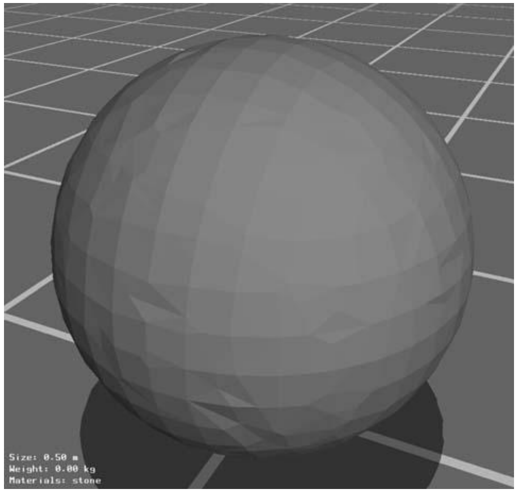
Curves & Surfaces

Today

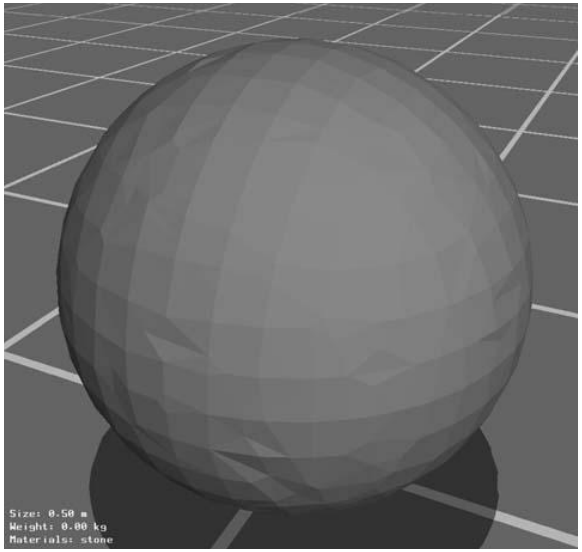
- Review
- Motivation
  - Limitations of Polygonal Models
  - Phong Normal Interpolation
  - Some Modeling Tools & Definitions
- Curves
- Surfaces / Patches
- Subdivision Surfaces
- Procedural Texturing

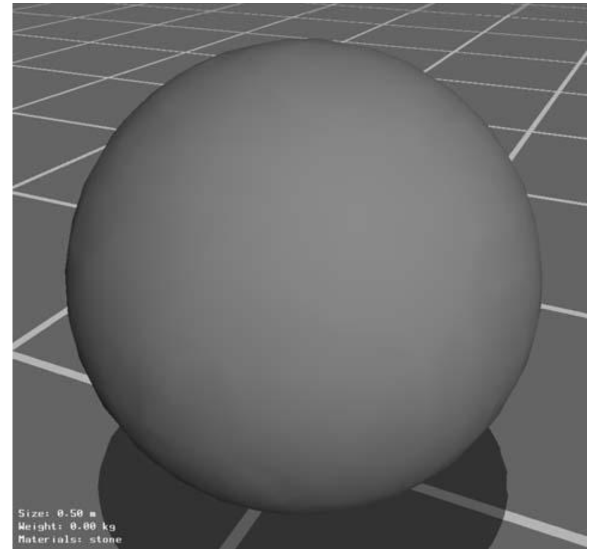
Limitations of Polygonal Meshes

- planar facets
- fixed resolution
- deformation is difficult
- no natural parameterization



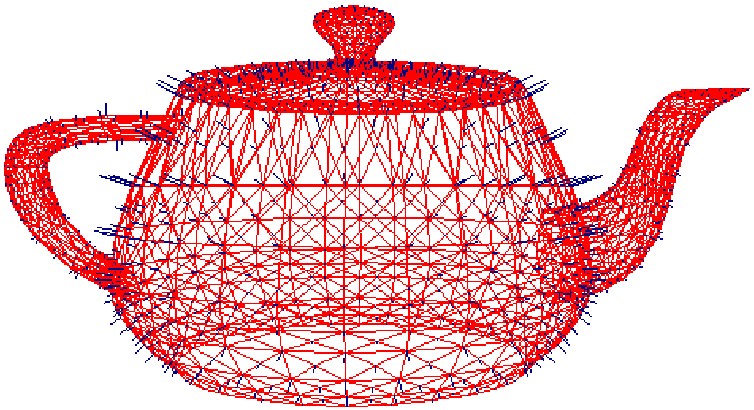
Can We Disguise the Facets?

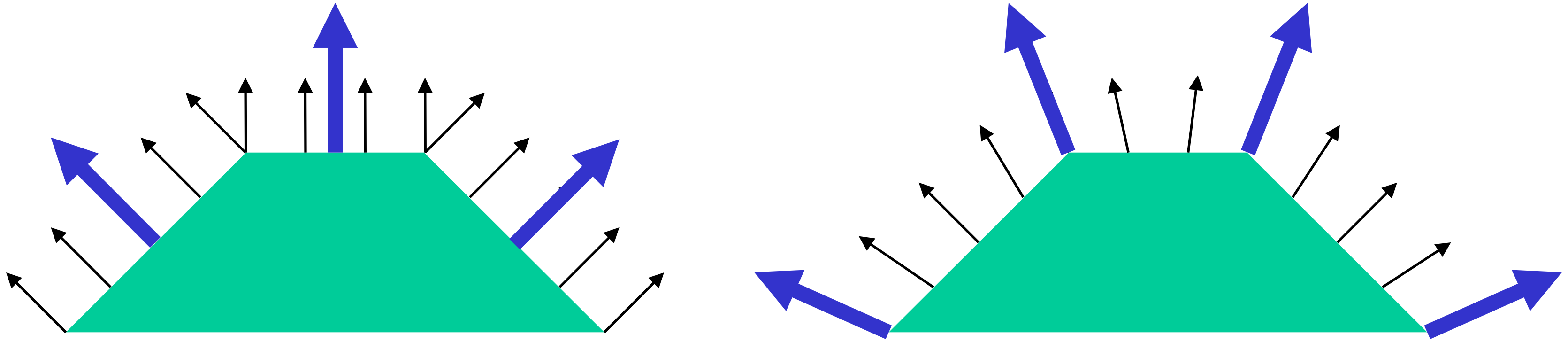


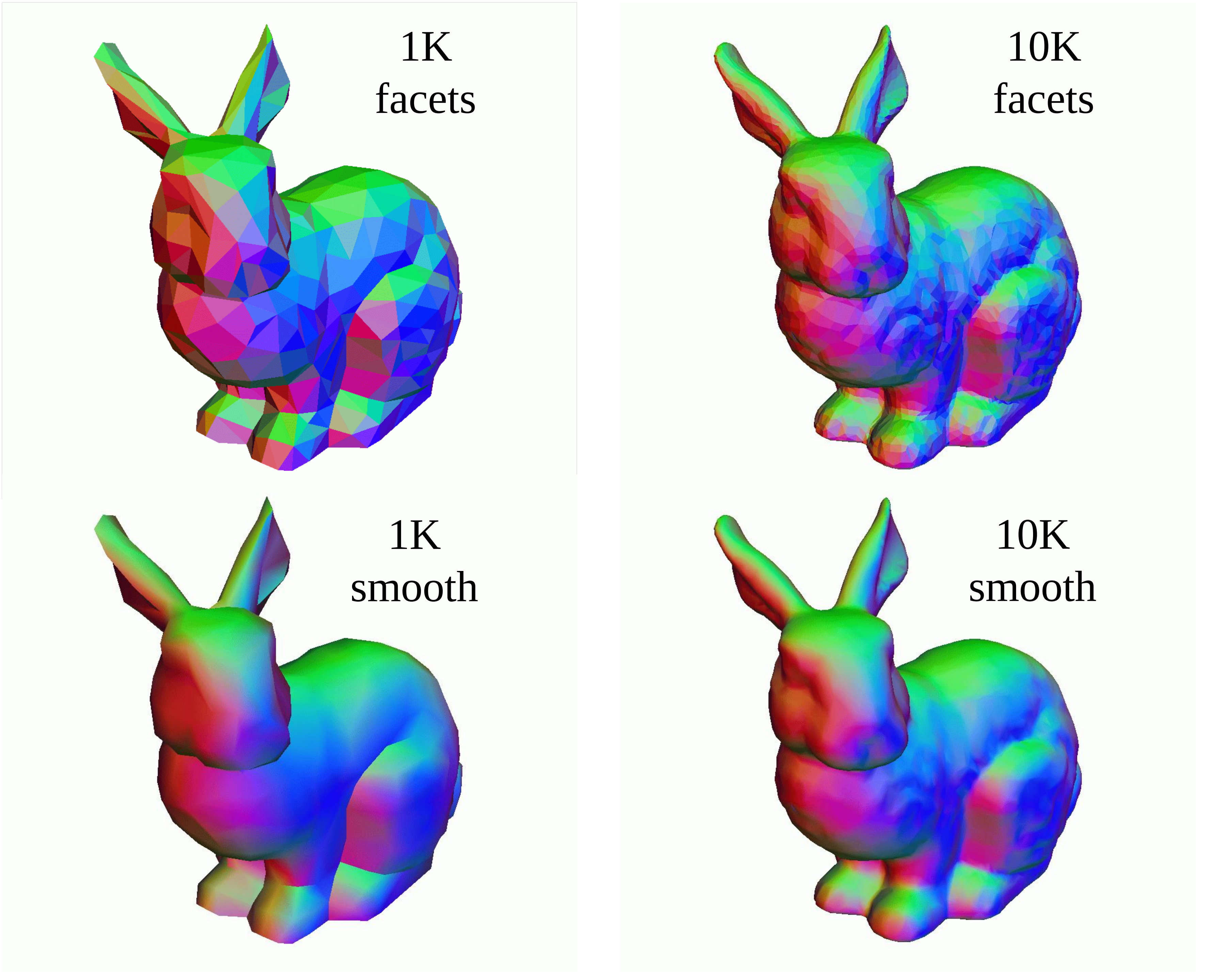


Phong Normal Interpolation

- Not Phong *Shading* from Assignment 3
- Instead of using the normal of the triangle, interpolate an averaged normal at each vertex across the face
- Must be renormalized

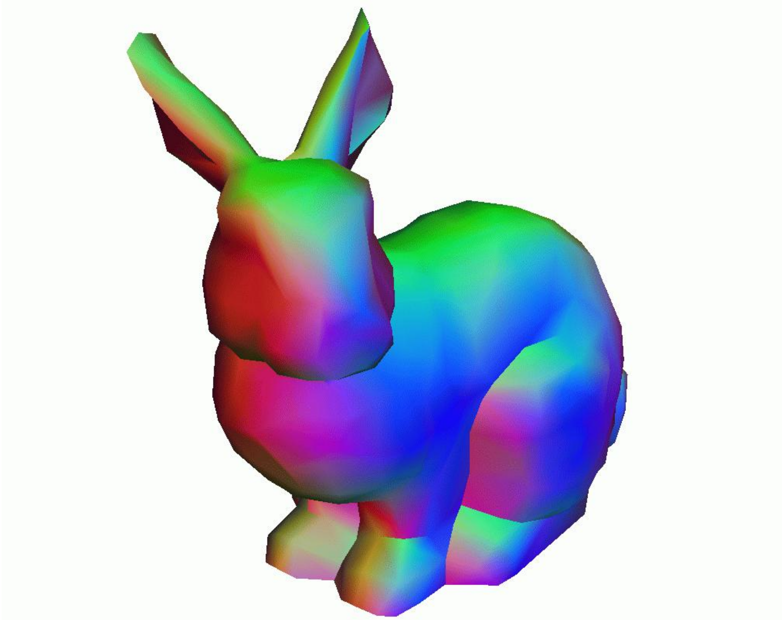




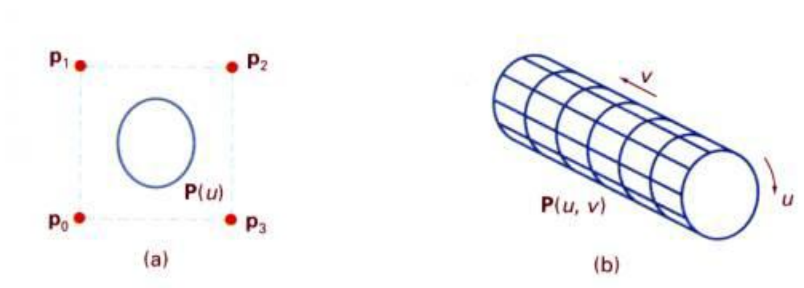

1K
facets
10K
facets
1K
smooth
10K
smooth

Better, but not always good enough

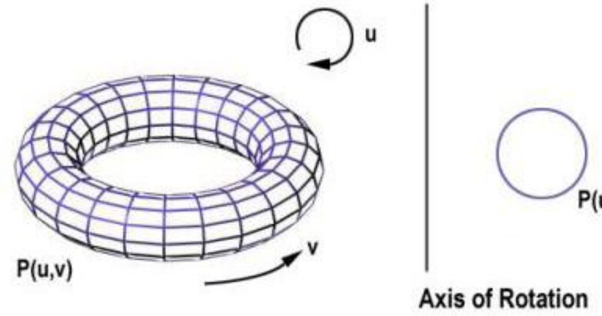
- Still low resolution (missing fine details)
- Still have polygonal silhouettes
- Intersection depth is planar
- Collisions in a simulation
- Solid Texturing
- ...



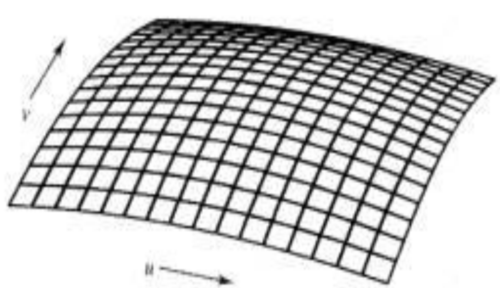
Some Non-Polygonal Modeling Tools



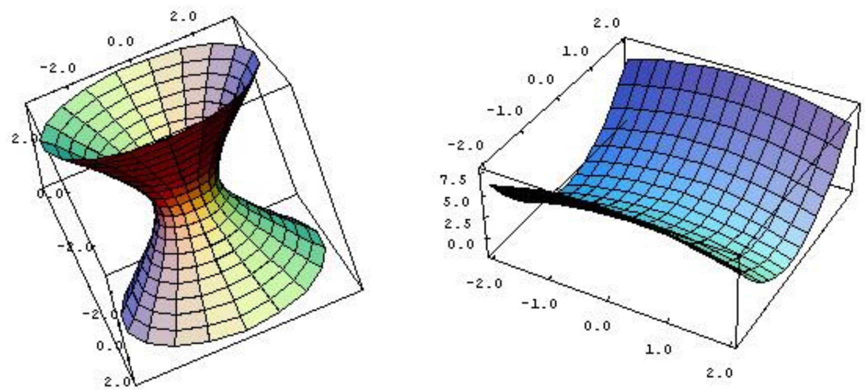
Extrusion



Surface of Revolution



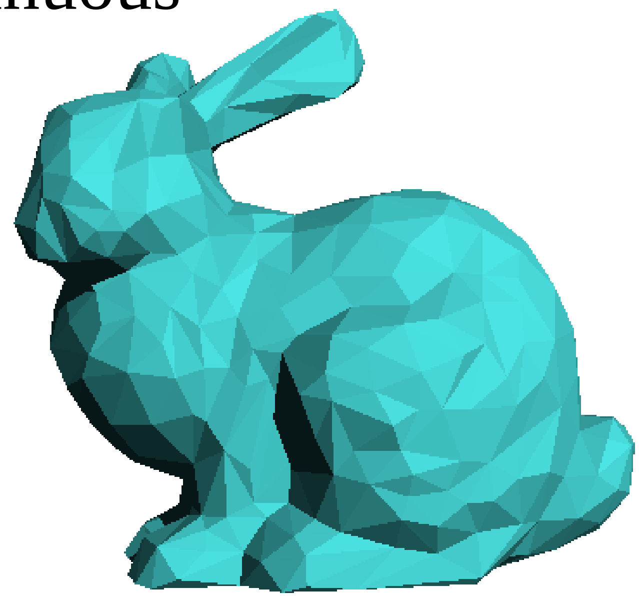
Spline Surfaces/Patches



Quadrics and other implicit polynomials

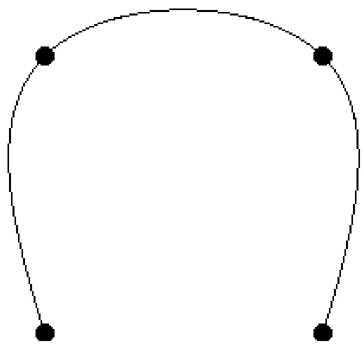
Continuity definitions:

- 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 2^{nd} derivative is continuous
  - Actually important for shading

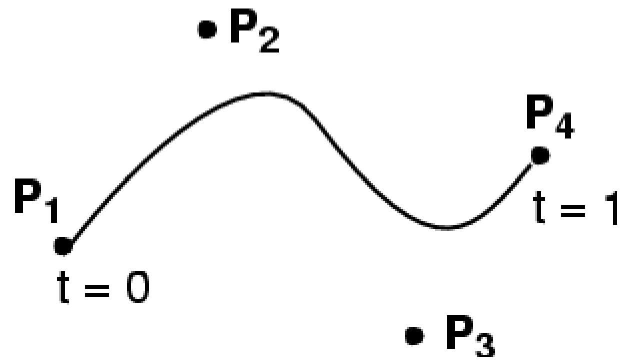


Definition: What's a Spline?

- Smooth curve defined by some control points
- Moving the control points changes the curve

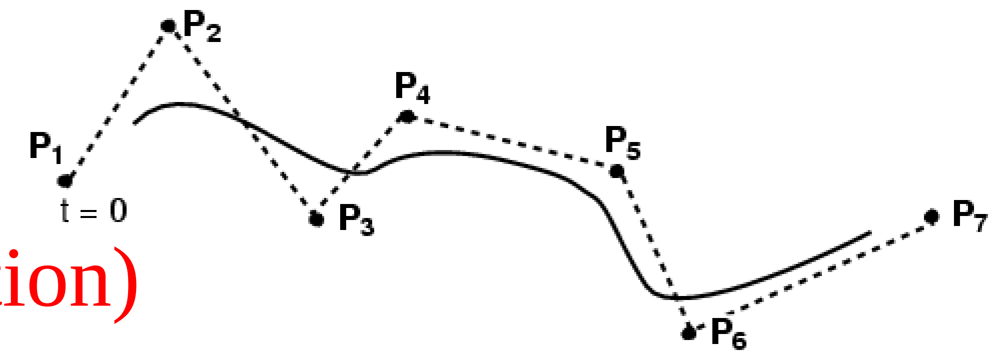


Interpolation

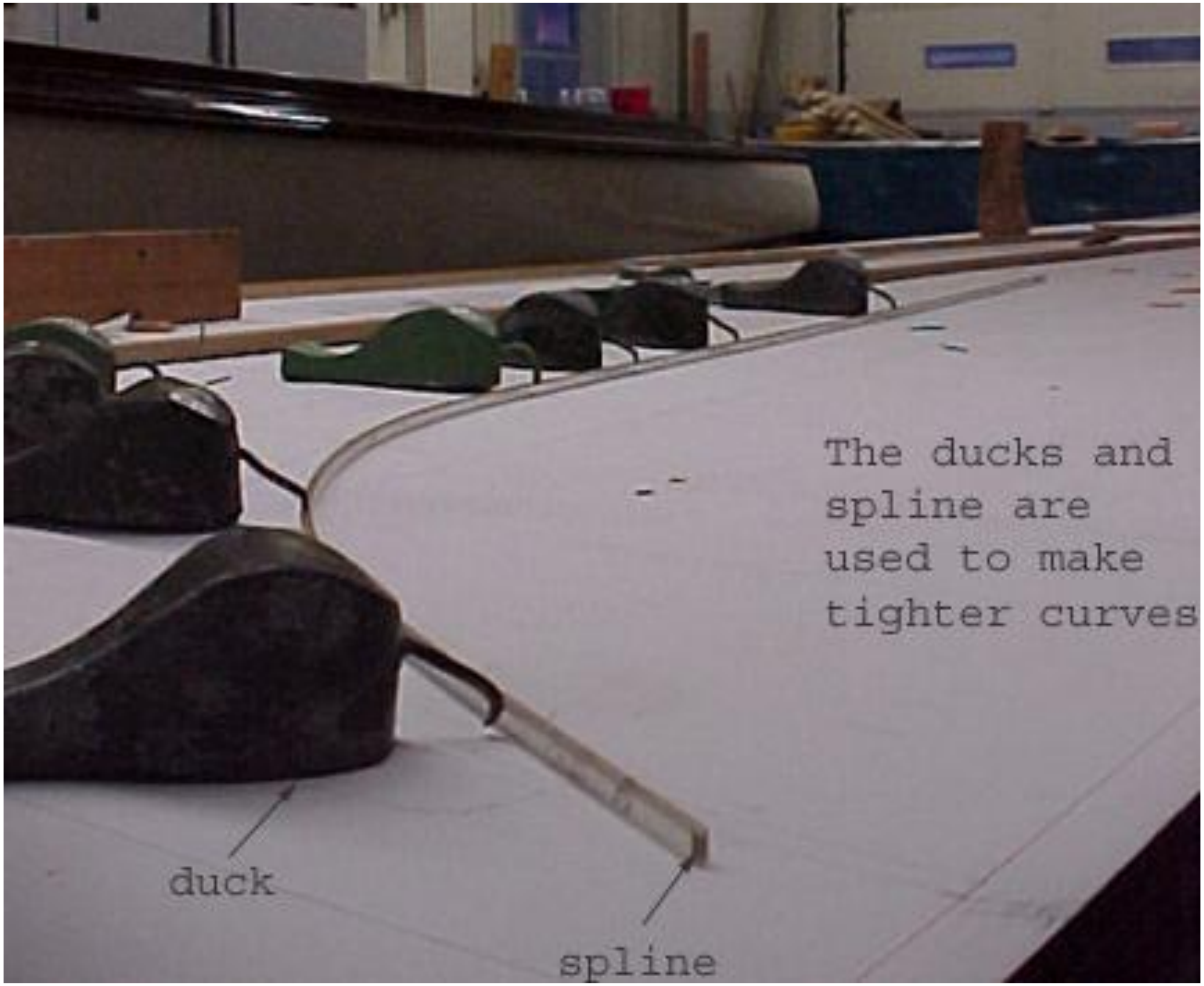


Bézier (approximation)

BSpline (approximation)



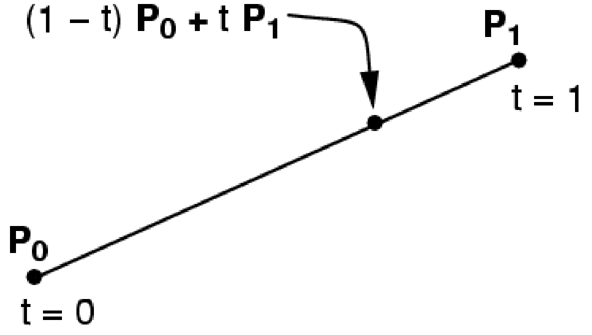
Interpolation Curves / Splines

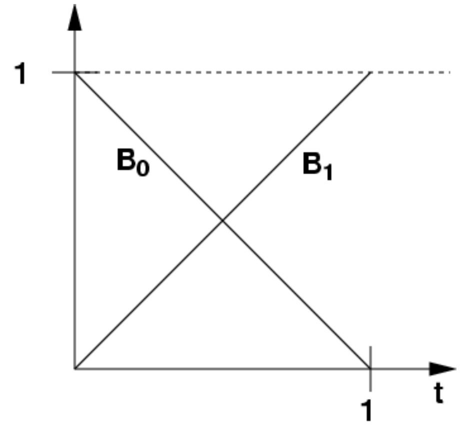


www.abm.org

Linear Interpolation

- Simplest "curve" between two points

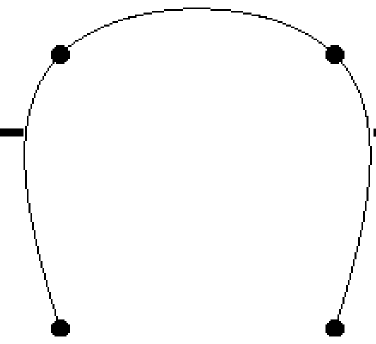




$$Q(t) = \begin{pmatrix} Q_x(t) \\ Q_y(t) \\ Q_z(t) \end{pmatrix} = \begin{pmatrix} (P_0) & (P_1) \end{pmatrix} \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} t \\ 1 \end{pmatrix}$$

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

Interpolation Curves

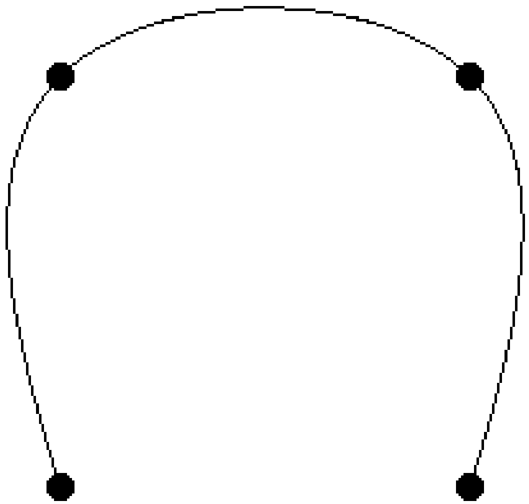


- Curve is constrained to pass through all control points
- Given points P_0, P_1, ... P_n, find lowest degree polynomial which passes through the points

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

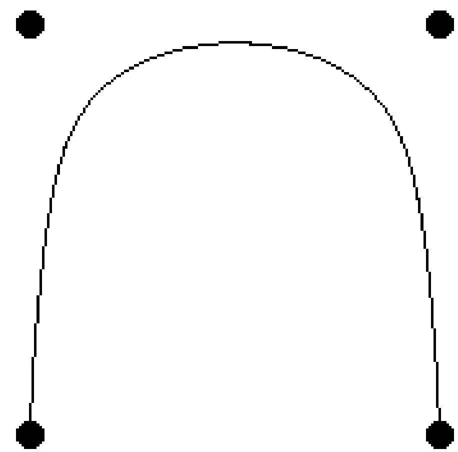
$$Q(t) = \mathbf{GBT(t)} = \text{Geometry } \mathbf{G} \cdot \text{Spline Basis } \mathbf{B} \cdot \text{Power Basis } \mathbf{T(t)}$$

Interpolation vs. Approximation Curves



Interpolation

curve must pass
through control points

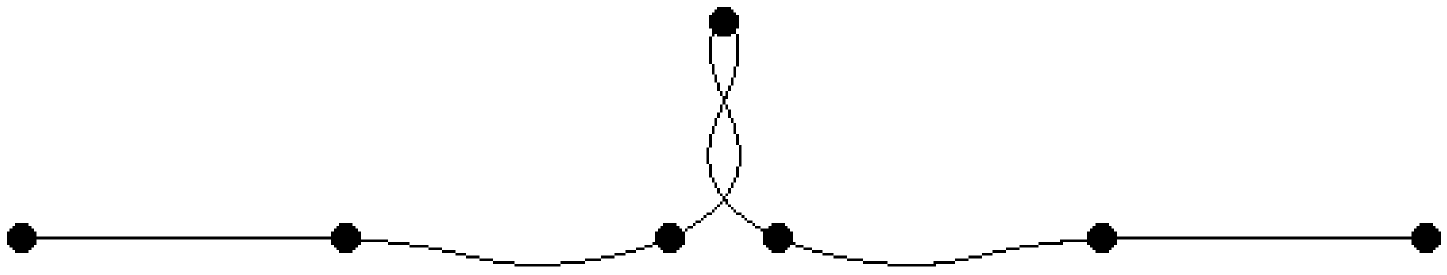


Approximation

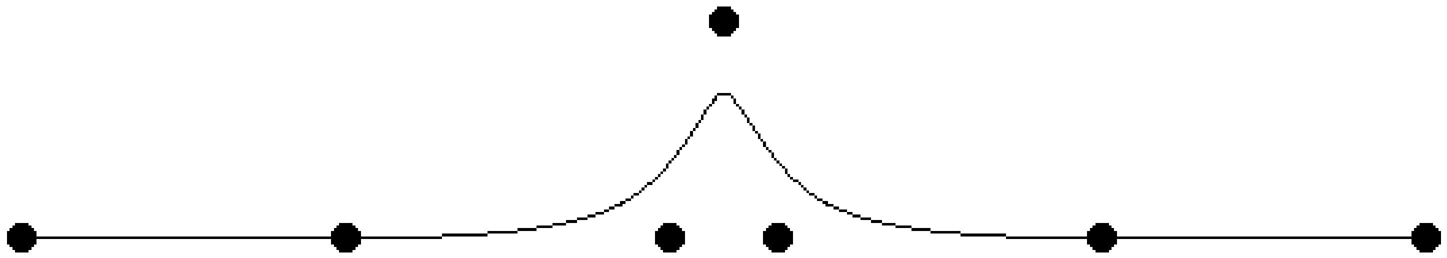
curve is influenced
by control points

Interpolation vs. Approximation Curves

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

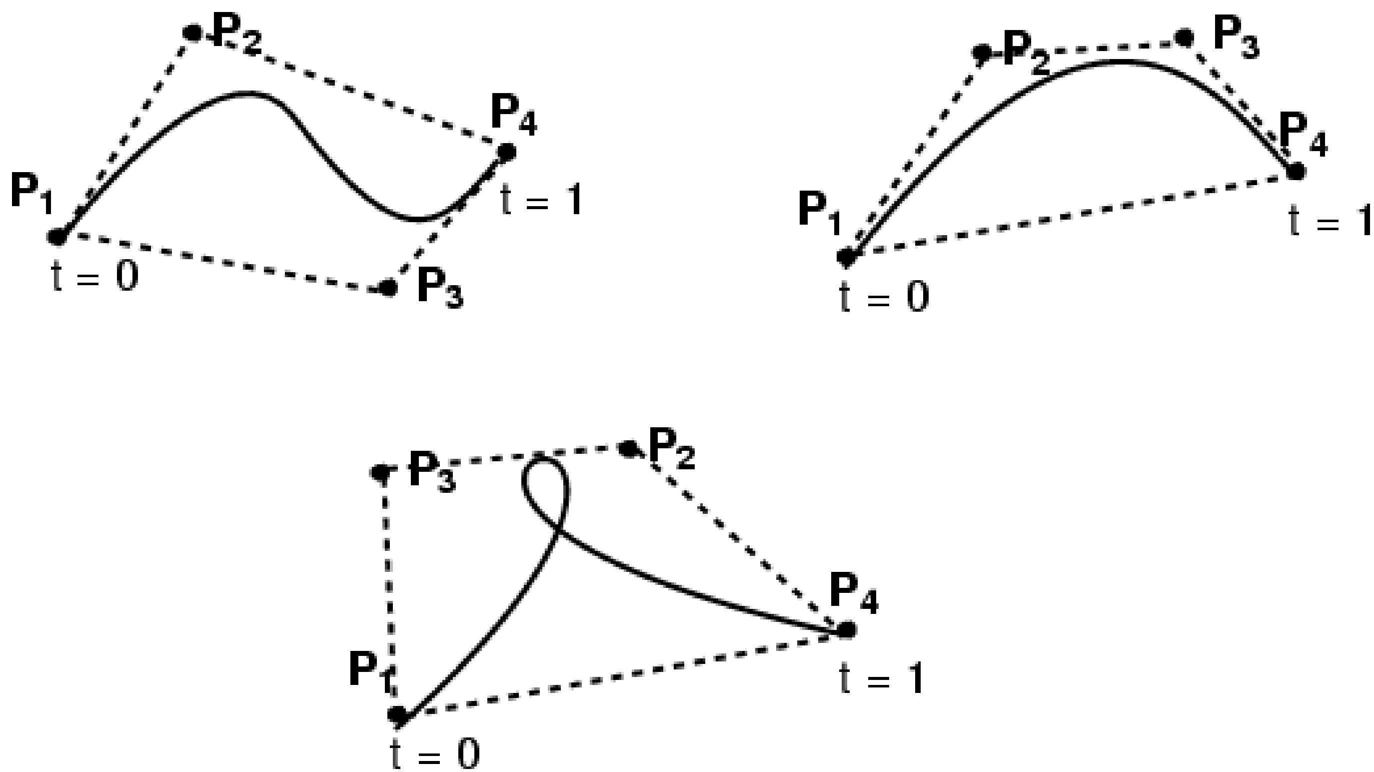


- Approximation Curve – more reasonable?



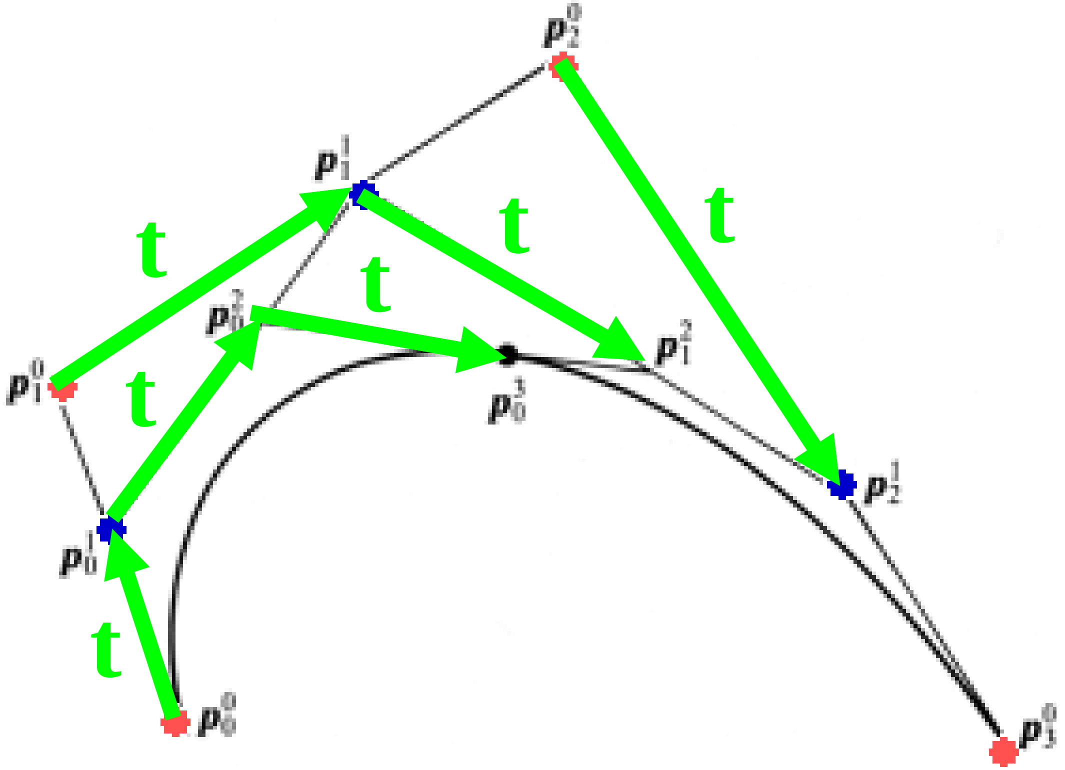
Cubic Bézier Curve

- 4 control points
- Curve passes through first & last control point
- Curve is tangent at $\mathbf{P}_0$ to ($\mathbf{P}_0$-$\mathbf{P}_1$) and at $\mathbf{P}_4$ to ($\mathbf{P}_4$-$\mathbf{P}_3$)

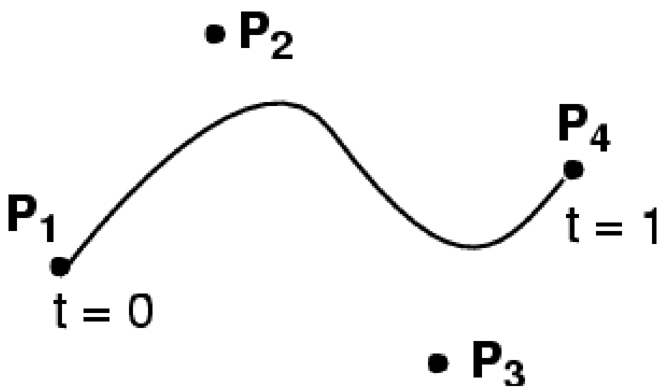


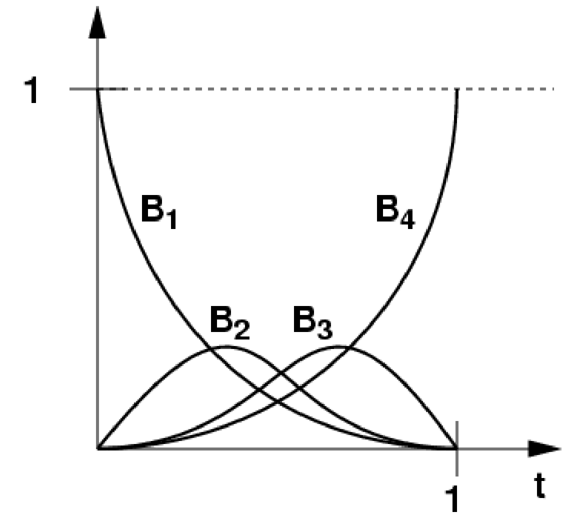
Cubic Bézier Curve

- de Casteljau's algorithm for constructing Bézier curves



Cubic Bézier Curve



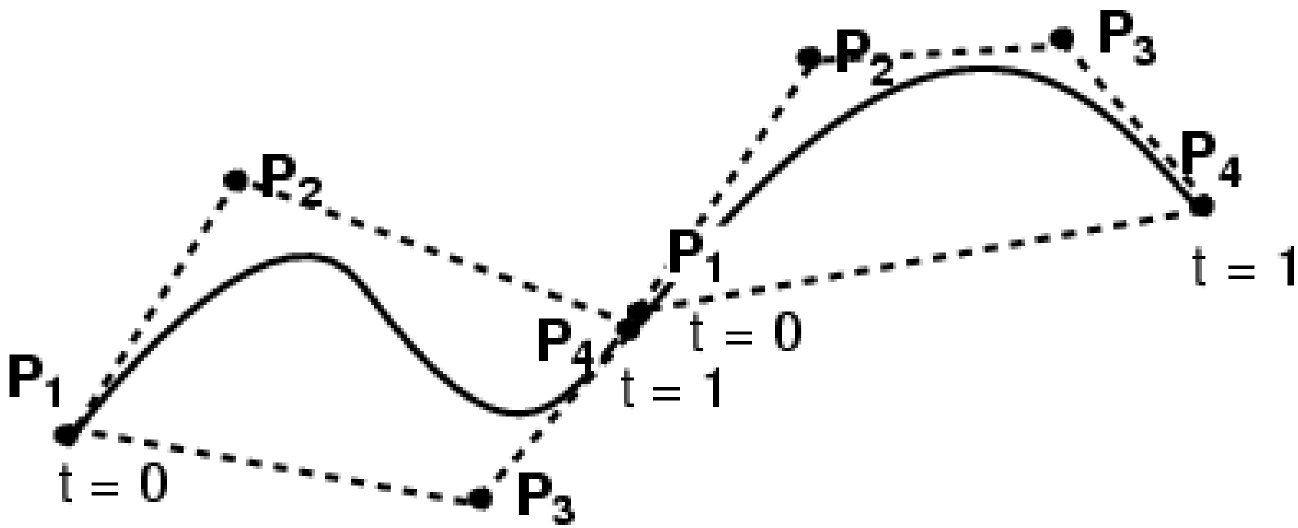


$$Q(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4$$

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

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

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

Higher-Order Bézier Curves

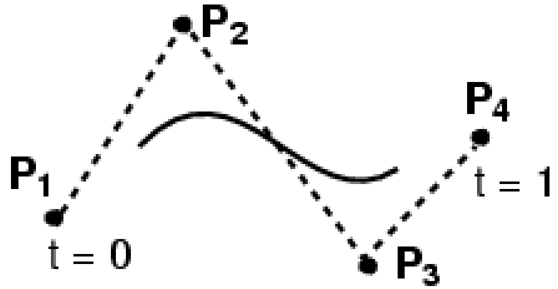
- > 4 control points
- Bernstein Polynomials as the basis functions

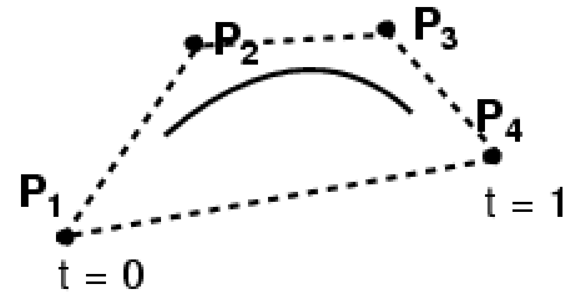
$$B_i^n(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}, \qquad 0 \le i \le n$$

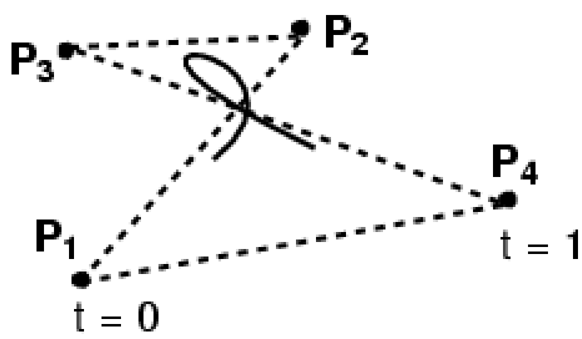
- Every control point affects the entire curve
  - Not simply a local effect
  - More difficult to control for modeling

Cubic BSplines

- $\geq$ 4 control points
- Locally cubic
- Curve is not constrained to pass through any control points

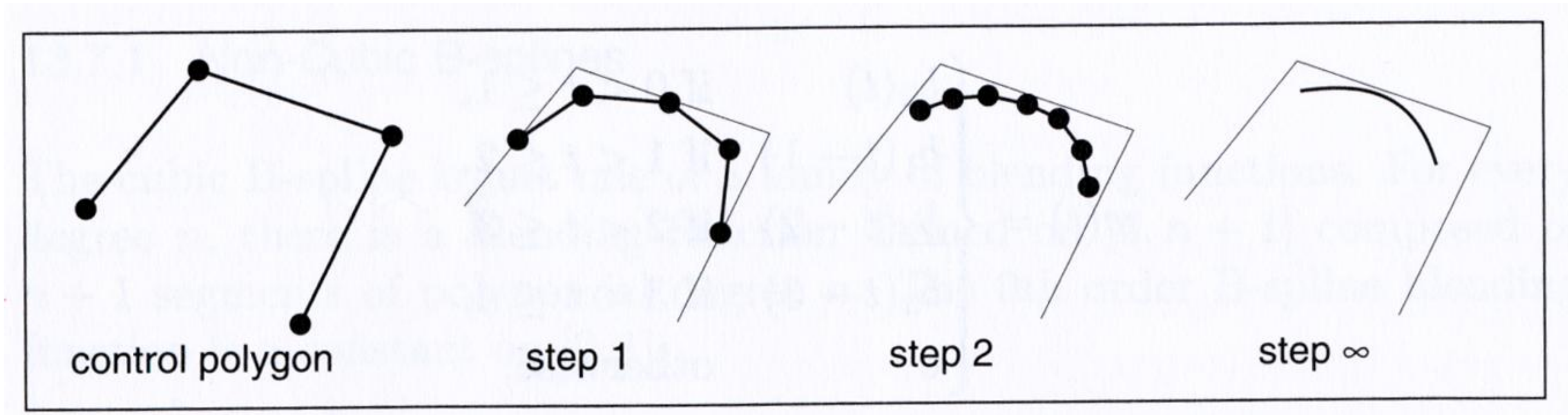


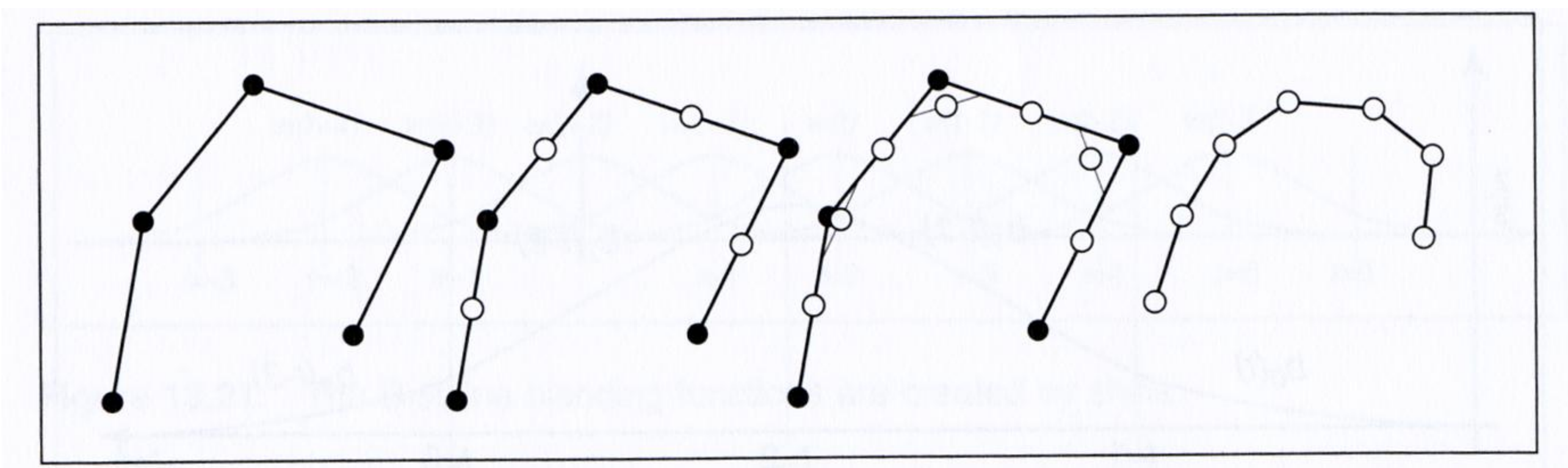




Cubic BSplines

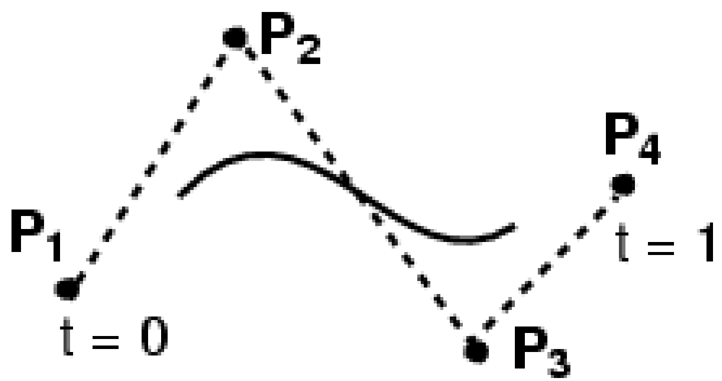
- Iterative method for constructing BSplines

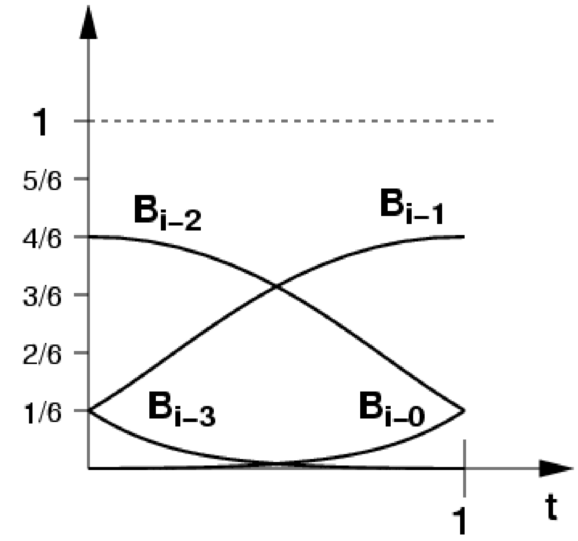




Shirley, Fundamentals of Computer Graphics

Cubic BSplines



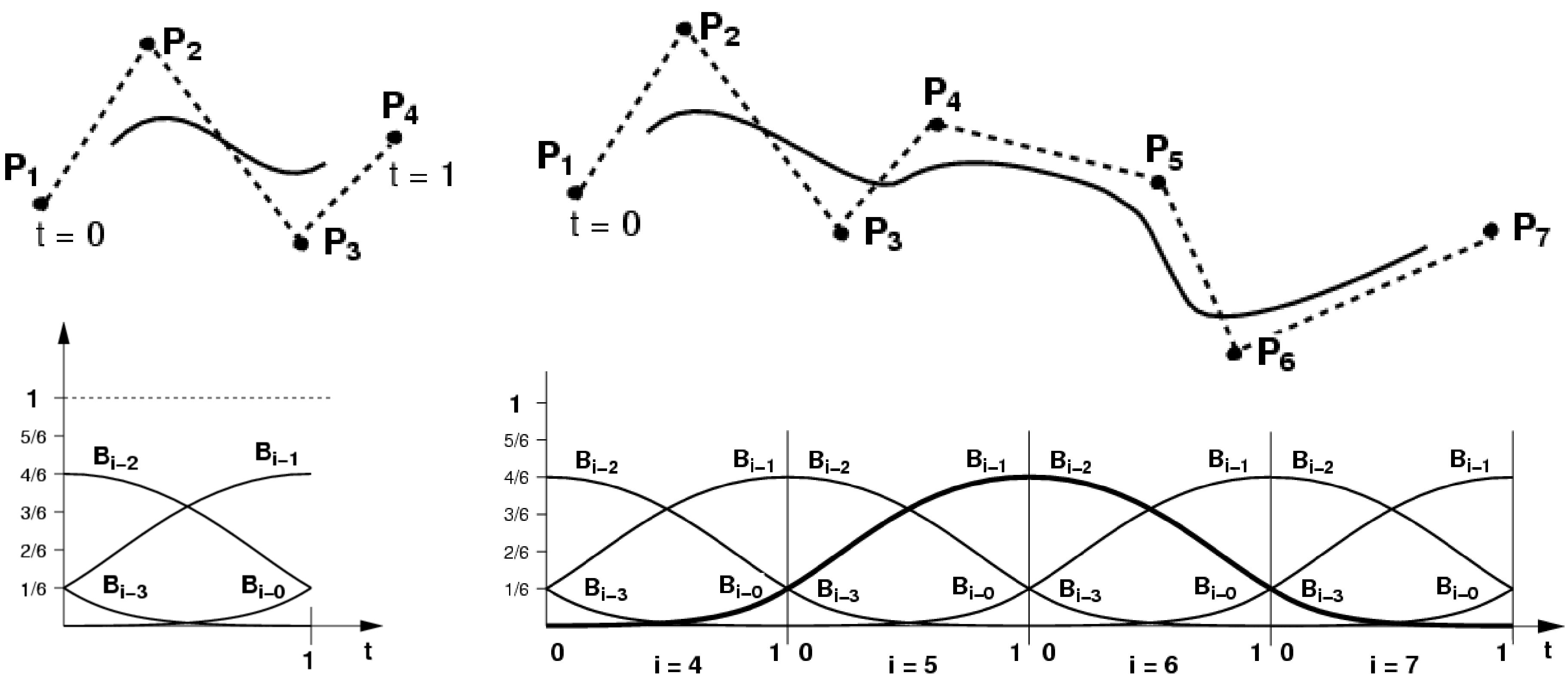


$$Q(t) = \frac{(1-t)^3}{6} P_{i-3} + \frac{3t^3 - 6t^2 + 4}{6} P_{i-2} + \frac{-3t^3 + 3t^2 + 3t + 1}{6} P_{i-1} + \frac{t^3}{6} P_i$$

$$Q(t) = \mathbf{GBT(t)} \qquad B_{B-Spline} = \frac{1}{6}\begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}$$

Cubic BSplines

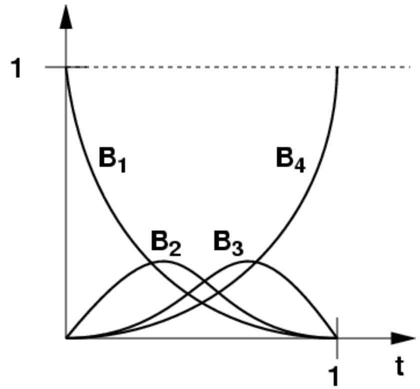
- can be chained together
- better control locally (windowing)

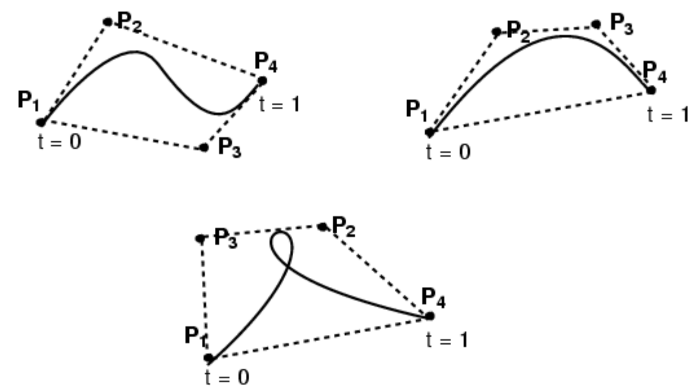


Bézier is not the same as BSpline

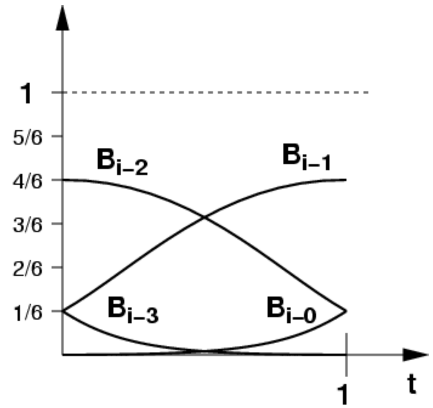
- Relationship to the control points is different

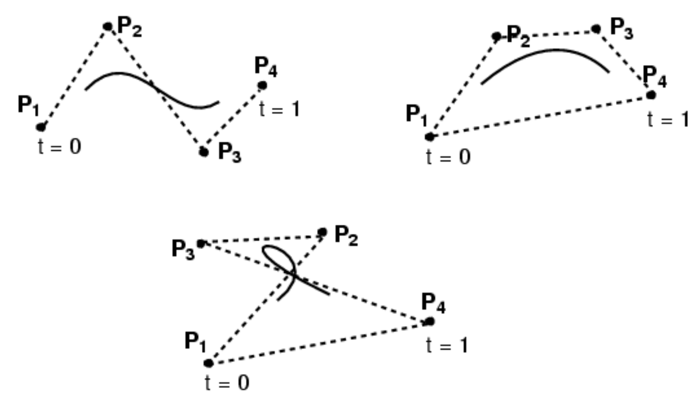
Bézier





BSpline





Bezier is not the same as Bspline

- But we can convert between the curves using the basis functions:

$$B_{Bezier} = \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

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

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

NURBS (generalized BSplines)

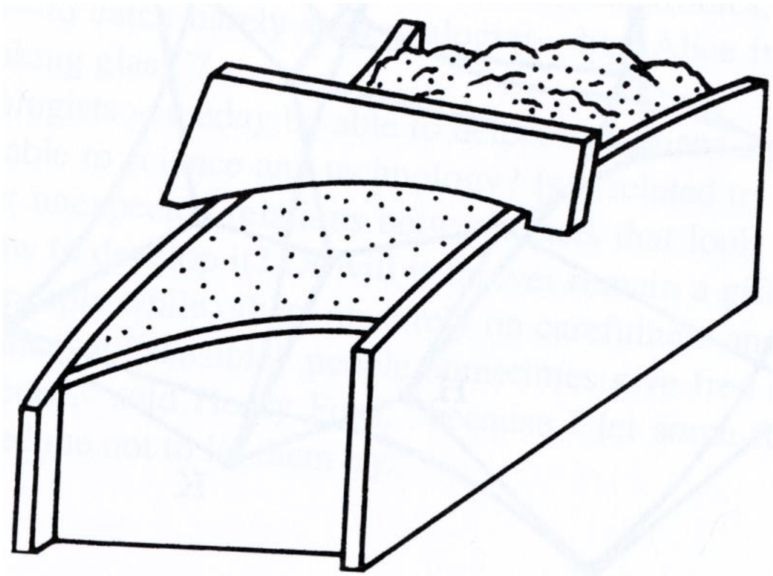
- BSpline: uniform cubic BSpline
- NURBS: Non-Uniform Rational BSpline
  - non-uniform = different spacing between the blending functions, a.k.a. knots
  - rational = ratio of polynomials (instead of cubic)

Tensor Product

- Of two vectors:

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \otimes \begin{bmatrix} b_1 & b_2 & b_3 & b_4 \end{bmatrix} = \begin{bmatrix} a_1 b_1 & a_2 b_1 & a_3 b_1 \\ a_1 b_2 & a_2 b_2 & a_3 b_2 \\ a_1 b_3 & a_2 b_3 & a_3 b_3 \\ a_1 b_4 & a_2 b_4 & a_3 b_4 \end{bmatrix}$$

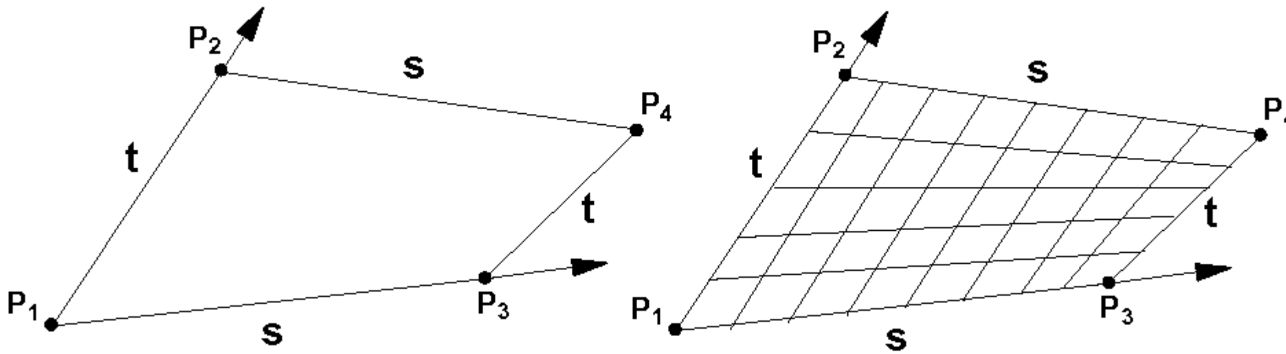
- Similarly, we can define a surface as the tensor product of two curves....



Farin, Curves and Surfaces for Computer Aided Geometric Design

Bilinear Patch

Bi-lerp a (typically non-planar) quadrilateral

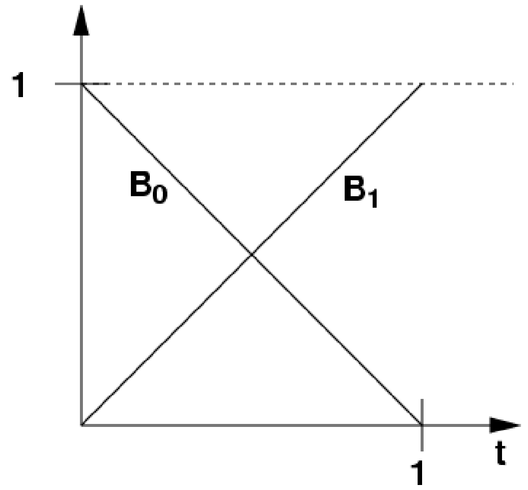


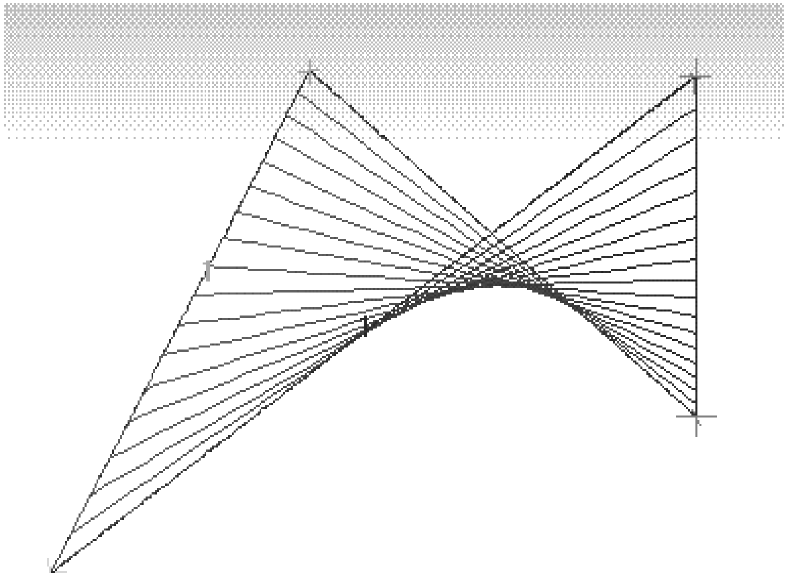
Notation: $\mathbf{L}(P_1, P_2, \alpha) \equiv (1-\alpha)P_1 + \alpha P_2$

$$Q(s,t) = \mathbf{L}(\mathbf{L}(P_1, P_2, t), L(P_3, P_4, t), s)$$

Bilinear Patch

- Smooth version of quadrilateral with non-planar vertices...





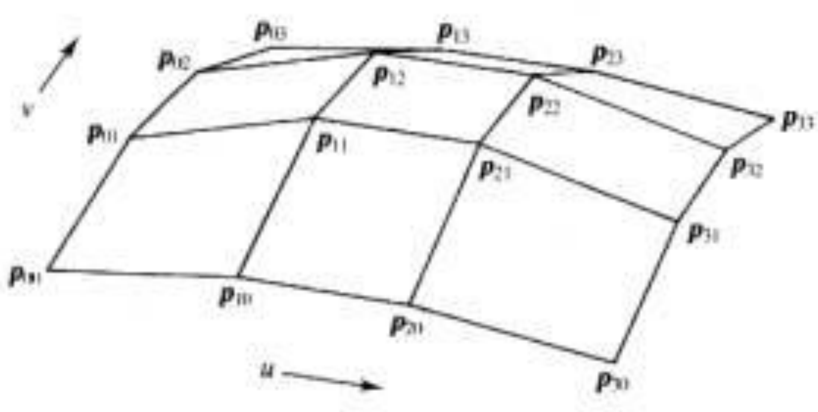
  – But will this help us model smooth surfaces?
  – Do we have control of the derivative at the edges?

Bicubic Bezier Patch

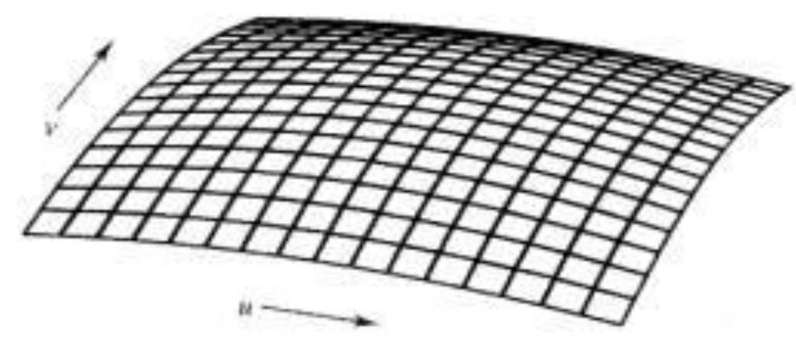
Notation: $\mathbf{CB}(P_1, P_2, P_3, P_4, \alpha)$ is Bézier curve with control points P_i evaluated at α

Define "Tensor-product" Bézier surface

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

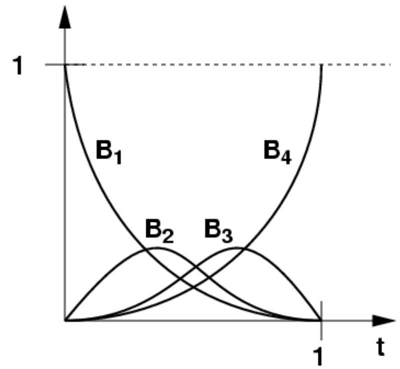


(a)

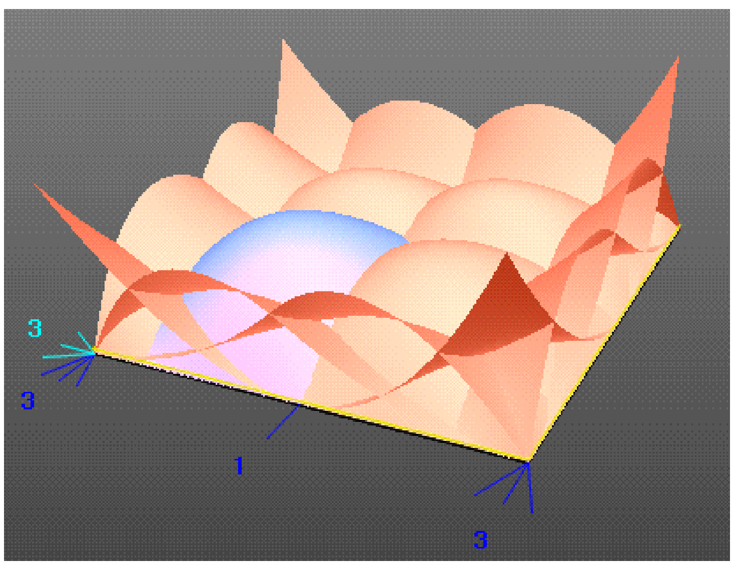


(b)

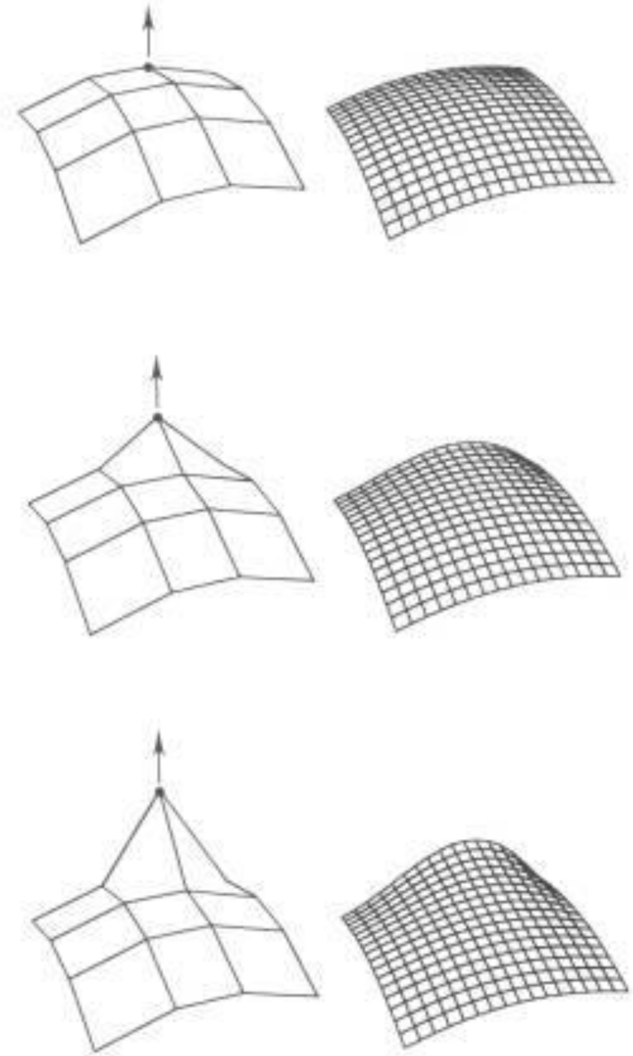
Editing Bicubic Bezier Patches



Curve Basis Functions

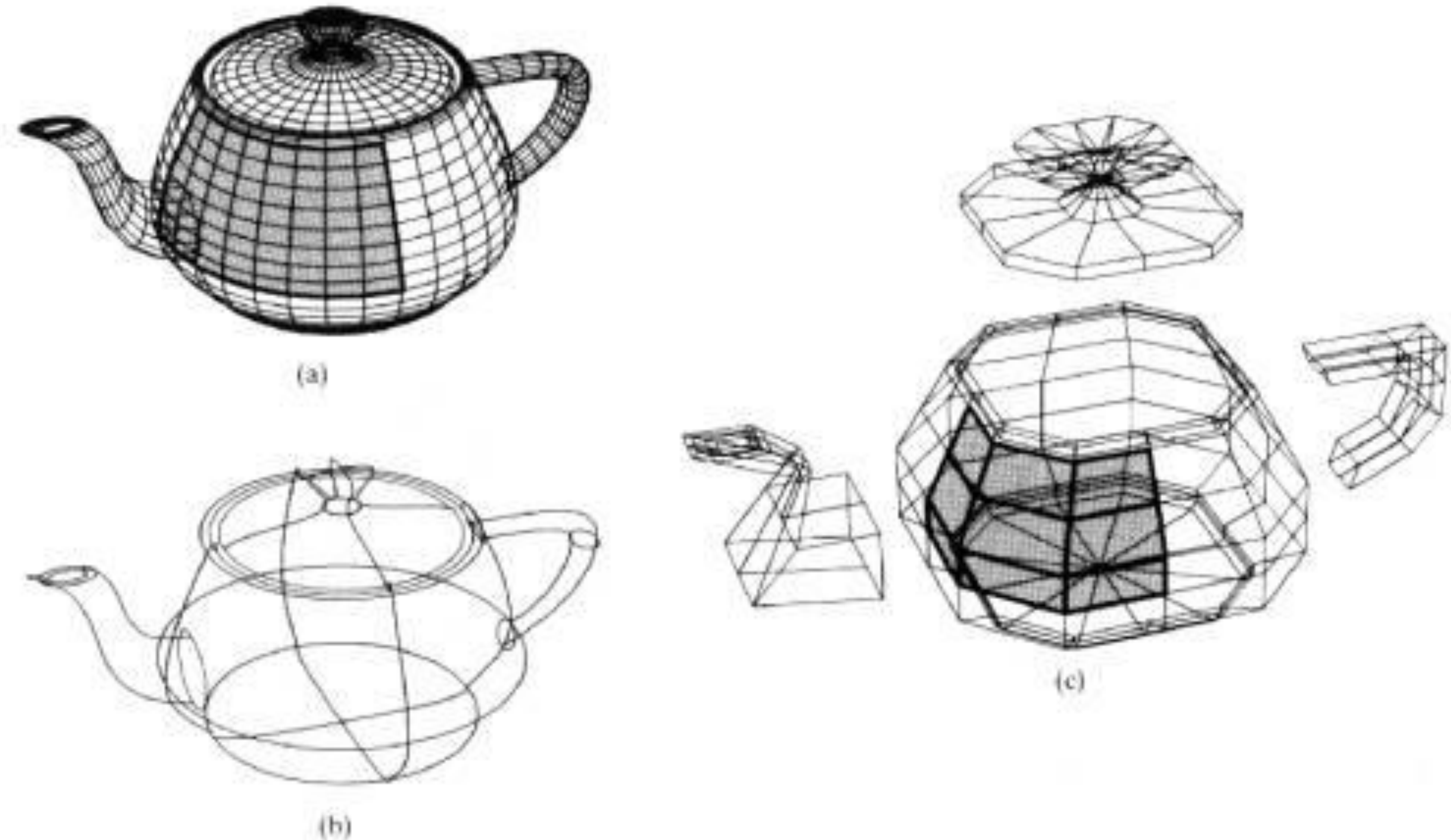


Surface Basis Functions



Modeling with Bicubic Bezier Patches

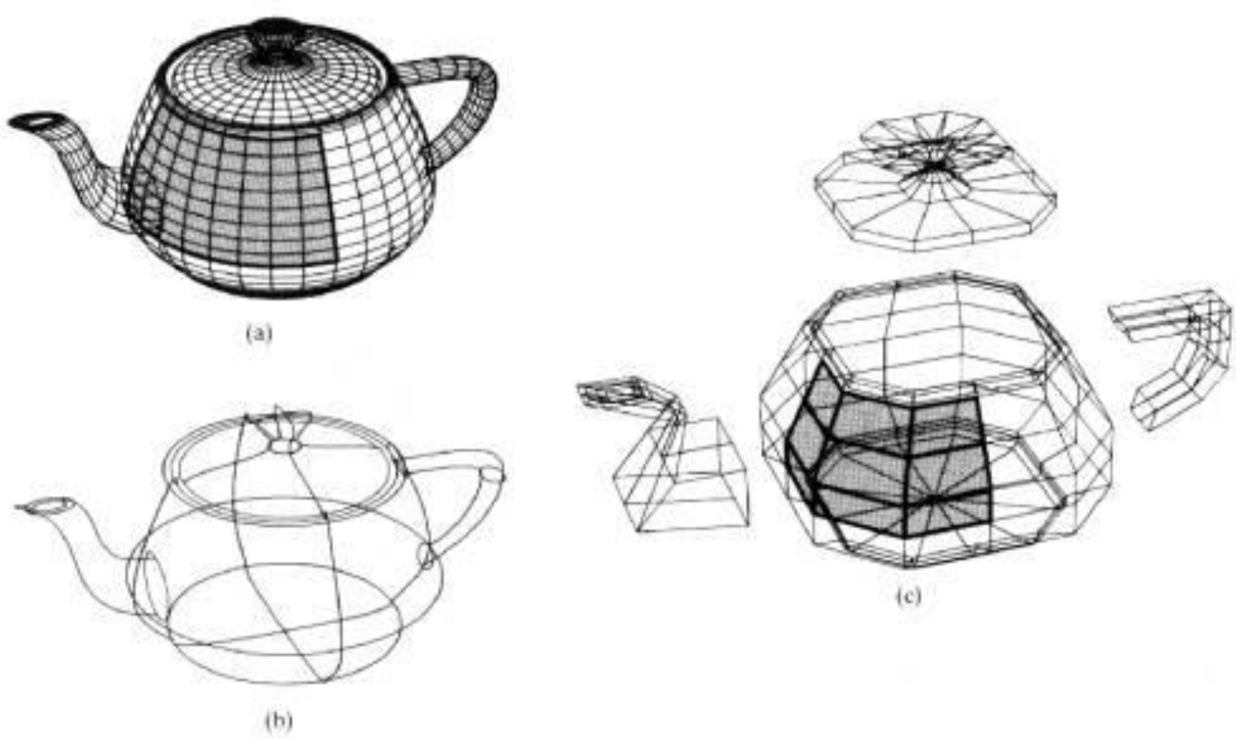
- Original Teapot specified with Bezier Patches



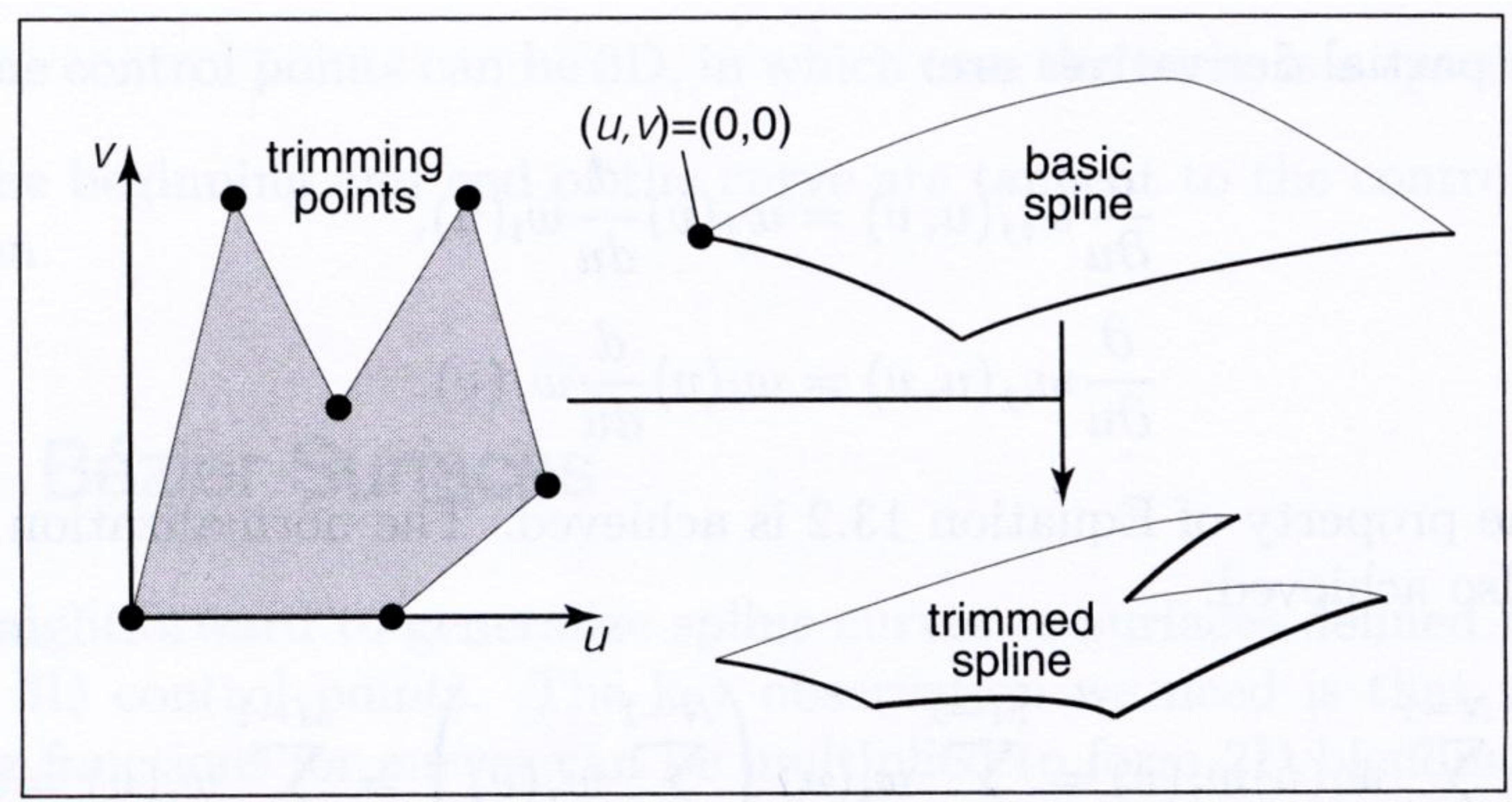
Modeling Headaches

- Original Teapot model is not "watertight":

  intersecting surfaces at spout & handle, no bottom, a hole at the spout tip, a gap between lid & base

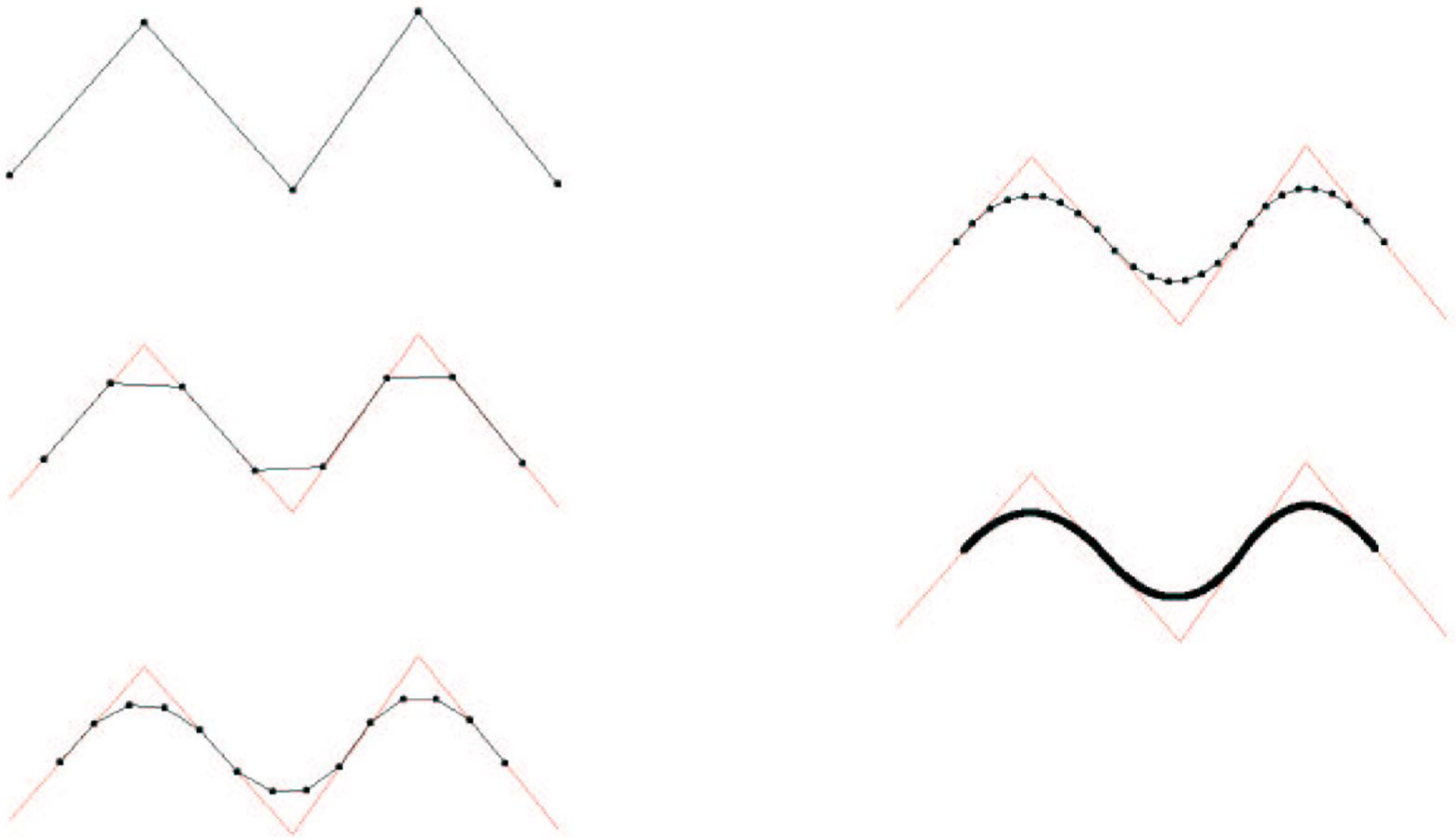


Trimming Curves for Patches



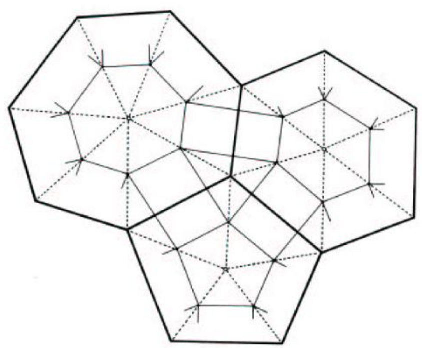
Shirley, Fundamentals of Computer Graphics

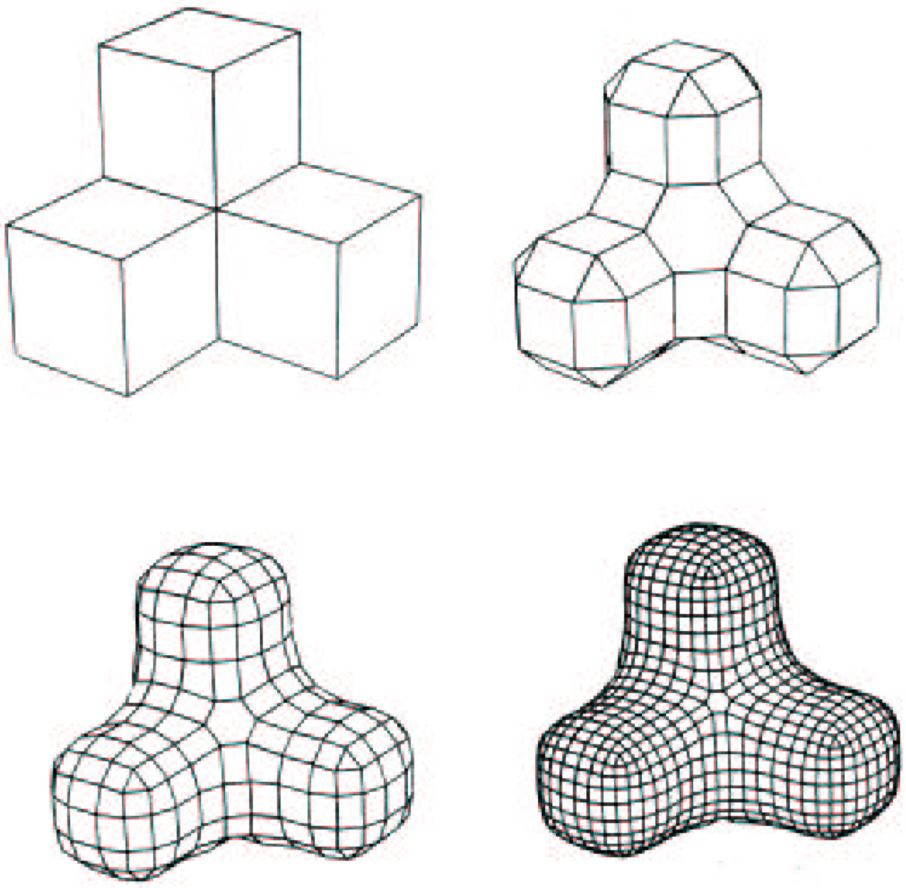
Chaikin's Algorithm



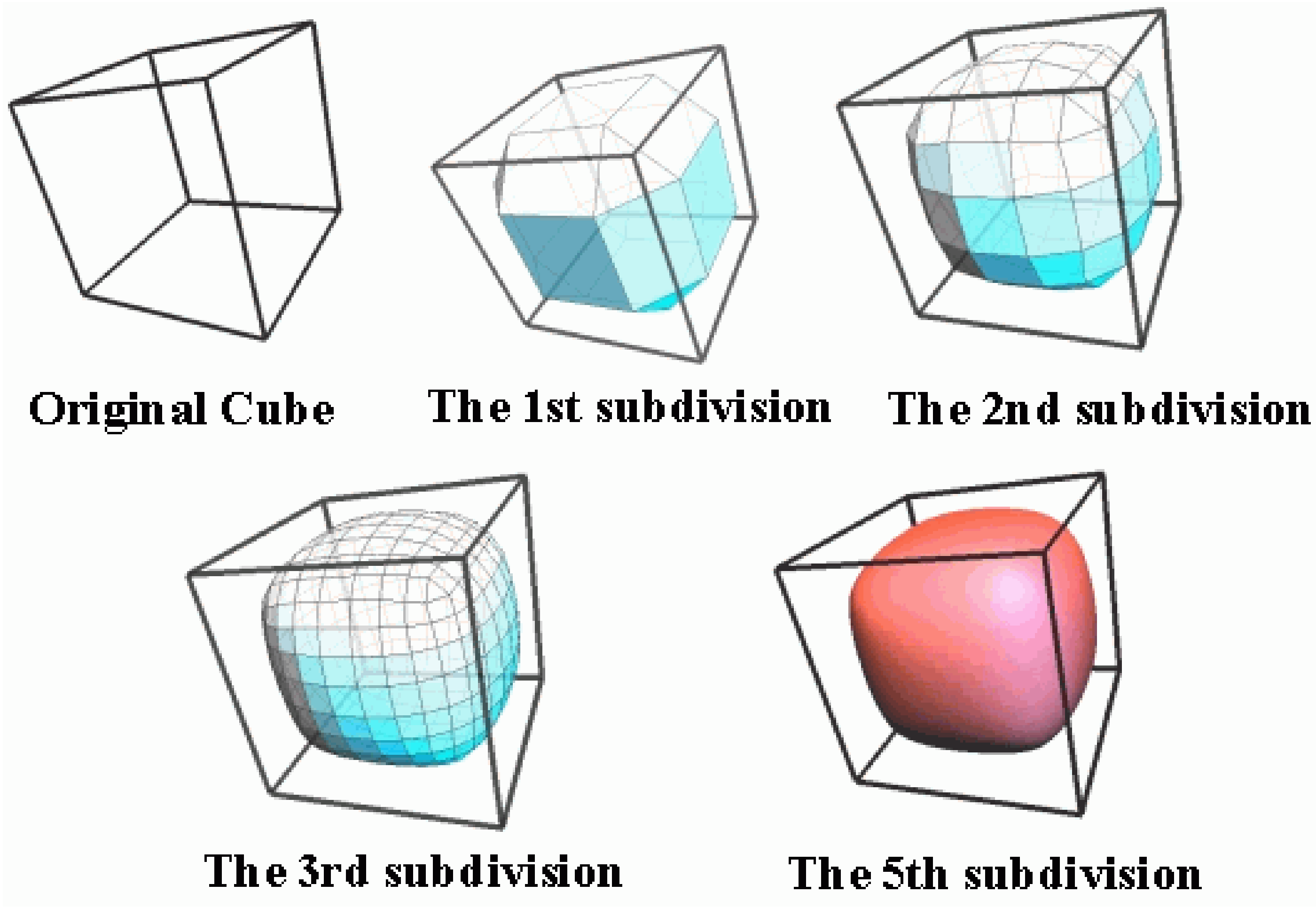
Doo-Sabin Subdivision

Idea: introduce a new vertex for each face
At the midpoint of old vertex, face centroid



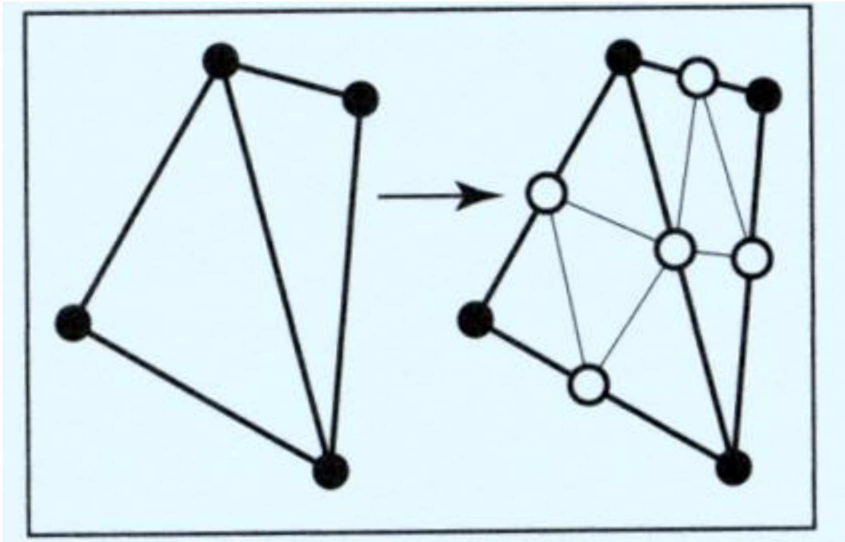


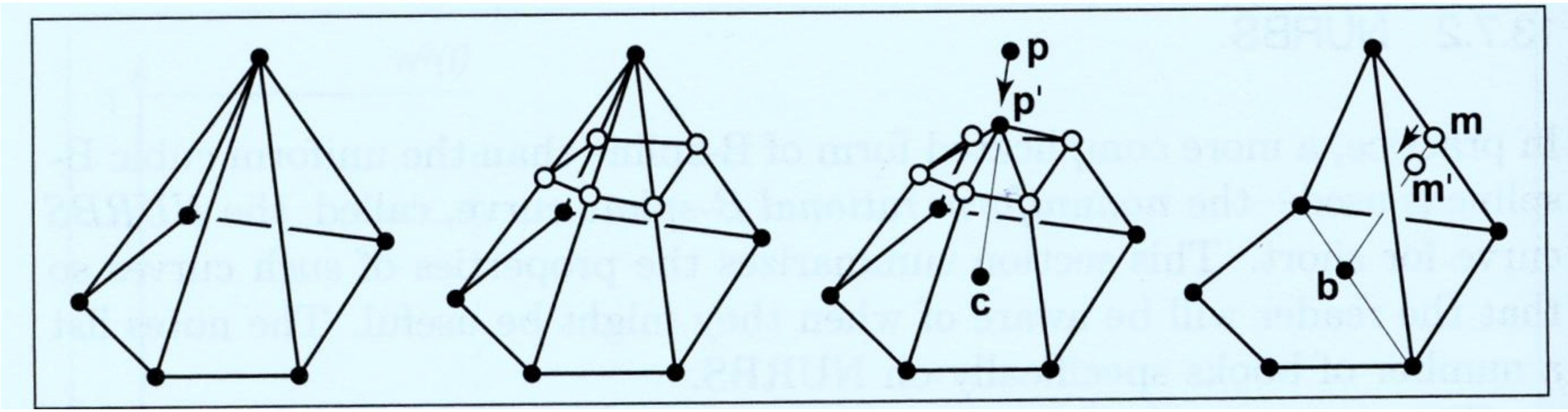
Doo-Sabin Subdivision



http://www.ke.ics.saitama-u.ac.jp/xuz/pic/doo-sabin.gif

Loop Subdivision

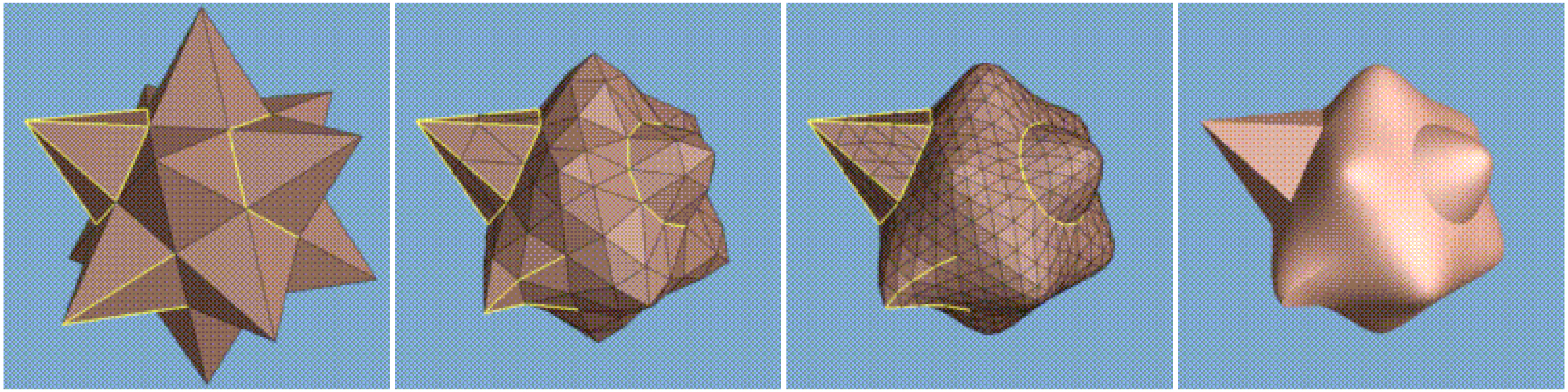




Shirley, Fundamentals of Computer Graphics

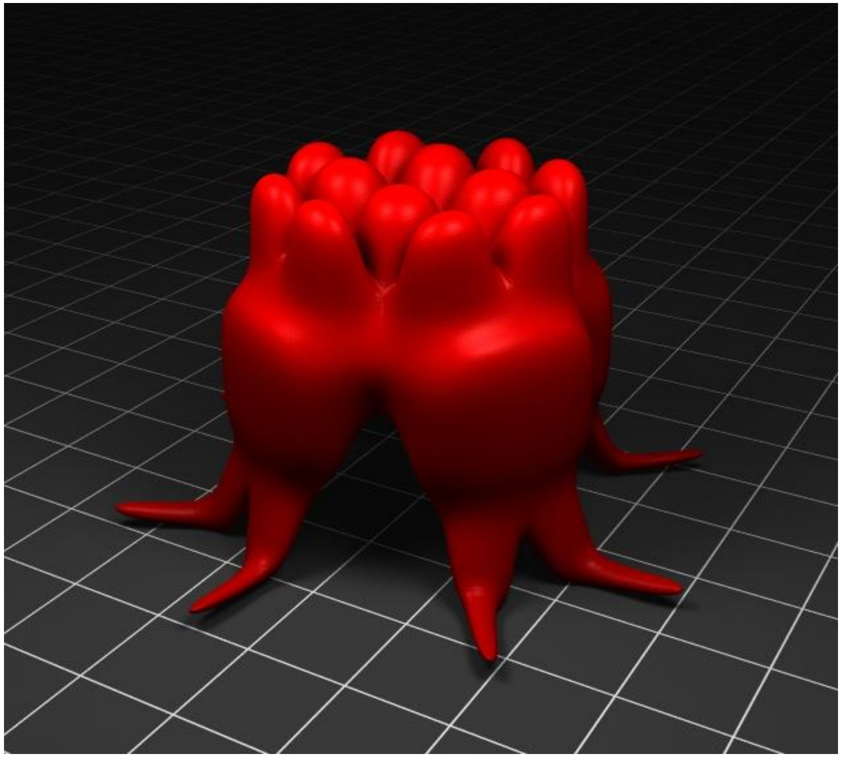
Loop Subdivision

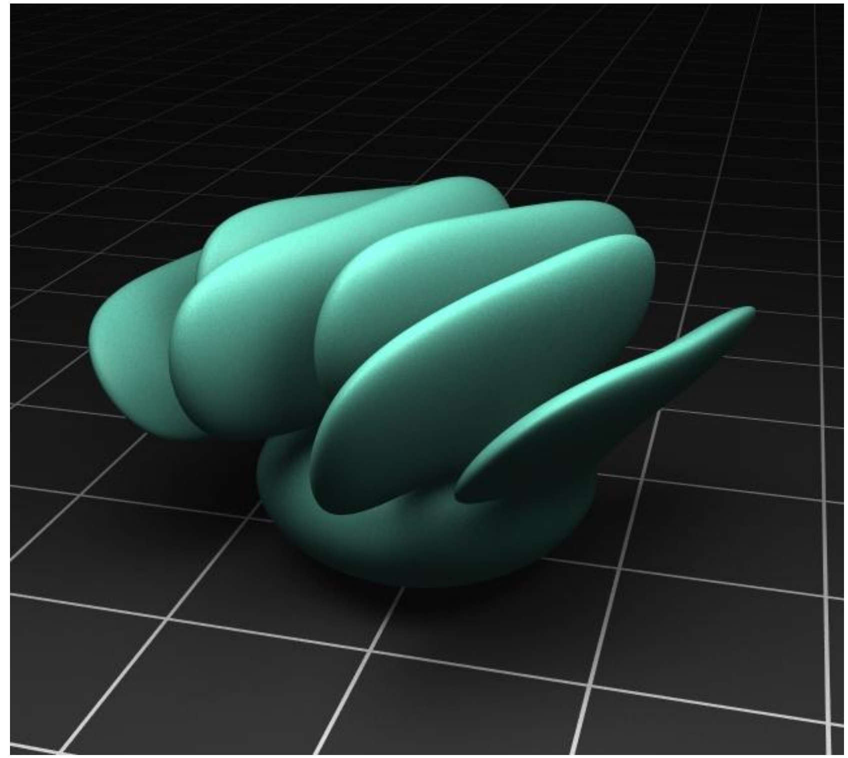
- Some edges can be specified as crease edges



http://grail.cs.washington.edu/projects/subdivision/

Weird Subdivision Surface Models





Justin Legakis

Procedural Textures

$f\ (x,y,z) \rightarrow \text{color}$

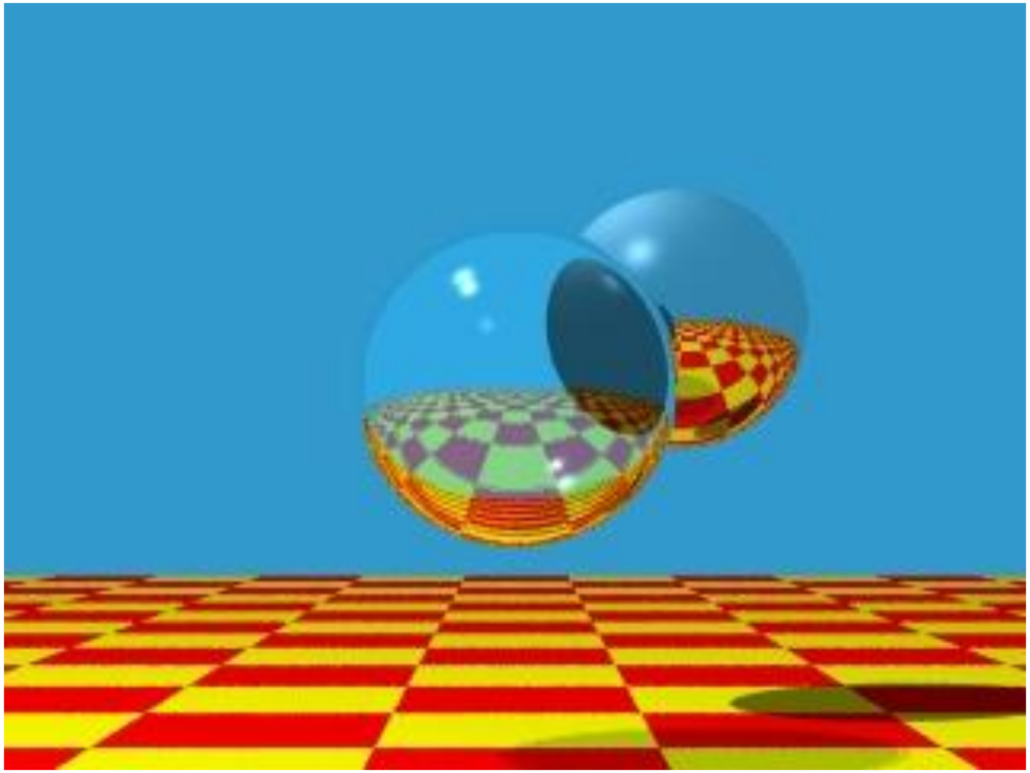
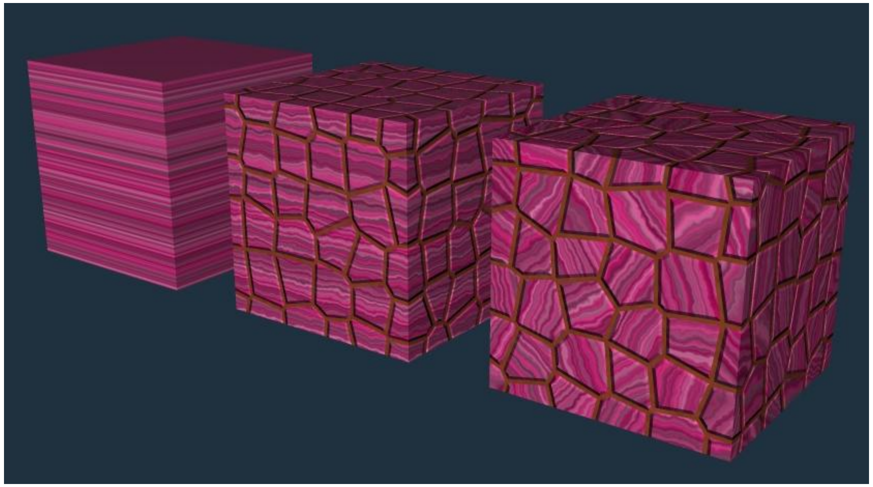


Image by Turner Whitted

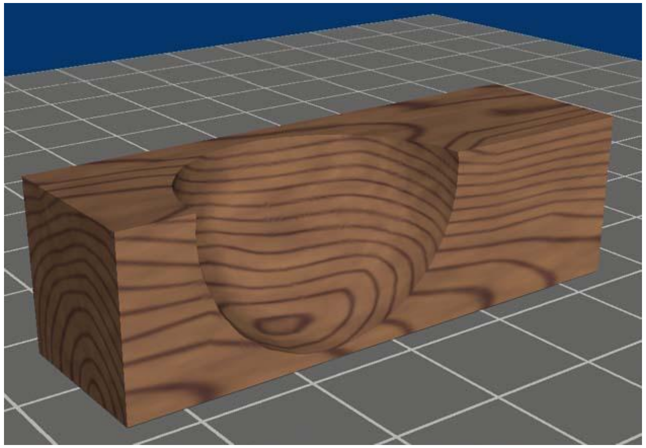
Procedural Solid Textures

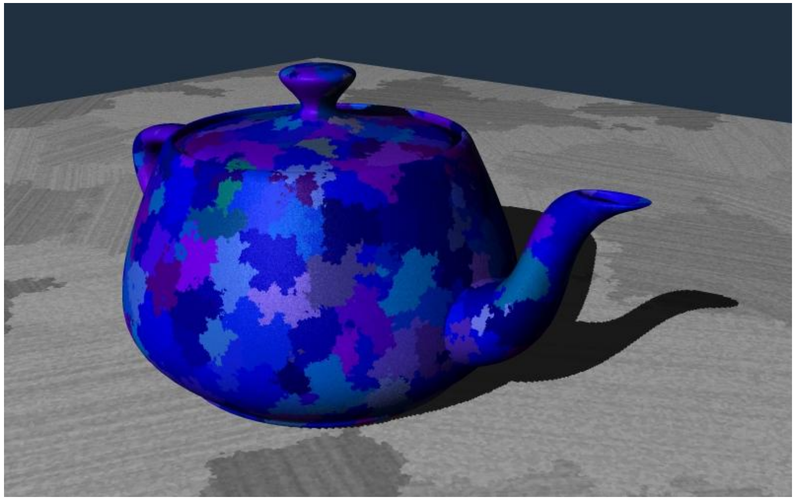
- Noise
- Turbulence

Ken Perlin



Justin Legakis





Justin Legakis

That's All