Introduction to Parametric Curves

J.C. Dill

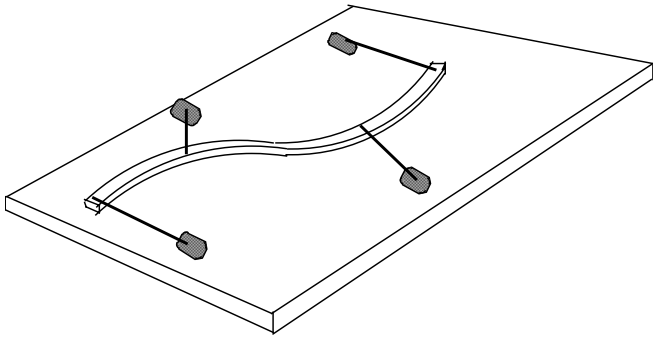
CurvSurf/Curves.doc 18Feb96

8Mar97: summary of Dec96 CGW article; 14feb99(cvt to pc) 27Oct00

References: Lee Ch 6
McMahon and Browne:
Zeid ch 5 section 4,6.1-3
Rogers & Adams, "Mathematical Elements for Computer Graphics"
Foley & van Dam, "Computer Graphics: Principles and Practices" Ch 11

A Physical Basis for Cubic Splines

A "spline" is a thin elastic beam, made out of plastic, wood or metal. It was used in drawing long smooth curves through a series of points, in the design of aircraft, ships and cars. The beam was positioned by attaching lead weights (called "ducks") to the beam at various points.



Euler's equation for a thin elastic beam is:

$$M(x) = EI/R(x)$$

where $M(x)$ = bending moment
E = Young's Modulus
I = moment of inertia
and $R(x)$ = radius of curvature

From elementary calculus:

$$1/R = y''/\left(1+y'^2\right)^{3/2} \approx y'' \text{ for } y' << 1 \text{ (small deflection)}$$

Substituting, we get $y'' = M(x) / EI$.

We also know from physics that M(x) is linear in x so we can write $M(x) = Ax + B$. Thus

$$y'' = (Ax + B)/EI$$

or

$$y = ax^3 + bx^2 + cx + d$$

That is, we have a cubic in x. This suggests that a cubic curve is a reasonable way to represent a smooth curve; ie that cubic polynomials are a reasonable for a mathematical spline. In general a mathematical spline is a piecewise polynomial of degree n, and has continuity of derivatives of order n-1 at the joints between segments (pieces). Thus a cubic should have C(2) continuity at the joints.

We use low degree polynomials both for computational reasons and because high degree polynomials tend to be unstable numerically. Since low degree poly's can only "span" a limited number of points, the technique we use is to join together, end-to-end fashion, as many curve segments aa are needed. Thus, continuity conditions at the endpoints of individual segments become important, in order to get sufficiently smooth joins.

Parametric form

A simple example is the 1st quadrant of the unit circle:

$$x = \cos\theta$$
$$y = \sin\theta \quad \text{for } 0 \le t \le 1$$

However parameterizations are not unique. For eg, letting t = tan(θ/2), we can derive

$$x = \left(1 - t^2\right) / \left(1 + t^2\right)$$
$$y = 2t / \left(1 + t^2\right) \quad 0 \le t \le 1$$

a second parametrization for the circle.

If we think of P(t) =(x(t), y)t)) as the path over time from 0 to 1, P' and P" are the velocity and acceleration. Different parameterizations however may give us different values for these. For the sine-cosine parametrization we observe:

$$P'(\theta) = (-\sin\theta, \cos\theta) \quad \text{and} \quad |P'(\theta)| = \text{sqrt}(\sin^2\theta + \cos^2\theta) = 1 \text{ everywhere}$$

but for the tan(θ/2) parametrization, we have

$$P'(t) = (-4t / (1 + t^2)2,\ 2(1 - t^2) / (1 + t^2)2) \text{ and } |P'(0)| = 2;\ |P'(1)| = 1$$

SO ... what functions should we use as the parametric functions?
We want functions x(t), y(t), z(t) that are easy to compute, flexible, and give us the shapes we need to cover.

- polynomials are a good class
- two types often used are the monomials or power basis, and Bezier basis.

The power basis is just $\{ u^i \}$, i = 0,1, ... We develop the Bezier basis below.

Parametric Cubic (Hermite) Curves

An algebraic form of the 3D (space) curve is simply: $x = x;$ $y = f(x);$ $z = g(x);$ What are the problems with this method? Some of these are solved with the implicit form: $f(x,y,z) = 0$. However the implicit representation also has drawbacks: what are they?

Let's look at power i=3, the cubic form, and develop a more intuitive, geometric way of dealing with the coefficients of the power basis. The Parametric form, **P**(t) = [*X(t) Y(t) Z(t)*] solves these problems and can be represented:

$$X(t) = a_x t^3 + b_x t^2 + c_x t + d_x$$
$$Y(t) = a_y t^3 + b_y t^2 + c_y t + d_y \qquad 0 \le t \le 1$$
$$Z(t) = a_z t^3 + b_z t^2 + c_z t + d_z \qquad \text{ie } t \text{ restricted to [0,1]}$$

Using matrix notation, we write

$$X(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \bullet \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}_x$$
$$= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \mathbf{C}_x$$
$$= \mathbf{T}\mathbf{C}_x$$

or

$$\mathbf{P}(t) = \mathbf{T}\,\mathbf{C}$$

The *tangent vector* is the (parametric) derivative of **P**(t):

$$\mathbf{P}'(t) = [X'(t) \;\; Y'(t) \;\; Z'(t)]$$
$$\text{where} \quad X'(t) = dX(t)/dt$$

Continuity Definition: Two curves which join at their end/start points have *geometric* continuity of order 0, written as $G^{(0)}$ continuity. If also the directions of the tangent vectors (but not necessarily the magnitudes) are equal, we say the curves have $G^{(1)}$ continuity. If the magnitudes are also the same they have $C^{(1)}$ or *parametric* continuity. The *tangent vector* is the velocity of a point on the curve with respect to the parameter *t*.

To determine a, b, c, and d, we convert to geometric form and match end points and slopes:

By inspection of the figure we see that

$X(0) = P_{1x}$ $\quad$ $X'(0) = R_{1x}$

$X(1) = P_{2x}$ $\quad$ $X'(1) = R_{2x}$

Then, since $X(t) = [\, t^3 \;\; t^2 \;\; t \;\; 1\,]\, C_x$ and $X'(t) = [3t^2 \;\; 2t \;\; 1 \;\; 0]\, C_x$,

we have that $X(0) = P_{1x} = [\, 0 \;\; 0 \;\; 0 \;\; 1\,]\, C_x$ $\quad$ $X'(0) = R_{1x} = [\, 0 \;\; 0 \;\; 1 \;\; 0]\, C_x$

$X(1) = P_{2x} = [\, 1 \;\; 1 \;\; 1 \;\; 1\,]\, C_x$ $\quad$ $X'(1) = R_{2x} = [\, 3 \;\; 2 \;\; 1 \;\; 0]\, C_x$

Combining all four, we get

$$\begin{bmatrix} P_1 \\ P_2 \\ R_1 \\ R_2 \end{bmatrix}_x = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \mathbf{C}_x$$

We simply invert the matrix to solve for $\mathbf{C}_x$:

$$\mathbf{C}_x = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ R_1 \\ R_2 \end{bmatrix}_x$$

$$= \mathbf{M}_h \mathbf{G}_{h_x}$$

where $\mathbf{M}_h$ = Hermite matrix, and

$\mathbf{G}_{h_x}$ = Hermite geometry matrix

Thus we have $X(t) = \mathbf{T}\, \mathbf{M}_h \mathbf{G}_{hx}$

$Y(t) = \mathbf{T}\, \mathbf{M}_h \mathbf{G}_{hy}$ $\quad$ or, combining, $\mathbf{P}(t) = \mathbf{T}\, \mathbf{M}_h \mathbf{G}_h$.

$Z(t) = \mathbf{T}\, \mathbf{M}_h \mathbf{G}_{hz}$

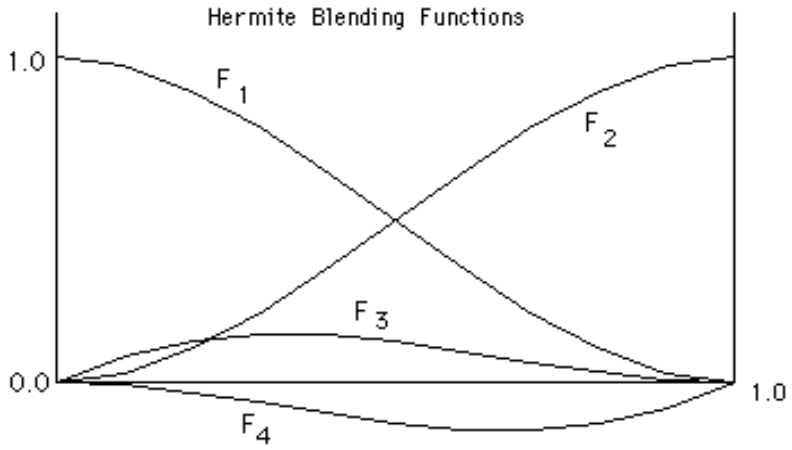
In other words, we have an expression giving the curve $\mathbf{P}(t)$ in terms of a parameter t and a geometry vector that has intuitive appeal.

Blending Functions

Let's gather terms that modify each of the geometric quantities:

$$
\begin{aligned}
X(t) &= \mathbf{T}\mathbf{M}_h\mathbf{G}_{h_x} \\
&= \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ R_1 \\ R_2 \end{bmatrix} \\
&= \left(2t^3 - 3t^2 + 1\right)P_{1_x} + \left(-2t^3 + 3t^2\right)P_{2_x} + \left(t^3 - 2t^2 + 1\right)R_{1_x} + \left(t^3 - t^2\right)R_{2_x} \\
&= \begin{bmatrix} F_1 & F_2 & F_3 & F_4 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ R_1 \\ R_2 \end{bmatrix}
\end{aligned}
$$

The functions F_i are called *Blending Functions* because they "blend" the geometric quantities P_1, P_2, R_1 and R_2 to produce the resultant curve. The Hermite blending functions are plotted in the figure below:

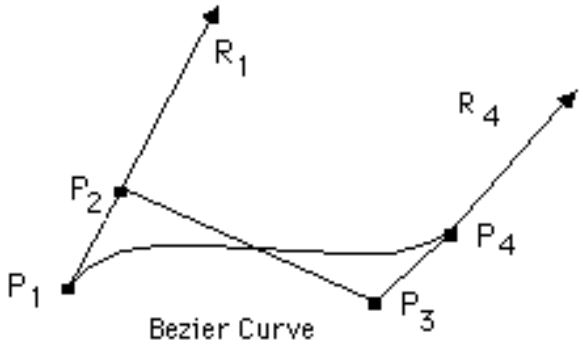


Observe that at P_1 we have $F_1(0) = 1$ and $F_2(0) = F_3(0) = F_2(0) = 0$ so that the curve passes through P_1. A similar argument shows it also passes through P_2.

Bezier Curves

We can develop these by modifying the Hermite form. The tangent vectors are defined in terms of intermediate points:

$$R_1 = 3(P_2 - P_1) \qquad R_4 = 3(P_4 - P_3)$$
$$= P'(0) \qquad = P'(1)$$



R1 = 3(P2 - P1) = P'(0) R4 = 3(P4 - P3) = P'(1)

Bezier curve passes through (interpolates) first, last points.

We note the Bezier curve passes through (interpolates) the first and last points. If we now write $\mathbf{G}_h$ in terms of the Bezier Geometry matrix, $\mathbf{G}_b$, we get

$$\mathbf{G}_h = \begin{bmatrix} P_1 \\ P_4 \\ R_1 \\ R_4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix} = \mathbf{M}_{hb}\mathbf{G}_b$$

Now since $X(t) = \mathbf{T}\,\mathbf{M}_h\,\mathbf{G}_{hx}$, we get

$$X(t) = \mathbf{T}\mathbf{M}_h\mathbf{M}_{hb}\mathbf{G}_{b_x}$$

so that letting $\mathbf{M}_b = \mathbf{M}_h\mathbf{M}_{hb}$ we get

$$X(t) = \mathbf{T}\mathbf{M}_b\mathbf{G}_{b_x}$$
$$= \mathbf{F}_b\mathbf{G}_{b_x}$$

where $\mathbf{F}_b$ is the Blending function vector (see plots below). The general form of a Bezier curve of degree n (with n+1 control vertices) is, writing $B_{n,i}$ for the i^{th} blending function,

$$X(t) = \sum_{i=0}^{n} P_{i_x} B_{n,i}(t) = \sum_{i=0}^{n} P_{i_x} \binom{n}{i} t^i (1-t)^{n-i}$$

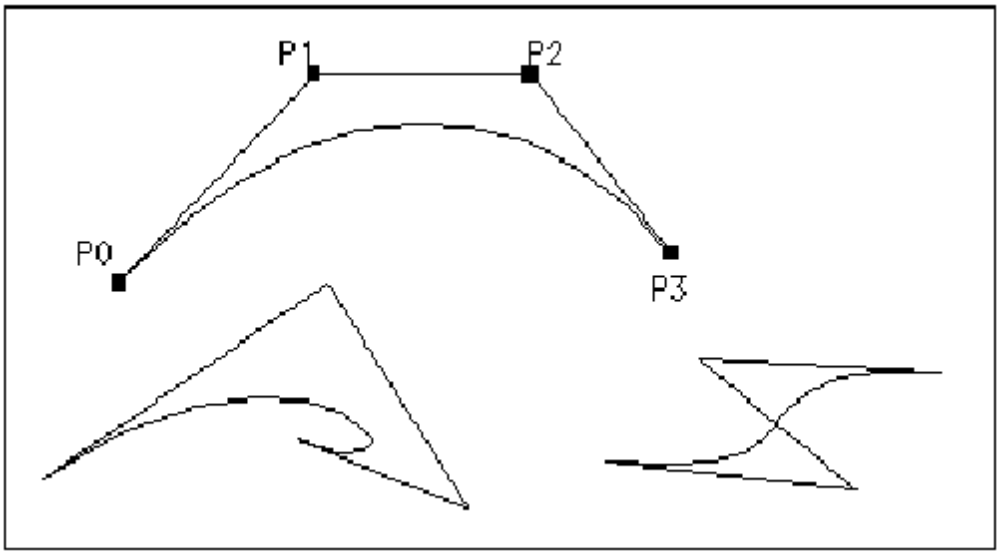
where $B_{n,i}$ is the i^{th} *Bernstein polynomial* of order n, and $\binom{n}{i} = \frac{n!}{i!(n-1)!}$.

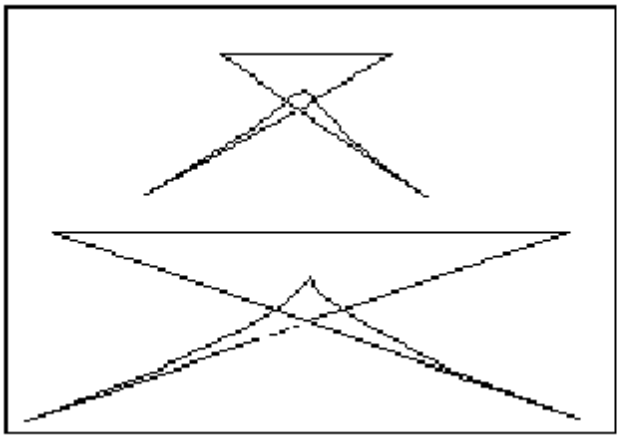
We note that the Bernstein functions sum to one and are everywhere positive (convex hull property[1], an interesting exercise to prove this!), ie

$$\sum_{i=0}^{n} B_{n,i}(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} = 1$$

This means we can think of the curve $Q(t) = [X(t)\ \ Y(t)]$ as the *weighted average* of the control points, weighted by the blending functions.

Here's some more examples:





[1]Convex Hull of a set of points is the polygon formed by putting a rubber band around points. The Convex Hull property of the Bspline curve is that the curve is contained within the convex hull of the control points.

Joining Bezier Curves

It's also interesting to see what's required to join Bezier curve segments with various degrees on continuity. For $C^{(0)}$, we just need the first point of segment two to be the same as the last point of segment one. Let's see what we need for $C^{(1)}$: consider two segments (in blending function form):

$$\mathbf{P}(t) = [(1-t)^3 \;\; 3t(1-t)^2 \;\; 3t^2(1-t) \;\; t^3\,] \bullet [P_0 \;\; P_1 \;\; P_2 \;\; P_3\,]^T$$

and

$$\mathbf{Q}(t) = [(1-t)^3 \;\; 3t(1-t)^2 \;\; 3t^2(1-t) \;\; t^3\,] \bullet [Q_0 \;\; Q_1 \;\; Q_2 \;\; Q_3\,]^T$$

For $C^{(0)}$ we need $Q_0 = P_3$. For $C^{(1)}$ we also need to look at the first derivatives:

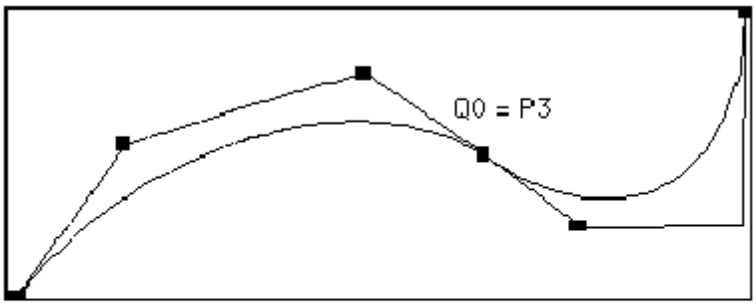
$\mathbf{P}'(t) = [-3(1-t)^2 \;\; 3(1-t)^2 - 6t(1-t) \;\; 6t(1-t) - 3t^2 \;\; 3t^2\,] \bullet [P_0 \;\; P_1 \;\; P_2 \;\; P_3\,]^T$

and similarly for $\mathbf{Q}'(t)$. Evaluating at the end of the first segment and at the start of the second:

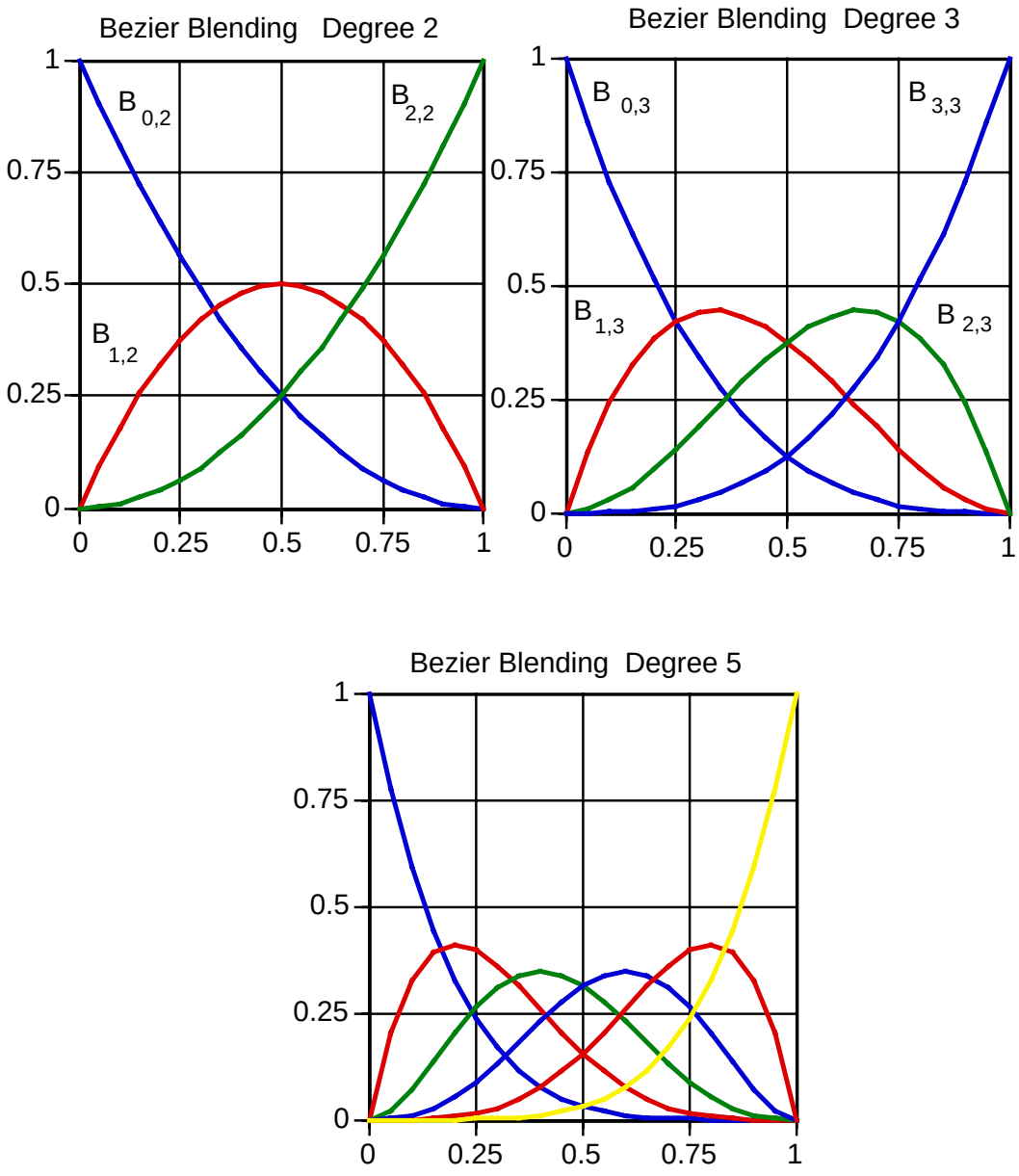
$$\mathbf{P}'(1) = 3(\mathbf{P}_3 - \mathbf{P}_2)$$

$$\mathbf{Q}'(0) = 3(\mathbf{Q}_1 - \mathbf{Q}_0)$$

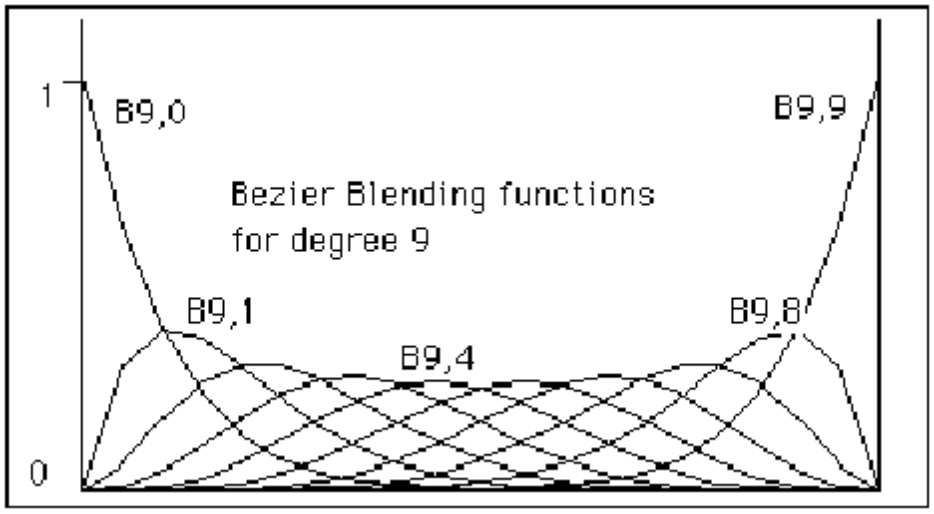
We get $C^{(1)}$ continuity if $\mathbf{P}'(1) = \mathbf{Q}'(0)$, i.e. if all three points, $\mathbf{P}_2$, $\mathbf{P}_3 = \mathbf{Q}_0$, and $\mathbf{Q}_1$ are co-linear as the following diagram illustrates.



For *curvature* continuity ($C^{(2)}$), we need *five* colinear points, which doesn't leave much freedom to move vertices for a cubic! This suggests for $C^{(2)}$ we should switch to *quintic* Bezier (and 6 points per segment).



Bezier Blending Functions of degree 2, 3, and 5



Degree Raising [optional]

We sometimes need to raise the accuracy of control of a Bezier curve. One way is to raise the degree of the curve, adding an appropriate number of control vertices. See Chiyokura for details. In outline:

To raise from degree n to n+1 we first set $P(t) = t\,P(t) + (1-t)\,P(t)$

Now $P(t) = \{ P0, P1, \ldots Pn \}$

Substituting and doing some algebra, we can show

$tP(t) = \{ 0, 1/(n+1)P0, 2/(n+1)P1, \ldots n/(n+1)Pn-1, Pn\}$

and

$(t-1)P(t) = \{ P0, n/(n+1)P1, (n-1)/(n+1)P2, \ldots 1/(n+1)Pn-1, 0\}$

Summing, we get

$Q(t) = \{ P0, (P0 + nP1)/(n+1), (2P1 + (n-1)P2)/(n+1), \ldots (nPn-1 + Pn-1)/(n+1), Pn \}$

for the new curve.

B-Spline Curves

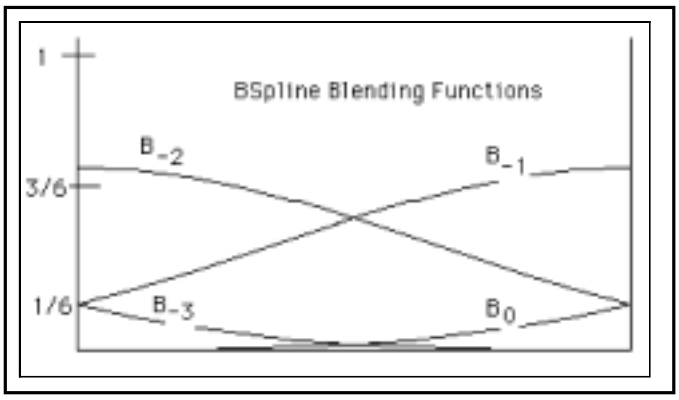
The B-spline is in a sense the smoothest of the three types considered so far. It has continuous tangents *and curvature*, but does not necessarily pass through *any* of the control points. It also exhibits the variation diminishing, or *local control* property, i.e. a control point only affects a limited portion of the curve. The equations are

$$X(t) = \mathbf{T}\,\mathbf{M}_s\,\mathbf{G}_{sx}$$

Where

$$M_s = \tfrac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

The Blending functions are shown in the sketch below; the "numbering" is quite different from Bezier etc.



Blending functions:

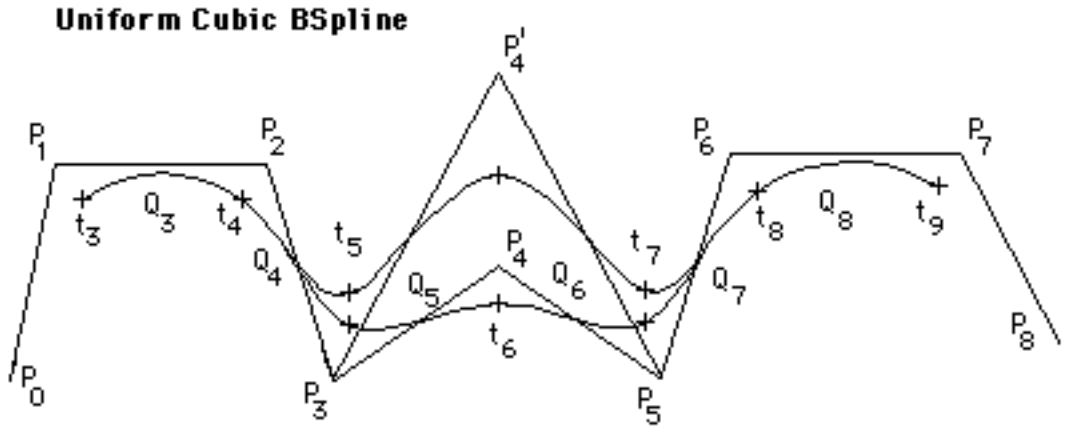
- non negative and sum = 1 => convex hull property
- can show $C^{(0)}$, $C^{(1)}$ and $C^{(2)}$ continuous

For a series of control points P_1, P_2, P_3, . . . P_n, a *sequence* of B-spline segments is used with a different geometry matrix between each pair of adjacent points. For P_i to P_{i+1} we have

$$G_{s_i} = \begin{bmatrix} P_{i-1} \\ P_i \\ P_{i+1} \\ P_{i-2} \end{bmatrix} \quad \text{for} \quad 2 \le i \le n-2$$

Example: A uniform cubic BSpline with 9 points (m=8). We show what happens when a vertex is moved:

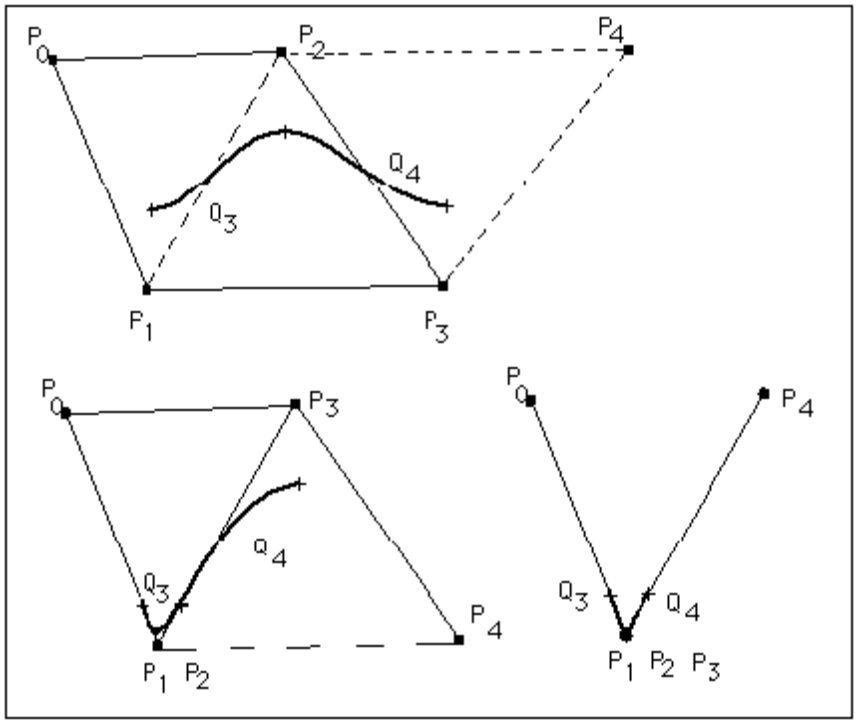
m+1 points	9 points	$P_0\ P_1\ P_2\ P_3\ P_4\ P_5\ P_6\ P_7\ P_8$
m-2 segments	6 segments	$Q_3\ Q_4\ Q_5\ Q_6\ Q_7\ Q_8$
m-1 knots	7 knots	$t_3\ t_4\ t_5\ t_6\ t_7\ t_8\ t_9$



For the uniform BSpline

- each Qi could be defined on its own $0 \le t_i < 1$ domain
- adjust parameter t so that $t = t_i + k$ so that parametric domains are sequential and Q_i is defined on $t_i \le t < t_{i+1}$
- knots defined at equal intervals of t; and we set $t_3 = 0$

Uniform B Spline: multiple control points:



NonUniform (NonRational) BSpline

Basic difference for non-uniform B-Spline: non uniform spacing between knot values For the *NonRational* Bspline, the curve is an ordinary simple polynomial (not a quotient of two poly's)

- advantage over uniform: can reduce continuity at a join from $C^{(2)}$ to $C^{(1)}$ to $C^{(0)}$ to 0
- if $C^{(0)}$, curve interpolates control vertices, but not forced to be a straight line on either side, as it is for *uniform* Bsplines
- Basic recurrence relation for BSplines:

$$N_{i,1}(t) = 1 \text{ for } t_i \le t < t_{i+1}$$
$$= 0 \text{ otherwise}$$

$$N_{i,n}(t) = \frac{(t - t_i)}{(t_{i+n-1} - t_i)} N_{i,n-1}(t) + \frac{(t_{i+n} - t)}{(t_{i+n-1} - t_{i+1})} N_{i+1,n-1}(t)$$

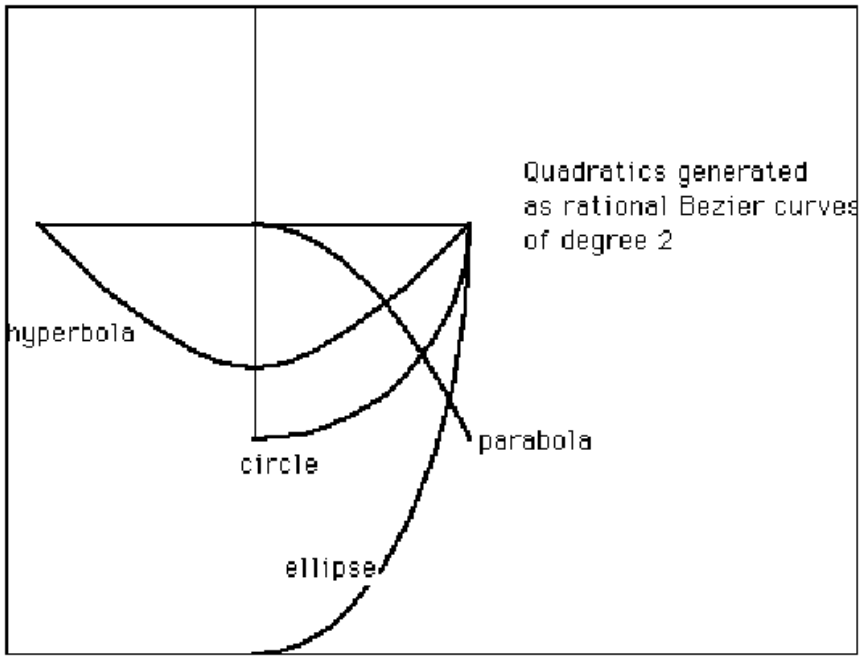
NonUniform Rational B-Splines

$x(t) = X(t)/W(t)$

- cubic polynomial with control points defined in homogenous coordinates, or
- curve is Q(t) = [X(t), Y(t), Z(t), W(t)] in homogenous space and we must project it back into ordinary 3 space to plot.

These are useful because:

- invariant under rotation, scale, translate, perspective
  - (Non-rational is invariant only under R, S, T)
  - thus perspective only needs be applied to control points!!
- can define quadrics exactly (Conic sections)
- very important for CAD



Comparison of Hermite, Bezier and B-spline curves

Hermite:

- passes through end points
- continuous tangents
- good for approximating existing points
- $C^{(1)}$ continuity
- reduce exactly to conic section
- poorly approximate asymptotic curves
- may have spurious oscillations unless carefully controlled
- cubic spline locally influenced by each data point
- 3rd derivative only piecewise continuous, thus discontinuities can induce unwanted inflection points. An oscillating curve may be $C^{(2)}$ continuos, but is not "fair".
- intuitive effect of varying tangent vector direction and magnitude
- lack of local control (change any point affects curve everywhere) (matrix inversion requires large effort for many points)

Bezier:

- passes through end points
- continuous tangents
- more intuitive for interactive adjustment
- convex hull property
- $C^{(1)}$ continuity

B-spline:

- need not pass through control points
- convex hull property
- variation diminishing (local control)
- $C^{(2)}$ continuity

A more advanced representation extends B-splines to allow exact representation of conics along with other advantages. It uses Non-uniform B-splines for more flexibility, and uses a rational form, thus the name: Non Uniform Rational B-Splines, or NURBS. An excellent introduction is on the Apple developers website:

http://devworld.apple.com/dev/techsupport/develop/issue25/schneider.html

Recent Articles

Smith, Susan. 1996. "The Challenges of Complex Surfaces", CGW Dec '96, pp27-35

A major problem is matching boundaries between surfaces. To avoid this, the European approach is to try to model a whole fender, for eg, with a single 21st order equation. The Japanese try to use very large numbers of very small 1st order (4 point) patches. Continuity of joins is key; they measure curvature continuity as the ability to pass bands o flight across the surface without disruption to the light bands.

Vendors

Top 5 in MCAD (1997):

- PTC (Pro/Engineer); uses CDRS developed originally by E&S
- IBM/Dassault Systemes (CATIA),
- Computer Vision (CADDS 5),
- SDRC (IDEAS)
- EDS (Unigraphics); UGS claims to deal with up to order 24

Others include Matra/Datavision, which uses the code from Cisigraph that uses uniform polynomials to order 21.