4: Parametric curves and surfaces

• Lecture 1: Euclidean, similarity, affine and projective transformations. Homogeneous coordinates and matrices. Coordinate frames. Perspective projection and its matrix representation.

• Lecture 2: Vanishing points. Horizons. Applications of projective transformations.

• Lecture 3: Convexity of point-sets, convex hull and algorithms. Conics and quadrics, implicit and parametric forms, computation of intersections.

• **Lecture 4**: **Interpolating and Approximating Splines: Cubic splines, Bezier curves, B-splines**

Introduction 4.1

• In the last lecture we saw that curves can be represented conveniently in parametric form. Eg in 2D

$$\mathbf{x}(p) = (\ x(p),\ \ y(p)\ )^\top$$

and we saw some "regular" curves like ellipses, and so on.

• In this lecture we introduce how to build "irregular" shaped curves in a piecewise fashion out of **splines**.

• Splines are widely used in graphics and CAD — indeed one of the pioneers of their computational use, Pierre Bézier, developed his ideas while working for Renault in the 1970's.

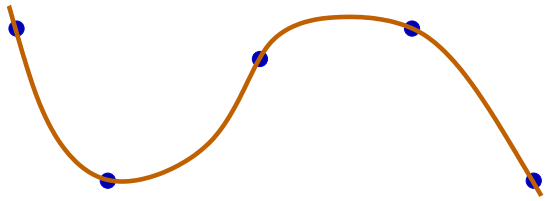
Example of an interpolating spline 4.2

An idea of what were are about can be gained by considering the following problem

How should one specify the path of a road through a number of villages ...

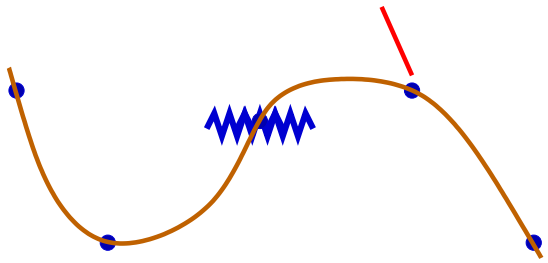
Roads made up of separate functions $\mathbf{x}_i$.

Continuous	$\mathbf{x}_i(\bar{p}_i) = \mathbf{x}_{i+1}(\bar{p}_i)$
Smooth	$\mathbf{x}'_i(\bar{p}_i) = \mathbf{x}'_{i+1}(\bar{p}_i)$
Cont 2nd d?	$\mathbf{x}''_i(\bar{p}_i) = \mathbf{x}''_{i+1}(\bar{p}_i)$



These are **interpolating** splines, because they actually go through the points. Later we will see approximating splines, where the points are **control points**, which define the shape of the curve, but do not necessary lie on the curve.

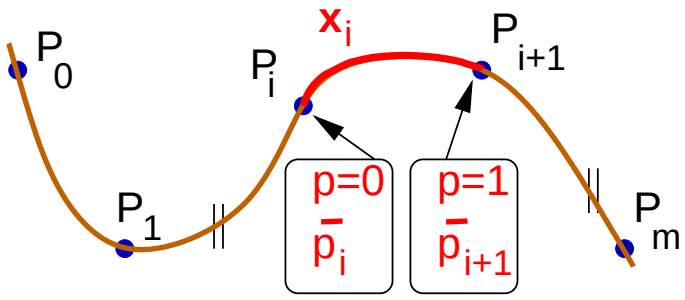
Other constraints might be applied ...
Eg, need to have particular orientation to meet other roads, rivers at right angles ...



Parameterization

• Convenient to use $\bar{p}$ as a global parameter along the entire curve, and p as a local parameter running from 0 to 1 between points.
That is, within spline i, the local parameter is

$$p = \frac{\bar{p} - \bar{p}_i}{\bar{p}_{i+1} - \bar{p}_i}$$



• Polynomials provide usable functions for the splines...
But what degree to use ?

Linear? — a "Roman Road" spline, so not smooth!
Quadratic? — let's see ...

Quadratic splines?

Think just with the $y(\bar{p})$ part of $\mathbf{x} = (x(\bar{p}), y(\bar{p}))^\top$

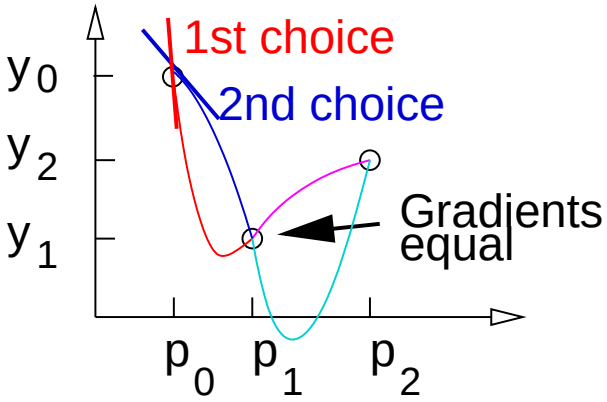
Using the global parameterization

$$\begin{aligned} y_0 &= a + b\bar{p}_0 + c\bar{p}_0^2 \\ y_1 &= a + b\bar{p}_1 + c\bar{p}_1^2 \\ y_0' &= b + 2c\bar{p}_0 \end{aligned}$$

Using the local parameterization

$$\begin{aligned} y_0 &= a \\ y_1 &= a + b + c \\ y_0' &= b \end{aligned}$$

This suggests:
• **Choose** gradient y_0' at P_0.
• Use gradient and y_0, y_1 to fit quadratic.
• This fixes gradient y_1' at P_1.
• Use gradient y_1' and y_1, y_2 to fit next
• ... and so on ...
• ... Then repeat for x part ...



• So for a continuous, smooth quadratic spline simply choose of gradient at the first (or any one) point, and **the entire curve is fixed**.

• Too sensitive to choice of gradient. Likely to oscillate.

Cubic splines 4.5

Now consider just the local parameterization of the ith cubic spline sequence. Again, we just look at y — similar equations are written for x — but you should realize that the a_i, b_i, etc are different.

$$\begin{aligned} & y_i(p) &=& a_i + b_i p + c_i p^2 + d_i p^3 \\ \Rightarrow\ & y_i(0) &=& a_i \\ & y_i(1) &=& a_i + b_i + c_i + d_i \\ & y_i'(0) &=& D_i = b_i \\ & y_i'(1) &=& D_{i+1} = b_i + 2c_i + 3d_i \end{aligned}$$

Re-arranging, and writing $y_i(1) = y_{i+1}$,

$$\begin{aligned} a_i &= y_i \\ b_i &= D_i \\ c_i &= 3(y_{i+1} - y_i) - 2D_i - D_{i+1} \\ d_i &= -2(y_{i+1} - y_i) + D_i + D_{i+1} \end{aligned}$$

We make the derivative equal at the junction of two splines. The is called Hermite interpolation. So, for a m segment spline, there are $m + 1$ degrees of freedom.

How are the derivatives specified? 4.6

1. By external constraints (eg, the perpendicular road)
2. Automatically:
   - For example, fit parabola to y_{i-1}, y_i, y_{i+1}, and use value of gradient at y_i as the gradient D_i.
   - Also need to choose end point gradients: eg, set $D_0 = 0$ and $D_m = 0$.
3. But, we could ask the 2nd derivatives to match as well. This is a **natural cubic spline**.

Natural cubic splines 4.7

Try a counting argument to show that we have sufficient d.o.f.
Again we think only about the y cubic: the x cubic is similiar.

• Each of the m segments between P_0 and P_m requires 4 parameters to specify the cubic in y. $\Rightarrow 4m$ UNKNOWNS.

At each of $m-1$ internal points on spline

$$y_{i-1}(1) = y_i \qquad y_i(0) = y_i \qquad y'_{i-1}(1) = y'_i(0) \qquad y''_{i-1}(1) = y''_i(0)$$

$\Rightarrow 4(m-1)$ CONSTRAINTS

At the end points

$$y_0(0) = y_0 \qquad y_m(1) = y_{m+1}$$

$\Rightarrow 2$ CONSTRAINTS

Now, $4m-2 < 4m$ so we can do it!

The extra two degrees of freedom are setting the second derivatives at the end points:

$$y''_0(0) = 0 \qquad y''_m(1) = 0$$

$\Rightarrow 2$ CONSTRAINTS

Computing Natural Cubic Splines 4.8

Don't solve for the cubic splines by inverting a $4m \times 4m$ matrix. Instead ...

Basic equations

$$\begin{aligned} y_i(p) &= a_i + b_i p + c_i p^2 + d_i p^3 \\ y'_i(p) &= b_i + 2c_i p + 3d_i p^2 \\ y''_i(p) &= 2c_i + 6d_i p \end{aligned}$$

Put in $p = 0$ and $p = 1$ into the first two on the left.

Leads to

$$\begin{aligned} a_i &= y_i \\ b_i &= D_i \\ c_i &= 3(y_{i+1} - y_i) - 2D_i - D_{i+1} \\ d_i &= -2(y_{i+1} - y_i) + D_i + D_{i+1} \end{aligned}$$

Now equate the 2nd derivs ($p = 1$ for the $(i-1)$th section, $p = 0$ for the ith ...

$$\begin{aligned} 2c_{i-1} + 6d_{i-1} &= 2c_i \\ \Rightarrow \quad 2[3(y_i - y_{i-1}) - 2D_{i-1} - D_i] &+ 6[-2(y_i - y_{i-1}) + D_{i-1} + D_i] \\ &= 2[3(y_{i+1} - y_i) - 2D_i - D_{i+1}] \\ \Rightarrow \quad D_{i-1} + 4D_i + D_{i+1} &= 3(y_{i+1} - y_{i-1}) \end{aligned}$$

Computing Natural Cubic Splines, ctd

To repeat

$$D_{i-1} + 4D_i + D_{i+1} = 3(y_{i+1} - y_{i-1})$$

At the first point, $y_0''(0) = 0$. So $c_0 = 0$ and $3(y_1 - y_0) - 2D_0 - D_1 = 0$, so

$$2D_0 + D_1 = 3(y_1 - y_0)$$

Similarly for the end section $y_m''(1) = 0$, so $2c_m + 6d_m = 0$ from which

$$D_{m-1} + 2D_m = 3(y_m - y_{m-1})$$

Gather all this together

$$\begin{bmatrix} 2 & 1 & & & & & \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & & \vdots & & & \\ & & & & 1 & 4 & 1 \\ & & & & & 1 & 2 \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ \vdots \\ D_{m-1} \\ D_m \end{bmatrix} = \begin{bmatrix} 3(y_1 - y_0) \\ 3(y_2 - y_0) \\ 3(y_3 - y_1) \\ \vdots \\ 3(y_m - y_{m-2}) \\ 3(y_m - y_{m-1}) \end{bmatrix}$$

Computing Natural Cubic Splines, ctd

Use row ops to simplify further to

$$\begin{bmatrix} 1 & \gamma_0 & & & & & \\ & 1 & \gamma_1 & & & & \\ & & 1 & \gamma_2 & & & \\ & & & & \vdots & & \\ & & & & & 1 & \gamma_{m-1} \\ & & & & & & 1 \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ \vdots \\ D_{m-1} \\ D_m \end{bmatrix} = \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \\ \Gamma_2 \\ \vdots \\ \\ \Gamma_m \end{bmatrix}$$

Then back substitute to find D_m, D_{m-1} and so in reverse order.

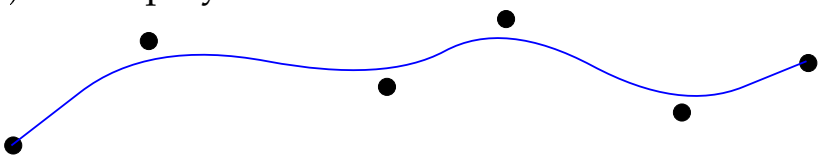
Having got the D's substitute in to

$$\begin{aligned} a_i &= y_i \\ b_i &= D_i \\ c_i &= 3(y_{i+1} - y_i) - 2D_i - D_{i+1} \\ d_i &= -2(y_{i+1} - y_i) + D_i + D_{i+1} \end{aligned}$$

Approximating curves as opposed to interpolating

4.11

- In 2D $x(u), y(u)$ each a polynomial in u.



- Local control of curve.
- Important examples are Bézier curves and B-splines.
- Used in graphics, CAD, drawing packages ...

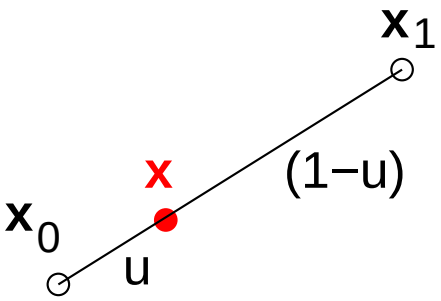
Bézier curves

4.12

Bezier curves were one of the earliest spline curves. The curves were originally devised using the *de Casteljau* construction.

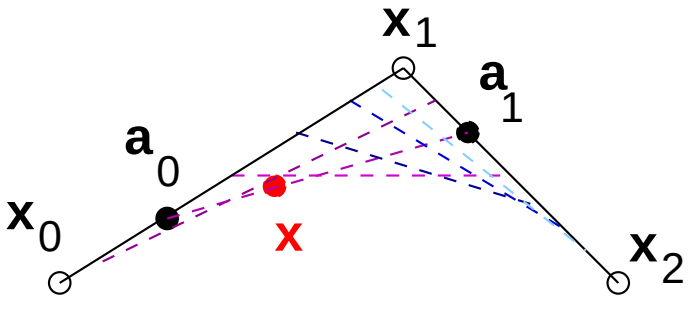
First consider a Bezier curve of degree 1, between P_0 and P_1

$$\mathbf{x} = (1-u)\mathbf{x}_0 + u\mathbf{x}_1$$



Now consider three control points, $P_{0,1,2}$ and set

$$\begin{aligned} \mathbf{a}_0 &= (1-u)\mathbf{x}_0 + u\mathbf{x}_1 \\ \mathbf{a}_1 &= (1-u)\mathbf{x}_1 + u\mathbf{x}_2 \\ \mathbf{x} &= (1-u)\mathbf{a}_0 + u\mathbf{a}_1 \end{aligned}$$



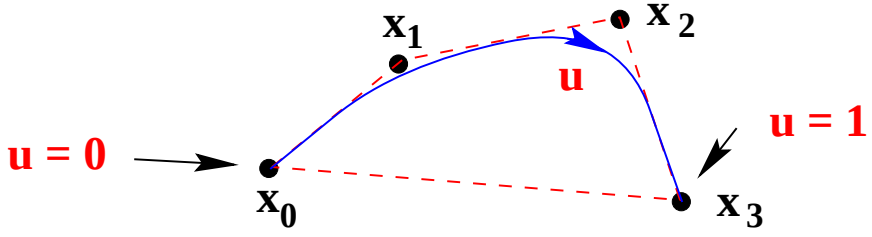
Then x follows a quadratic Bezier curve

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

Carrying on ... 4.13

Now consider four control points, $P_{0,1,2,3}$ and set

$$\begin{aligned}
\mathbf{a}_0 &= (1-u)\mathbf{x}_0 + u\mathbf{x}_1 \\
\mathbf{a}_1 &= (1-u)\mathbf{x}_1 + u\mathbf{x}_2 \\
\mathbf{a}_2 &= (1-u)\mathbf{x}_2 + u\mathbf{x}_3 \\
\mathbf{b}_0 &= (1-u)\mathbf{a}_0 + u\mathbf{a}_1 \\
\mathbf{b}_1 &= (1-u)\mathbf{a}_1 + u\mathbf{a}_2 \\
\mathbf{x} &= (1-u)\mathbf{b}_0 + u\mathbf{b}_1
\end{aligned}$$



Then $\mathbf{x}$ follows a cubic Bezier curve

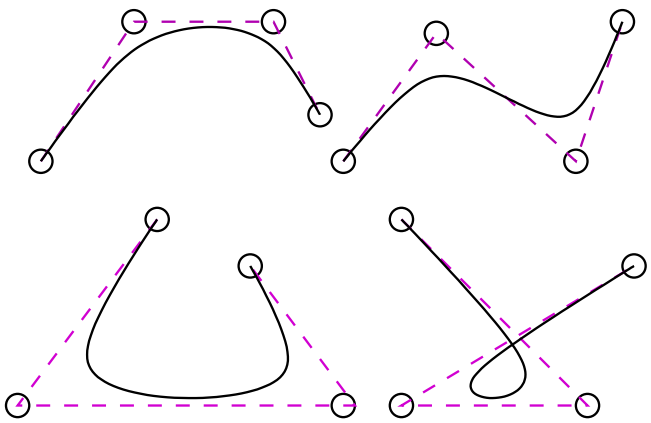
$$\mathbf{x}(u) = (x(u), y(u))^\top = (1-u)^3\mathbf{x}_0 + 3u(1-u)^2\mathbf{x}_1 + 3u^2(1-u)\mathbf{x}_2 + u^3\mathbf{x}_3 \quad \text{for } 0 \le u \le 1$$

Bézier curves of degree 3 4.14

$$\mathbf{x}(u) = (1-u)^3\mathbf{x}_0 + 3u(1-u)^2\mathbf{x}_1 + 3u^2(1-u)\mathbf{x}_2 + u^3\mathbf{x}_3 \quad \text{for } 0 \le u \le 1$$

Properties

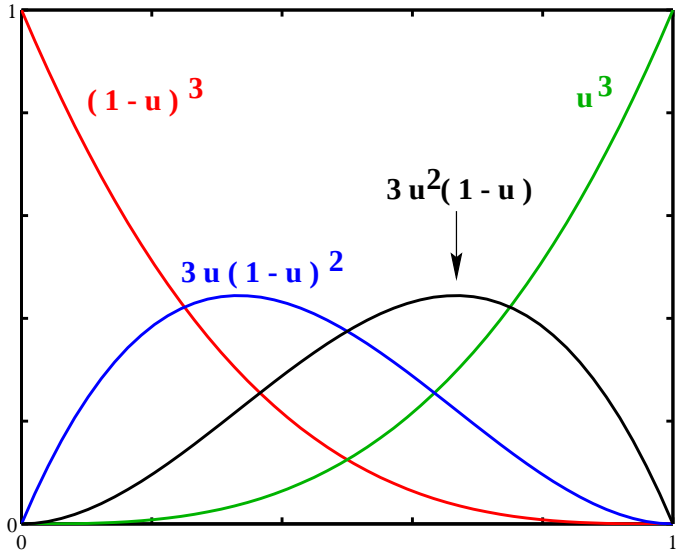
- $\mathbf{x}(0) = \mathbf{x}_0$ and $\mathbf{x}(1) = \mathbf{x}_3$, so that a Bézier curve interpolates its end control points.
- $\frac{d\mathbf{x}(0)}{du} = 3(\mathbf{x}_1 - \mathbf{x}_0)$. Geometric significance?
- The curve lies inside the convex hull of its control points.
- The control points need not be planar.
- Is it always smooth
- It can have inflections.
- It can self-cross.



Blending functions — actually Bernstein basis functions 4.15

$$\mathbf{x}(u) = (1-u)^3\mathbf{x}_0 + 3u(1-u)^2\mathbf{x}_1 + 3u^2(1-u)\mathbf{x}_2 + u^3\mathbf{x}_3$$

- One can think of the coefficients of the control points as blending functions
- These are symmetric under $u \to (1-u)$.



Bézier curves of degree N 4.16

The de Casteljau construction can be expressed as

$$\mathbf{x}(u) = \sum_{n=0}^{N} f_n(u)\mathbf{x}_n \quad \text{where} \quad f_n(u) = \begin{pmatrix} N \\ n \end{pmatrix} (1-u)^{N-n}u^n \quad \text{and} \quad 0 \le u \le 1$$

e.g. for $N = 3$ (cubic Bézier):

$$\begin{aligned} f_0(u) &= (1-u)^3 \\ f_1(u) &= 3(1-u)^2u \\ f_2(u) &= 3(1-u)u^2 \\ f_3(u) &= u^3 \end{aligned}$$

Note that (what is the geom significance?)

$$\sum_{n=0}^{N} f_n(u) = (u + (1-u))^N = 1 \quad \text{and} \quad f_n(u) \ge 0 \quad \text{for} \quad 0 \le u \le 1$$

Matrix form 4.17

Another representation is

$$\mathbf{x}(u) = (f_0(u)\ f_1(u)\ f_2(u)\ f_3(u)) \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} \qquad f_0(u) = (1-u)^3 = \begin{pmatrix} 1\ u\ u^2\ u^3 \end{pmatrix} \begin{pmatrix} 1 \\ -3 \\ +3 \\ -1 \end{pmatrix}$$

Collecting these together:

$$\begin{aligned} \mathbf{x}(u) &= \begin{pmatrix} 1\ u\ u^2\ u^3 \end{pmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} \\ &= \mathtt{U}\ \mathtt{M}_{\text{Bezier}}\ \mathtt{X} \end{aligned}$$

$\mathtt{M}_{\text{Bezier}}$ is the Bezier basis matrix for order 3.

Example 4.18

Where is $\mathbf{x}(0.5)$?

$$\begin{aligned} \mathbf{x}(\tfrac{1}{2}) &= \begin{pmatrix} 1\ \frac{1}{2}\ \frac{1}{4}\ \frac{1}{8} \end{pmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} \\ &= \frac{1}{8}(1\ 3\ 3\ 1) \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} = \frac{1}{8}\mathbf{x}_0 + \frac{3}{8}\mathbf{x}_1 + \frac{3}{8}\mathbf{x}_2 + \frac{1}{8}\mathbf{x}_3 \end{aligned}$$

So, it is a "centre-weighted centroid".

Parametric surface (Tensor Product) 4.19

We would like to represent a surface using control points and blending functions, in the same manner as a curve

$$\mathbf{x}(u) = (x(u), y(u), z(u))^\top = \sum_n f_n(u)\mathbf{x}_n$$

For a surface, we need two parameters:

$$\mathbf{x}(u,v) = (x(u,v), y(u,v), z(u,v))^\top = \sum_{mn} g_{m,n}(u,v)\mathbf{x}_{m,n}$$

with $\mathbf{x}_{m,n}$ a grid of 3D points.

Natural choice

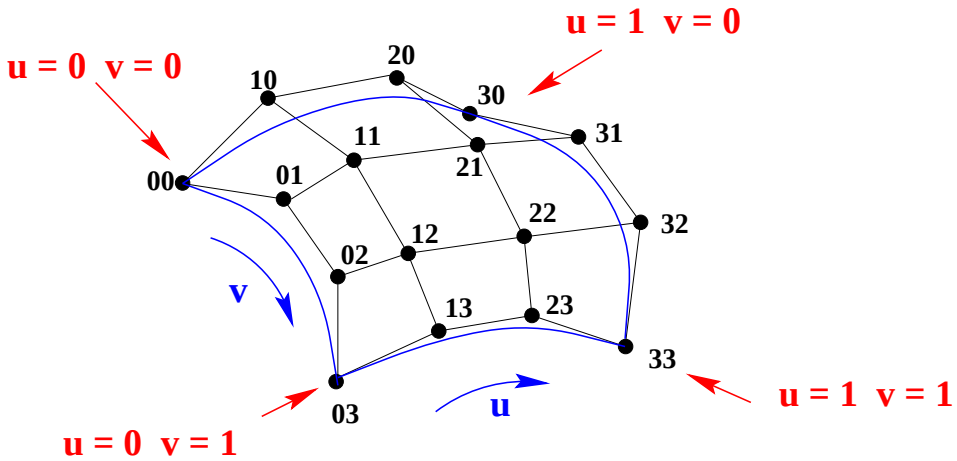
$$g_{m,n}(u,v) = f_m(u)f_n(v)$$

This is simply a "product" of Bézier curves

$$\mathbf{x}(u,v) = \sum_m f_m(u)\left(\sum_n f_n(v)\mathbf{x}_{m,n}\right)$$

The 16 control points for a Bézier bicubic patch 4.20

$$\mathbf{x}(u,v) = \sum_{mn} f_m(u)f_n(v)\mathbf{x}_{m,n}$$



Properties

- $\mathbf{x}(0,0) = \mathbf{x}_{0,0},\ \mathbf{x}(0,1) = \mathbf{x}_{0,3}\ldots$ so that corners are pinned.
- Convex hull property

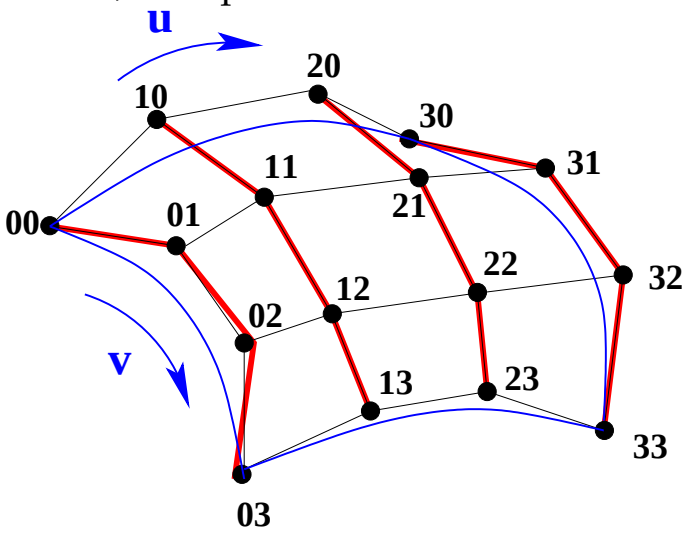
$$\sum_{m=0}^{N}\sum_{n=0}^{N} g_{m,n}(u,v) = \left(\sum_{m=0}^{N} f_m(u)\right)\left(\sum_{n=0}^{N} f_n(v)\right) = 1$$

Another way of picturing the process

4.21

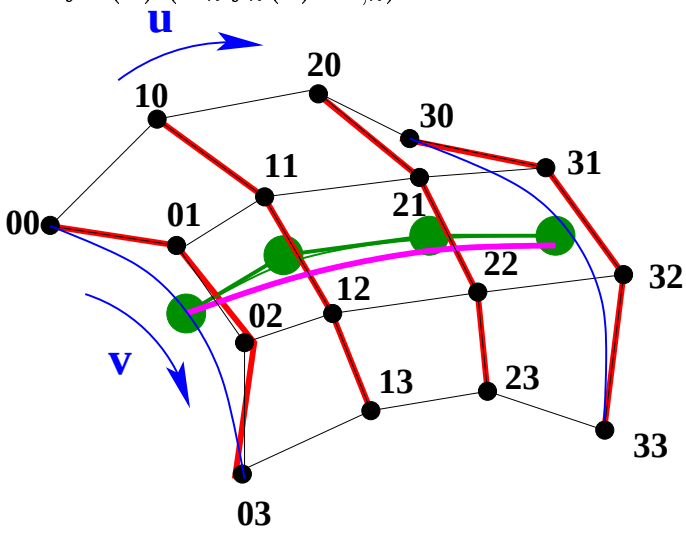
You might prefer to look at the expression $\mathbf{x}(u,v) = \sum_m f_m(u)\left(\sum_n f_n(v)\mathbf{x}_{m,n}\right)$ in a rather different way.

For $m = 0, 1, 2, 3$ $\left(\sum_n f_n(v)\mathbf{x}_{m,n}\right)$ for $0 \le v \le 1$ defines four separate Bezier curves, each parallel to the v axis.



Choose a value of v and you have four points on these curves.

Then construct a Bezier curve out of these for $0 \le v \le 1$ — ie $\sum_m f_m(u)\left(\sum_n f_n(v)\mathbf{x}_{m,n}\right)$.



Then pick a point on this curve by choosing u.

Matrix form

4.22

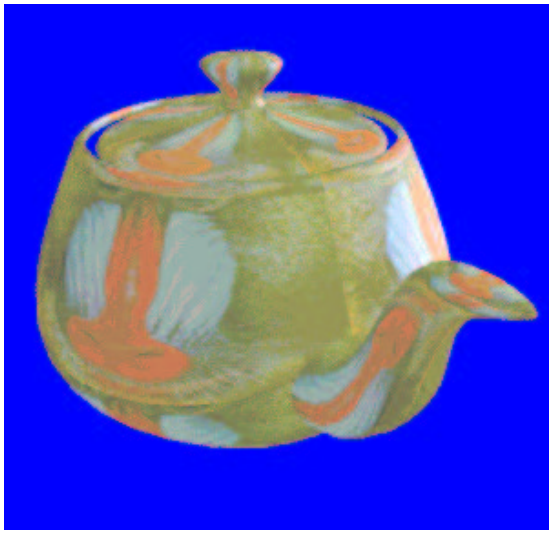
Another representation is using matrices. Recall $\mathbf{x}(u) = \mathtt{U\,M\,X}$.

$$\mathbf{x}(u,v) = \mathtt{U\,M\,B\,M}^\top\,\mathtt{V}^\top \qquad \text{4 loops}$$

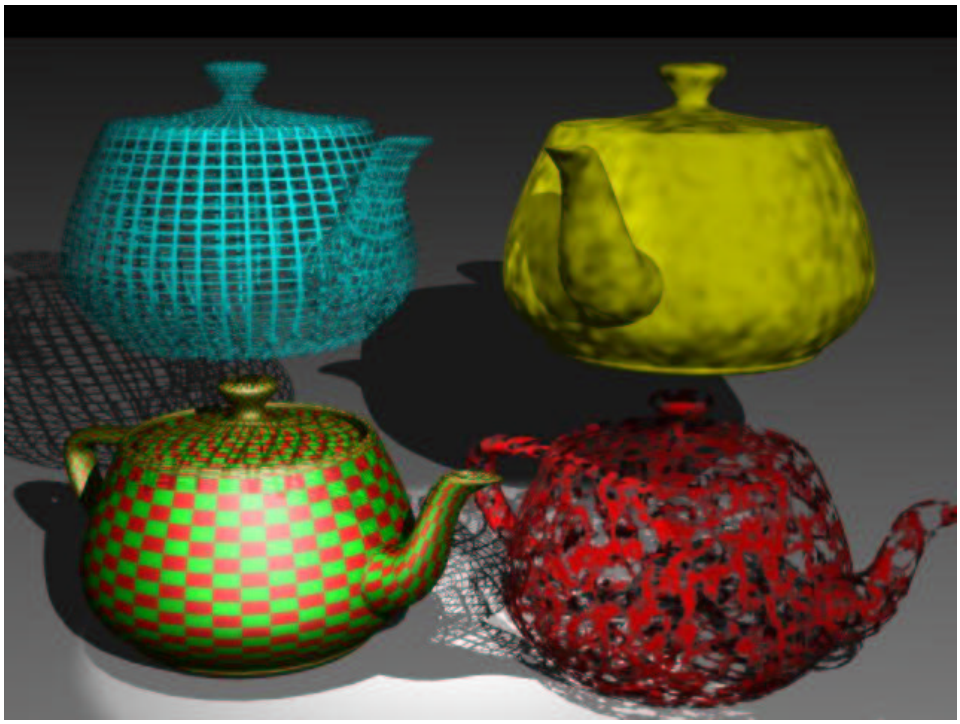
where

$$\mathtt{B} = \begin{bmatrix} \mathbf{x}_{0,0} & \mathbf{x}_{0,1} & \mathbf{x}_{0,2} & \mathbf{x}_{0,3} \\ \mathbf{x}_{1,0} & \mathbf{x}_{1,1} & \mathbf{x}_{1,2} & \mathbf{x}_{1,3} \\ \mathbf{x}_{2,0} & \mathbf{x}_{2,1} & \mathbf{x}_{2,2} & \mathbf{x}_{2,3} \\ \mathbf{x}_{3,0} & \mathbf{x}_{3,1} & \mathbf{x}_{3,2} & \mathbf{x}_{3,3} \end{bmatrix} \qquad \mathtt{V} = \begin{pmatrix} 1 & v & v^2 & v^3 \end{pmatrix}$$

Graphics application of Bézier tensor product surface 4.23



More teapots ... 4.24



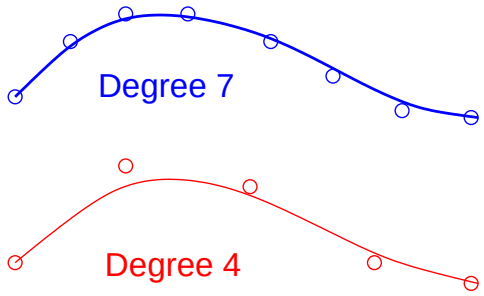
Other properties of Bezier curves 4.25

Switching to higher degree A Bezier curve of degree N is also a Bezier curve of degree $N' = N + 1$.

This means that the $N + 1$ control points can be replaced by $N + 2$ control points in different positions and the curve shape remains unchanged, and hence one can increase the degree arbitrarily.

Here are the expressions for moving the control points to increase the degree by 1.

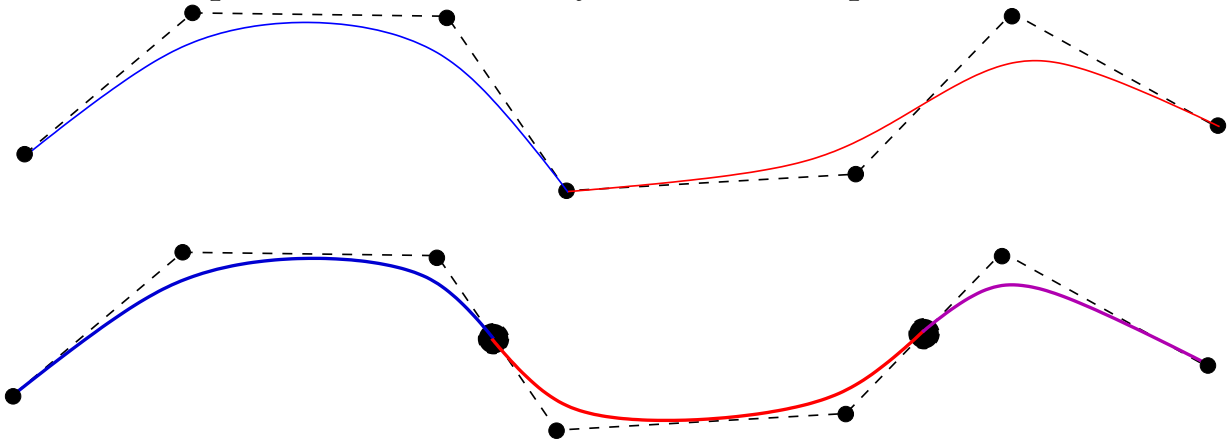
$$\begin{aligned} \mathbf{x}'_0 &= \mathbf{x}_0 \\ \mathbf{x}'_i &= \frac{i}{N+1}\mathbf{x}_{i-1} + \left(1 - \frac{i}{N+1}\right)\mathbf{x}_i \\ & \qquad i = 1, \ldots, N \\ \mathbf{x}'_{N+1} &= \mathbf{x}_N \end{aligned}$$



Piecewise Bezier curves 4.26

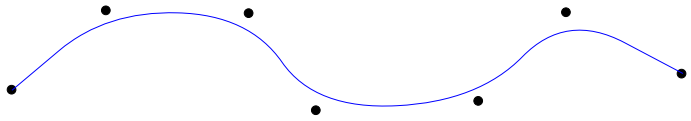
Suppose we require many control points ...

- Piecewise Bézier curves
- Need an extra point inserted midway. This will be a point of inflexion.



B-Splines

4.27



• Like draftsperson's spline, but approximating.

• A B-spline consists of several curve segments, each represented by a polynomial.

• Controlled smoothness: e.g. cubic B-splines are $\mathcal{C}^2$, or can be made to be less: $\mathcal{C}^1$ or $\mathcal{C}^0$.

Key advantage:

• **Local control**: moving a control point only affects part of the curve.

• This isn't the case with a Bezier curve of high degree. But often it is said not to be the case with, say, cubic Bezier curves. But introducing the "extra" points does provide decoupling ...

Example - a *cubic B-spline*

4.28

• $m+1$ control points, $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m$, with $m \geq 3$.

• $m-2$ cubic polynomial curve segments $\mathbf{c}_3, \mathbf{c}_4, ..., \mathbf{c}_m$.

• Each curve segment is defined as

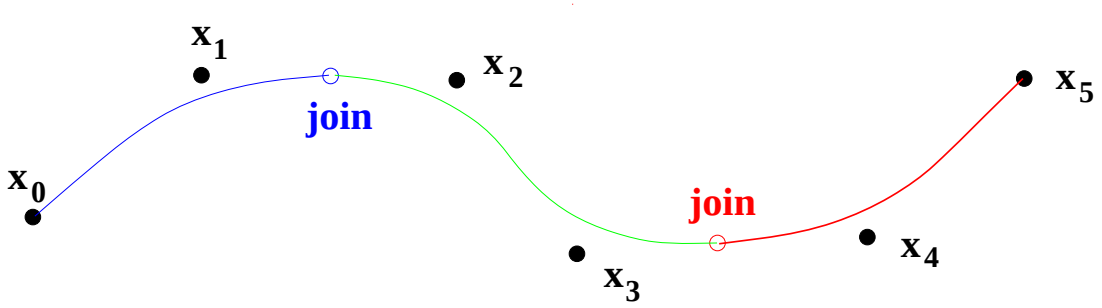
$$\mathbf{x}_i(u) = \begin{pmatrix} 1 & u & u^2 & u^3 \end{pmatrix} \mathtt{B} \begin{pmatrix} \mathbf{x}_{i-3} \\ \mathbf{x}_{i-2} \\ \mathbf{x}_{i-1} \\ \mathbf{x}_i \end{pmatrix} \quad \text{with} \quad \mathtt{B} = \frac{1}{6} \begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

• If the spline is closed then $\mathtt{B}$ is the same for all curve segments and $\mathbf{x}_0 = \mathbf{x}_m$.

• If the spline is open then the matrices $\mathtt{B}$ can be modified at the curve ends so that the curve segments interpolate the end control points.

B-splines, ctd

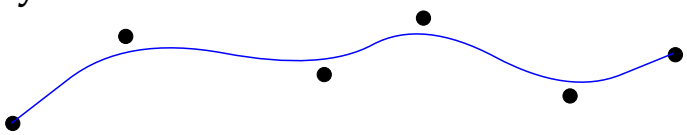
4.29



—— influenced by $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$

—— influenced by $\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5$

Convex hull property



Spline demo: http://www.geocities.com/msheinrichs/curve.html

Tracking application of B-splines

4.30

- Used in computer vision as "Active Contours". Here the local control is a bonus.

- See Blake and Isard "Active Contours" Springer 1999.