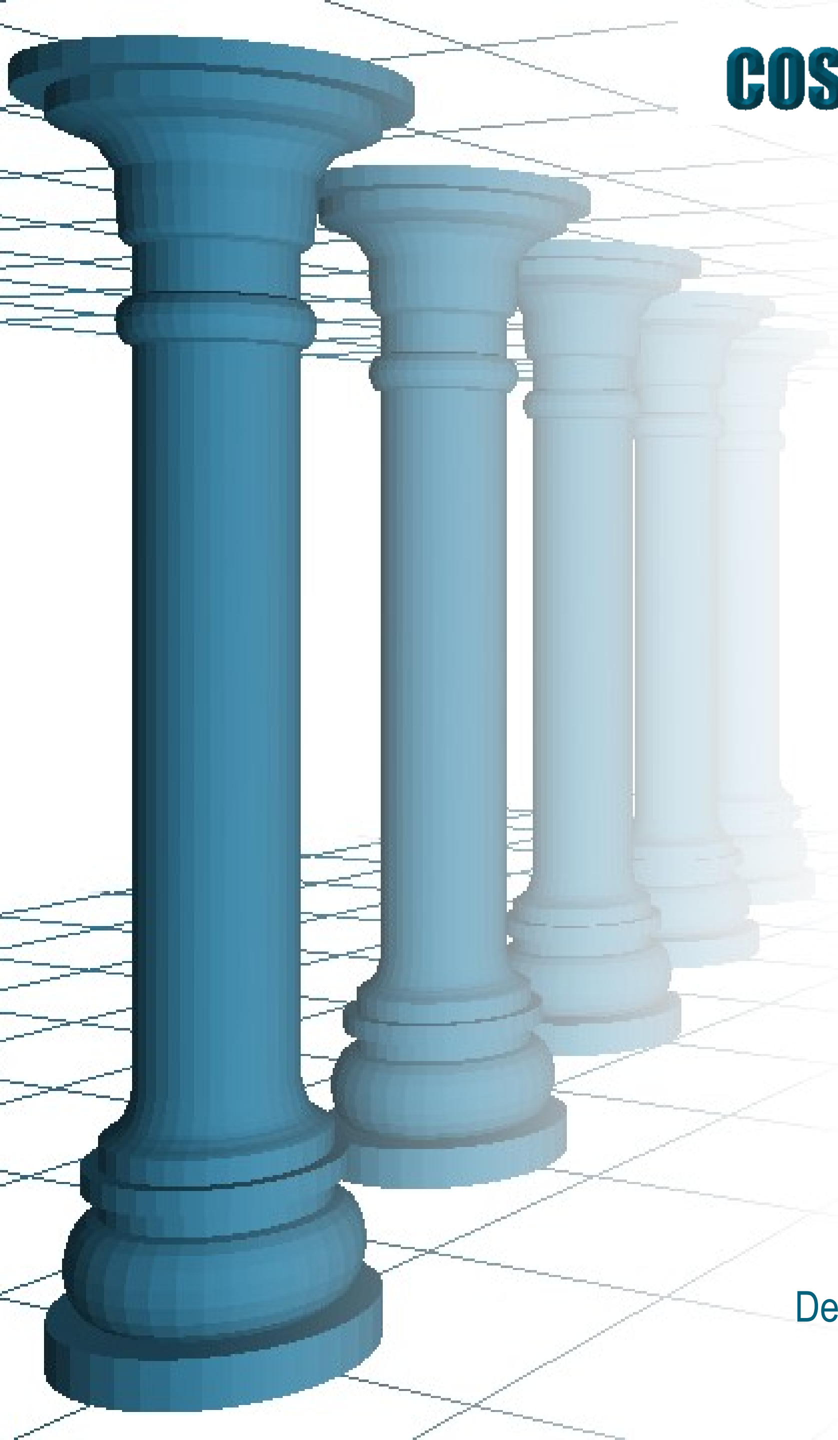


COSC363 Computer Graphics

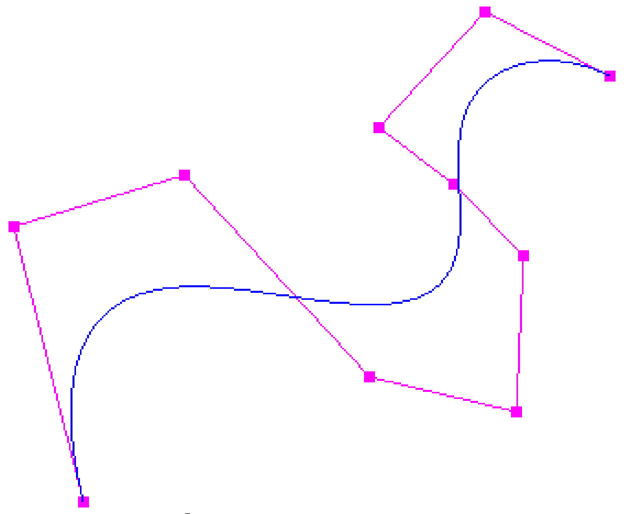
11
Bezier Surfaces

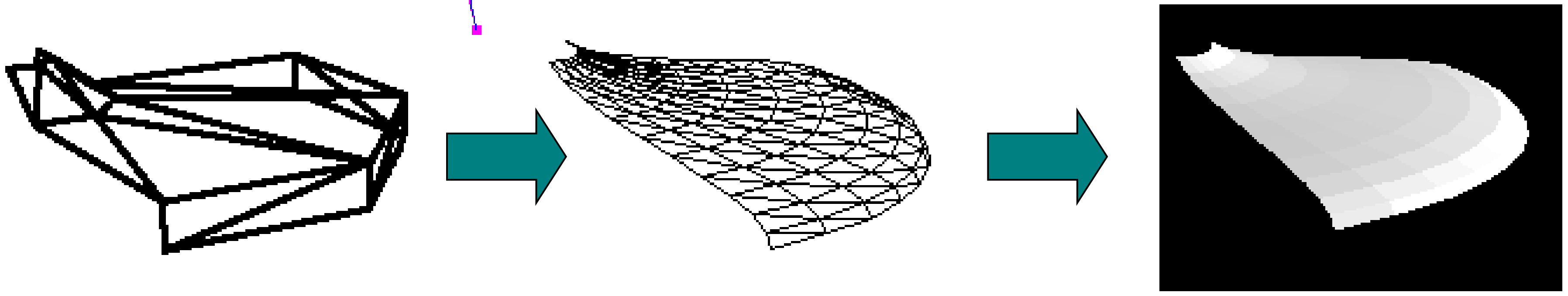
Level up your crafting skill

R. Mukundan (mukundan@canterbury.ac.nz)
Department of Computer Science and Software Engineering
University of Canterbury, New Zealand.

Motivation

We often require methods for procedurally generating complex surface geometries using only a coarse definition of their shapes specified by a small set of points (2D) or polygons (3D).





Curves and Surfaces

- Key idea:

  We can combine a set of points using parametric functions to generate a smooth and continuous curve.

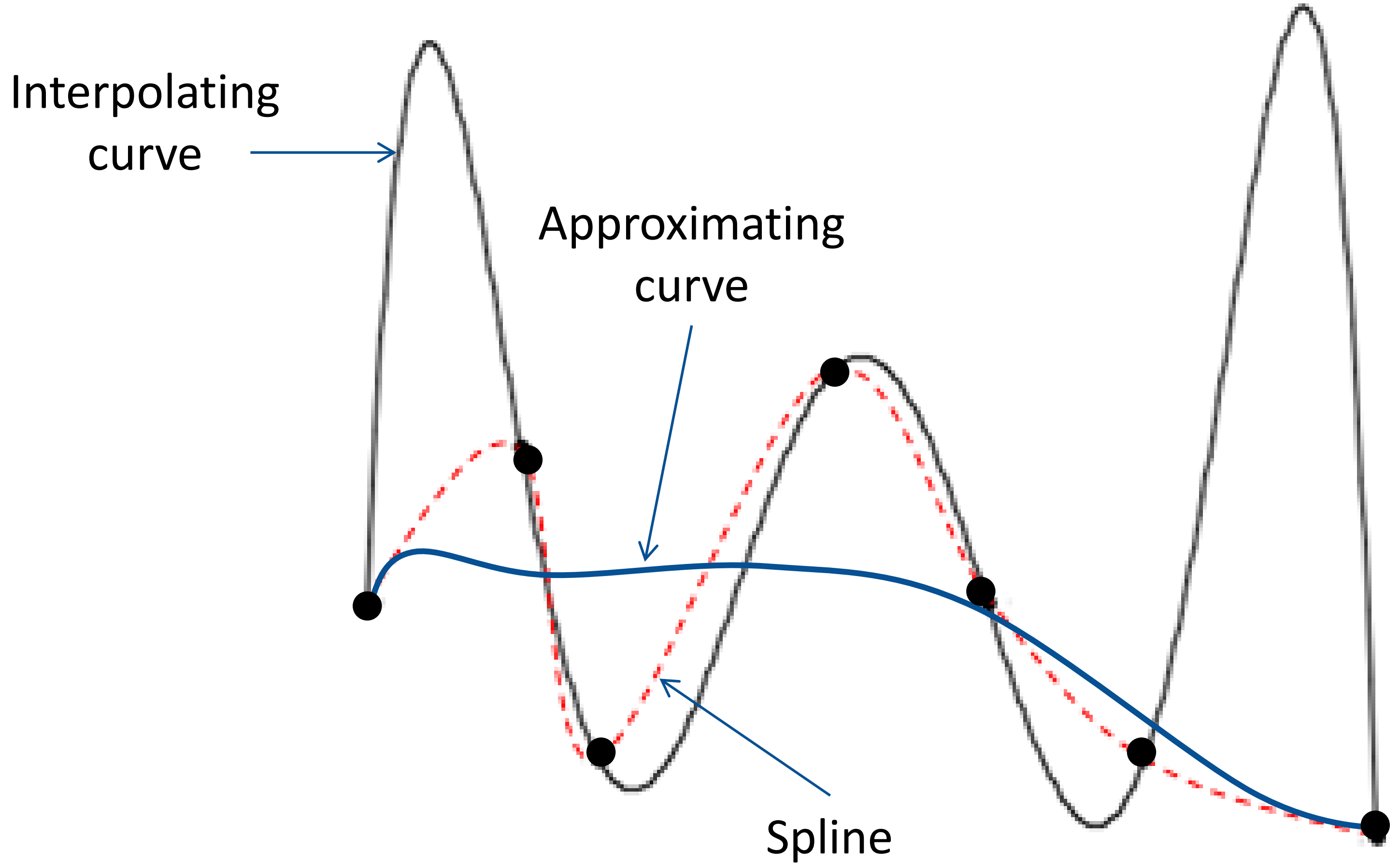
  Given a set of *n* points $P_0, P_1, \ldots P_{n-1}$, and parametric functions $f_0(t)$, $f_1(t)$, … $f_{n-1}(t)$, we can generate *any* number of points along a curve given by

  $P(t) = P_0 f_0(t) + P_1 f_1(t) + \ldots + P_{n-1} f_{n-1}(t)$
- The points P_i are called "control points" and the functions "blending functions".
- Main requirement: The generated curve should be continuous and closely following the shape defined by the control points.

Curves and Surfaces

- There are three types of modelling curves:
  - Interpolating curve: A single polynomial curve that passes through *every* control point.
  - Approximating curve: A polynomial curve that passes through only a few of the control points.
  - Splines: Piecewise polynomial curves that have a specified degree of smoothness at the connecting points (knots)
- Applications:
  - Surface design (Bezier Surfaces, B-Splines, NURBS)
  - Keyframe animation, Interactive drawing (Catmull-Rom splines)
  - Vertex blending (Hermite polynomials)

Curves and Surfaces

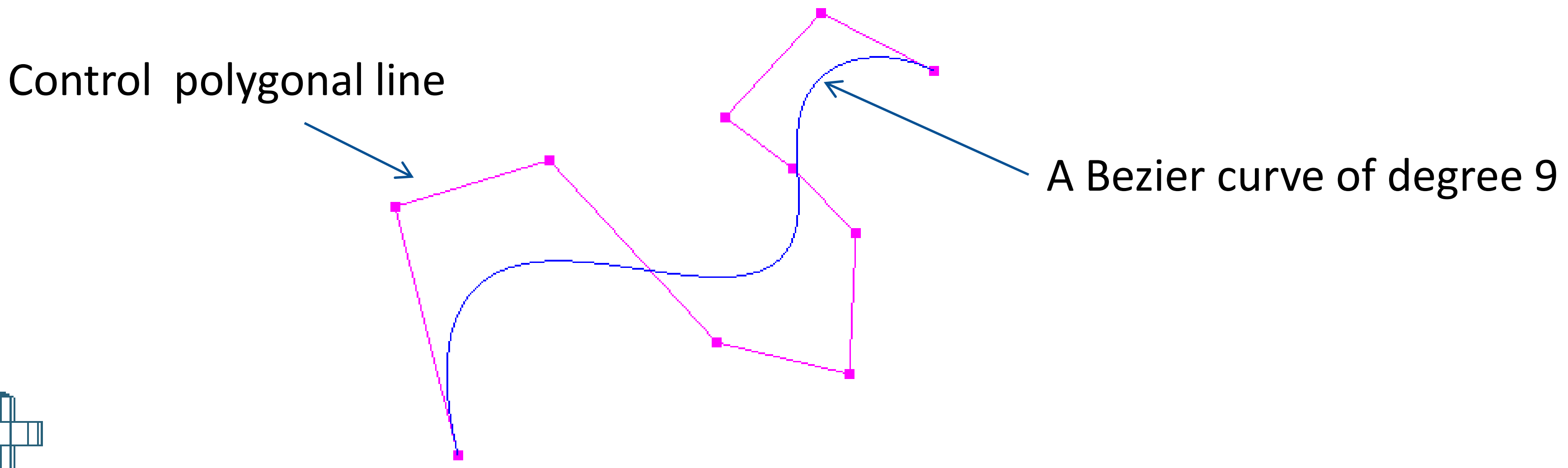


Interpolating Curves

- Polynomial interpolation theorem: Given n points (x_i, y_i), i = 1..n on the xy-plane, where all x_i's are distinct, there exists a unique polynomial $f(x)$ of degree $n-1$ such that $f(x_i) = y_i$.
- Interpolation curves are generated using a combination of Lagrange polynomials of degree $n-1$.
- Polynomial interpolation curves are not very useful as they tend to have large overshoots and undesirable oscillations. They cannot be used for representing the shape defined by the control points.
- They also become numerically unstable when n becomes large.

Approximating Curves

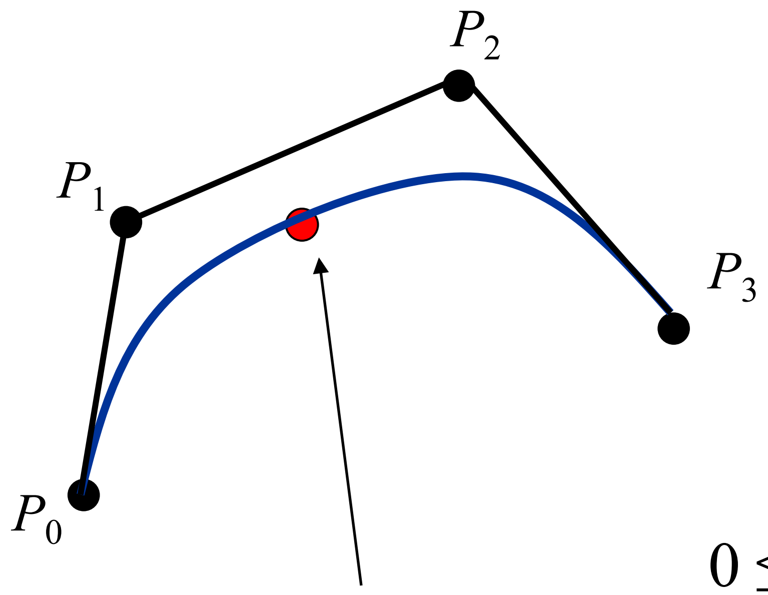
- Hermite curves and Bezier curves are examples of approximating curves.
- Bezier curves are defined using **Bernstein polynomials** as blending functions.
- Bezier curves are guaranteed to pass through only the first and the last control points.



Cubic Bezier Curve

- A cubic Bezier curve is defined using 4 control points.
- The blending functions are Bernstein polynomials of degree 3:

$$f_0^{(3)}(t) = (1-t)^3$$
$$f_1^{(3)}(t) = 3(1-t)^2 t$$
$$f_2^{(3)}(t) = 3(1-t)t^2$$
$$f_3^{(3)}(t) = t^3$$



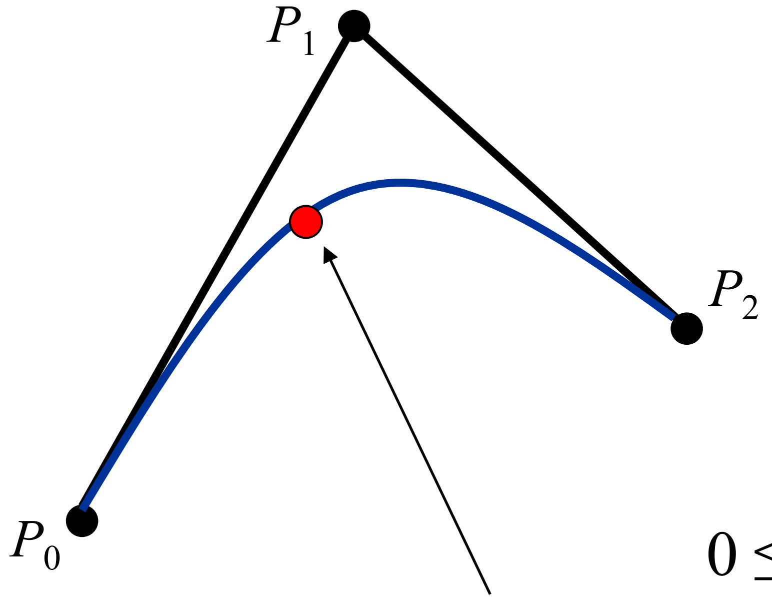
$0 \le t \le 1.$

$$\begin{aligned} x &= (1-t)^3 x_0 + 3(1-t)^2 t\, x_1 + 3(1-t)t^2 x_2 + t^3 x_3 \\ y &= (1-t)^3 y_0 + 3(1-t)^2 t\, y_1 + 3(1-t)t^2 y_2 + t^3 y_3 \\ z &= (1-t)^3 z_0 + 3(1-t)^2 t\, z_1 + 3(1-t)t^2 z_2 + t^3 z_3 \end{aligned}$$

Quadratic Bezier Curve

- A quadratic Bezier curve is defined using 3 control points.
- The blending functions are Bernstein polynomials of degree 2:

$$f_0^{(2)}(t) = (1-t)^2$$
$$f_1^{(2)}(t) = 2(1-t)t$$
$$f_2^{(2)}(t) = t^2$$



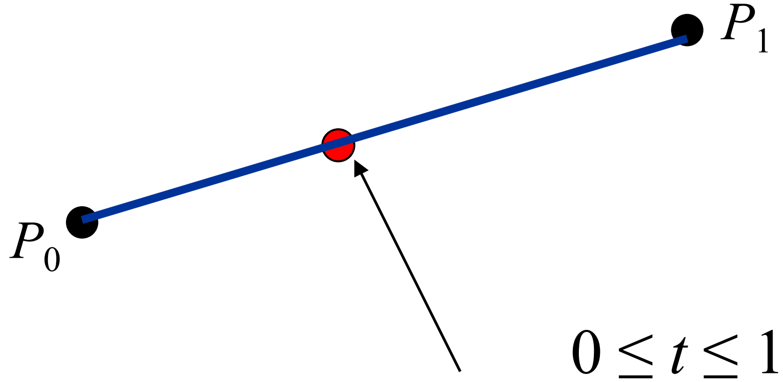
$$0 \le t \le 1.$$

$$\boxed{\begin{aligned} x &= (1-t)^2\, x_0 + 2(1-t)t\, x_1 + t^2 x_2 \\ y &= (1-t)^2\, y_0 + 2(1-t)t\, y_1 + t^2 y_2 \\ z &= (1-t)^2\, z_0 + 2(1-t)t\, z_1 + t^2 z_2 \end{aligned}}$$

Linear Bezier Curve

- A linear Bezier curve is just a straight line defined using 2 control points.
- The blending functions are Bernstein polynomials of degree 1:

$$f_0^{(1)}(t) = (1-t)$$
$$f_1^{(1)}(t) = t$$



$$0 \le t \le 1.$$

$$\boxed{\begin{aligned} x &= (1-t)\,x_0 + t\,x_1 \\ y &= (1-t)\,y_0 + t\,y_1 \\ z &= (1-t)\,z_0 + t\,z_1 \end{aligned}}$$

Nth degree Bezier Curve

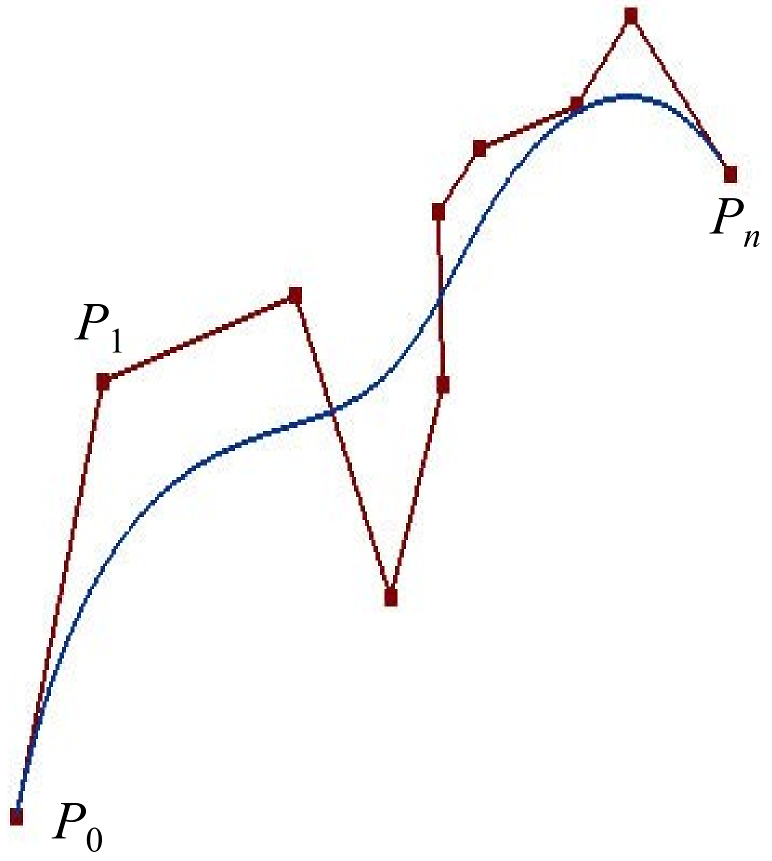
- An *n*th degree Bezier curve is defined using *n*+1 control points.
- The blending functions are Bernstein polynomials of degree *n*.

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

$$i = 0,\ 1,\ 2\ldots\ n.$$

$$P(t) = \sum_{i=0}^{n} f_i^{(n)}(t)P_i$$

$$0 \le t \le 1.$$



Properties of Bezier Curves

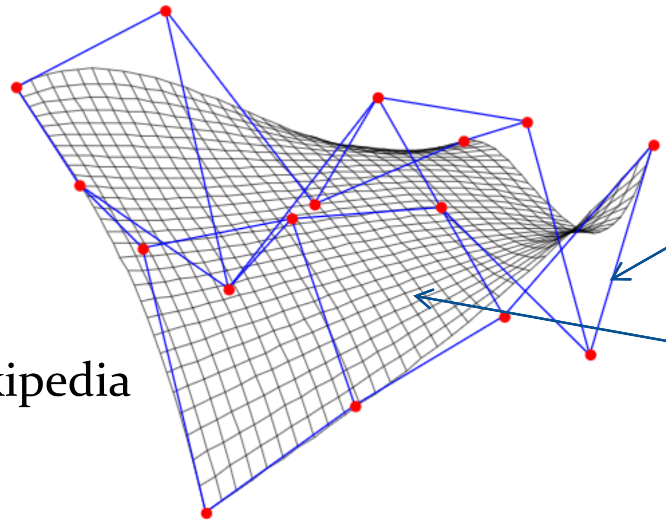
- The Bernstein polynomials form a partition of unity (sum of the polynomial values = 1), and therefore a Bezier curve is always generated using a convex combination of control points.
- The Bezier curve always lies within the convex hull of the control points.
- The Bezier curve passes through the first and the last control points, and has the corresponding edges of the control polygonal line as its tangents.
- The curve is affine invariant: An affine transformation of a Bezier curve is the same as the Bezier curve generated using the transformed control points.

From Curves to Patches

- The method for generating one-parameter Bezier curves *P*(*t*) can be extended to get two-parameter Bezier surfaces *P*(*u*, *v*) by using a grid of control points instead of a polygonal line. Control Points: $\{P_{ij},\ i = 0,1,\ldots n,\quad j = 0,1,\ldots m\}$.
- The generated surface is called a ***Bezier surface patch***.

$$p(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} f_i^{(n)}(u) f_j^{(m)}(v) P_{ij}$$

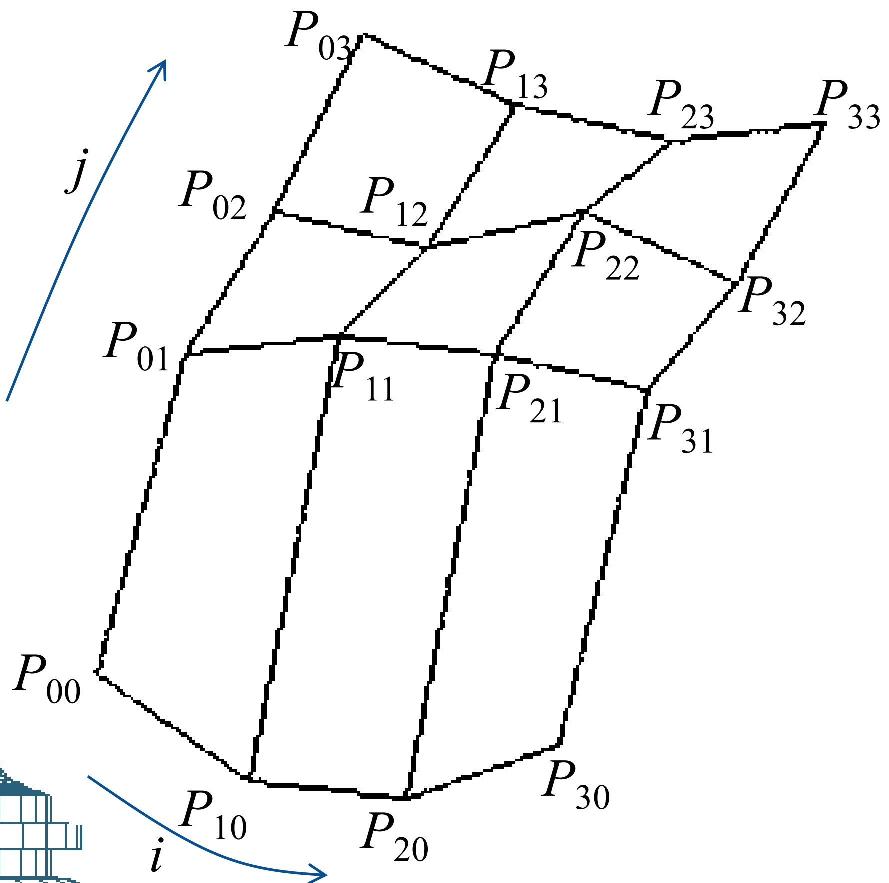
$$0 \le u \le 1$$
$$0 \le v \le 1$$



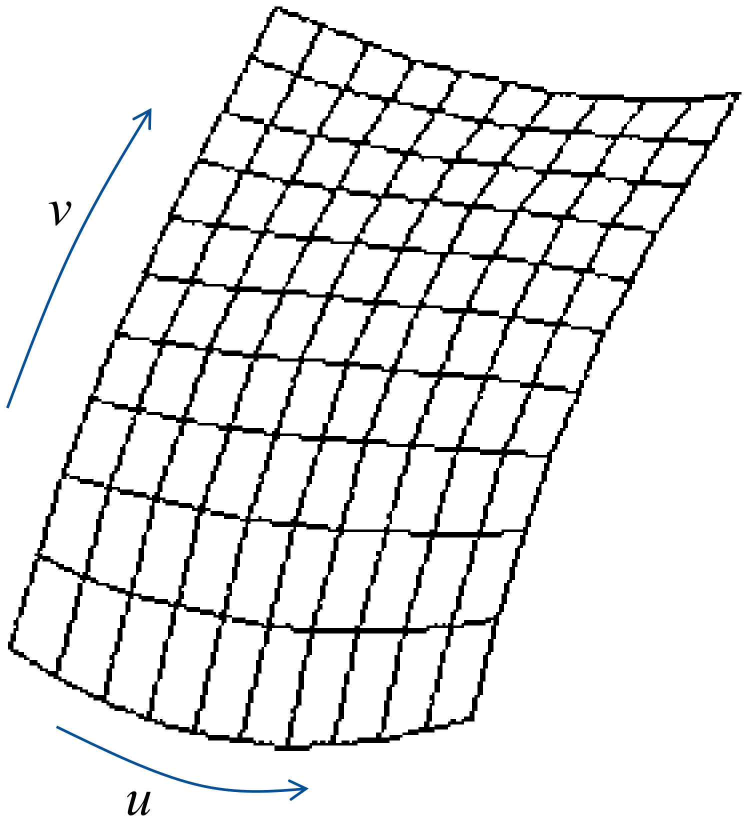
Source: Wikipedia

Bezier Surface Patch

- A 4x4 grid of control points is used to generate a bi-cubic Bezier patch.



4x4 Control patch



Bi-cubic Bezier surface patch

Bi-cubic Bezier Surface Patch

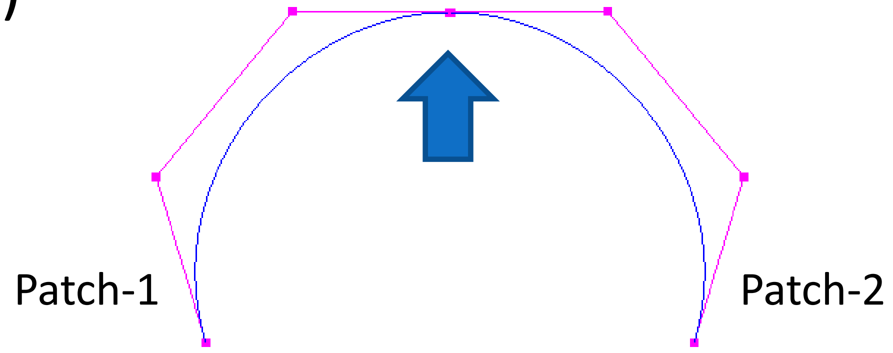
A *Bi-cubic Bezier surface patch* is generated using two sets of Bernstein polynomials of degree 3: $B_i^3(u)$ and $B_j^3(v)$.

$$p(u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} f_i^{(3)}(u)\, f_j^{(3)}(v)\, P_{ij} \qquad \begin{array}{l} 0 \le u \le 1 \\ 0 \le v \le 1 \end{array}$$

$$\begin{aligned} p(u,v) = {} & (1-u)^3 \left\{(1-v)^3 P_{00} + 3(1-v)^2 v P_{01} + 3(1-v)v^2 P_{02} + v^3 P_{03}\right\} \\ & + 3(1-u)^2 u \left\{(1-v)^3 P_{10} + 3(1-v)^2 v P_{11} + 3(1-v)v^2 P_{12} + v^3 P_{13}\right\} \\ & + 3(1-u)u^2 \left\{(1-v)^3 P_{20} + 3(1-v)^2 v P_{21} + 3(1-v)v^2 P_{22} + v^3 P_{23}\right\} \\ & + u^3 \left\{(1-v)^3 P_{30} + 3(1-v)^2 v P_{31} + 3(1-v)v^2 P_{32} + v^3 P_{33}\right\} \end{aligned}$$

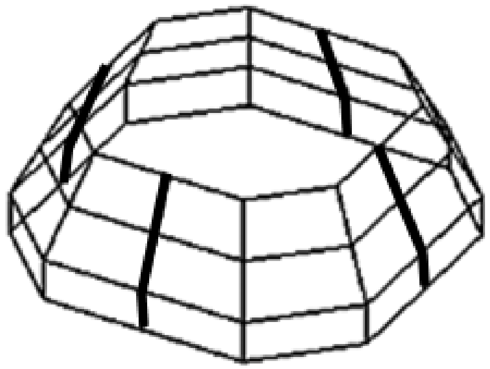
Modelling Complex Shapes

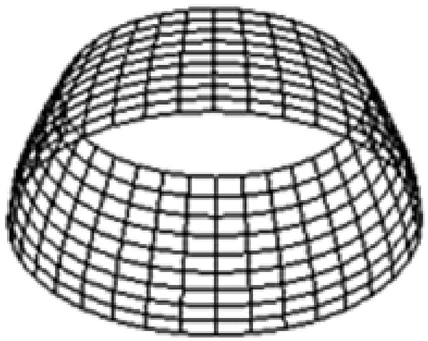
- A coarse description of a shape could be specified using several 4x4 control patches. Each control patch is then used to create a Bezier surface patch.
- We can get a smooth surface *if* the Bezier patches do not have any discontinuity along the edges where two patches meet. This can be achieved by making sure that the corresponding polygonal edges are collinear (for curves) or the corresponding polygons of the control patches are coplanar (for surfaces)



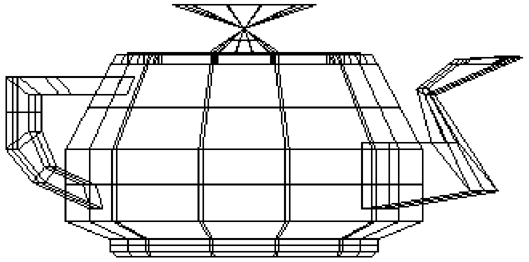
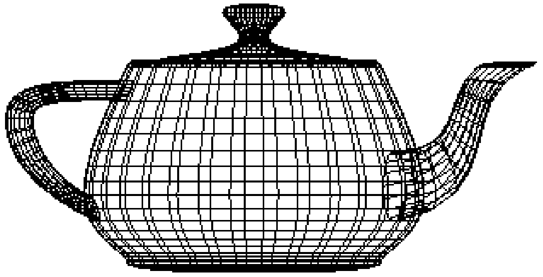
Modelling Complex Shapes

- The Utah Teapot is defined using 32 control patches each of size 4x4.
- The grids are defined such that polygons on either side of an edge where two grids meet are coplanar.
- Bi-cubic Bezier surface patches are generated for each control patch to get the shape of the teapot.


4 Control patches are joined together along edges shown as thick lines

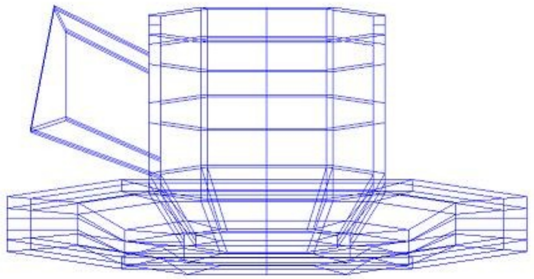
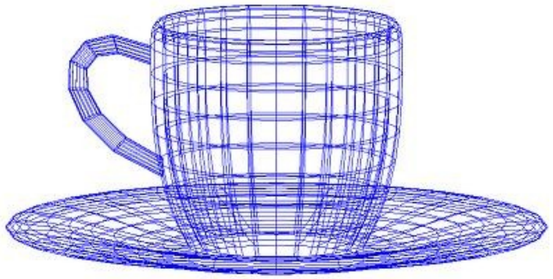

Teapot's upper body

The Teapot Family

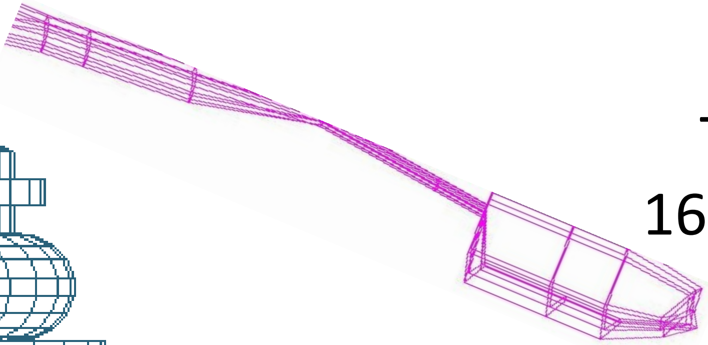
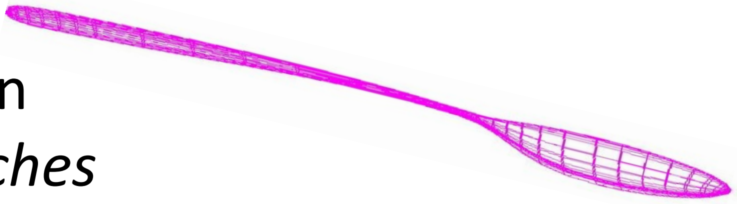
The Teapot

32 *Control patches*

The Teacup

26 *Control patches*

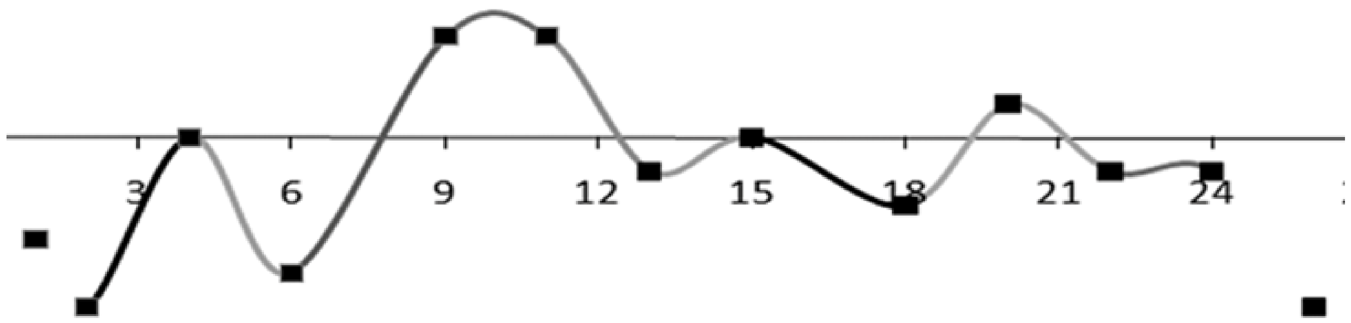



The Teaspoon

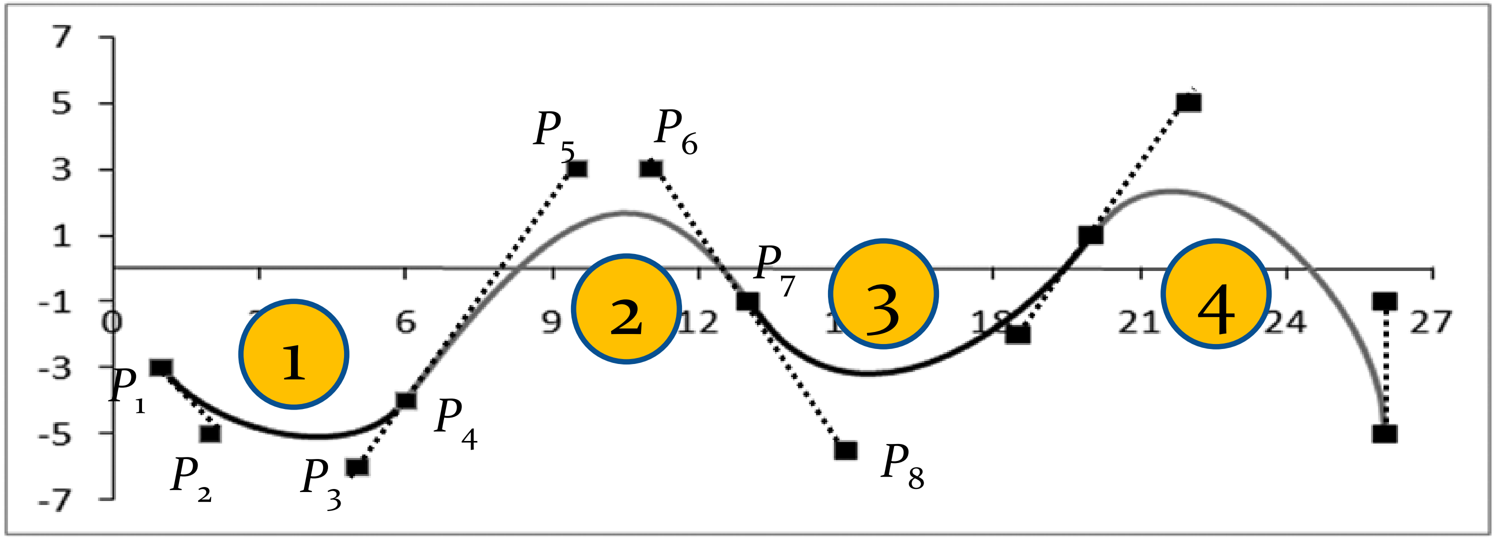
16 *Control patches*

Splines

- Bezier curves or surfaces of a high degree do not allow local control. Changing the position of one control point affects the whole curve.
- We therefore model surfaces by connecting together several Bezier patches of low degree.
- Splines are piecewise smooth curves that connect together at points called knots. At the knots, splines are required to satisfy continuity constraints.
- Splines allow fine local control of a shape.



Cubic Bezier Splines



- The above spline is made up of 4 cubic Bezier curves.
- The spline passes through the points P_1, P_4, P_7, P_{10}, P_{13}
- The remaining points control the shape, and may be used to give tangential continuity.

Cubic Parametric Splines

- At the knots, the curves must have tangential continuity.
- Cubic polynomial curves have continuous first and second order derivatives.
- The Cubic Bezier Spline on the previous slide has the following limitations:
  - The user must make sure that the point P_3, P_4, P_5 etc. are collinear.
  - The parameter *t* does not vary continuously over the whole curve.
  - It is not possible to interactively reshape the curve by moving the knots alone.