

Animation Systems: 3. Interpolation

Charles A. Wüthrich

CogVis/MMC, Faculty of Media

Bauhaus-University Weimar



Parametric curves

- Curves and surfaces can have explicit, implicit, and parametric representations.
 - Explicit equations are of the form $y=f(x)$
 - Implicit equations of the form $f(x,y)=0$
 - Parametric equations are of the form
- Parametric representations are the most common in computer graphics and animation.
- They are independent from the axes

$$\begin{cases} x = f(t) \\ y = g(t) \end{cases}$$

Parametric curves

- Parametrization is not unique: take a look at the straight line:

$$L(P_0, P_1) = P_0 + u(P_1 - P_0) = \\ (1-u)P_0 + uP_1, \\ u \in [0, 1]$$

$$L(P_0, P_1) = v(P_1 - P_0)/2 + \\ (P_1 + P_0)/2, v \in [-1, 1]$$

- They represent the same line

- Parameterizations can be changed to lie between desired bounds.

To reparameterize from $u \in [a, b]$ to $w \in [0, 1]$,

we can use

$w = (u - a) / (b - a)$, which gives $u = w(b - a) + a$.

- Thus, we have:

$$P(u), u \in [a, b] = \\ P(w(b - a) + a), w \in [0, 1]$$

Linear interpolation

- Consider the straight line passing through P_0 and P_1 :
$$P(u) = (1-u)P_0 + uP_1$$
- Since $(1-u)$ and u are functions of u , one can rewrite the eq. above as
$$P(u) = F_0(u)P_0 + F_1(u)P_1$$
- Note that $F_0(u) + F_1(u) = 1$
- $F_0(u)$ and $F_1(u)$ are called *blending functions*.
- Alternatively, one can rewrite the function as
$$P(u) = (P_1 - P_0)u + P_0$$

$$P(u) = \underline{a}_1 u + \underline{a}_0$$
- This called the algebraic form of the equation

Linear interpolation

- One can also rewrite these equations in matrix notation:
- Note that the last one of these equations decomposes the equation in the product of variables (U), coefficients (M) and geometric information (B)

$$P(u) = \begin{bmatrix} F_0(u) \\ F_1(u) \end{bmatrix} \begin{bmatrix} P_0 & P_1 \end{bmatrix} = FB^T$$

$$P(u) = \begin{bmatrix} u & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_0 \end{bmatrix}$$

$$P(u) = \begin{bmatrix} u & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \end{bmatrix} = U^T MB = FB = U^T A$$

Arc length

- Note that there is not necessarily a linear relation between the parameter u and the arc length described by the curve
- For example, also the equation

$$P(u) = P_0 + ((1-u)u + u)(P_1 - P_0)$$

represents the same straight line, but the relationship between u and the arc length is non linear.

- This means that there is not necessarily an obvious relationship between changes in parameter and distance travelled and changes in the parameter

Derivatives of a curve

- Any parametric curve of polynomial order can be expressed in the form

$$P(u) = U^T M B$$

- Since only the matrix U contains the variable, then it is easy to compute the derivative of a parametric curve
- For a curve of third degree we have

$$P(u) = U^T M B = [u^3 \ u^2 \ u \ 1] M B$$

$$P'(u) = U'^T M B = [3u^2 \ 2u \ 1 \ 0] M B$$

$$P''(u) = U''^T M B = [6u \ 2 \ 0 \ 0] M B$$

Hermite interpolation

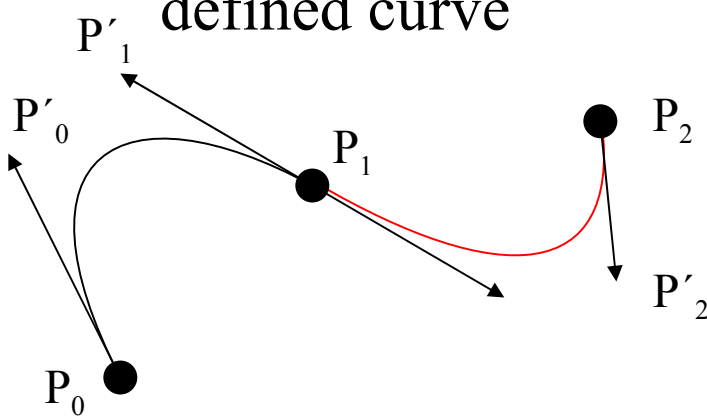
- Hermite interpolation generates a cubic polynomial between two points.
- Here, to specify completely the curve the user needs to provide two points P_0 and P_1 and the tangent to the curve in these two points P'_0 P'_1
- Remember, we write in the form $P(u)=U^TMB$

- For Hermite interpolation we have

$$U^T \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}$$
$$M = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$
$$B = \begin{bmatrix} P_0 \\ P_1 \\ P'_0 \\ P'_1 \end{bmatrix}$$

Hermite interpolation

- Suppose that an interpolation curve is wanted passing through n points P_0, P_1, \dots, P_n .
- The interpolation curve through them can be defined as a piecewise defined curve
- In fact, if one ensures that the resulting curve is not only continuous at the joints, but also that
 - Its tangent (=velocity)
 - Its second order derivative (= acceleration)are continuous, then the curve can be used also in animations.



Continuity: parametric and geometric

- For a piecewise defined curve, there are two main ways of defining the continuity at the borders of the single intervals of definition
 - 1st order parametric continuity (C^1): the end tangent vector at the two ends must be exactly the same
 - 1st order geometric continuity (G^1): the direction of the tangent must be the same, but the magnitudes may differ
 - Similar definitions for higher order continuity (C^2 - G^2)
- Parametric continuity is sensitive to the „velocity“ of the parameter on the curve, geometric continuity is not

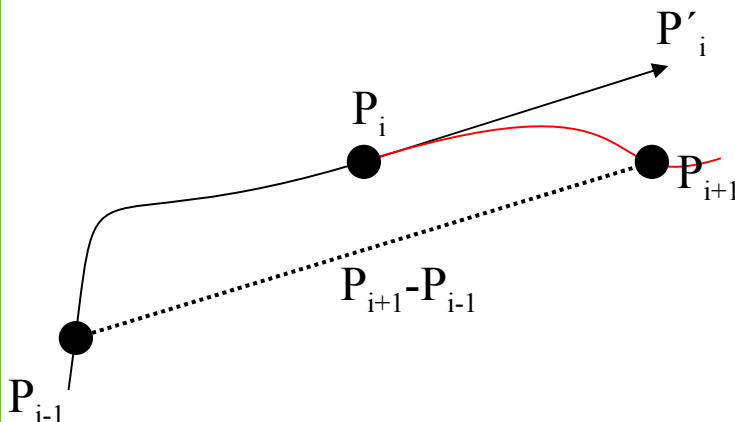
Catmull-Rom spline

- A Catmull-Rom spline is a special Hermite curve where the tangent of the middle points is computed as one half the vector joining the previous control point to the next one

- $P'_i = 1/2(P_{i+1} - P_{i-1})$
- From this we deduce:

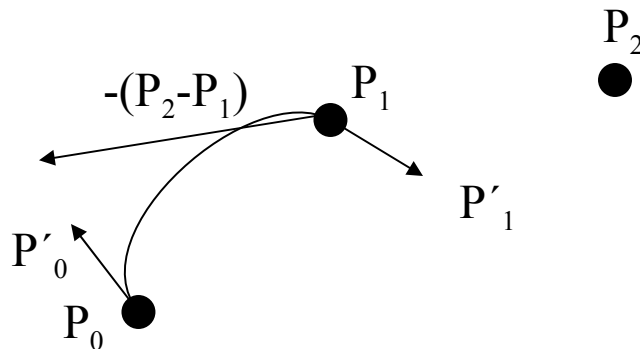
$$U^T = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}$$
$$M = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i+2} \end{bmatrix}$$



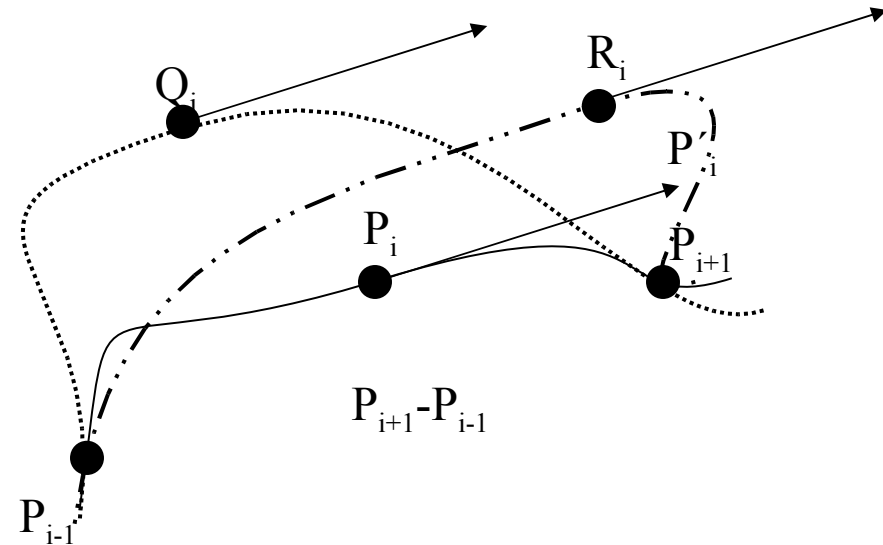
Catmull-Rom spline

- If one wants to write the complete Catmull-Rom spline, one needs a method to find the tangents at the initial and final points
- One method used involves subtracting P_2 from P_1 and then using the point obtained as the direction of the tangent
- $P'_0 = \frac{1}{2}(P_1 - (P_2 - P_1) - P_0) = \frac{1}{2}(2P_1 - P_2 - P_0)$



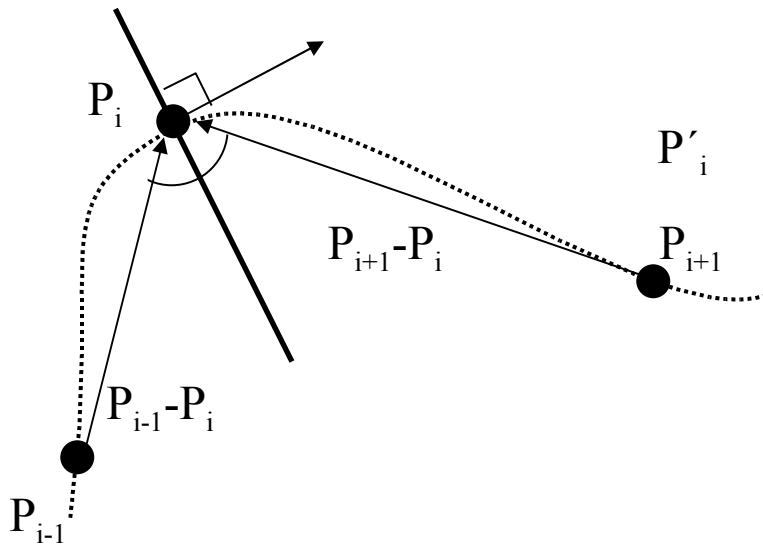
Catmull-Rom spline

- Advantage of Catmull-Rom splines: fast and simple computations
- Disadvantage: tangent vector is not this flexible: for example, all curves below have same tangent in P_i



Catmull-Rom spline

- A simple alternative is to compute the tangent at the point as the \perp of the bisector of the angle formed by $P_{i-1}-P_i$ and $P_{i+1}-P_i$
- Another modification is to not impose same tangent length at the points, but different lengths on the two sides of the joint.
- The tangent vectors can be scaled for example by the ratio of the distance between current point and former point and the distance between former and next point.
- This obtains more „adaptable“ tangents, but trades also off C^1 continuity



Four point form

- Suppose you have 4 points $P_0P_1P_2P_3$ and to want a cubic segment fitting through them.
- One can set up a linear system of equations through the points and solve

$$P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \\ m_{30} & m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

$$\begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix} = \begin{bmatrix} u_0^3 & u_0^2 & u_0 & 1 \\ u_1^3 & u_1^2 & u_1 & 1 \\ u_2^3 & u_2^2 & u_2 & 1 \\ u_3^3 & u_3^2 & u_3 & 1 \end{bmatrix} \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \\ m_{30} & m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

Four point form

- In the case that you want the parameter values at the points to be $(0, 1/3, 2/3, 1)$, the matrix is

$$M = \frac{1}{2} \begin{bmatrix} -9 & 27 & -27 & 9 \\ 18 & -45 & 36 & -9 \\ -11 & 18 & -9 & 2 \\ 2 & 0 & 0 & 0 \end{bmatrix}$$

With this form it is difficult to join segments with C^1 continuity

Blended parabolas

- One other method is through blending two overlapping parabolas
- The blending is done by taking the first 3 points to define a parabola, then the 2nd, 3rd and 4th point to define a second parabola, and then linearly interpolate the parabolas
- This is the resulting matrix for equally spaced points in parametric space

$$M = \frac{1}{2} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix}$$

Bezier curves

- Another way of defining a curve is to define it through two endpoints, which are interpolated, and two interior points, which control the shape.
- Bezier curves use the two additional control points to define the tangent
- $P'(0)=3(P_1-P_0)$
 $P'(1)=3(P_3-P_2)$

- The corresponding matrix will be

$$M = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

which corresponds to the basic functions UM

$$B_0(t)=(1-t)^3$$

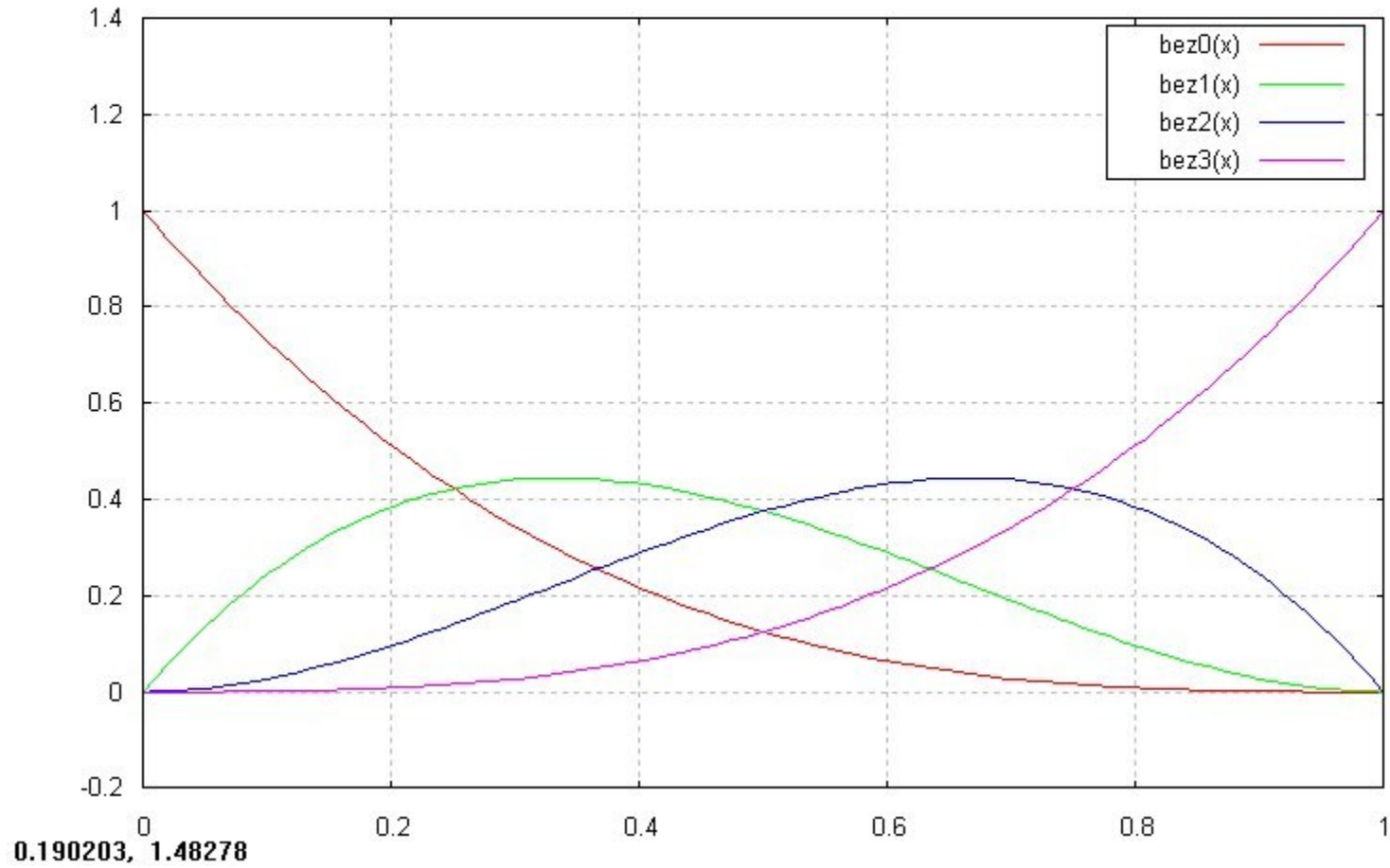
$$B_1(t)=3t(1-t)^2$$

$$B_2(t)=3t^2(1-t)$$

$$B_3(t)=t^3$$

Bezier curves

The cubic Bezier basis functions



Bezier curves

- In fact, Bezier curves can be of any order. The basis functions are

$$B_{in}(t) = t^i(1-t)^{n-i}n!/i!/(n-i)!$$

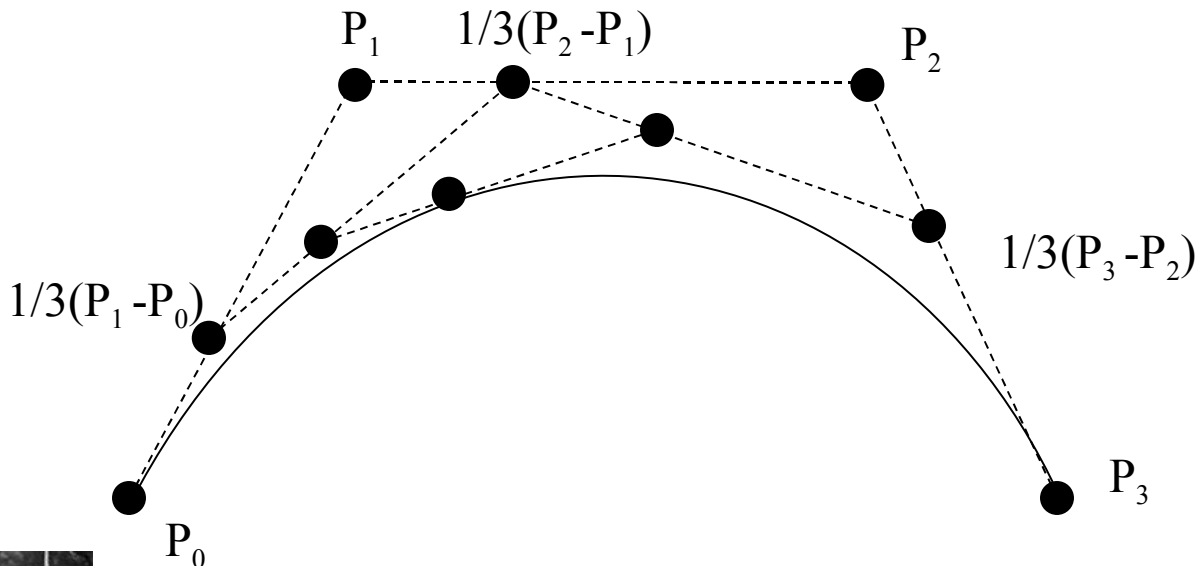
Where n is the degree and $i=0, \dots, n$.

- And the Bezier curve passing through the points P_0, P_1, \dots, P_n is

$$Q(T) = \sum_{i=0, \dots, n} B_{in}(t)P_i$$

Bezier curves: De Casteljeau construction

- De Casteljeau came up with a geometric method for constructing a Bezier Curve
- The figure illustrates the construction of a point at $t=1/3$ for a curve of 3rd degree



Uniform B-splines

- Uniform B-splines are most flexible type of curves, and also more difficult to understand
- They detach the order of the resulting polynomial from the number of control points. Suppose we have a number N of control points.
- Bezier curves are a special case of B-splines
- One starts by defining a uniform knot vector $[0,1,2,\dots,N+k-1]$, where k is the degree of the B-spline curve and n the number of control points.
- Knots are uniformly spaced.
- If k is the degree of the B-spline, then each single component of the B-spline will be defined between the consecutive control points $P_i, P_{i+1}, \dots, P_{i+k}$.
- The next bit will be defined between $P_{i+1}, P_{i+2}, \dots, P_{i+k+1}$

Uniform B-splines

- The equation for k -order B-spline with $N+1$ control points (P_0, P_1, \dots, P_N) is
- $N_{i,k}$ is a polynomial of order k (degree $k-1$) on each interval $t_i < t < t_{i+1}$.

$$P(t) = \sum_{i=0, \dots, N} N_{i,k}(t) P_i, \\ t_{k-1} \leq t \leq t_{N+1}$$

- In a B-spline each control point is associated with a basis function $N_{i,k}$ which is given by the recurrence relations
- k must be at least 2 (linear) and can be not more, than $n+1$ (the number of control points).
- A knot vector $(t_0, t_1, \dots, t_{N+k})$ must be specified. Across the knots basis functions are C^{k-2} continuous.

$$N_{i,k}(t) = \\ N_{i,k-1}(t) (t - t_i) / (t_{i+k-1} - t_i) + \\ N_{i+1,k-1}(t) (t_{i+k} - t) / (t_{i+k} - t_{i+1}), \\ N_{i,1} = \{1 \text{ if } t_i \leq t \leq t_{i+1}, \\ 0 \text{ otherwise } \}$$

Uniform B-splines

- B-spline basis functions as like as Bezier ones are nonnegative $N_{i,k} \geq 0$ and have "partition of unity" property
- Since $N_{i,k} = 0$ for $t \leq t_i$ or $t \geq t_{i+k}$ a control point P_i influences the curve only for $t_i < t < t_{i+k}$.

$$\sum_{i=0, N} N_{i,k}(t) = 1,$$

$$t_{k-1} < t < t_{n+1}$$

therefore

$$0 \leq N_{i,k} \leq 1.$$

B-splines

- Depending on the relative spaces between knots in parameter spaces, we can have uniform or non-uniform B-splines
- The shapes of the $N_{i,k}$ basis functions are determined entirely by the *relative* spacing between the knots $(t_0, t_1, \dots, t_{N+k})$.
- Scaling or translating the knot vector has no effect on shapes of basis functions and B-spline.
- Knot vectors are generally of 3 types:
 - *Uniform knot vectors* are the vectors for which
$$t_{i+1} - t_i = \text{const},$$
e.g. $[0, 1, 2, 3, 4, 5]$.
 - *Open Uniform knot vectors* are uniform knot vectors which have k -equal knot values at each end:
$$t_i = t_0, \quad i < k$$
$$t_{i+1} - t_i = \text{const}, \quad k-1 \leq i < n+1$$
$$t_i = t_{k+n}, \quad i \geq n+1$$
eg $[0, 0, 0, 1, 2, 3, 4, 4, 4]$ ($k=3, N=5$)
 - *Non-uniform knot vectors*. This is the general case, the only constraint is the standard $t_i \leq t_{i+1}$.

B-splines

- The main properties of B-splines
 - composed of $(n-k+2)$ Bezier curves of k -order joined C^{k-2} continuously at knot values $(t_0, t_1, \dots, t_{n+k})$
 - each point affected by k control points
 - each control point affected k segments
 - inside convex hull
 - affine invariance
 - uniform B-splines don't interpolate deBoor control points $(\mathbf{P}_0, \mathbf{P}_1, \dots, \mathbf{P}_N)$

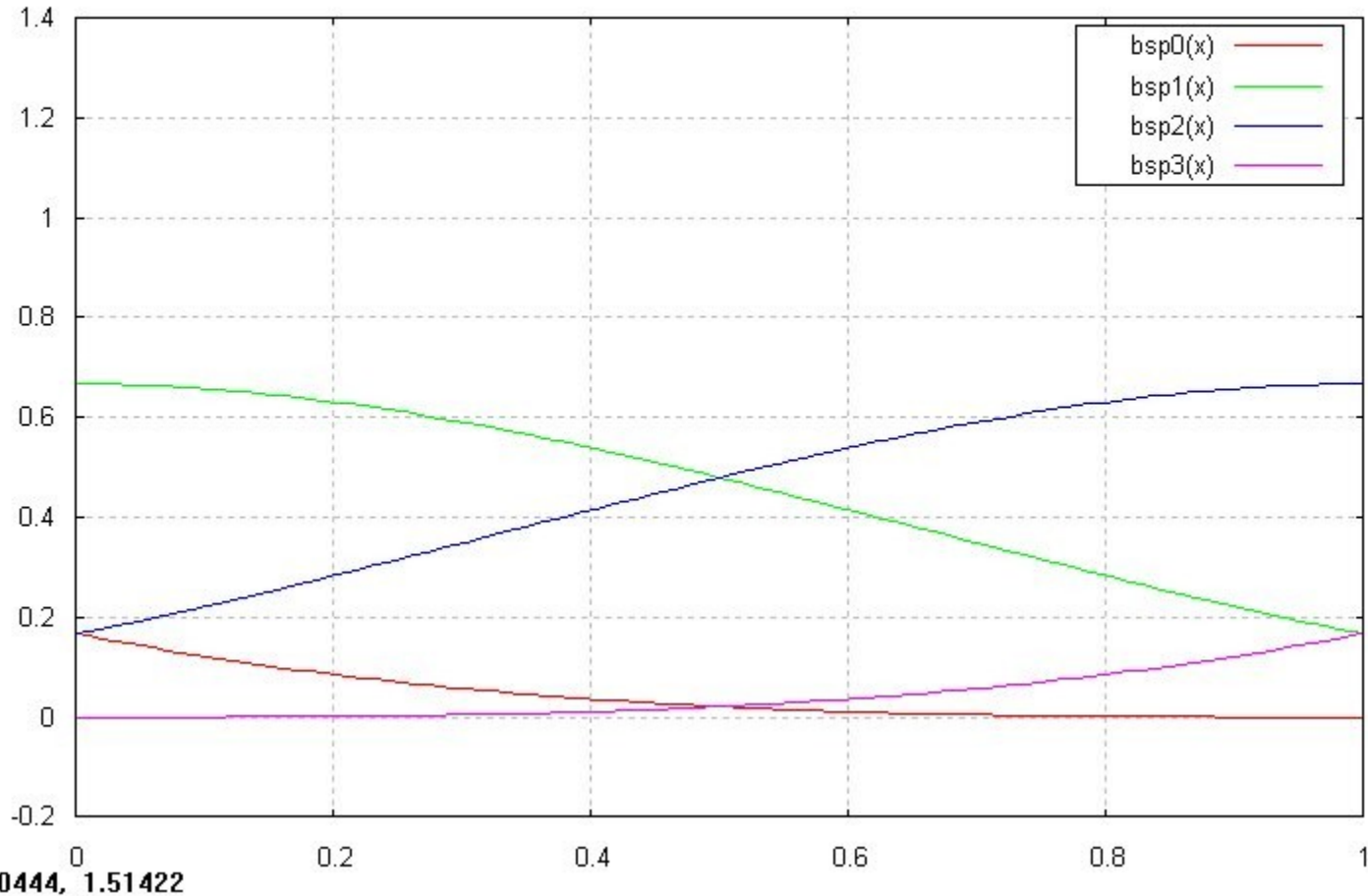
Uniform 3rd order B-splines

- For a B-spline of order 3, and the three control points $P_i, P_{i+1}, P_{i+2}, P_{i+3}$ we have that the B-spline can be written as
- The curves defined by increasing $i=0, \dots, N-3$ will define a C^2 -continuous curve

$$P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{bmatrix}$$

Uniform B-splines

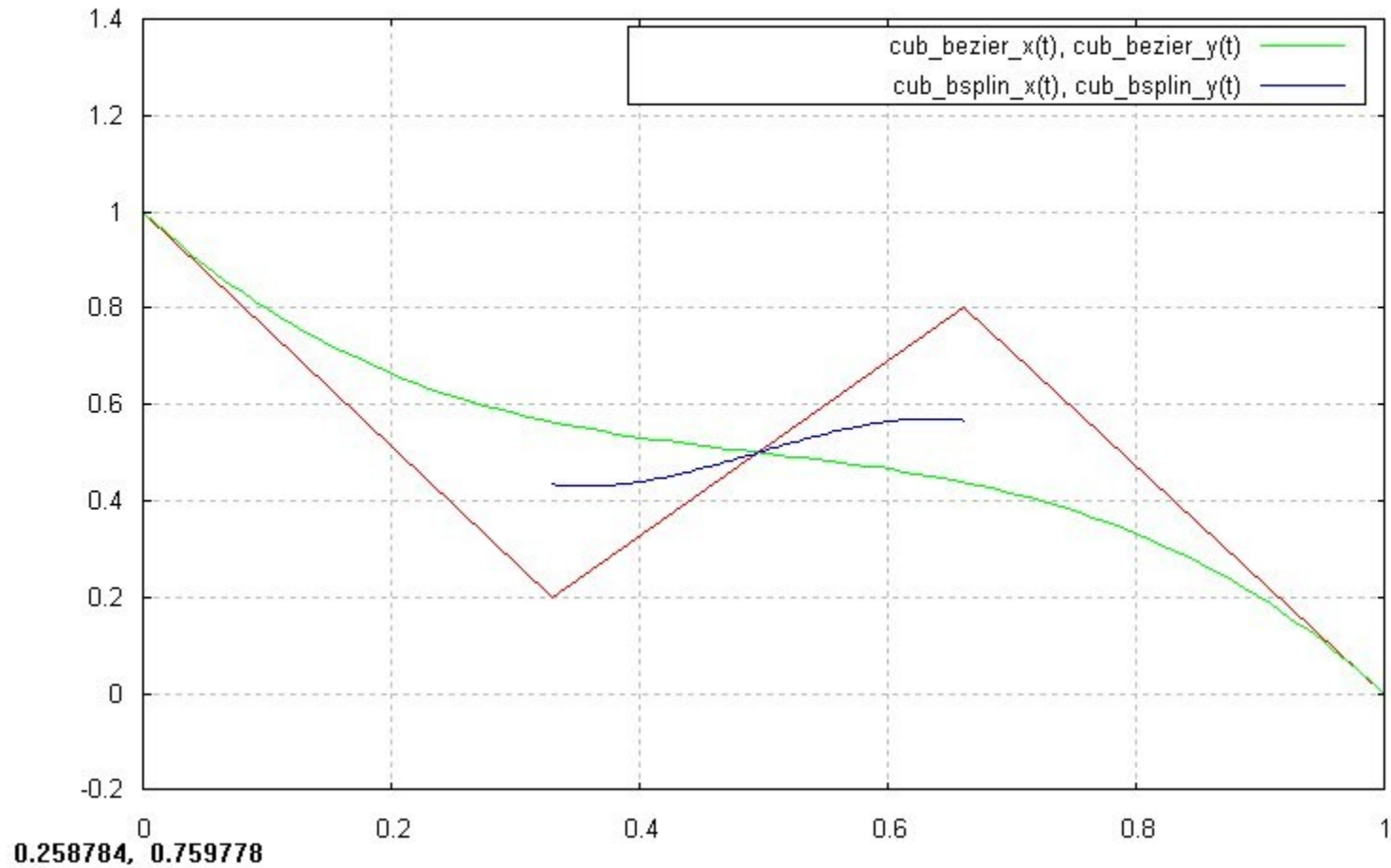
The cubic uniform B-spline basis functions



0.490444, 1.51422

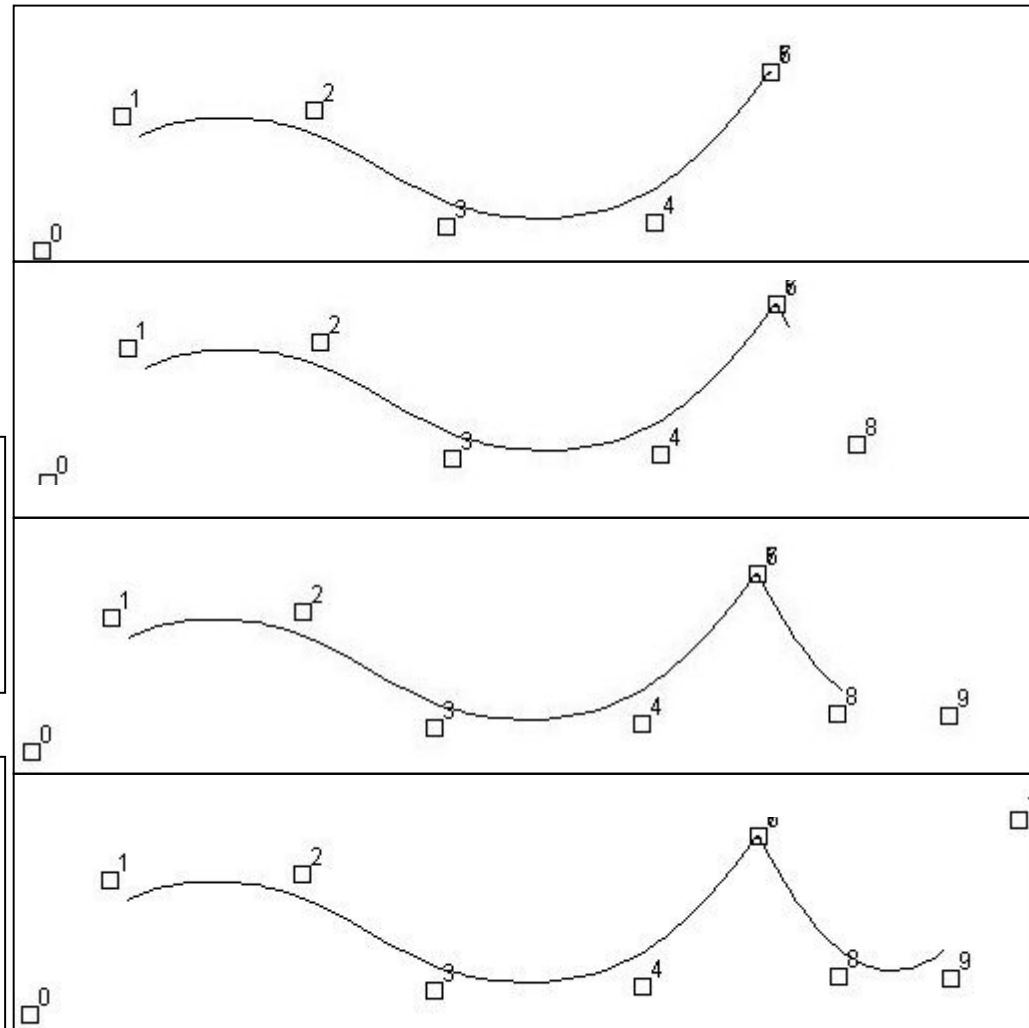
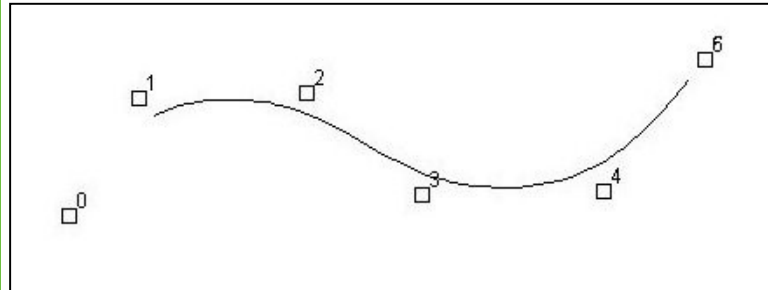
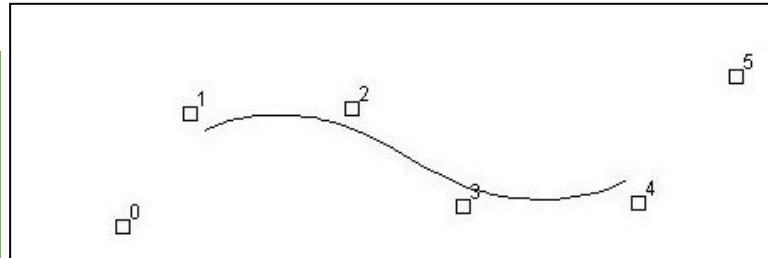
Bezier and B-spline curves

The cubic Bezier/Bspline basis functions in use



B-splines: multiple knots

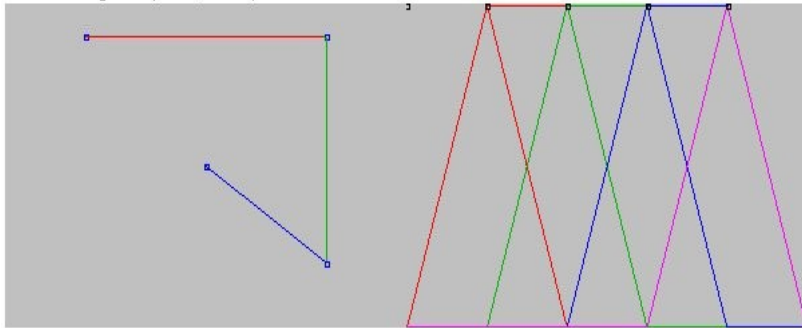
- Knots can be made to coincide to obtain cusps and passing through a desired point



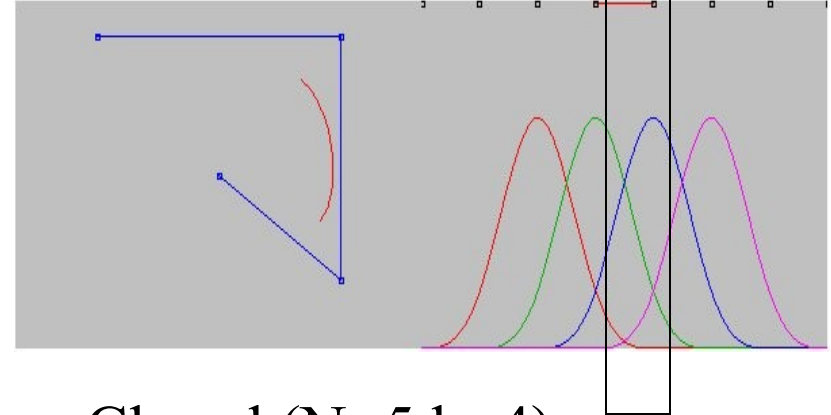
Uniform B-splines: examples

- For a given order k , uniform B-splines are shifted copies of one another since all the knots are equispaced

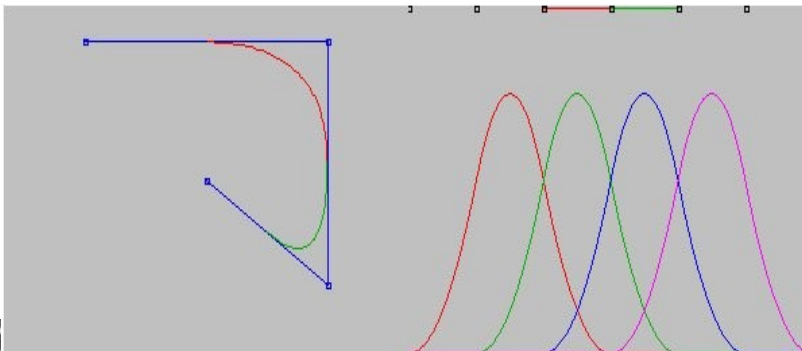
- Linear ($N=3, k=2$)



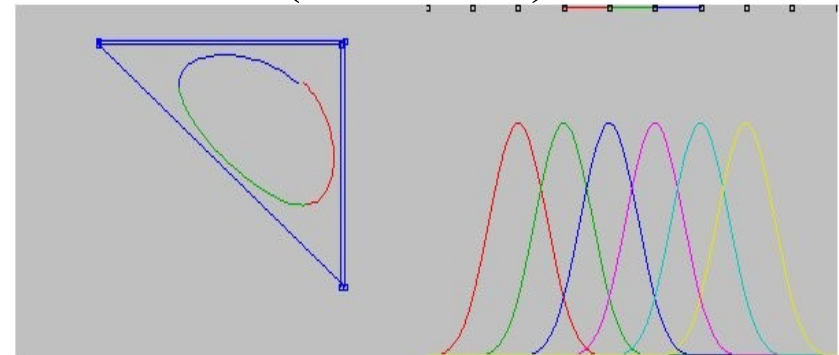
- Cubic ($N=3, k=4$)



- Quadratic ($N=3, k=3$)



- Closed ($N=5, k=4$)



NURBS

- Stands for non-uniform rational B-splines
 - Non-uniform: knots are not at same distance
 - Rational: it's a fraction, with B-splines at the numerator and denominator
- Advantages: one can express circular arcs with NURBS
- Disadvantages: lots of computational effort

NURBS

- Recall, that the B-spline is weighted sum of its control points

$$\mathbf{P}(t) = \sum_{i=0, \dots, N} N_{i,k}(t) \mathbf{P}_i, \\ t_{k-1} \leq t \leq t_{N+1}$$

and the weights $N_{i,k}$ have the "partition of unity" property

$$\sum_{i=0, \dots, N} N_{i,k}(t) = 1.$$

- As weights $N_{i,k}$ depend on the knot vector only, it is useful to add to every control point one more weight w_i which can be set independently

$$\mathbf{P}(t) = \frac{\sum_{i=0, \dots, N} w_i N_{i,k}(t) \mathbf{P}_i}{\sum_{i=0, \dots, N} w_i N_{i,k}(t)}.$$

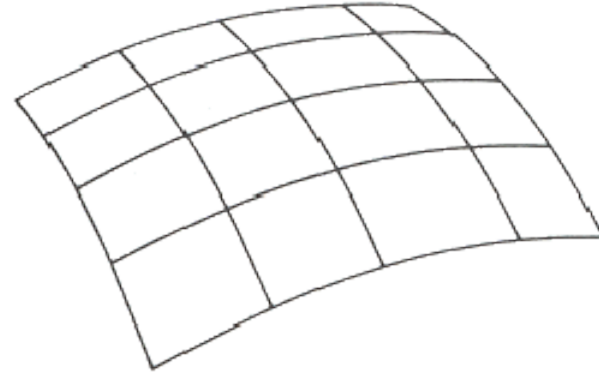
- Increasing a weight w_i makes the point more influence and attracts the curve to it.
- The denominator in the 2nd equation normalizes weights, so we will get the 1st equation if we set $w_i = \text{const}$ for all i .
- Full weights $w_i N_{i,k}$ satisfy the "partition of unity" condition again.

Global vs local control

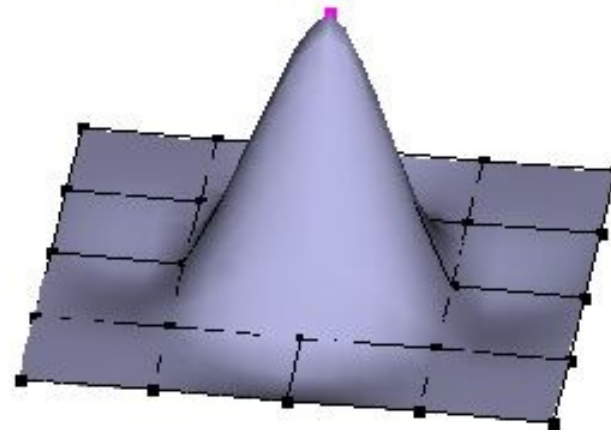
- Depending on the curve formulation, moving a control point can have different effects
 - Local control: in this case the effect of the movement is limited in its influence along the curve
 - Global control: moving a point redefines the whole curve
- Local control is the most desirable for manipulating a curve
- Almost all of the piecewise defined curves have local control
- Only exception: Hermite curves enforcing C^2 continuity

Modeling with splines

- 3D Splines can be used to represent object boundaries by piecewise defined „patches“ joined at their definition edges so that they are continuous at the joins, like a „patchwork“
- Splines are very flexible in shape modeling
- But what is behind spline patches?



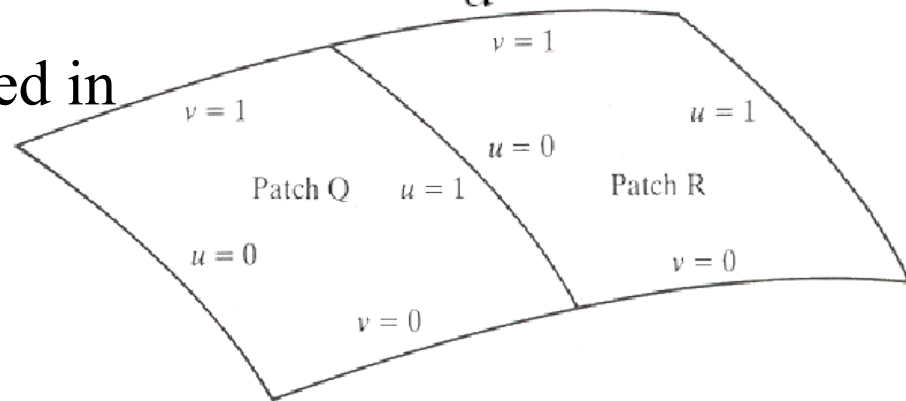
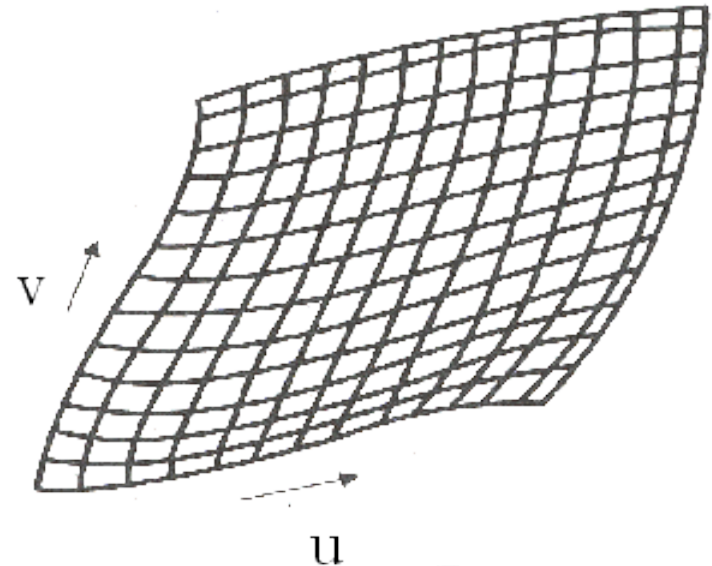
Courtesy T. Funkhouser,
Princeton University



Courtesy Russian Academy of
Sciences

BRep representation: patches

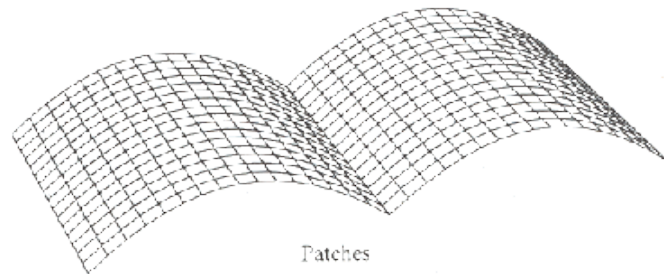
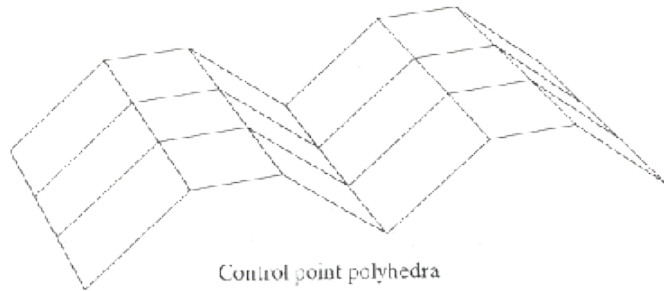
- Here the idea is to find families of piecewise parametric functions that allow a good control on shape
- Patches are joined at the edges so as to achieve the desired continuity
- Each patch is represented in parametric space



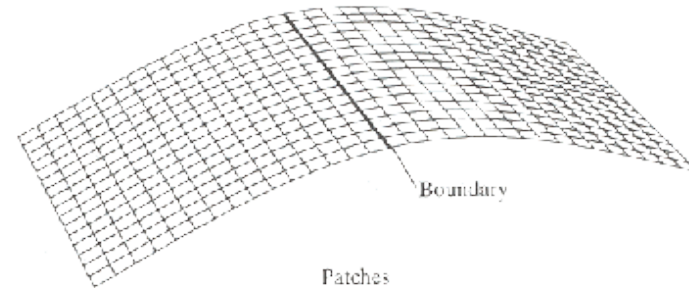
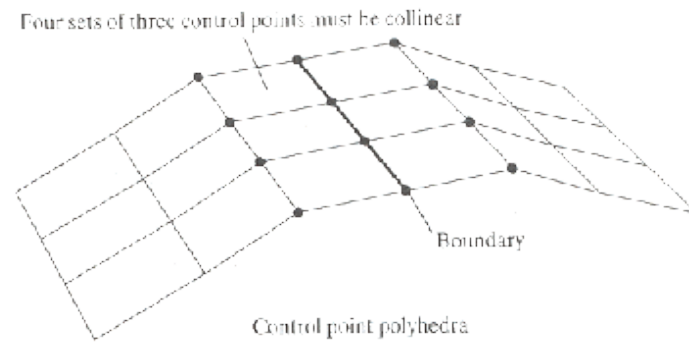
Courtesy T. Funkhouser,
Princeton University

BRep representation: patches

- C^0 continuity



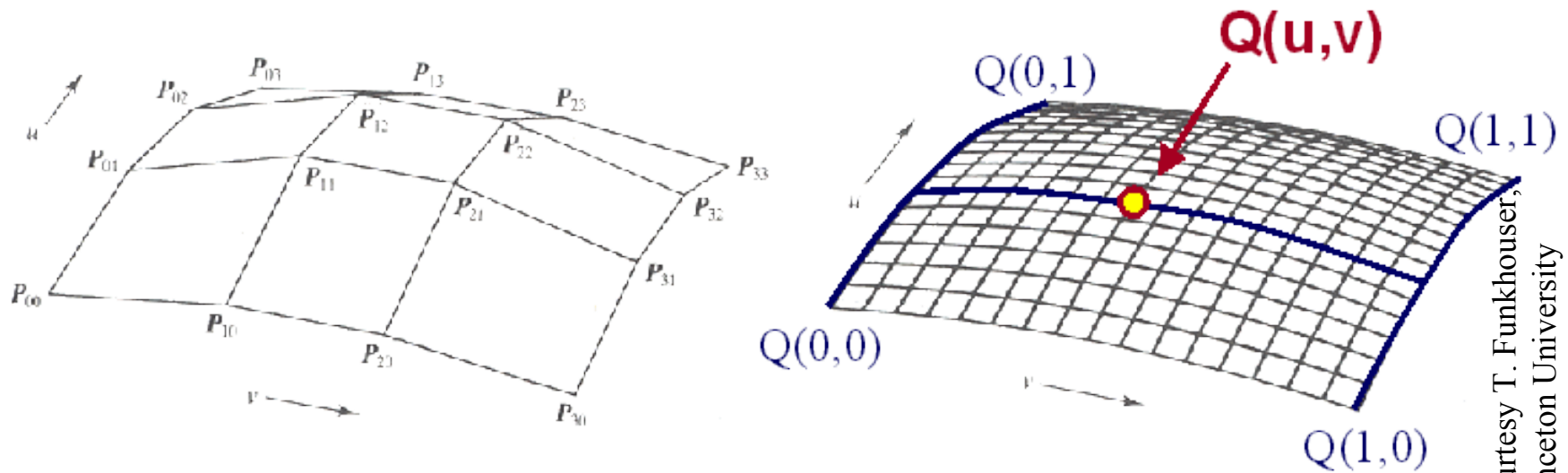
- C^1 continuity



Courtesy T. Funkhouser,
Princeton University

Spline patches

- A point Q on a patch is the tensor product of parametric functions defined by control points



Courtesy T. Funkhouser,
Princeton University

Spline patches

- A point Q on any patch is defined by multiplying control points by polynomial blending functions

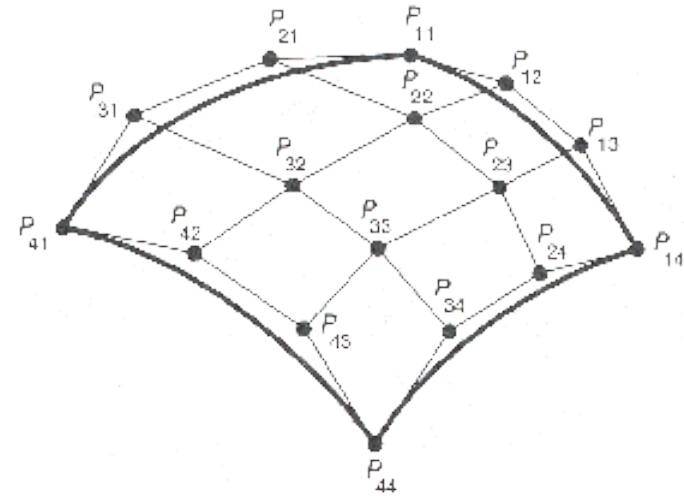
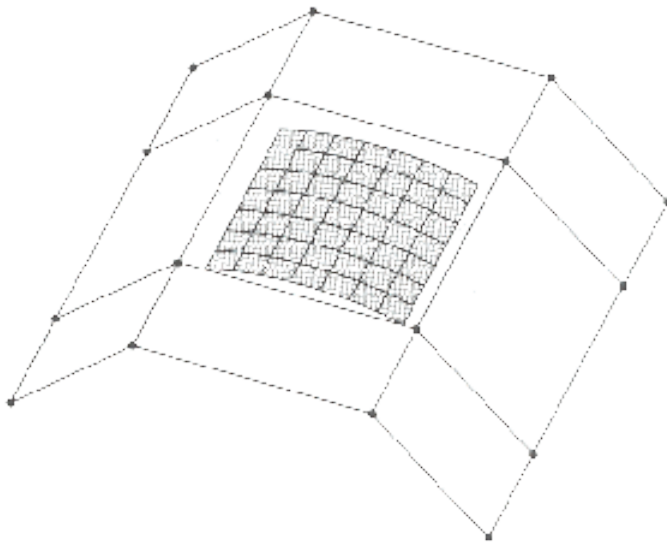
$$Q(u, v) = UM \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix} M^T V^T$$

- What about M then? M describes the blending functions for a parametric curve of third degree

Spline patches

$$M_{B-spline} = \begin{bmatrix} -1/6 & 1/2 & -1/2 & 1/6 \\ 1/2 & -1 & 1/2 & 0 \\ -1/2 & 0 & 1/2 & 0 \\ 1/6 & 2/3 & 1/6 & 0 \end{bmatrix}$$

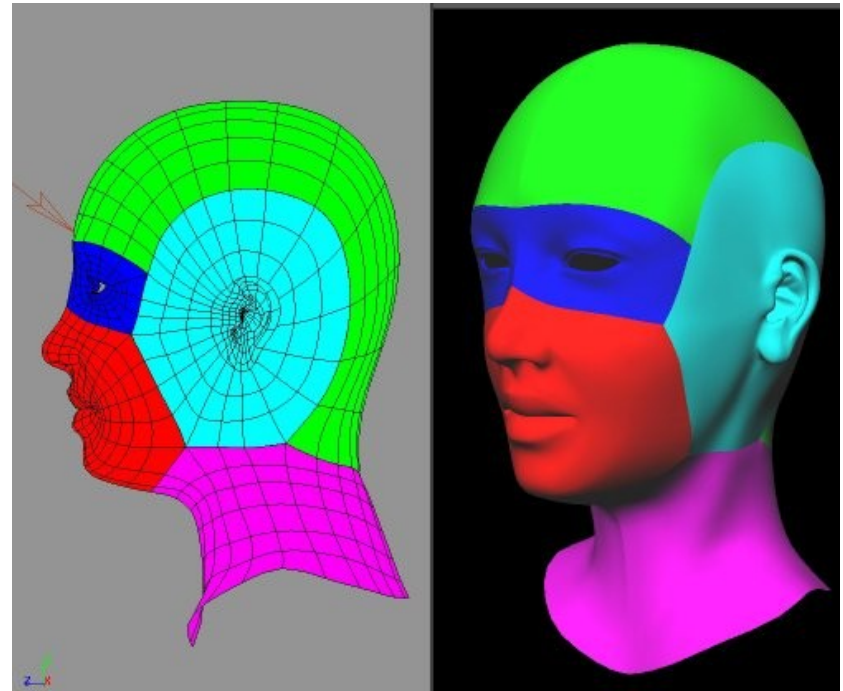
$$M_{Bezier} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$



Courtesy T. Funkhouser,
Princeton University

Spline patches

- Third order patches allow the generation of free form surfaces, and easy controllability of the shape
- Why third order functions?
 - Because they are the minimal order curves allowing inflection points
 - Because they are the minimal order curves allowing to control the curvature (= second order derivative)



Courtesy Softimage Co.

End



Copyright (c) 1988 ILM

+++ Ende - The end - Finis - Fin - Fine +++ Ende - The end - Finis - Fin - Fine +++