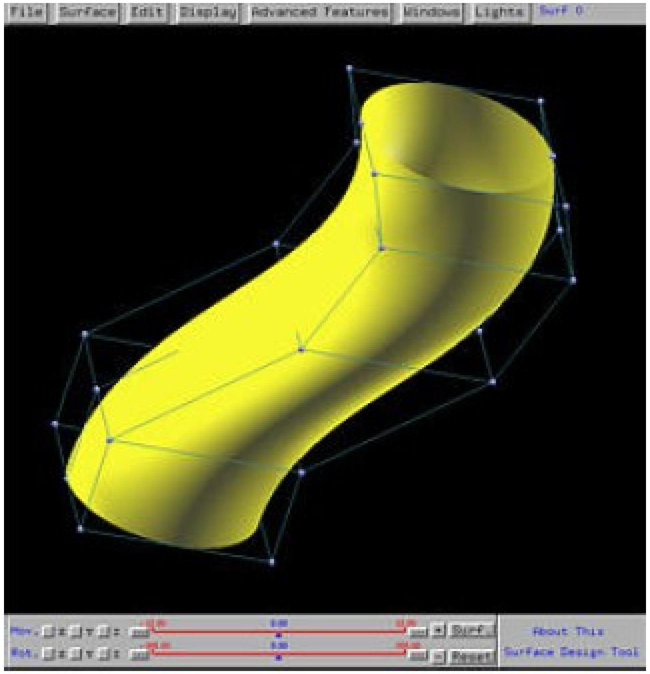
From curves to surfaces

- So far have discussed spline curves in 2D
  - this already provides of the mathematical machinery for several ways of building curved surfaces
- Building surfaces from 2D curves
  - extrusions and surfaces of revolution
- Building surfaces from 2D and 3D curves
  - generalized swept surfaces
- Building surfaces from spline patches
  - generalizing spline curves to spline patches
- Also to think about: generating triangles

Sweeping

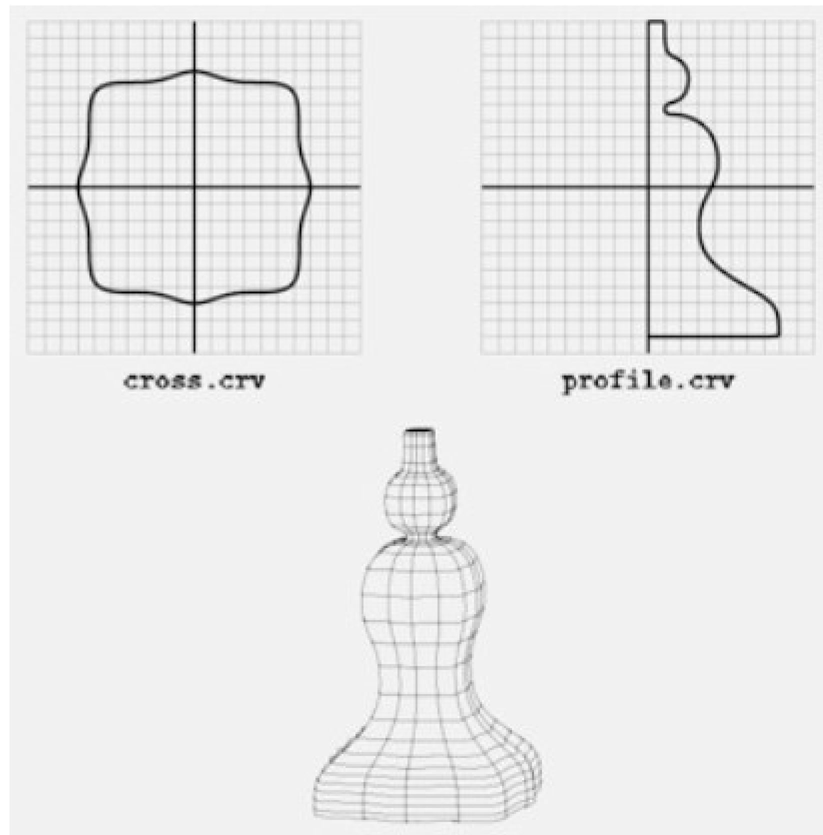
- Surface defined by a *cross section* moving along a *spline*
- Simple version: a single 3D curve for spine and a single 2D curve for the cross section





Sweeping

- General swept surfaces
  - varying radius
  - varying cross-section
  - curved axis

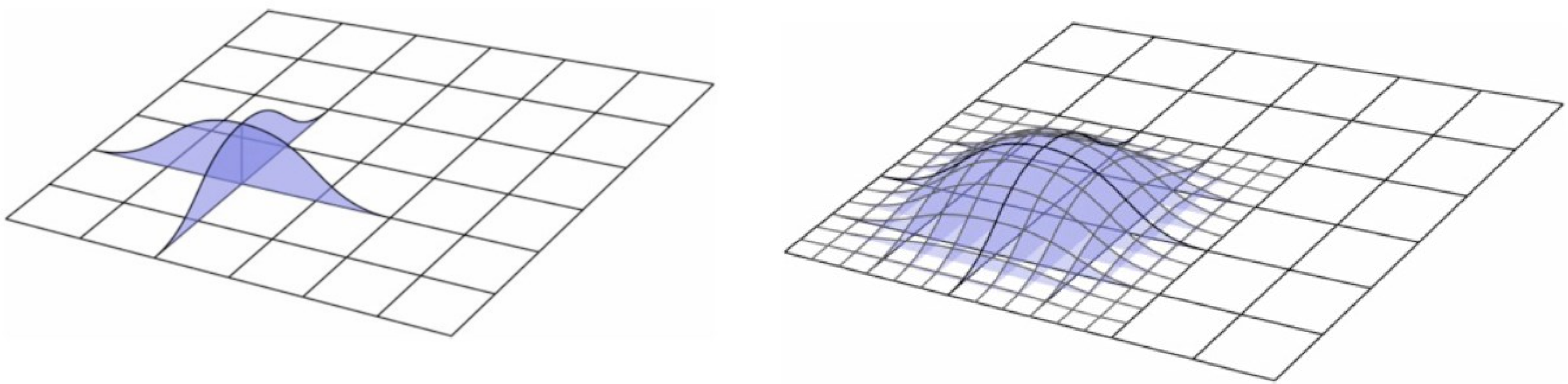


[Snyder 1992]

From curves to surface patches

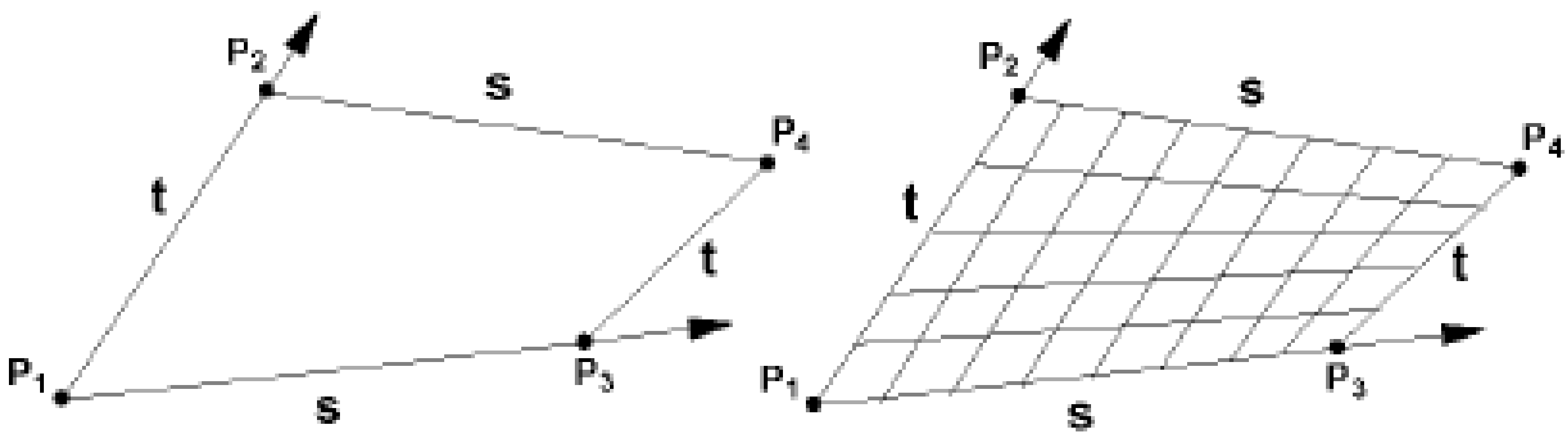
- Curve was sum of weighted 1D basis functions
- Surface is sum of weighted 2D basis functions
  - construct them as separable products of 1D functions.
  - choice of different splines
    - spline type
    - order
    - closed/open (B-spline)

Product construction



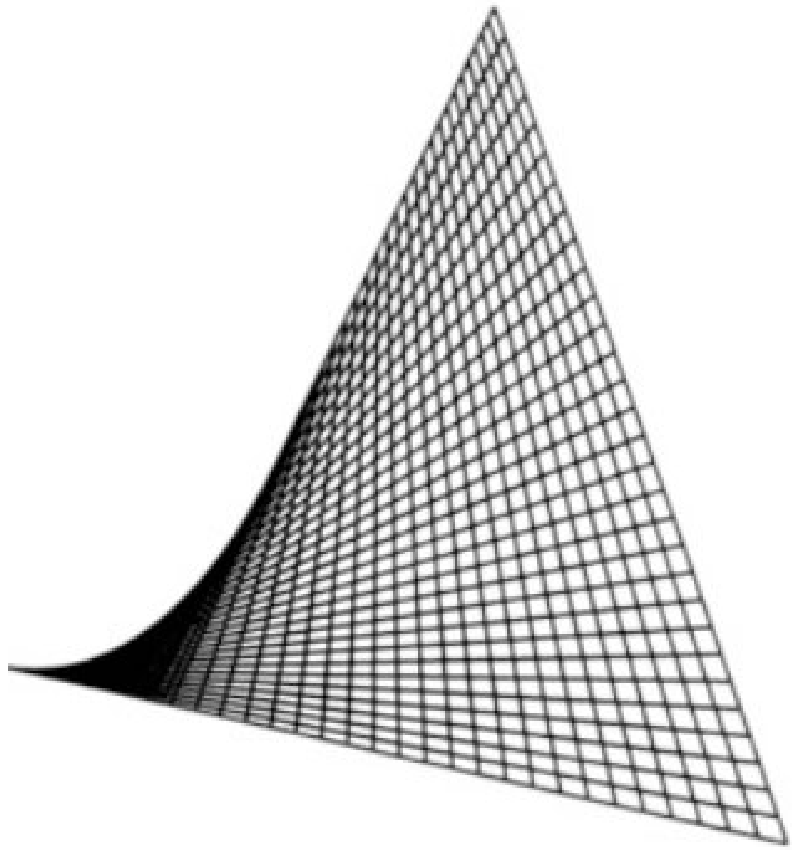
Spline Surface

- We can define a surface as the tensor product of two curves
- Bilinear Surface patch
- $L(P_1, P_2, \alpha) = (1 - \alpha)P_1 + \alpha P_2$
- $Q(s,t) = L(L(P_1, P_2, t), L(P_3, P_4, t), s)$



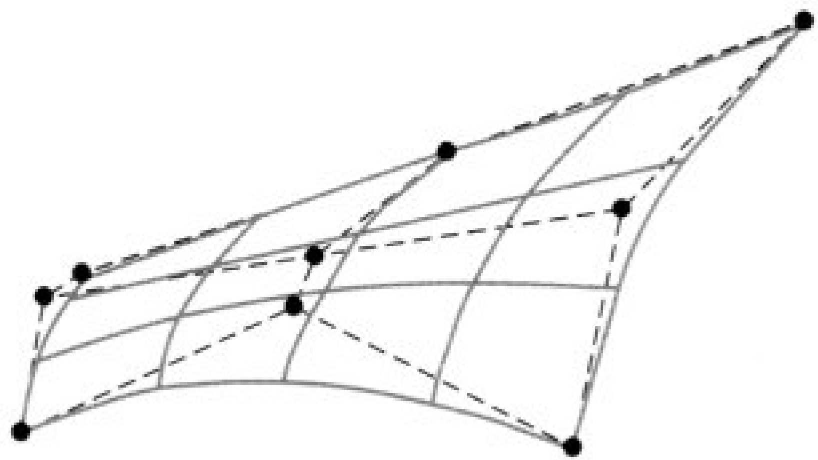
Bilinear patch

- 4 points, cross product of two linear segments



Biquadratic Bézier patch

- Cross product of quadratic Bézier curves

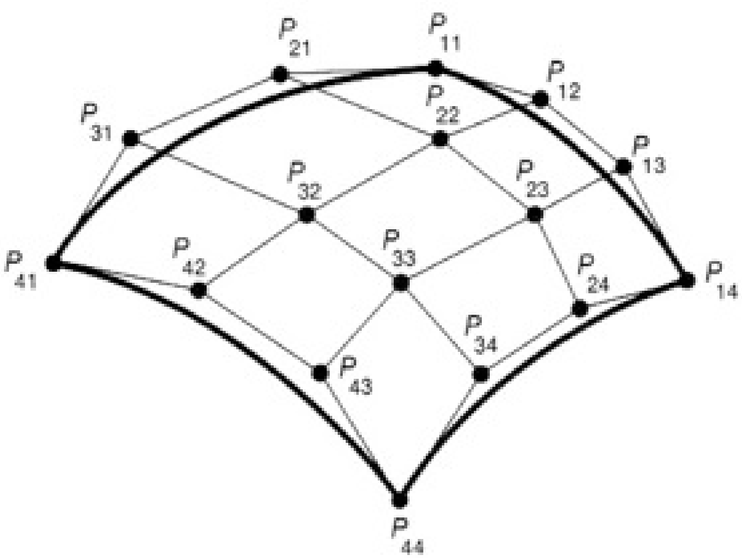


(a)

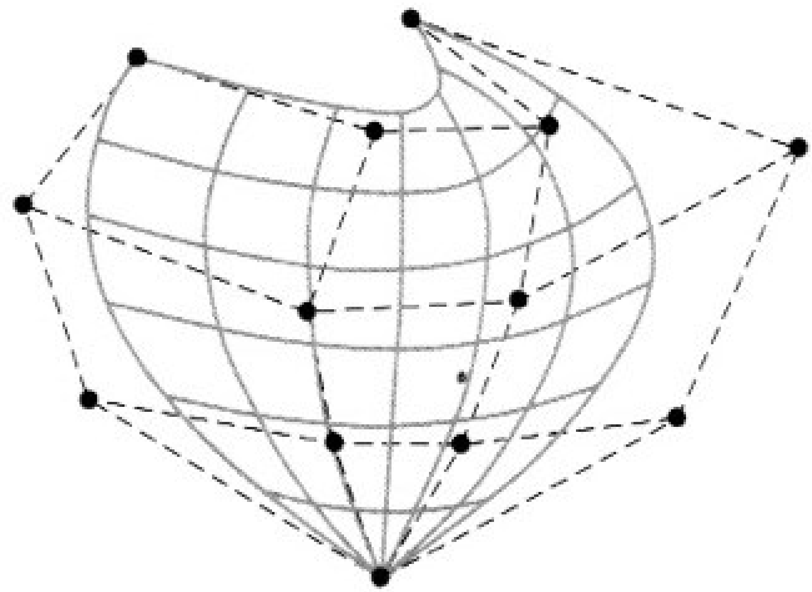
[Hearn & Baker]

Bicubic Bézier patch

- Cross product of two cubic Bézier segments



[Foley et al.]



(b)

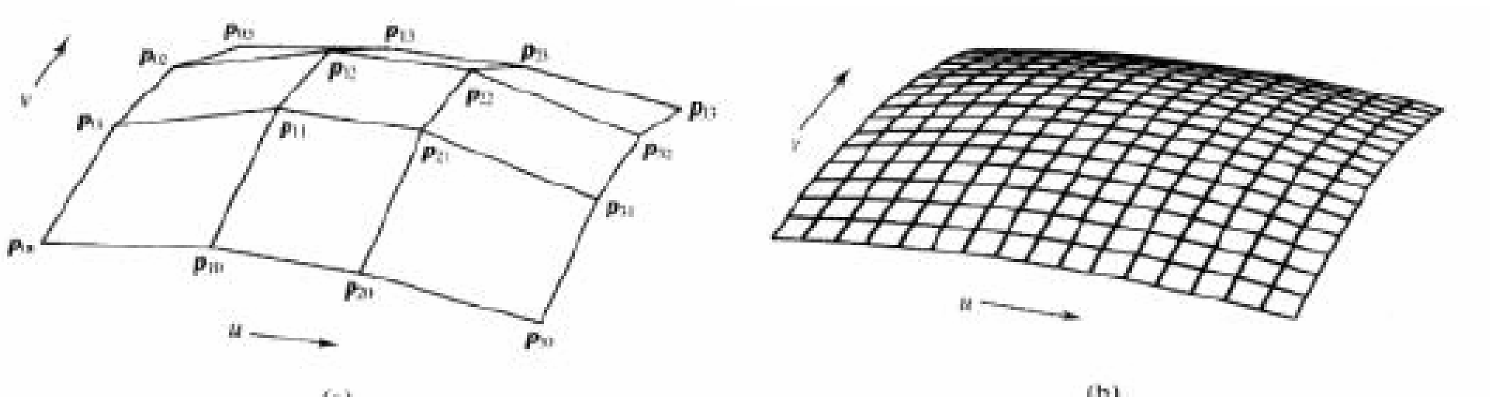
[Hearn & Baker]

Bicubic Bézier patch

Notation: $\mathbf{CB}(P_1, P_2, P_3, P_4, \alpha)$ is Bézier curve with control points P_i evaluated at α

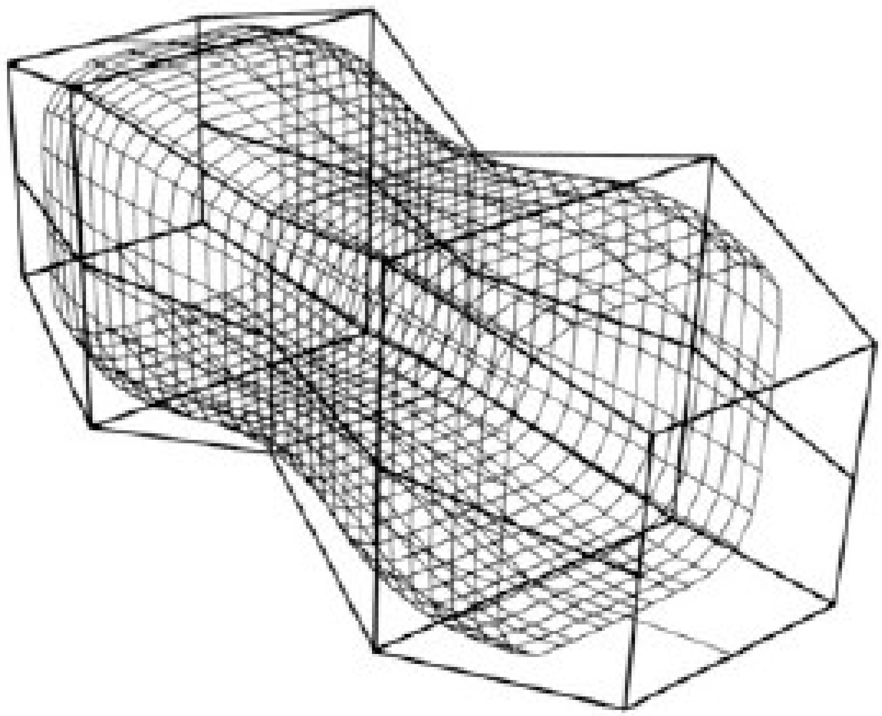
Define "Tensor-product" Bézier surface

$$
\begin{aligned}
Q(s,t) = \mathbf{CB}(\quad & \mathbf{CB}(P_{00}, P_{01}, P_{02}, P_{03}, t), \\
& \mathbf{CB}(P_{10}, P_{11}, P_{12}, P_{13}, t), \\
& \mathbf{CB}(P_{20}, P_{21}, P_{22}, P_{23}, t), \\
& \mathbf{CB}(P_{30}, P_{31}, P_{32}, P_{33}, t), \\
& s)
\end{aligned}
$$



Cylindrical B-spline surfaces

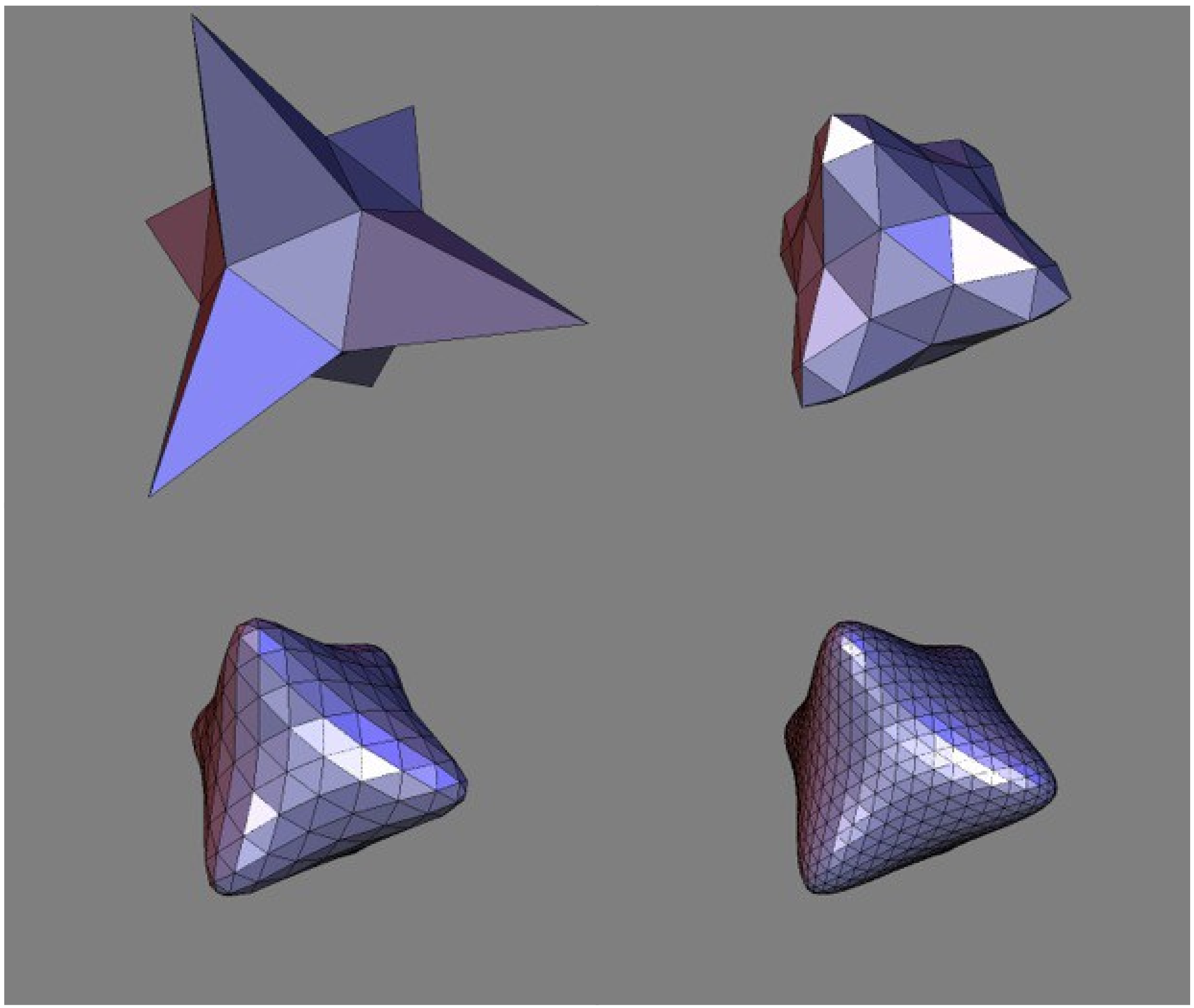
- Cross product of closed and open cubic B-splines



(b)

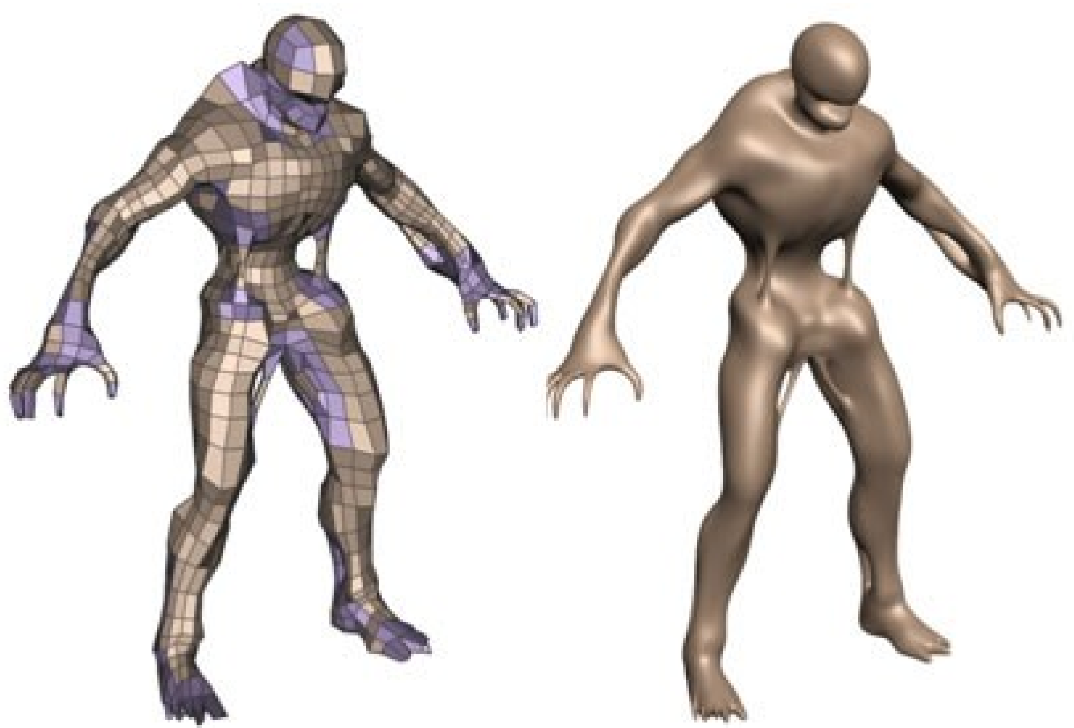
[Rogers]

Subdivision surface



Subdivision surface

- Subdivision surfaces
  - based on polygon meshes (quads or triangles)
  - rules for subdividing surface by adding new vertices

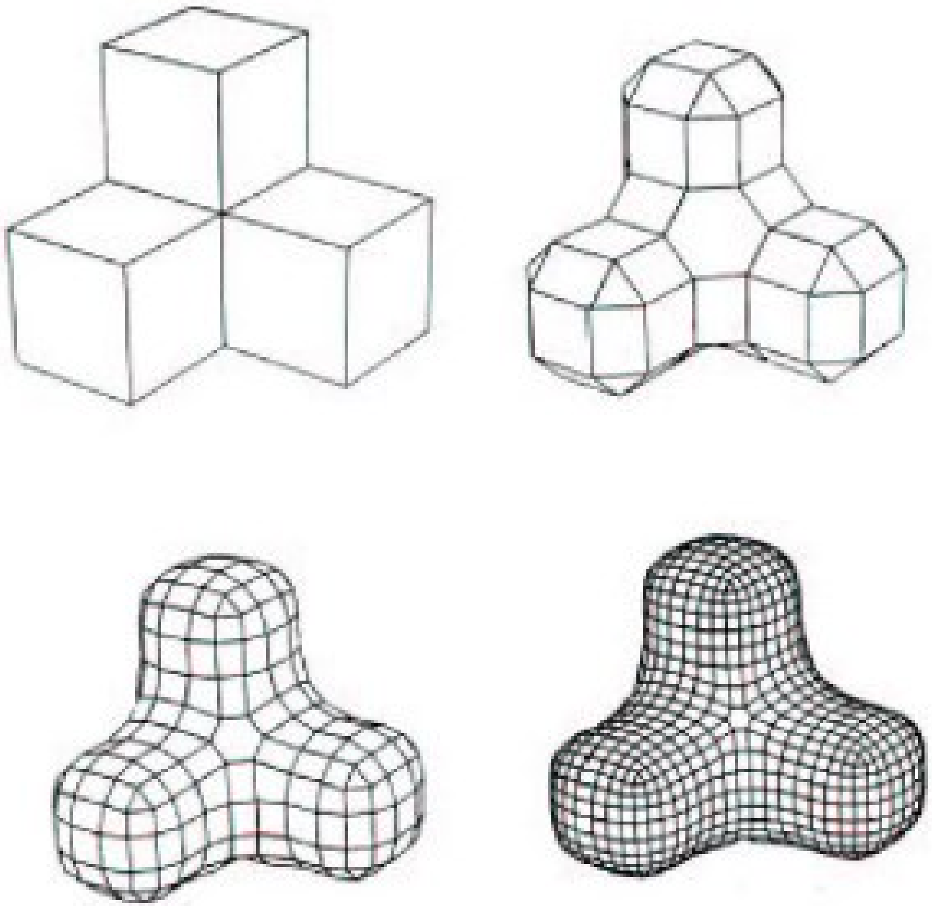


Generalizing from curves to surfaces

- Two parts to subdivision process
- Subdividing the mesh (computing new topology)
  - For curves: replace every segment with two segments
  - For surfaces: replace every face with some new faces
- Positioning the vertices (computing new geometry)
  - For curves: two rules (one for *odd* vertices, one for *even*)
    - New vertex's position is a weighted average of positions of old vertices that are nearby along the sequence
  - For surfaces: two kinds of rules (still called odd and even)
    - New vertex's position is a weighted average of positions of old vertices that are nearby in the mesh

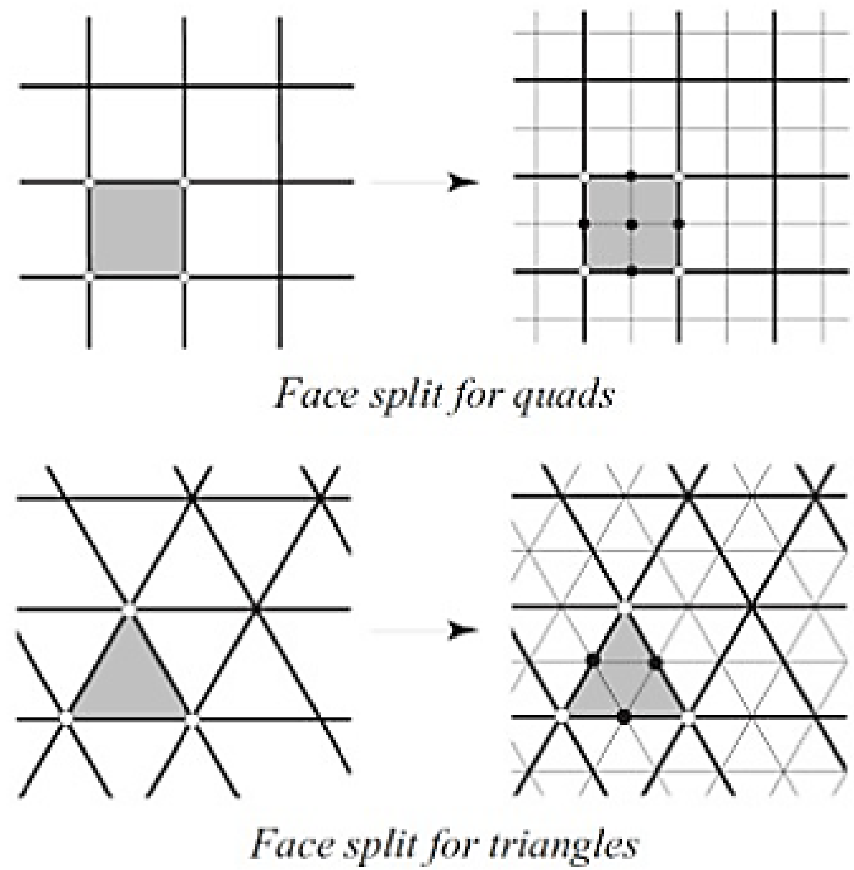
Subdivision Surface

- Chaikin's Algorithm



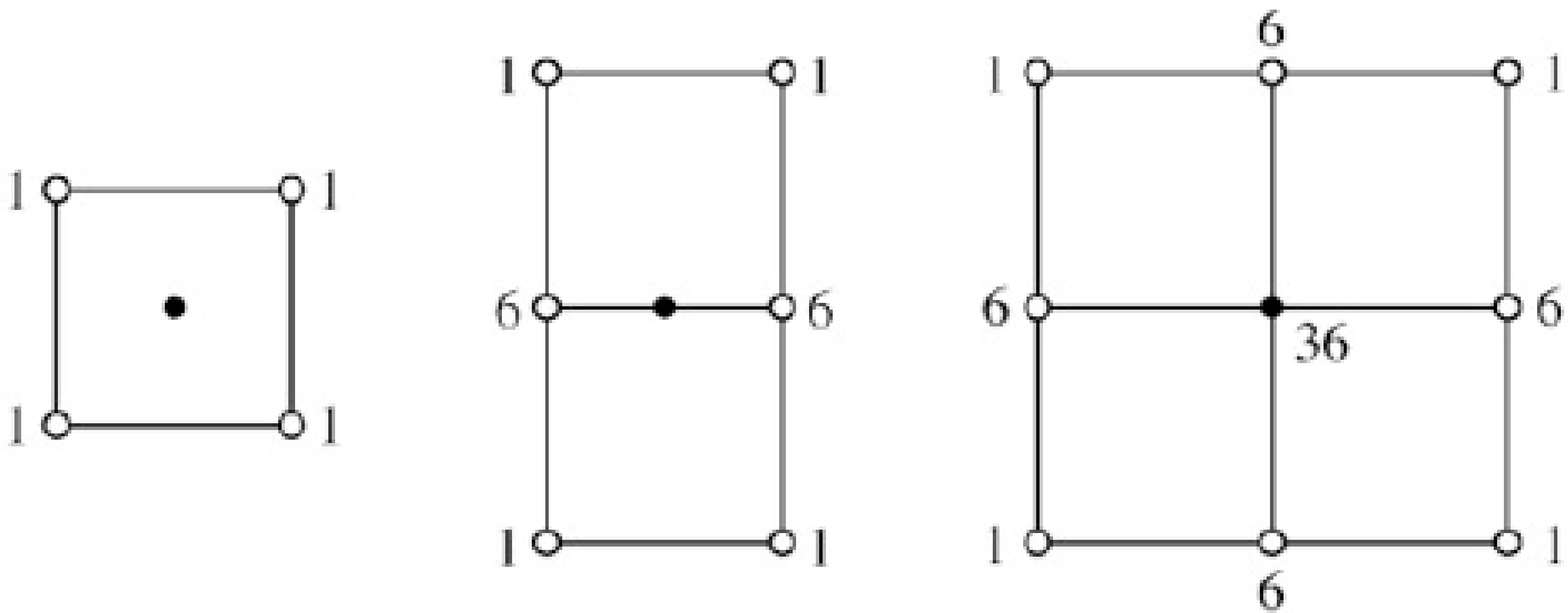
Subdivision of meshes

- Quadrilaterals
  - Catmull-Clark 1978
- Triangles
  - Loop 1987

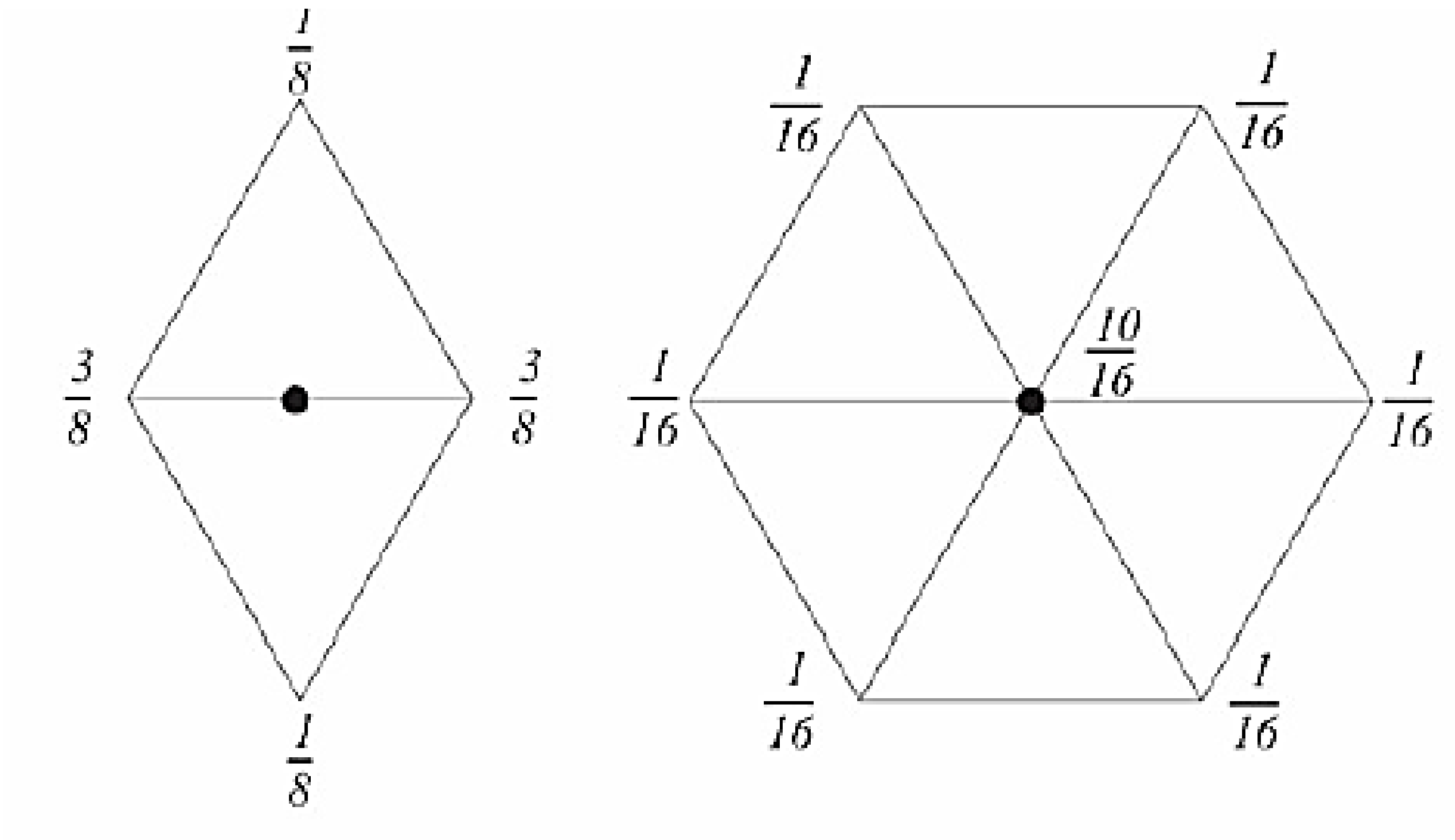


[Schröder & Zorin SIGGRAPH 2000 course 23]

Catmull-Clark regular rules



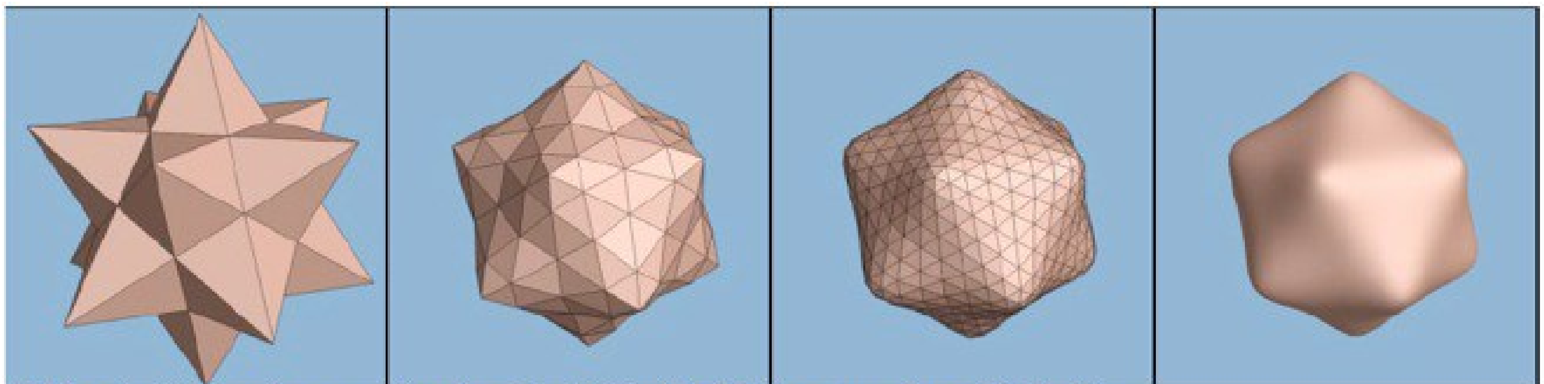
Loop regular rules



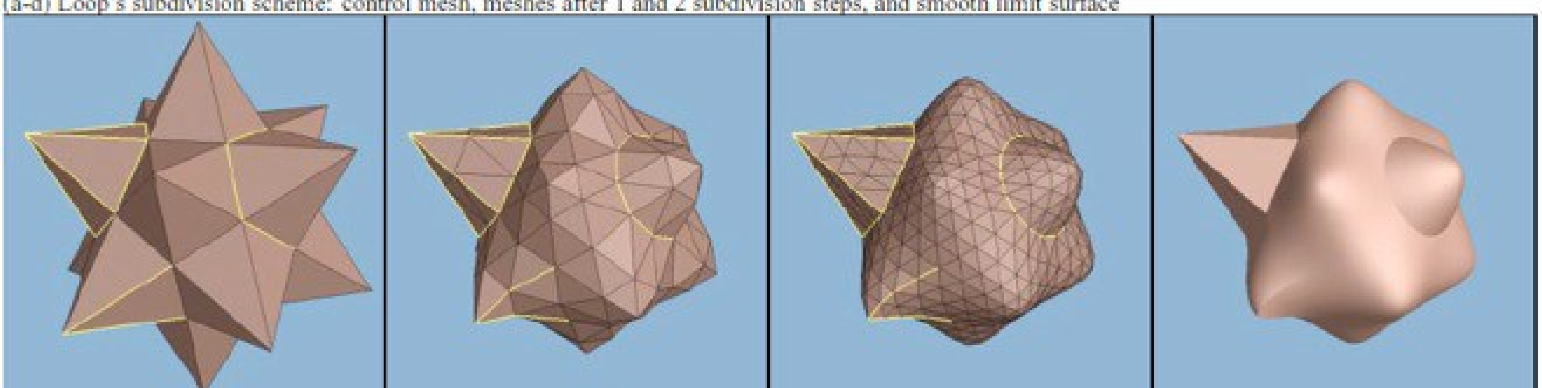
[Schröder & Zorin SIGGRAPH 2000 course 23]

How about the boundary?

- Treat it with other rules
- Crease rule



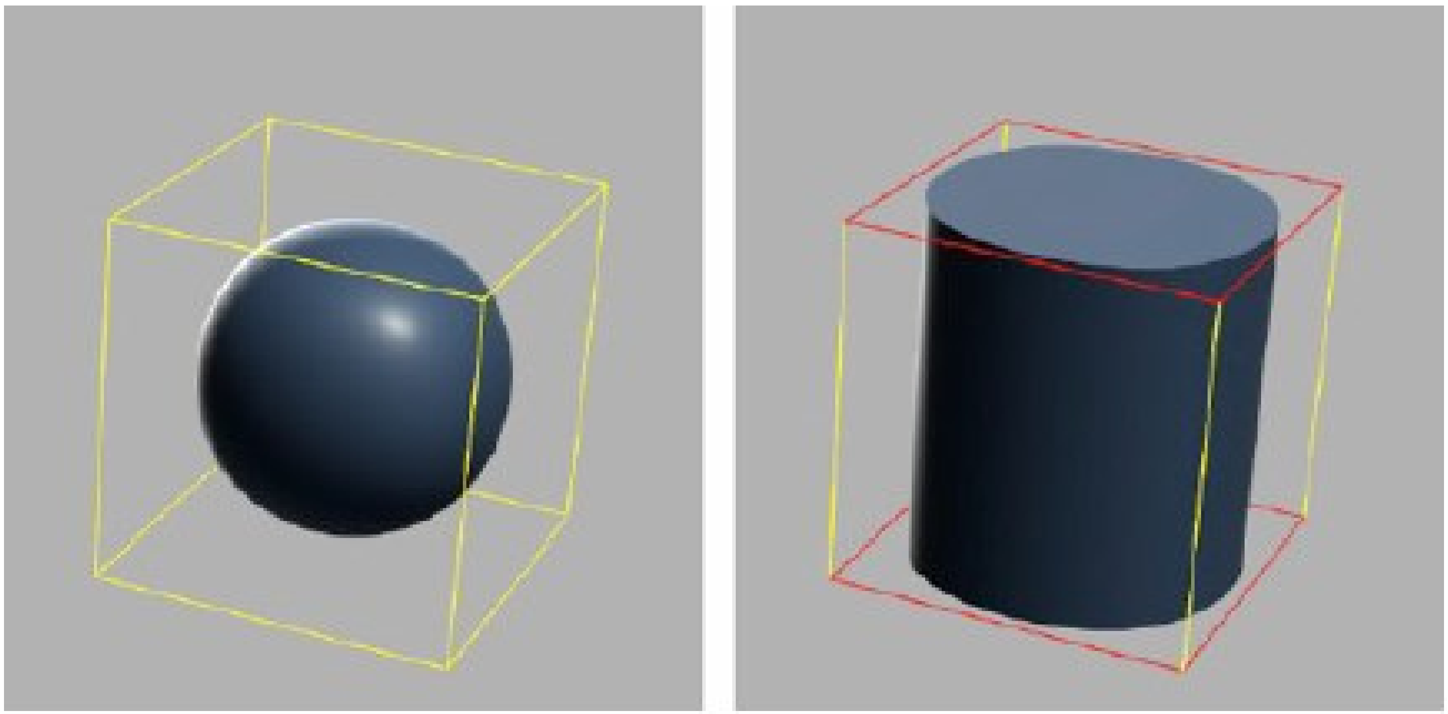
(a-d) Loop's subdivision scheme: control mesh, meshes after 1 and 2 subdivision steps, and smooth limit surface



(e-h) Our piecewise smooth subdivision scheme: tagged control mesh, meshes after 1 and 2 subdivision steps, and piecewise smooth limit surface

[Hugues Hoppe]

Catmull-Clark with creases



[DeRose et al. SIGGRAPH 1998]

Subdivision vs Splines

- In regular regions, behavior is identical
- At extraordinary vertices, achieve C1
  - near extraordinary, different from splines
- Linear everywhere
  - mapping from parameter space to 3D is a linear combination of the control points

OpenGL support

- Legacy
  - Evaluators
- Modern
  - DIY

References

- Steve Marschner, CS4620/5620 Computer Graphics, Cornell
- Cutler and Durand, MIT EECS 6.837
- Tom Thorne, COMPUTER GRAPHICS, The University of Edinburgh
- Elif Tosun, Computer Graphics, The University of New York
- C.-K. Shene, CS3621 Introduction to Computing with Geometry Notes, Michigan Technological University

- Questions?