# Varying the Betas in Beta-splines

*Brian A. Barsky*

Berkeley Computer Graphics Laboratory
Computer Science Division
University of California
Berkeley, California 94720 USA
(415) 642-9838

*John C. Beatty*

Computer Graphics Laboratory
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
(519) 886-1351

University of California, Berkeley TR CSD 82/112
University of Waterloo TR CS-82-49
December 1982

## *ABSTRACT*

The Beta-spline introduced recently by Barsky is a generalization of the uniform cubic B-spline: parametric discontinuities are introduced in such a way as to preserve continuity of the unit tangent and curvature vectors at joints (*geometric continuity*) while providing bias and tension parameters, independent of the position of control vertices, by which the shape of a curve or surface can be manipulated. We introduce a practical method by which different values of each can be specified at every joint. This involves the use of a special form of quintic Hermite interpolation to yield values of the bias and tension at each point along a curve, the actual position being determined by substituting these values into the equations for a uniformly-shaped Beta-spline. We explore the properties of the resulting piecewise polynomial curves and surfaces. An important characteristic is their local response when either the position of a control vertex or the value of a shape parameter is altered.

There is also a conceptually simple and obvious way to directly generalize the equations defining the uniformly-shaped Beta-splines so that each shape parameter may have a distinct value at every joint. Unfortunately, the curves which result lack many desirable properties.

Keywords: Beta-splines, computer aided design, geometric continuity, polynomial splines, tension.

---

This work was supported by funds from the United States National Science Foundation (Grant ECS-82-04381), the Defense Advanced Research Projects Agency of the United States (Contract N00039-82-C-0235), the Natural Sciences and Engineering Research Council of Canada (Grants A3022 & G0651), and the National Research Council of Canada (Contract OSU81-00439).

## 1. Introduction

In the last decade it has become apparent that the use of straight line segments and planar polygons to approximate curved lines and surfaces has limited the state-of-the-art in computer graphics. Even with the most sophisticated continuous shading models, polygonal techniques generally result in visually objectionable images. Mach bands are apparent at the borders between adjacent polygons, and there is always a telltale jaggedness to polygonal silhouettes. Also, polygonal methods often require excessive amounts of storage and the "resolution" at which a polygonal database is stored is fixed, independent of the eventual display, as opposed to curved surface techniques in which the resulting image can be computed to whatever level of detail the situation demands.

Early work by Coons [Coons64a, Coons67a] and Bézier [Bézier70a, Bézier77a] introduced the use of nonlinear parametric polynomial representations for the *segments* and *patches* which are stitched together to form such *piecewise* curves and surfaces, establishing their viability. More recently, Riesenfeld [Riesenfeld73a, Gordon74a] has advocated the use of B-splines to represent such polynomials on the grounds of greater flexibility and efficiency.

Parametric B-splines have many advantages. Among them is the ability to control the degree of continuity at the joints between adjacent curve segments, and at the borders between surface patches, independent of the order of the segments or the number of control vertices. However, the notion of parametric first or second degree continuity at joints does not always correspond to intuition or to a physically desired effect. For piecewise cubic curves and bicubic surfaces these parametric continuity constraints can be replaced by the more meaningful requirements of continuous unit tangent and curvature vectors. Doing so introduces certain constrained discontinuities in the first and second parametric derivatives. These are expressed in terms of *bias* and *tension* parameters called $\beta_1$ and $\beta_2$ in [Barsky81a], and give rise to *Beta-spline* curves and surfaces.

The application of tension to the cubic interpolatory spline was first analytically modeled in [Schweikert66a]. An alternative development was given in [Cline74a] and generalized in [Pilcher73a]. A detailed derivation of the generalized form based on a variational principle is given in [Barsky82a]. The result of this approach is a spline curve which is no longer piecewise *polynomial*, but piecewise *exponential*; that is, each curve segment is expressed in terms of exponential functions.

From a computational standpoint, however, polynomial functions are much more desirable than exponentials. For this reason Nielson developed the Nu-spline, a polynomial alternative to the exponential spline under tension [Nielson74a, Nielson74b]. It is derived in detail in [Barsky82a] from the cubic Hermite basis functions, thereby emphasizing its relation to the conventional cubic interpolatory spline.

Unfortunately, neither of the above-mentioned interpolating spline curve representations provide *local control*; that is, the capability of modifying one portion of the curve without altering the remainder. Local control is inherent in the B-spline formulation [Riesenfeld73a, Barsky82b], and thus this would be a good starting representation upon which to apply tension. In an approach that preserved the variation diminishing property of the B-spline scheme [Barsky82b], Lane experimented with this idea by adding knots to a nonuniform B-spline curve in the region of desired tension [Lane77a].

The Beta-spline, then, is a new curve and surface representation having an inherent capability to model tension and containing the uniform cubic B-splines as a special case. This representation generalizes previous work on the mathematical modeling of tension insofar as it is based on the *shape parameters* $\beta_1$ and $\beta_2$.

The basic theory of Beta-splines was developed in [Barsky81a, Barsky82c]. This theory was expressed in terms of shape parameters which could either be uniform or varying throughout the curve or surface being defined. For the purposes of computer aided geometric design, the ability to alter shape

parameters is particularly useful because it provides an additional means of manipulating a curve or surface. In this paper we will extend previous work on varying shape parameters.

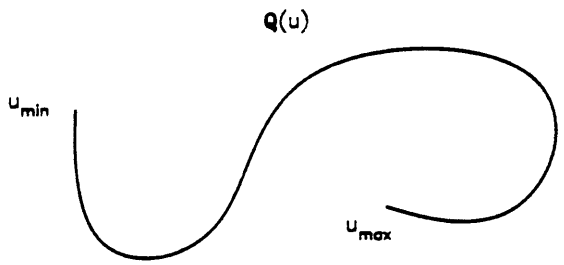
We begin in the next two sections with a review of pertinent terminology, notation and definitions. In section 4 we will quickly review the fundamental definition of a Beta-spline from [Barsky81a, Barsky82c]. This will lead naturally to a discussion of two methods for varying the $\beta$ parameters in a Beta-spline curve, one of which is of substantial practical interest. We will also discuss a means of defining Beta-spline surfaces in which the shape parameters can be locally altered.

## 2. Preliminaries

It is usually convenient to represent a two-dimensional curve parametrically as

$$\mathbf{Q}(u) = ( X(u), Y(u) )$$

where X(u) and Y(u) are single-valued functions of the parameter u which yield the x- and y-coordinates, respectively, of a point on the curve in question for any value of u.




Although polynomials are computationally efficient and easy to work with, it is not usually possible to define a satisfactory curve using single polynomials for X(u) and Y(u). Instead it is customary to break the curve into some number $m$ of pieces called *segments*, each defined by a separate polynomial, and hook the segments together to form a *piecewise polynomial* curve. The parameter u then varies between some initial minimum value $u_0$ and some final maximum value $u_m$ as we move along the curve; the values of u which correspond to joints between segments are called *knots*. Usually we write a sequence of knot values in nondecreasing order as

$$u_{min} = u_0 \leq u_1 \leq \cdots \leq u_m = u_{max}$$

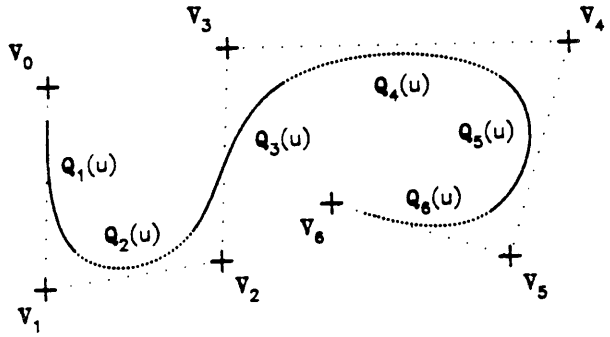
so that the parametric functions X(u) and Y(u) are each composed of m polynomial segments, the first covering the interval of u ranging from $u_0$ to $u_1$, the second covering values from $u_1$ to $u_2$, and so on. Usually X(u) and Y(u) are required to satisfy some continuity constraints at the joints between successive polynomial segments; if the $0^{th}$ through $d^{th}$ derivatives are everywhere continuous (in particular, at the joints), then X and Y are said to be $C^d$ continuous.

For our purposes it will be convenient to assume that the knots are a consecutive sequence of integers, with $u_i = i$; this is called a *uniform knot sequence*.

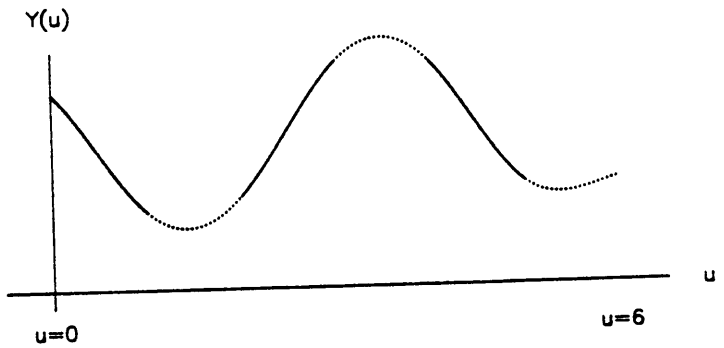
Also, it will usually be simpler when discussing the $i^{th}$ segment to write down $X(u-u_i)$ rather than X(u), and we shall generally do so; the re-parametrization is easily accomplished by substitution. Thus for the segment between $u_i$ and $u_{i+1}$ we might write $Y(u) = u^2$, so that $u = 0$ corresponds to the left end of the segment and $u = 1$ to the right end, rather than

$$Y(u) = (u-u_i)^2 = u^2 - 2u_iu + u_i^2$$

There are a variety of ways in which to actually define a specific curve. We will be concerned with techniques in which the user provides a number of *control vertices*, near which the curve is to pass. For example:



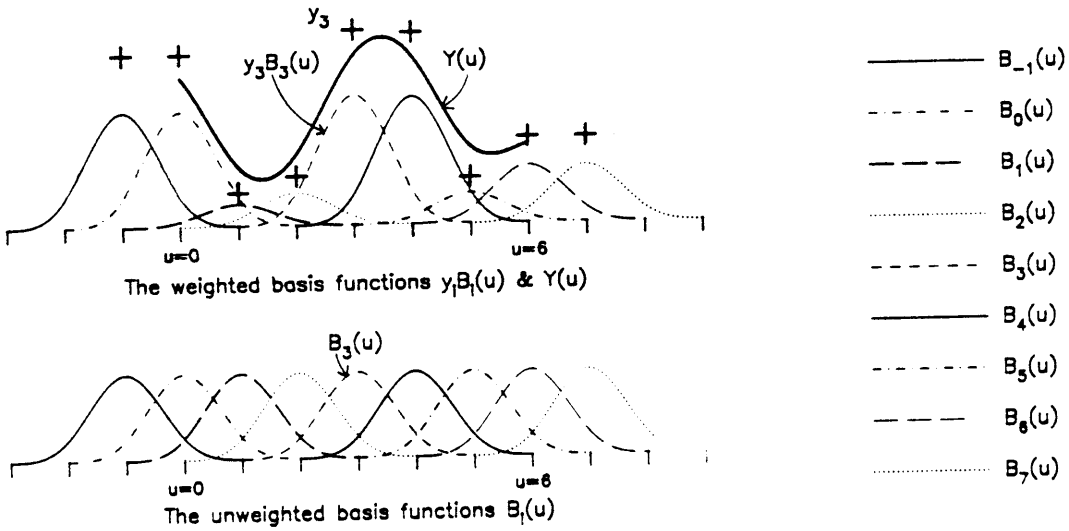

Moving the control vertices then alters the curve. It is easiest to explain this process by considering the definition of an individual coordinate such as Y(u), with X(u) being defined analogously. Thus Y(u) in the previous figure looks like this when plotted against u.




Let us denote the control vertices by $\mathbf{V}_i = (x_i, y_i)$. Generically one writes

$$X(u) = \sum_i x_i B_i(u) \qquad \text{and} \qquad Y(u) = \sum_i y_i B_i(u) \tag{E1}$$

so that Y(u), for example, is a weighted sum of the $B_i$: (For reasons explained in section 5.5 vertices $\mathbf{V}_0$ and $\mathbf{V}_6$ appear twice in the following figure.)




In vector notation this becomes

$$\mathbf{Q}(u) = \sum_i \mathbf{V}_i B_i(u) = \sum_i ( x_i B_i(u), y_i B_i(u) ) \qquad \text{(E2)}$$

The functions $B_i(u)$ are called *basis functions*. One generally chooses to work with some class of basis functions because of particular properties it possesses, or imparts to the curves it can be used to define.

For example: when manipulating a curve it is often desirable that the position of each control vertex affect only a limited portion of the curve. Such *local control* makes it possible to change one part of a curve without altering other portions of the curve whose design is already satisfactory. To obtain local control it is sufficient if each $B_i(u)$ is nonzero only for a small range of u. (This was the case for the basis functions appearing in the figure above.)

In summary, then, we have decomposed the problem of defining a 2D curve $\mathbf{Q}(u)$ into the problems of defining individual coordinate functions X(u) and Y(u); for 3D curves we simply add a third coordinate function Z(u), which is handled in exactly the same way as the first two. The coordinate functions are themselves weighted sums of basis functions, where the weights are the coordinates of control vertices. Our principle objective is to define appropriate basis functions, and to analyze the properties of those basis functions and the curves they define.

## 3. Uniform Cubic B-Splines

The Beta-splines we will introduce shortly are a generalization of the "uniform cubic B-splines," and it is both natural and informative to begin with a brief development of the uniform cubic B-spline basis functions.[1]

[1] Our definition focuses on those properties of the cubic B-splines which are most pertinent to our development of the Beta-splines. A more general definition of the B-splines in terms of divided differences, from which their properties are most easily derived, may be found in [deBoor78a]. For computational purposes they are also sometimes defined directly by the recurrence relation usually used to evaluate them [Cox72a, deBoor72a], which is also derived in [deBoor78a].

The B-spline curves which we are now considering are assembled from cubic polynomials which have positional, first derivative and second derivative continuity at the joints between successive segments, so that they satisfy the equations

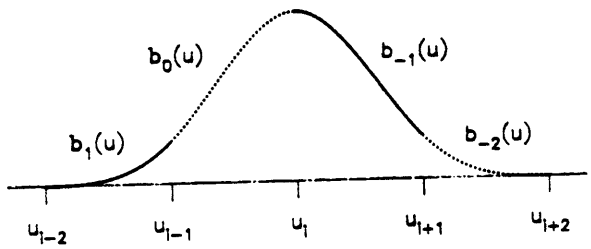
$$\mathbf{Q}_{i-1}(u_i) = \mathbf{Q}_i(u_i) \tag{E3}$$

$$\mathbf{Q}_{i-1}^{(1)}(u_i) = \mathbf{Q}_i^{(1)}(u_i) \tag{E4}$$

$$\mathbf{Q}_{i-1}^{(2)}(u_i) = \mathbf{Q}_i^{(2)}(u_i) \tag{E5}$$

where $\mathbf{Q}_i(u)$ is the portion of the curve $\mathbf{Q}(u)$ we are defining which lies between knots $u_{i-1}$ and $u_i$.

We could achieve the desired continuity if the basis functions with which we define the curve were themselves $C^2$ continuous piecewise cubic polynomials with knots at the $u_i$, since a weighted sum of such basis functions will also be a $C^2$ continuous piecewise cubic polynomial. Locality can be obtained if all but a small number of the polynomial segments defining a basis function are identically zero.




Suppose, then, that each segment of the basis function is a cubic polynomial of the form

$$a + bu + cu^2 + du^3$$

having four coefficients. If the nonzero portion of our cubic B-spline basis function B(u) consists (from right to left) of four such *basis segments* $b_{-2}(u)$, $b_{-1}(u)$, $b_0(u)$ and $b_1(u)$, then there are sixteen coefficients to determine. By assumption B(u) is identically zero for $u \le u_{i-2}$ and for $u \ge u_{i+2}$, so the first and second derivatives $B^{(1)}(u)$ and $B^{(2)}(u)$ are also identically zero outside the interval $(u_{i-2},u_{i+2})$. The requirement that positions, first derivatives, and second derivatives match at each knot $u_i$ then implies that

$$\begin{array}{lll}
0 = b_1(0) & 0 = b_1^{(1)}(0) & 0 = b_1^{(2)}(0) \\
b_1(1) = b_0(0) & b_1^{(1)}(1) = b_0^{(1)}(0) & b_1^{(2)}(1) = b_0^{(2)}(0) \\
b_0(1) = b_{-1}(0) & b_0^{(1)}(1) = b_{-1}^{(1)}(0) & b_0^{(2)}(1) = b_{-1}^{(2)}(0) \\
b_{-1}(1) = b_{-2}(0) & b_{-1}^{(1)}(1) = b_{-2}^{(1)}(0) & b_{-1}^{(2)}(1) = b_{-2}^{(2)}(0) \\
b_{-2}(1) = 0 & b_{-2}^{(1)}(1) = 0 & b_{-2}^{(2)}(1) = 0
\end{array} \tag{E6}$$

where for simplicity each segment is individually parametrized so that $u = 0$ corresponds to its left endpoint and $u = 1$ corresponds to its right endpoint. These comprise fifteen constraints. We additionally require that

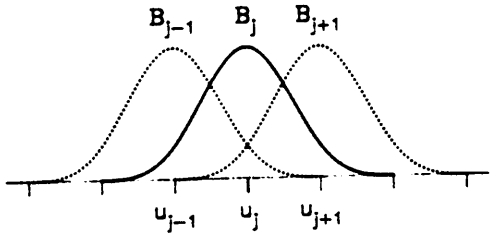
$$b_1(0) + b_0(0) + b_{-1}(0) + b_{-2}(0) = 0$$

Because $b_1(0) = 0$ this simplifies to

$$b_0(0) + b_{-1}(0) + b_{-2}(0) = 1$$

Since our knots are equally spaced, this amounts to assuming that when we add together a weighted

sequence of basis functions $B_j$, each of which is a copy of B shifted so as to be centered at $u_j$, the three basis functions $B_{j-1}$, $B_j$ and $B_{j+1}$ which are nonzero at $u_j$ sum to one. Such an assumption is said to be a *normalizing condition* and serves to define the function B(u) uniquely.




We now have sixteen equations in sixteen unknowns (that's why we assumed that our basis function had four cubic segments), and we may solve for the coefficients a, b, c and d of the four segments $b_1$, $b_0$, $b_{-1}$, and $b_{-2}$ comprising our basis function B. Doing so yields the polynomials

$$b_1(u) = \frac{1}{6} u^3 \tag{E7}$$

$$b_0(u) = \frac{1}{6} ( 1 + 3u + 3u^2 - 3u^3 )$$

$$b_{-1}(u) = \frac{1}{6} ( 4 - 6u^2 + 3u^3 )$$

$$b_{-2}(u) = \frac{1}{6} ( 1 - 3u + 3u^2 - u^3 )$$

These four segments define the *uniform cubic B-spline basis function;* again, the term *uniform* means that the knots are equally spaced. To determine a curve, we select a set of control vertices $\mathbf{V}_i$ and use them to define the curve

$$\mathbf{Q}(u) = \sum_i \mathbf{V}_i B_i(u) = \sum_i ( x_i B_i(u), y_i B_i(u) ) \tag{E8}$$

in which each $B_i$ is simply a copy of B, shifted so that it peaks at $u = u_i$, and the weighting is given by the control vertices

$$\mathbf{V}_i = ( x_i, y_i )$$

Notice that because the basis functions are nonzero on only four successive intervals, if $u_{i-1} \le u \le u_i$ then

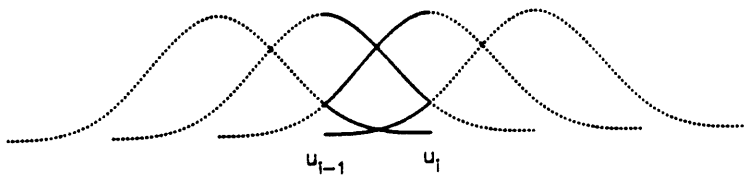
$$\mathbf{Q}_i(u) = \sum_{r=-2}^{r=+1} \mathbf{V}_{i+r} B_{i+r}(u) = \mathbf{V}_{i-2} B_{i-2}(u) + \mathbf{V}_{i-1} B_{i-1}(u) + \mathbf{V}_i B_i(u) + \mathbf{V}_{i+1} B_{i+1}(u) \tag{E9}$$

If we replace each basis function $B_j(u)$ by the particular segment which pertains to the interval $[u_{i-1}, u_i)$, then (E9) can be written as

$$\mathbf{Q}_i(u) = \sum_{r=-2}^{r=+1} \mathbf{V}_{i+r} b_r(u) = \mathbf{V}_{i-2} b_{-2}(u) + \mathbf{V}_{i-1} b_{-1}(u) + \mathbf{V}_i b_0(u) + \mathbf{V}_{i+1} b_1(u) \tag{E10}$$

Notice that the segments of our basis function are numbered from right to left because that is the order in which they appear when summed to form a curve: the leftmost control vertex weights the rightmost basis segment, and so on. Equation (E10) also reflects the convenience of parametrizing each basis segment

from u = 0 at its left end; since the basis functions are all translates of one another, this convention allows us to use the same formulas in defining each basis function, and hence in computing each curve segment.




The four basis functions which are nonzero on the $i^{th}$ interval

## 4. Uniformly-Shaped Beta-Splines

The details of what follows may be found in [Barsky81a].

The *unit tangent vector* of a curve $\mathbf{Q}(u)$ is

$$\hat{\mathbf{T}}(u) = \frac{\mathbf{Q}^{(1)}(u)}{|\mathbf{Q}^{(1)}(u)|} \tag{E11}$$

and the *curvature vector* is

$$\mathbf{K}(u) = \kappa(u)\hat{\mathbf{N}}(u) = \kappa(u)\frac{\hat{\mathbf{T}}^{(1)}(u)}{|\hat{\mathbf{T}}^{(1)}(u)|}$$

where $\kappa(u)$ is the curvature of $\mathbf{Q}$ at u and $\hat{\mathbf{N}}(u)$ is a unit vector pointing from $\mathbf{Q}(u)$ towards the center of the osculating circle at $\mathbf{Q}(u)$.[3] $\hat{\mathbf{T}}(u)$ and $\mathbf{K}(u)$ capture the physical meaningful notions of the direction and curvature at a point on the curve. It is shown in [Barsky81a, Barsky82c] that

$$\mathbf{K}(u) = \frac{\mathbf{Q}^{(1)}(u) \times \mathbf{Q}^{(2)}(u) \times \mathbf{Q}^{(1)}(u)}{|\mathbf{Q}^{(1)}(u)|^4} \tag{E12}$$

Of course, $\mathbf{Q}(u)$, $\hat{\mathbf{T}}(u)$ and $\mathbf{K}(u)$ are easily seen to be continuous away from the joints of a piecewise polynomial; from the equations (E11) and (E12) it is possible to show that in order for $\mathbf{Q}(u)$, $\hat{\mathbf{T}}(u)$ and $\mathbf{K}(u)$ to also be continuous at the joint between two consecutive curve segments $\mathbf{Q}_{i-1}(u)$ and $\mathbf{Q}_i(u)$ of $\mathbf{Q}(u)$, which we call $G^2$ or *second degree geometric continuity,* it is sufficient that

$$\mathbf{Q}_{i-1}(1) = \mathbf{Q}_i(0) \tag{E13}$$

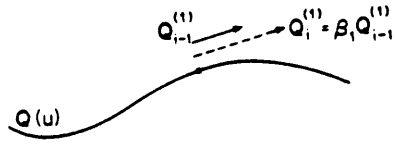
$$\beta_1 \mathbf{Q}_{i-1}^{(1)}(1) = \mathbf{Q}_i^{(1)}(0) \tag{E14}$$

$$\beta_1^2 \mathbf{Q}_{i-1}^{(2)}(1) + \beta_2 \mathbf{Q}_{i-1}^{(1)}(1) = \mathbf{Q}_i^{(2)}(0) \tag{E15}$$

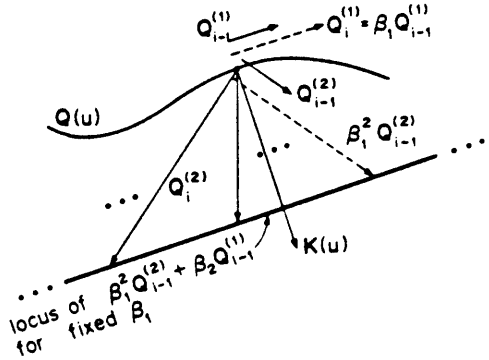
at every knot $u_i$ and for any $\beta_1$ and $\beta_2$. These equations are, by definition, less restrictive than simple continuity of position and parametric derivatives, which is the special case in which $\beta_1 = 1$ and $\beta_2 = 0$.

Equation (E13) simply enforces positional continuity. Equation (E14) requires that the first parametric derivative vectors from the left and right at a joint be colinear, but allows their magnitudes to differ. There is an instantaneous change in velocity at the joint, but not a change in direction.

[3] The *osculating circle* at Q(u) is the circle whose first and second derivative vectors agree with those of Q at u. The curvature $\kappa(u)$ is then the reciprocal of the radius of this osculating circle.




Equation (E15) reflects the fact that a sufficient condition for curvature continuity is that $\mathbf{Q}_i^{(2)}(0) = \beta_1^2 \mathbf{Q}_{i-1}^{(2)}(1)$. However, $\mathbf{Q}_i^{(2)}(0)$ may have an additional component directed along the tangent since acceleration along the tangent does not "deflect" a point traveling along the curve, and so does not affect the curvature there.




Let us again consider a basis function composed of four cubic polynomial basis segments, as we did for the uniform cubic B-splines, but this time we ask that they satisfy the geometric constraints (E13), (E14) and (E15) instead of the parametric constraints (E3), (E4) and (E5). The equations which result are

$$\begin{aligned} 0 &= b_1(0) & 0 &= b_1^{(1)}(0) \\ b_1(1) &= b_0(0) & \beta_1 b_1^{(1)}(1) &= b_0^{(1)}(0) \\ b_0(1) &= b_{-1}(0) & \beta_1 b_0^{(1)}(1) &= b_{-1}^{(1)}(0) \\ b_{-1}(1) &= b_{-2}(0) & \beta_1 b_{-1}^{(1)}(1) &= b_{-2}^{(1)}(0) \\ b_{-2}(1) &= 0 & \beta_1 b_{-2}^{(1)}(1) &= 0 \end{aligned} \qquad \text{(E16)}$$

$$\begin{aligned} 0 &= b_1^{(2)}(0) \\ \beta_1^2 b_1^{(2)}(1) + \beta_2 b_1^{(1)}(1) &= b_0^{(2)}(0) \\ \beta_1^2 b_0^{(2)}(1) + \beta_2 b_0^{(1)}(1) &= b_{-1}^{(2)}(0) \\ \beta_1^2 b_{-1}^{(2)}(1) + \beta_2 b_{-1}^{(1)}(1) &= b_{-2}^{(2)}(0) \\ \beta_1^2 b_{-2}^{(2)}(1) + \beta_2 b_{-2}^{(1)}(1) &= 0 \end{aligned}$$

To obtain sixteen equations we again require, for the same reason, that

$$\begin{aligned} \sum_{r=-2}^{r=+1} B_{i+r}(u) &= b_1(0) + b_0(0) + b_{-1}(0) + b_{-2}(0) \\ &= b_0(0) + b_{-1}(0) + b_{-2}(0) = 1 \end{aligned}$$

yielding a total of sixteen equations in sixteen unknowns. For any particular values of $\beta_1$ and $\beta_2$ these

equations can be solved numerically (as in the B-spline case) to obtain explicit formulae for the polynomials comprising the basis segments. This is not very practical, however, since we do not want to solve a new system every time we wish to alter one of the $\beta$ parameters. Instead we can solve this system symbolically, using Vaxima [Bogen77a, Fateman82a], to obtain the following symbolic representation of the basis segments for all values of $\beta_1$ and $\beta_2$.[4]

$$b_1(u) = \frac{1}{\delta}\left\{ 2u^3 \right\} \tag{E17}$$

$$b_0(u) = \frac{1}{\delta}\left\{ 2 + (6\beta_1)u + (3\beta_2 + 6\beta_1^2)u^2 - (2\beta_2 + 2\beta_1^2 + 2\beta_1 + 2)u^3 \right\}$$

$$b_{-1}(u) = \frac{1}{\delta}\left\{ (\beta_2 + 4\beta_1^2 + 4\beta_1) + (6\beta_1^3 - 6\beta_1)u - (3\beta_2 + 6\beta_1^3 + 6\beta_1^2)u^2 + (2\beta_2 + 2\beta_1^3 + 2\beta_1^2 + 2\beta_1)u^3 \right\}$$

$$b_{-2}(u) = \frac{1}{\delta}\left\{ (2\beta_1^3) - (6\beta_1^3)u + (6\beta_1^3)u^2 - (2\beta_1^3)u^3 \right\}$$

where

$$\delta = \beta_2 + 2\beta_1^3 + 4\beta_1^2 + 4\beta_1 + 2$$

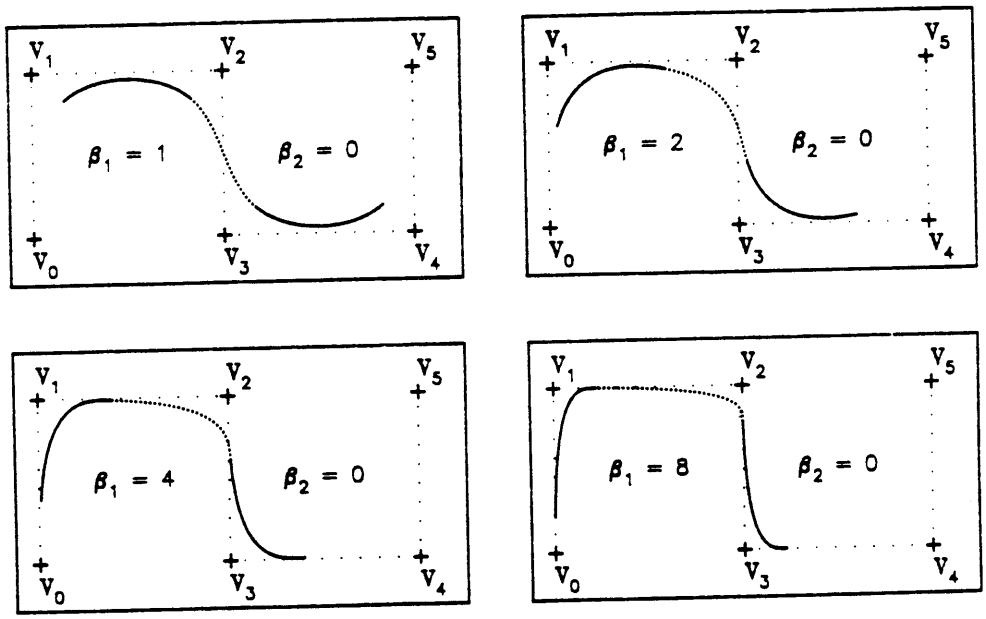
Notice that if we substitute $\beta_1 = 1$ and $\beta_2 = 0$ into the Beta-spline constraint equations (E16) we obtain the B-spline constraint equations (E6), and that substituting these values into the Beta-spline basis segments (E17) we obtain the B-spline basis segments (E7). For other values of $\beta_1$ and $\beta_2$ the Beta-spline basis segments fail to be $C^2$ continuous at knots, although they do satisfy equations (E16) and are therefore $G^2$ continuous.

Equations (E17) can, of course, be evaluated more rapidly if they are factored. Since for any particular values of $\beta_1$ and $\beta_2$ they are cubic polynomials in u, forward differencing can also be used where appropriate. The efficient evaluation of these equations is discussed in [Barsky81a, Barsky82d].

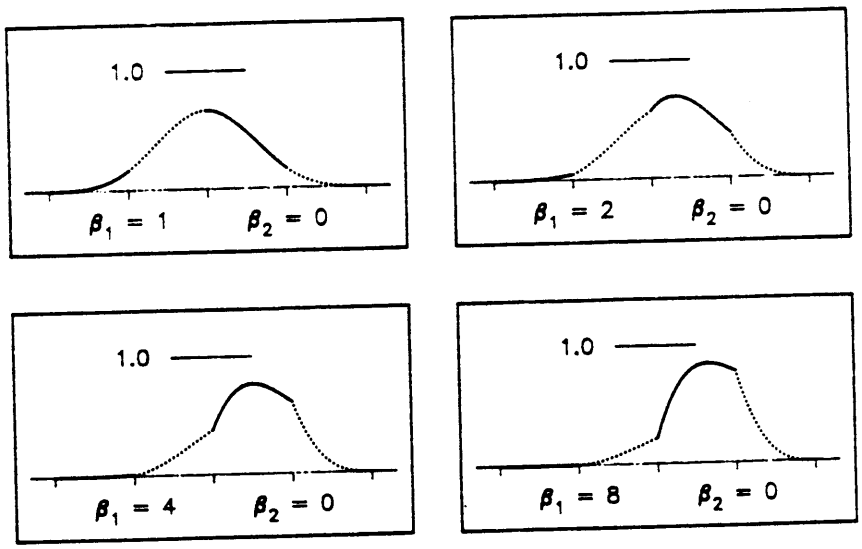
We shall refer to a Beta-spline curve whose segments are defined by equations (E9) and (E17) as a *uniformly-shaped Beta-spline* in order to distinguish it from the Beta-spline curves which will be defined subsequently.

Increasing $\beta_1$ increases the "velocity" with which we traverse a curve immediately to the right of a joint, with respect to the "velocity" just left of the joint, thus serving to *bias* the curve; values in excess of one cause the unit tangent vector at the joint (which is, of course, continuous) to have greater influence to the right than to the left, in that the curve will "continue in the direction of the tangent" longer in the rightmost segment. Values of $\beta_1$ from ranging from one down to zero have the reciprocal effect, causing the curve to lie close to the tangent longer to the left of a joint than to the right.

[4] Vaxima code to solve equations (E16) has been banished to an appendix (programs P1, P2 and P3). Code referred to subsequently appears there as well.

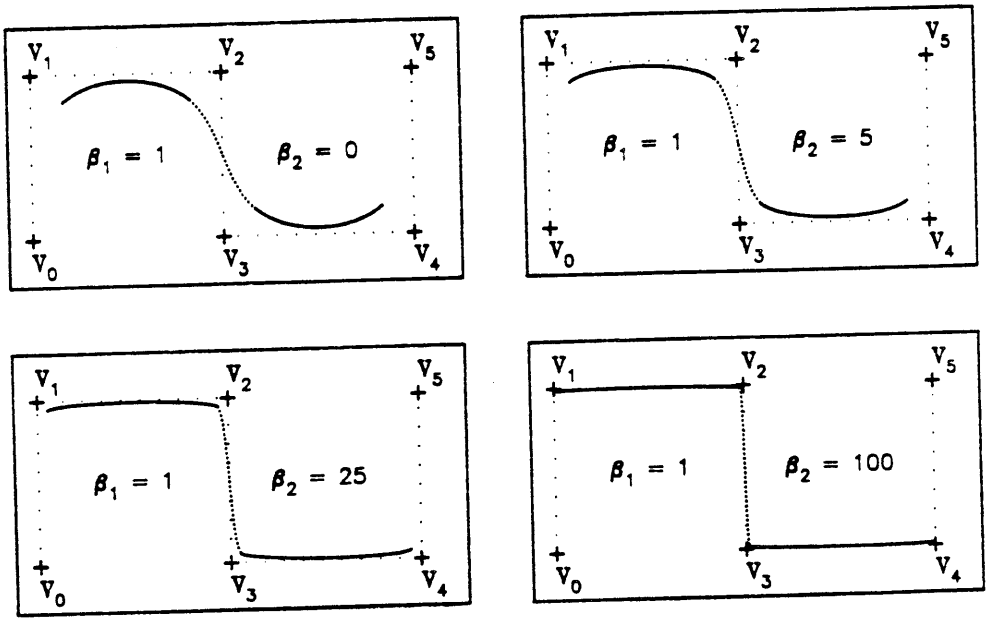



It is instructive to examine the basis functions used to define these curves:

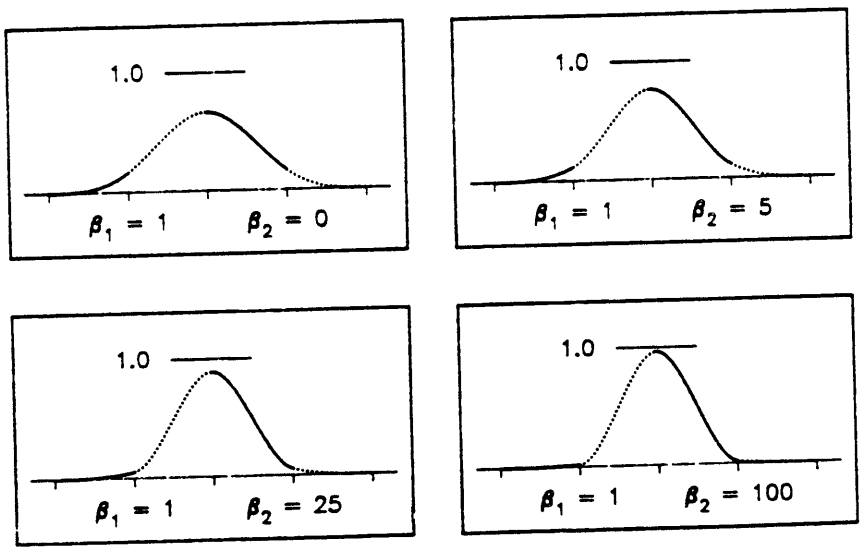



Each is computed for a distinct value of $\beta_1$, which determines the change in the absolute value of the slope at each knot. Notice that since the same basis function is used for each of X(u) and Y(u), any continuous basis function whose first derivative is continuous except for a jump of some arbitrary value ($\beta_1$) at the knots suffices to define a curve with unit tangent continuity.

The $\beta_2$ parameter serves to control *tension* in the curve: altering the value of $\beta_2$ moves the joint between $\mathbf{Q}_{i-1}(u)$ and $\mathbf{Q}_i(u)$ along a vector which passes through the $i^{th}$ control vertex. For example, increasingly positive values move this joint towards $\mathbf{V}_i$ and flatten the curve.




The corresponding basis functions are:




Notice that as $\beta 2$ increases the peak of the basis function approaches one and the "tails" of the basis function, lying in the leftmost and rightmost intervals of its support, approach zero. More generally, the curve itself converges to the control polygon as $\beta 2$ goes to infinity, as well as interpolating the control vertices.

This behavior is predictable from equations (E17). As $\beta 2$ is increased, the basis segments converge to

$$b_1(u) = 0$$

$$b_0(u) = (3u^2 - 2u^3)$$

$$b_{-1}(u) = 1 - (3u^2 - 2u^3)$$

$$b_{-2}(u) = 0$$

If we let $t = (3u^2 - 2u^3)$, it is easy to see that in the limit we obtain a curve which varies linearly between each successive pair of control vertices.

$\beta_1$ also serves, to some extent, as a "biased" tension parameter. If for any value of $\beta_2$ we allow $\beta_1$ to become arbitrarily large then the basis segments converge to

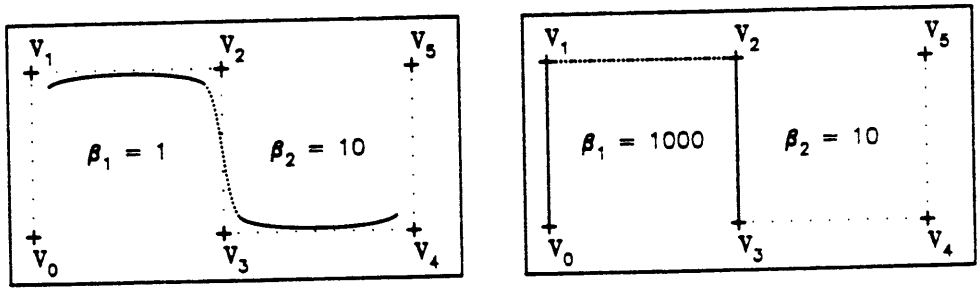
$$b_1(u) = 0$$

$$b_0(u) = 0$$

$$b_{-1}(u) = (3u - 3u^2 + u^3)$$

$$b_{-2}(u) = 1 - (3u - 3u^2 + u^3)$$

If these are weighted by $\mathbf{V}_{i+1}$, $\mathbf{V}_i$, $\mathbf{V}_{i-1}$ and $\mathbf{V}_{i-2}$ to define the $i^{th}$ segment $\mathbf{Q}_i(u)$ then this segment of the curve converges to a straight line between $\mathbf{V}_{i-2}$ and $\mathbf{V}_{i-1}$ as $\beta_1$ increases.




If $\beta_2$ has the value zero and we allow $\beta_1$ to approach zero then we obtain symmetrical behavior:

$$b_1(u) = u^3$$

$$b_0(u) = 1 - u^3$$

$$b_{-1}(u) = 0$$

$$b_{-2}(u) = 0$$

In this case $\mathbf{Q}_i(u)$ is, in the limit, a straight line running from $\mathbf{V}_i$ to $\mathbf{V}_{i+1}$. However, if $\beta_2$ is nonzero then as $\beta_1$ approaches zero the basis segments converge to

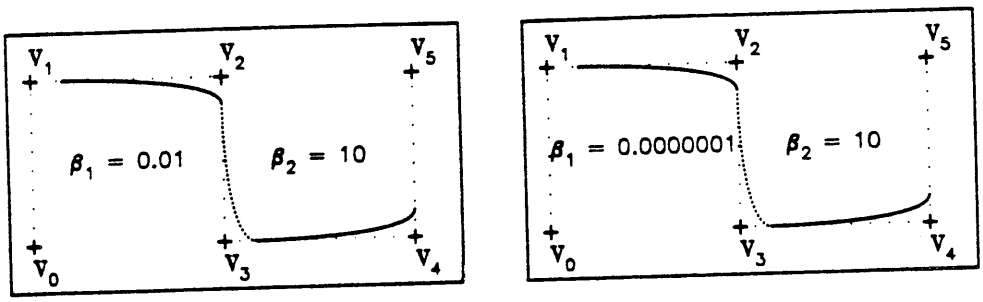
$$b_1(u) = \frac{1}{\beta_2 + 2} \, 2u^3$$

$$b_0(u) = \frac{1}{\beta_2 + 2} \left\{ 2 + 3\beta_2 u^2 - (2\beta_2 + 2)u^3 \right\}$$

$$b_{-1}(u) = \frac{1}{\beta_2 + 2} \left\{ \beta_2 - 3\beta_2 u^2 + 2\beta_2 u^3 \right\}$$

$$b_{-2}(u) = 0$$

Thus as $\beta_1$ approaches zero $\mathbf{Q}_i(u)$ does not, in general, approach a straight line unless $\beta_2$ is zero.




$\beta_1$ and $\beta_2$ may be altered, independent of the control vertices, in order to change the shape of the curve. In the curves we have been discussing a single value of $\beta_1$ is used for the entire curve, and similarly for $\beta_2$. We would prefer, if possible, to specify distinct values of $\beta_1$ and $\beta_2$ at each joint. Before discussing how this can be done, we indicate briefly how uniformly-shaped Beta-spline surfaces can be constructed from uniformly-shaped curves.

### 4.1. Surfaces

The formation of uniform Beta-spline surfaces is a natural and straightforward generalization of the uniform Beta-spline curves. We want to form our surface as a weighted sum of basis functions, as in (E8), but now X, Y and Z must be functions of two independent variables:

$$\mathbf{Q}(u,v) = \sum_{i,j} \mathbf{V}_{i,j} B_{i,j}(u,v) = \sum_{i,j} \left( x_{i,j} B_{i,j}(u,v),\ y_{i,j} B_{i,j}(u,v),\ z_{i,j} B_{i,j}(u,v) \right) \tag{E18}$$

For weights we again use the x-, y- and z-coordinates of what is now a two-dimensional array of control vertices called the *control mesh* or *control graph*. In order to obtain locality we would like the new basis functions $B_{i,j}(u,v)$ to be nonzero only for a small range of u and v. An easy way to arrange this is to let $B_{i,j}(u,v) = B_i(u) B_j(v)$, where $B_i(u)$ and $B_j(v)$ are simply the univariate basis functions (E17) which we developed for the Beta-spline curves. Since each is nonzero only over four successive intervals, if $u_{i-1} \le u \le u_i$ and $u_{j-1} \le v \le u_j$ we can rewrite (E18) as

$$\mathbf{Q}(u,v) = \sum_{r=-2}^{1} \sum_{s=-2}^{1} \mathbf{V}_{i+r,j+s} B_{i+r}(u) B_{j+s}(v) \tag{E19}$$

If we rewrite this in terms of basis segments instead of basis functions and adopt the convention that the portion of $\mathbf{Q}(u,v)$ defined by this set of values for u and v is denoted by $\mathbf{Q}_{i,j}(u,v)$, then we can write

$$\mathbf{Q}_{i,j}(u,v) = \sum_{r=-2}^{1} \sum_{s=-2}^{1} \mathbf{V}_{i+r,j+s} b_r(u) b_s(v) \tag{E20}$$

so that $\mathbf{Q}_{i,j}(u,v)$, the $i,j^{th}$ *patch*, is completely determined by sixteen control vertices. The separability of $B_{i,j}(u,v)$ into $B_i(u)$ and $B_j(v)$ can be useful. For example, we can expand (E20) as

$$\begin{aligned}
\mathbf{Q}_{i,j}(u,v) = {} & \left[ \mathbf{V}_{i-2,j+1}\, b_{-2}(u) + \mathbf{V}_{i-1,j+1}\, b_{-1}(u) + \mathbf{V}_{i,j+1}\, b_0(u) + \mathbf{V}_{i+1,j+1}\, b_{-2}(u) \right] b_1(v) + \\
& \left[ \mathbf{V}_{i-2,j}\, b_{-2}(u) + \mathbf{V}_{i-1,j}\, b_{-1}(u) + \mathbf{V}_{i,j}\, b_0(u) + \mathbf{V}_{i+1,j}\, b_{-2}(u) \right] b_0(v) + \\
& \left[ \mathbf{V}_{i-2,j-1}\, b_{-2}(u) + \mathbf{V}_{i-1,j-1}\, b_{-1}(u) + \mathbf{V}_{i,j-1}\, b_0(u) + \mathbf{V}_{i+1,j-1}\, b_{-2}(u) \right] b_{-1}(v) + \\
& \left[ \mathbf{V}_{i-2,j-2}\, b_{-2}(u) + \mathbf{V}_{i-1,j-2}\, b_{-1}(u) + \mathbf{V}_{i,j-2}\, b_0(u) + \mathbf{V}_{i+1,j-2}\, b_{-2}(u) \right] b_{-2}(v)
\end{aligned} \tag{E21}$$

From this it is clear that if we fix u at some arbitrary value between 0 and 1 then we can write (E21) as

$$\mathbf{Q}_{i,j,u}(v) = \mathbf{W}_{-2} b_{-2}(v) + \mathbf{W}_{-1} b_{-1}(v) + \mathbf{W}_0 b_0(v) + \mathbf{W}_1 b_1(v)$$

where

$$\begin{aligned}
\mathbf{W}_1 &= \mathbf{V}_{i-2,j+1}\, b_{-2}(u) + \mathbf{V}_{i-1,j+1}\, b_{-1}(u) + \mathbf{V}_{i,j+1}\, b_0(u) + \mathbf{V}_{i+1,j+1}\, b_{-2}(u) \\
\mathbf{W}_0 &= \mathbf{V}_{i-2,j}\, b_{-2}(u) + \mathbf{V}_{i-1,j}\, b_{-1}(u) + \mathbf{V}_{i,j}\, b_0(u) + \mathbf{V}_{i+1,j}\, b_{-2}(u) \\
\mathbf{W}_{-1} &= \mathbf{V}_{i-2,j-1}\, b_{-2}(u) + \mathbf{V}_{i-1,j-1}\, b_{-1}(u) + \mathbf{V}_{i,j-1}\, b_0(u) + \mathbf{V}_{i+1,j-1}\, b_{-2}(u) \\
\mathbf{W}_{-2} &= \mathbf{V}_{i-2,j-2}\, b_{-2}(u) + \mathbf{V}_{i-1,j-2}\, b_{-1}(u) + \mathbf{V}_{i,j-2}\, b_0(u) + \mathbf{V}_{i+1,j-2}\, b_{-2}(u)
\end{aligned}$$

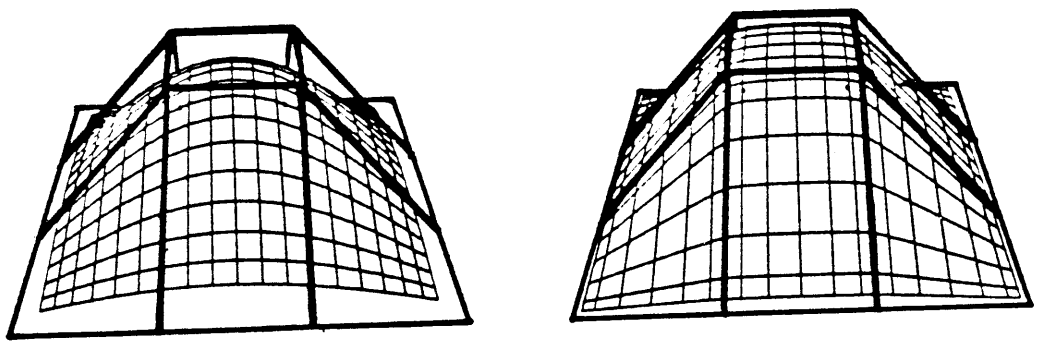
Thus $\mathbf{Q}_{i,j,u}(v)$ is simply the uniformly-shaped Beta-spline curve segment defined by the "control vertices" $\mathbf{W}_{-2}$, $\mathbf{W}_{-1}$, $\mathbf{W}_0$ and $\mathbf{W}_1$. It is not hard to see that $\mathbf{Q}_{i,j+1,u}(v)$, in the next patch "up", is given by

$$\mathbf{Q}_{i,j+1,u}(v) = \mathbf{W}_{-1}\, b_{-2}(v) + \mathbf{W}_0\, b_{-1}(v) + \mathbf{W}_1\, b_0(v) + \mathbf{W}_2\, b_1(v)$$

where

$$\mathbf{W}_2 = \mathbf{V}_{i-2,j+2}\, b_{-2}(u) + \mathbf{V}_{i-1,j+2}\, b_{-1}(u) + \mathbf{V}_{i,j+2}\, b_0(u) + \mathbf{V}_{i+1,j+2}\, b_1(u)$$

This is simply the second segment in a uniformly-shaped Beta-spline curve defined by the "control vertices" $\mathbf{W}_{-2}$, $\mathbf{W}_{-1}$, $\mathbf{W}_0$, $\mathbf{W}_1$ and $\mathbf{W}_2$. It follows immediately that this curve is $G^2$ continuous. Since a completely analogous argument can be made with respect to u by factoring the $b_r(u)$ out of (E20) instead of the $b_s(v)$, the uniformly-shaped Beta-spline surface we have defined is $G^2$ continuous along lines of constant u and v.



Uniformly shaped surfaces: $\beta 2$ is 0 on the right and 25 on the right.

## 5. Interpolating Distinct $\beta$'s

Let $\alpha 1_i$ and $\alpha 2_i$ be the values of $\beta 1$ and $\beta 2$ to be associated with the joint between $\mathbf{Q}_{i-1}(u)$ and $\mathbf{Q}_i(u)$. We would like to use the basis segments given by equations (E17), making $\beta 1$ and $\beta 2$ functions of u in such a way as to interpolate between the $\alpha 1$'s and $\alpha 2$'s at each end of a segment while preserving $G^2$ continuity of the curve.

Let us consider the following derivative with respect to u of a representative term of (E17),[5]

$$\frac{c\,[\beta(u)]^p\, u^q}{\delta(u)} \tag{E22}$$

where c is a constant. Its first parametric derivative is

$$\frac{c\,q\,[\beta(u)]^p\, u^{q-1}}{\delta(u)} + \frac{c\,p\,[\beta(u)]^{p-1}\,\beta^{(1)}(u)\, u^q}{\delta(u)} - \frac{c\,[\beta(u)]^p\,\delta^{(1)}(u)\, u^q}{\delta(u)^2} \tag{E23}$$

[5] We will use $\beta(u)$ rather than $\beta 1(u)$ or $\beta 2(u)$ when the argument applies to both and no confusion can occur because products of $\beta 1$ and $\beta 2$ do not arise. Similarly, $\alpha_i$ will be used to represent both $\alpha 1_i$ and $\alpha 2_i$

where

$$\delta(u) = \beta_2(u) + 2[\beta_1(u)]^3 + 4[\beta_1(u)]^2 + 4\beta_1(u) + 2$$
$$\delta^{(1)}(u) = \beta_2^{(1)}(u) + 6[\beta_1(u)]^2\beta_1^{(1)}(u) + 8\beta_1(u)\beta_1^{(1)}(u) + 4\beta_1^{(1)}(u) \quad \text{(E24)}$$

Examination of (E23) and (E24) reveals that the second and third terms of (E23) involve products with $\beta_1^{(1)}(u)$ or $\beta_2^{(1)}(u)$, while the first term of (E23) would constitute the complete parametric derivative if $\beta_1$ and $\beta_2$ were not functions of u. If we were to compute $\beta_1(u)$ and $\beta_2(u)$ by interpolating between the $\alpha 1_i$'s and $\alpha 2_i$'s in such a way as to cause $\beta_1^{(1)}(u)$ and $\beta_2^{(1)}(u)$ to be zero at each joint then equations (E14) would hold and $G^1$ continuity would be preserved.

Similarly, the second parametric derivative of (E22) is

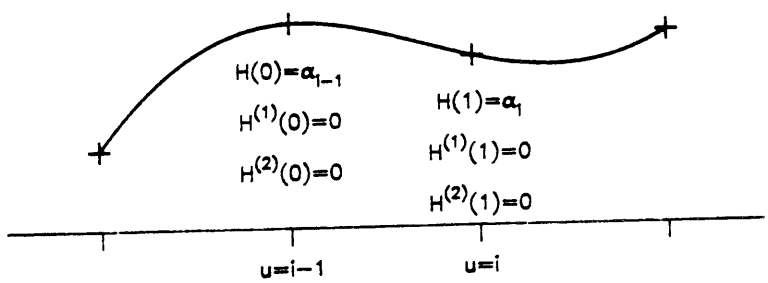
$$\frac{c(n-1)n[\beta(u)]^m u^{n-2}}{\delta(u)} \quad \text{(E25)}$$
$$- \frac{c[\beta(u)]^m \delta^{(2)}(u) u^n}{\delta(u)^2} + \frac{2c[\beta(u)]^m \delta^{(1)}(u)^2 u^n}{\delta(u)^3}$$
$$- \frac{2cm[\beta(u)]^{m-1}\beta^{(1)}(u)\delta^{(1)}(u)u^n}{\delta(u)^2} - \frac{2cn[\beta(u)]^m \delta^{(1)}(u) u^{n-1}}{\delta(u)^2}$$
$$+ \frac{cm[\beta(u)]^{m-1}\beta^{(2)}(u)u^n}{\delta(u)} + \frac{c(m-1)m[\beta(u)]^{m-2}\beta^{(1)}(u)^2 u^n}{\delta(u)}$$
$$+ \frac{2cmn[\beta(u)]^{m-1}\beta^{(1)}(u)u^{n-1}}{\delta(u)}$$

where

$$\delta^{(2)}(u) = \beta_2^{(2)}(u) + 6[\beta_1(u)]^2\beta_1^{(2)}(u) + 8\beta_1(u)\beta_1^{(2)}(u)$$
$$+ 4\beta_1^{(2)}(u) + 12\beta_1(u)\beta_1^{(1)}(u)^2 + 8\beta_1^{(1)}(u)^2$$

Again, only the first term of (E25) lacks a product with at least one of $\beta_1^{(1)}(u)$, $\beta_2^{(1)}(u)$, $\beta_1^{(2)}(u)$ or $\beta_2^{(2)}(u)$, and the first term would constitute the complete second parametric derivative if $\beta_1$ and $\beta_2$ were not functions of u. Thus arranging that all four derivatives have the value zero at joints should be sufficient to preserve $G^2$ continuity of the curve. This is easily accomplished in the following manner.

Suppose that we use a polynomial $H(\alpha_{i-1},\alpha_i;u)$ to interpolate between $\alpha_{i-1}$ and $\alpha_i$.




We have six constraints, since we would like

$$H(\alpha_{i-1},\alpha_i;0) = \alpha_{i-1}$$
$$H(\alpha_{i-1},\alpha_i;1) = \alpha_i$$

$$H^{(1)}(\alpha_{i-1},\alpha_i;0) = 0$$

$$H^{(1)}(\alpha_{i-1},\alpha_i;1) = 0$$

$$H^{(2)}(\alpha_{i-1},\alpha_i;0) = 0$$

$$H^{(2)}(\alpha_{i-1},\alpha_i;1) = 0$$

This suggests the use of a fifth degree polynomial (which has, of course, six coefficients). If

$$H(\alpha_{i-1},\alpha_i;u) = a + bu + cu^2 + du^3 + eu^4 + fu^5$$

then the above equations take the form

$$H(\alpha_{i-1},\alpha_i;0) = \alpha_{i-1} = a$$

$$H(\alpha_{i-1},\alpha_i;1) = \alpha_i = a + b + c + d + e + f$$

$$H^{(1)}(\alpha_{i-1},\alpha_i;0) = 0 = b$$

$$H^{(1)}(\alpha_{i-1},\alpha_i;1) = 0 = b + 2c + 3d + 4e + 5f$$

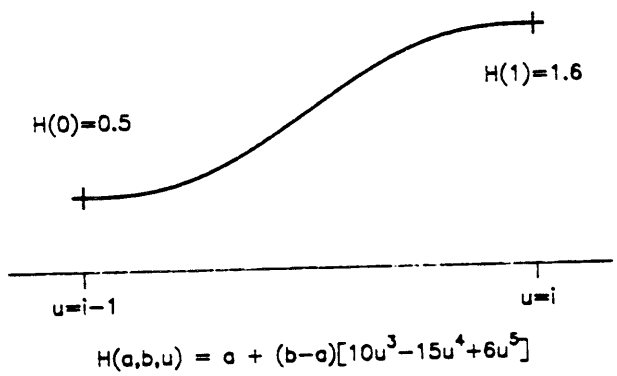
$$H^{(2)}(\alpha_{i-1},\alpha_i;0) = 0 = 2c$$

$$H^{(2)}(\alpha_{i-1},\alpha_i;1) = 0 = 2c + 6d + 12e + 20f$$

It is straightforward to obtain the polynomial

$$\begin{aligned}\beta_i(u) = H(\alpha_{i-1},\alpha_i;u) &= \alpha_{i-1} + 10(\alpha_i-\alpha_{i-1})u^3 - 15(\alpha_i-\alpha_{i-1})u^4 + 6(\alpha_i-\alpha_{i-1})u^5 \\ &= \alpha_{i-1} + (\alpha_i-\alpha_{i-1})[10u^3 - 15u^4 + 6u^5]\end{aligned} \tag{E26}$$

which satisfies these equations; this is, in fact, a special case of quintic Hermite interpolation. By the argument given above, the use of (E26) to interpolate $\beta_1$ and $\beta_2$ in (E17) preserves $G^2$ continuity of the curve. (This fact is verified directly by Vaxima program P4.)




It is, of course, possible that the derivative terms appearing in (E22) and (E25) might sum in such a way as to yield $G^2$ continuous curves even though the derivatives were nonzero; we have not ruled this out for all other interpolation schemes. However, it is not hard to produce examples which demonstrate that neither linear interpolation, nor cubic Hermite interpolation, nor even quintic Hermite interpolation for constant but nonzero derivatives works. Thus $C^2$ continuity of $\beta_1(u)$ and $\beta_2(u)$ is not sufficient to ensure $G^2$ continuity. (Program P5 computes a counter-example establishing this fact.)

We shall refer to the curves whose segments are defined by equations (E9) and (E17), where $\beta_1(u)$

and $\beta_2(u)$ are interpolated by equation (E26), as *continuously-shaped Beta-splines.*

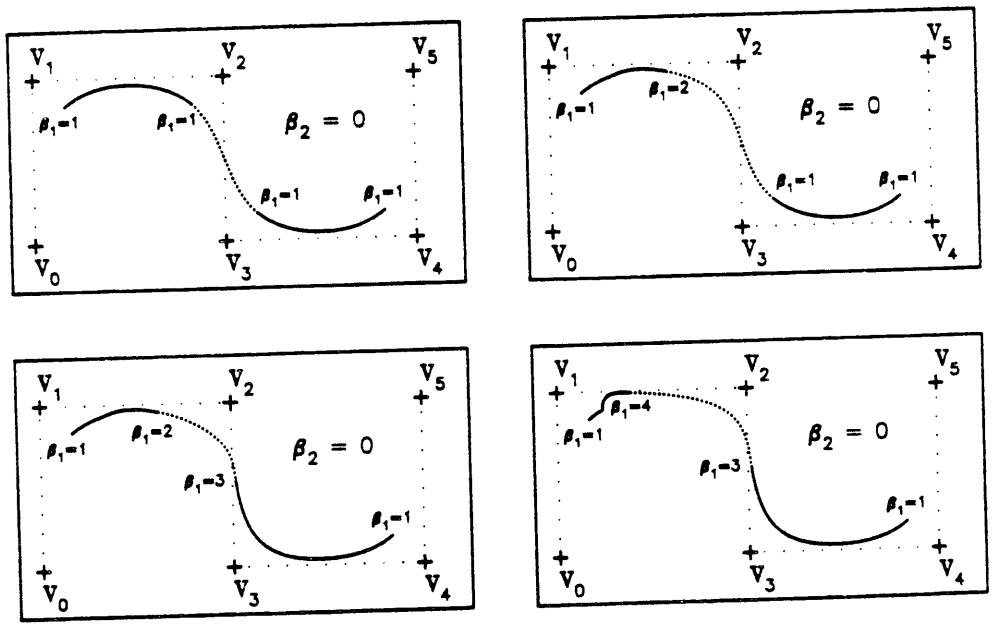
### 5.1. Locality

Just as for the uniformly-shaped Beta-splines, each basis function is nonzero only over four successive intervals. Since each control vertex is used to weight a particular basis function, moving a control vertex will alter only the four corresponding curve segments. These are, of course, consecutive.

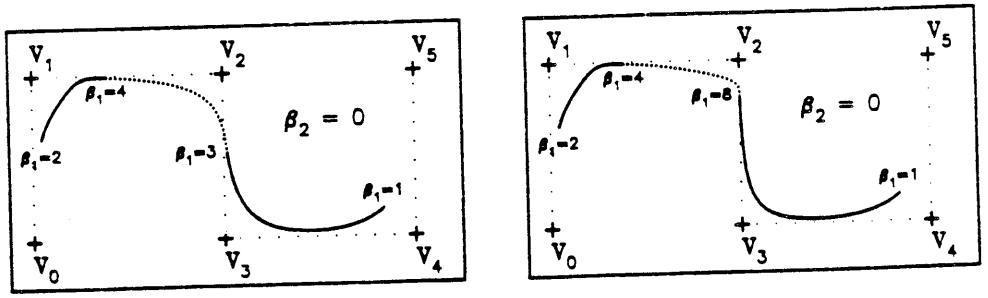
The effect of altering an $\alpha$ value is more localized still. The $\alpha$ value at a particular joint determines how the corresponding $\beta$ parameter is interpolated over the segments which meet at that joint, so that only two curve segments are changed.

### 5.2. Bias

The following figure illustrates a few of the effects which can be obtained by altering $\alpha_1$'s.




Although the resulting curves are often visually satisfying, their extreme locality with respect to changes in the shape parameters can necessitate "kinks" if there are large differences in the $\alpha$ values for consecutive control vertices. A modest reduction in the size of the jumps ameliorates the effect.



### 5.3. Tension

Since this scheme interpolates the $\alpha_i$, the discussion of tension in [Barsky81a, Barsky82c] is equally applicable here:

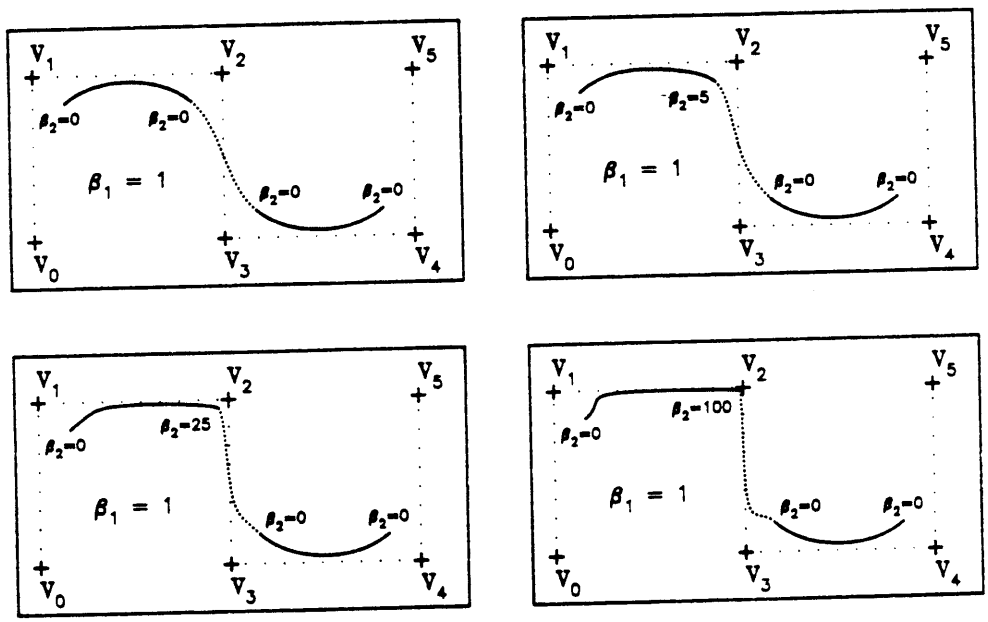
$$\mathbf{Q}_i(0) - \mathbf{V}_i = \mathbf{Q}_{i-1}(1) - \mathbf{V}_i = \frac{(\mathbf{C} - c\mathbf{V}_i)}{(c + \alpha 2_i)}$$

is the vector from the $i^{th}$ control vertex to the joint between $\mathbf{Q}_{i-1}(u)$ and $\mathbf{Q}_i(u)$, where

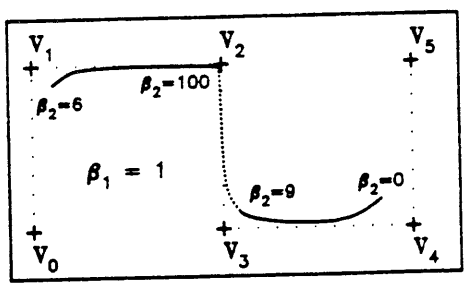
$$\mathbf{C} = 2\alpha 1_i^3 \mathbf{V}_{i-1} + 4\alpha 1_i(\alpha 1_i + 1)\mathbf{V}_i + 2\mathbf{V}_{i+1}$$

$$c = 2\alpha 1_i^3 + 4\alpha 1_i^2 + 4\alpha 1_i + 2$$

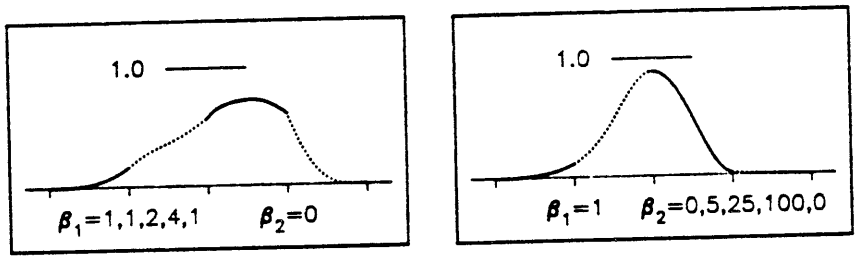
Altering $\alpha 2_i$ merely changes the length of this vector: values approaching -c "push" the joint arbitrarily far away from $\mathbf{V}_i$, while large positive or negative values draw the joint arbitrarily close to $\mathbf{V}_i$. Hence $\alpha 2_i$ serves as a tension parameter, just as for uniformly-shaped Beta-spline curves.




Again, wildly disparate values of $\beta 2$ for adjacent control vertices can produce unsightly kinks. These can be removed, if that is desirable, by smaller adjustments in neighboring $\beta$ values, as shown below.




For comparison with earlier figures we give some examples of continuously-varying basis functions.




Because each point on a continuously shaped Beta-spline curve also lies on the uniformly shaped curve defined by the same control points and the values of $\beta 1$ and $\beta 2$ at that point on the continuously shaped curve, $\alpha 2$ values can be used to locally force a curve to converge to the control polygon if they are increased arbitrarily.

### 5.4. Convex Hull

Like the uniformly-shaped Beta-splines, continuously-shaped Beta-spline curves possess a convex hull property in that the $i^{th}$ segment lies within the convex hull of control vertices $\mathbf{V}_{i-2}$, $\mathbf{V}_{i-1}$, $\mathbf{V}_i$ and $\mathbf{V}_{i+1}$, so long as $\beta 1$ and $\beta 2$ are nonnegative. The argument, as we shall see, is straightforward. Recall that since each basis function is nonzero over four intervals,

$$\mathbf{Q}_i(u) = \mathbf{V}_{i-2}\,b_{-2}(u) + \mathbf{V}_{i-1}\,b_{-1}(u) + \mathbf{V}_i\,b_0(u) + \mathbf{V}_{i+1}\,b_1(u) \qquad \text{(E27)}$$

Now for any given value of u, $\beta 1(u)$ and $\beta 2(u)$ yield some particular value of $\beta 1$ and $\beta 2$. By simply summing equations (E17) we see that for every such $\beta 1$, $\beta 2$ and u

$$b_1(u) + b_0(u) + b_{-1}(u) + b_{-2}(u) = 1$$

Next we must verify that these basis segments are nonnegative for all u in the interval [0,1]. If we rewrite equations (E17) in the form

$$b_1(u) = \frac{1}{\delta}\left\{ 2u^3 \right\}$$

$$b_0(u) = \frac{1}{\delta}\left\{ 2\beta 1^2 u^2(3-u) + 2\beta 1\, u(3-u^2) + \beta 2\, u^2(3-2u) + 2(1-u^3) \right\}$$

$$b_{-1}(u) = \frac{1}{\delta}\left\{ 2\beta 1^3 u((1-u)(2-u)+1) + 2\beta 1^2(u^3-3u^2+2) + 2\beta 1(u^3-3u+2) + \beta 2(2u^3-3u^2+1) \right\}$$

$$b_{-2}(u) = \frac{1}{\delta}\left\{ 2\beta 1^3(1-u)^3 \right\}$$

where

$$\delta = \beta 2 + 2\beta 1^3 + 4\beta 1^2 + 4\beta 1 + 2$$

for $\beta 1 \geq 0$, $\beta 2 \geq 0$, and $u \in [0,1]$, it is easy to see by inspection that $b_1(u)$, $b_0(u)$, and $b_{-2}(u)$ are nonnegative. For $b_{-1}(u)$, elementary consideration of the zeros of the derivatives 3u(u-2), 3(u-1)(u-1) and 6u(u-1) of $u^3-3u^2+2$, $u^3-3u+2$, and $2u^3-3u^2+1$ yields the same conclusion. Since $\beta 1$ and $\beta 2$ are actually interpolated by (E26), it is necessary to show that

$$\beta_i(u) = \alpha_{i-1} + (\alpha_i-\alpha_{i-1})[10u^3 - 15u^4 + 6u^5] \geq 0$$

if $\alpha_{i-1} \geq 0$, $\alpha_i \geq 0$, and $u \in [0,1]$. Consider

$$\beta_i^{(1)}(u) = 30(\alpha_i - \alpha_{i-1})u^2(1-u)^2$$

Clearly the slope changes sign only at $u=0$ and $u=1$. Since

$$\beta_i(0.5) = \frac{\alpha_i + \alpha_{i+1}}{2} \geq 0 \quad \text{if } \alpha_{i-1}, \alpha_i \geq 0,$$

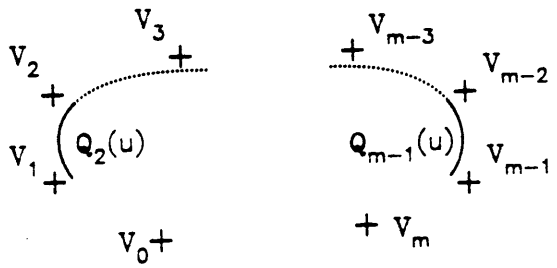
$\beta_i(u)$ must be nonnegative on [0,1] so long as the $\alpha_i$ are nonnegative.

Hence so long as $\alpha 1_i \geq 0$ and $\alpha 2_i \geq 0$, $\mathbf{Q}_i(u)$ lies within the convex hull of $\mathbf{V}_{i-2}$, $\mathbf{V}_{i-1}$, $\mathbf{V}_i$ and $\mathbf{V}_{i+1}$.

### 5.5. End Conditions

A properly defined curve segment is the sum of four weighted basis functions, as in equation (E27). Thus m+1 control vertices $\mathbf{V}_0, \cdots, \mathbf{V}_m$ can be used to define m-2 segments, which we index as $\mathbf{Q}_2(u), \cdots, \mathbf{Q}_{m-1}(u)$. The Beta-spline curve then begins[6] at

$$\mathbf{Q}_2(0) = \frac{1}{\delta(0)} \left\{ 2\alpha 1_i^3 \mathbf{V}_0 + (\delta(0) - 2\alpha 1_i^3 - 2)\mathbf{V}_1 + 2\mathbf{V}_2 \right\}$$




Thus the curve does not, in general, begin at a control vertex, or even at a point along the line segment from $\mathbf{V}_0$ to $\mathbf{V}_1$. In order to obtain better control of the beginning of the curve, one therefore often treats the ends of the curve specially.

Let $\mathbf{Q}(u)$ be a continuously-shaped Beta-spline with $\beta 1 = \alpha 1_i$ and $\beta 2 = \alpha 2_i$ at the joint between the $i^{th}$ and $i+1^{st}$ segments. Let $\mathbf{R}(u)$ be a uniformly-shaped Beta-spline curve defined by the same control vertices, but with $\beta 1 = \alpha 1_i$ and $\beta 2 = \alpha 2_i$ throughout. By the definition of $\mathbf{Q}(u)$ we must have $\mathbf{Q}(u) = \mathbf{R}(u)$, $\mathbf{Q}^{(1)}(u) = \mathbf{R}^{(1)}(u)$ and $\mathbf{Q}^{(2)}(u) = \mathbf{R}^{(2)}(u)$ at the joint in question. Hence the analysis of end conditions in [Barsky81a] applies immediately to the continuously-shaped Beta-splines. For convenience we summarize these results.

- A Double First Vertex. We define an additional segment at the beginning of the curve by

$$\mathbf{Q}_1(u) = \mathbf{V}_0[b_{-2}(u) + b_{-1}(u)] + \mathbf{V}_1 b_0(u) + \mathbf{V}_2 b_1(u)$$

  $\mathbf{Q}_1(u)$ begins at a point lying along the line segment from $\mathbf{V}_0$ to $\mathbf{V}_1$, at which point it is tangent to that line and has zero curvature.

- A Triple First Vertex. We define two additional segments at the beginning of the curve by

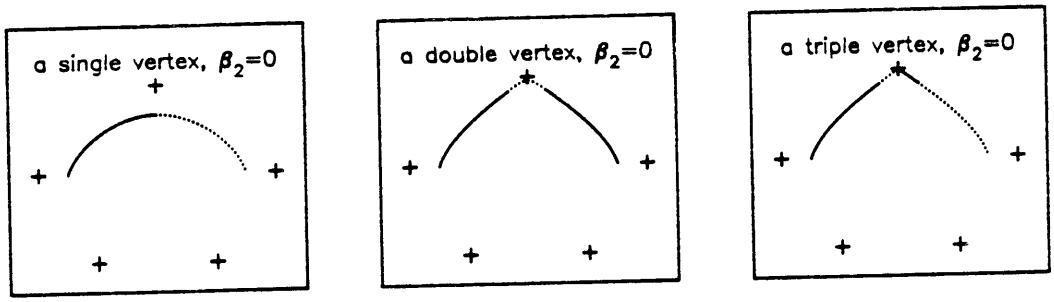
$$\mathbf{Q}_0(u) = \mathbf{V}_0[b_{-2}(u) + b_{-1}(u) + b_0(u)] + \mathbf{V}_2 b_1(u)$$

$$\mathbf{Q}_1(u) = \mathbf{V}_0[b_{-2}(u) + b_{-1}(u)] + \mathbf{V}_1 b_0(u) + \mathbf{V}_2 b_1(u)$$

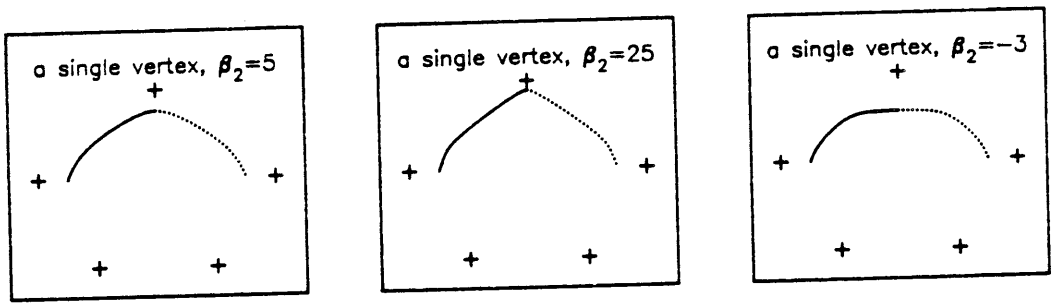
[6] The terminal point of the curve is analyzed in an exactly analogous manner, and we therefore omit explicit treatment of it.

The curve then begins at $\mathbf{Q}_0(0) = \mathbf{V}_0$ and the first segment of the curve is a short straight line. The behavior of the second segment $\mathbf{Q}_1(u)$, which has a double first vertex, is described above.

The analysis of double and triple vertices is equally applicable on the interior of a curve. Triple interior vertices are particularly interesting since they result in a cusp:




This cusp is not a violation of $G^2$ continuity because, or at least in the sense that, the first parametric derivative vector has the value (0,0) at the joint that coincides with the interpolated control vertex where the cusp occurs, so that the unit tangent vector is not defined. Multiple vertices give a tension-like effect, and it is instructive to compare the effect of repeating a vertex with the effect of altering $\beta 2$ there:




An alternate way of controlling the beginning of a curve is to automatically define a *phantom vertex* $\mathbf{V}_{-1}$ and a corresponding initial segment

$$\mathbf{Q}_1(u) = \mathbf{V}_{-1}\,b_{-2}(u) + \mathbf{V}_0\,b_{-1}(u) + \mathbf{V}_1\,b_0(u) + \mathbf{V}_2\,b_1(u)$$

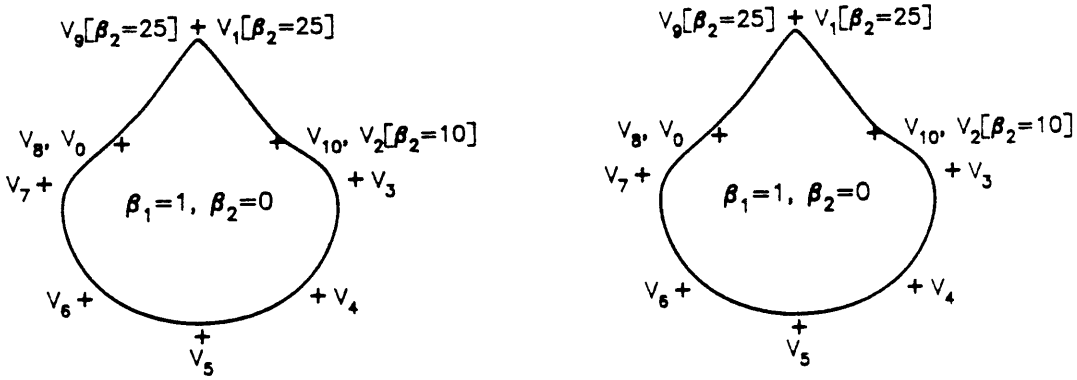
in such a way as to satisfy some requirement. We may ask that:

- $\mathbf{Q}_1(0)$ interpolate some furnished point (generally resulting in nonzero curvature);
- $\mathbf{Q}_1(0)$ interpolate $\mathbf{V}_0$ (at which point the curvature is then zero);
- $\mathbf{Q}_1^{(1)}(0)$ have some specified value (generally resulting in nonzero curvature);
- $\mathbf{Q}_1^{(2)}(0)$ have some specified value (generally resulting in nonzero curvature);
- $\mathbf{Q}_1^{(2)}(0)$ be zero, resulting in zero curvature at $\mathbf{Q}_1(0)$.

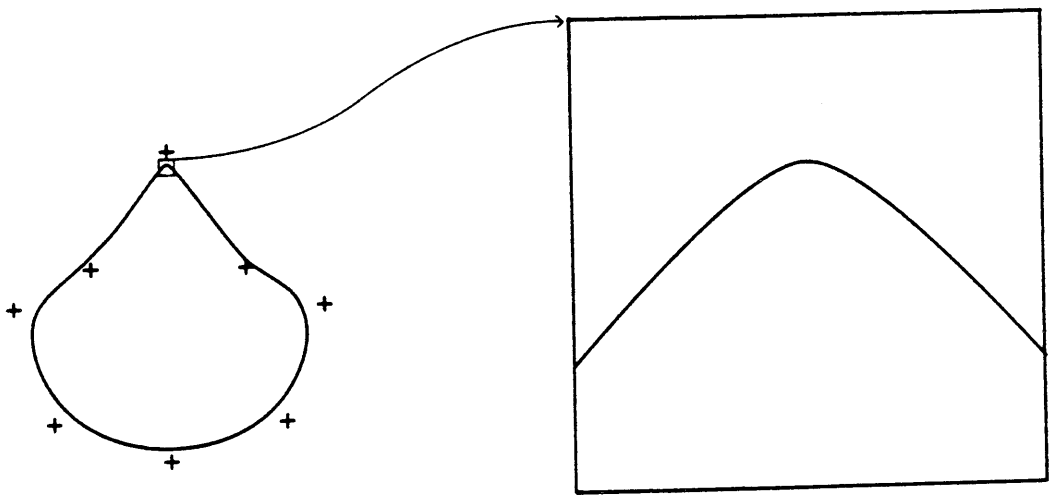
All of these techniques involve extending the curve by one or two segments at either end. This implies the existence of additional joints and associated $\alpha$ values. Hence the sequence of control vertices is extended in order to specify behavior at the ends of the curve, and additional $\alpha 1$ and $\alpha 2$ values must be specified as well. These may take any nonnegative value without affecting the behavior described above. In practice it is probably easiest simply to replicate $\alpha$ values as well as vertices.

The curves we have discussed so far are *open* curves, which is to say that the two endpoints do not, in general, coincide. A $G^2$-continuous *closed* curve whose endpoints do meet and which is $G^2$-continuous

there as well is obtained if the first three control vertices are identical to the last three and the same values of $\beta_1$ and $\beta_2$ are used at the joint between the beginning and the ending of the curve.




Although it may appear in this figure that the join near $\mathbf{V}_1$ is a cusp, by zooming in on the join we can see that in fact curvature continuity is maintained.



Again, the arguments establishing these results appear in [Barsky81a] and the details have therefore been omitted.

### 5.6. Evaluation

Using factorizations given in [Barsky81a] and [Barsky82d], the Beta-spline basis segments (E17) can be evaluated in 28 multiplication/divisions and 21 addition/subtractions. If a *single point* on $\mathbf{Q}(u)$ is to be determined, the evaluation of the right hand side of (E9) in d dimensions then requires 4d multiplications and 3d additions. The total cost for evaluating a point on a uniformly-shaped 2D Beta-spline curve is therefore 36 multiplication/divisions and 27 addition/subtractions; a 3D uniformly-shaped Beta-spline curve requires 40 multiplication/divisions and 30 addition/subtractions.

For a continuously-shaped Beta-spline curve, equation (E26) can be evaluated in 6 multiplications and 4 addition/subtractions if it is factored into the form

$$H(\alpha_{i-1},\alpha_i,u) = \alpha_{i-1} + (\alpha_i-\alpha_{i-1})[10+(6u-15)u]u^3$$

Since both $\beta_1(u)$ and $\beta_2(u)$ must be computed, both $H(\alpha 1_{i-1},\alpha 1_i,u)$ and $H(\alpha 2_{i-1},\alpha 2_i,u)$ must be evaluated. However, since $[10+(6u-15)u]u^3$ need only be evaluated once, the total cost of interpolation is 7 multiplications and 6 addition/subtractions. The additional cost for a single evaluation by this technique of a continuously-shaped Beta-spline curve, beyond that required to evaluate a uniformly-shaped curve, is therefore about 20%.

More often we wish to evaluate a sequence of points along each segment in order to render a curve. If we compute these points by repeatedly evaluating the basis functions as described above, then a uniformly-shaped 2D Beta-spline segment can be evaluated at r values of u in 16+ 20r multiplication/divisions and 14+ 13r addition/subtractions while its 3D counterpart requires 16+ 24r multiplication/divisions and 14+ 16r addition/subtractions. The corresponding cost to evaluate a 2D continuously-shaped Beta-spline curve is 36r multiplication/divisions and 2+ 31r addition/subtractions, while in 3D the cost is 41r multiplication/divisions and 2+ 34r addition/subtractions. The difference between the evaluation of uniformly- and continuously-shaped Beta-spline curves results from the need to re-evaluate the coefficients of the polynomials forming the basis segments, owing to the fact that $\beta_1$ and $\beta_2$ are no longer constant, as well as from the cost of actually performing the interpolation [Barsky81a, Barsky82d].

If instead we first sum the terms in equations (E10) so as to compute the coefficients of X(u) and Y(u), and then use Horner's rule (nested multiplication), then the evaluation of a 2D uniformly-shaped Beta-spline segment at r points requires 49+ 6r multiplication/divisions and 38+ 6r addition/subtractions while the 3D curve requires 65+ 9r multiplication/divisions and 50+ 9r addition/subtractions. A modified version of this algorithm which computes continuously-shaped Beta-spline curves requires 55r multiplication/divisions and 2+ 48r addition/subtractions in 2D and 75r multiplication/divisions and 2+ 63r addition/subtractions in 3D.

A third alternative is to use forward differencing techniques. For large values of r the evaluation of a 2D uniformly-shaped curve in this way is almost a factor of 17 faster than the evaluation of a continuously-shaped curve using Horner's rule, although it is subject to cumulative roundoff error. While in principle forward differencing is applicable to the continuously-shaped Beta-splines as well, in fact it is impractical since each coordinate is the quotient of an $18^{th}$ and a $15^{th}$ degree polynomial. Where cost is a crucial factor it may be desirable to fix $\beta_1$ at one and manipulate $\beta_2$ alone. Doing so significantly reduces the expense of evaluating equations (E17) after interpolating $\beta_2$; each coordinate is then the quotient of an $8^{th}$ and a $5^{th}$ degree polynomial.

There are other possibilities. The basis functions of a uniformly-shaped Beta-spline are translates of one another, and need only be evaluated for the first segment drawn if they are saved and reused. In the case of continuously-shaped Beta-splines, each joint is associated with distinct values of $\beta_1$ and $\beta_2$, so that in general each basis function has a different shape and must be individually evaluated.

Altering an existing curve can be done much more efficiently. If a control vertex is moved then only four segments of the curve must be recomputed, since the basis function which the vertex weights is nonzero on only four successive intervals. Since the vertex is usually moved several times in succession, it is advantageous to save the basis segments as they are first evaluated to avoid recomputing them. Moreover, the portions of the computation for each segment which are actually dependent on the vertex which is being moved may be segregated from those portions of the computation which are not, and which therefore need not be recomputed.

Altering an $\alpha$ parameter necessitates recomputing only two intervals, although all of the basis segments in each must be re-evaluated.

### 5.7. Surfaces

Continuously-shaped Beta-spline curves can be elegantly generalized to define surfaces which preserve $G^2$ continuity at the boundaries between adjacent patches. The generalization we shall present allows the user to specify a bias and tension parameter at each corner of a patch; of course, patches which share a corner make use of the same $\alpha$ values at that corner. The technique is to generalize the univariate interpolation formula (E26) to a bivariate formula in such a way that:

- the $\alpha$ values at the four corners of a patch are interpolated;
- two patches which share an edge will have the same $\beta$ values along that edge;
- the first and second partial derivatives of $\beta_1(u,v)$ and $\beta_2(u,v)$ across a patch boundary will be zero.

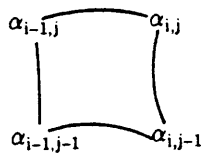
This last property will allow us to ignore (at boundaries) all but one of the terms which arise in computing the partial derivatives of a Beta-spline surface in which $\beta_1(u,v)$ and $\beta_2(u,v)$ are allowed to vary, so that the properties of a uniformly-shaped Beta-spline surface will be inherited by our continuously-shaped surface.

Thus our first consideration is to develop a bivariate interpolating formula. It is at least plausible that we would like lines of constant u or of constant v on a continuously-shaped surface to be continuously-shaped curves. Along such curves we would then expect $\beta_1$ and $\beta_2$ to vary as they do along continuously-shaped Beta-spline curves. For convenience let us write equation (E26) in two pieces as

$$s = 10u^3 - 15u^4 + 6u^5$$

$$H(\alpha_{i-1}, \alpha_i; u) = (1-s)\alpha_{i-1} + s\alpha_i$$

and along the top and bottom boundaries of the patch interpolate the $\alpha$ values




with our customary formula to obtain

$$\alpha_{top} = H(\alpha_{i-1,j}, \alpha_{i,j}; u) = (1-s)\alpha_{i-1,j} + s\alpha_{i,j}$$

$$\alpha_{bot} = H(\alpha_{i-1,j-1}, \alpha_{i,j-1}; u) = (1-s)\alpha_{i-1,j-1} + s\alpha_{i,j-1}$$

This yields values of $\alpha$ at parametric distance u from the left edge along the top and bottom of the patch. To interpolate in the v direction across the interior of the patch it is natural to again use the formula

$$H(\alpha_{bot}, \alpha_{top}; v) = (1-t)\alpha_{bot} + t\alpha_{top}$$

with

$$t = 10v^3 - 15v^4 + 6v^5$$

Substituting, we obtain the desired bivariate interpolation formula

$$\beta_{i,j}(u,v) = (1-s)(1-t)\alpha_{i-1,j-1} + s(1-t)\alpha_{i,j-1} + (1-s)t\alpha_{i-1,j} + st\alpha_{i,j} \qquad \text{(E28)}$$

with

$$s = 10u^3 - 15u^4 + 6u^5$$

$$t = 10v^3 - 15v^4 + 6v^5$$

(We emphasize that s and t are used here for notational convenience.) $\beta_{i,j}(u,v)$ has some rather attractive properties:

- it interpolates $\alpha_{i-1,j-1}$, $\alpha_{i,j-1}$, $\alpha_{i-1,j}$ and $\alpha_{i,j}$;
- along any of the four borders of a patch it reduces to the univariate interpolating formula (E26);
- the first and second partial derivatives of $\beta_{i,j}(u,v)$ with respect to v for $v = 0$ and $v = 1$ (*i.e.* across a vertical patch boundary) are zero, as are the first and second partial derivatives with respect to u for $u = 0$ and $u = 1$.

Now let us define a continuously-shaped Beta-spline surface patch $\mathbf{Q}_{i,j}$ by equation (E20) except that we let $\beta_1$ and $\beta_2$ be functions of u and v, using equation (E28) to interpolate between $\alpha$ values associated with the corners of each patch. To simplify the notation we shall actually discuss $\mathbf{Q}_{2,2}$ and $\mathbf{Q}_{2,3}$, which are defined by the control vertex mesh

$$\begin{array}{cccc} \mathbf{V}_{0,4} & \mathbf{V}_{1,4} & \mathbf{V}_{2,4} & \mathbf{V}_{3,4} \\ \mathbf{V}_{0,3} & \mathbf{V}_{1,3} & \mathbf{V}_{2,3} & \mathbf{V}_{3,3} \\ \mathbf{V}_{0,2} & \mathbf{V}_{1,2} & \mathbf{V}_{2,2} & \mathbf{V}_{3,2} \\ \mathbf{V}_{0,1} & \mathbf{V}_{1,1} & \mathbf{V}_{2,1} & \mathbf{V}_{3,1} \\ \mathbf{V}_{0,0} & \mathbf{V}_{1,0} & \mathbf{V}_{2,0} & \mathbf{V}_{3,0} \end{array}$$

(The generalization for an arbitrary patch is straightforward.) Since the $b_r(u)$ and $b_s(v)$ are now functions of $\beta_1(u,v)$ and $\beta_2(u,v)$, we write equation (E21) for $\mathbf{Q}_{2,3}$ as

(E29)

$$\begin{aligned} \mathbf{Q}_{2,3}(u,v) = \\ & [\, \mathbf{V}_{0,4}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,4}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,4}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,4}\, b_1(\beta_1,\beta_2;u) \,]\, b_1(\beta_1,\beta_2;v) \quad + \\ & [\, \mathbf{V}_{0,3}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,3}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,3}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,3}\, b_1(\beta_1,\beta_2;u) \,]\, b_0(\beta_1,\beta_2;v) \quad + \\ & [\, \mathbf{V}_{0,2}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,2}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,2}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,2}\, b_1(\beta_1,\beta_2;u) \,]\, b_{-1}(\beta_1,\beta_2;v) \quad + \\ & [\, \mathbf{V}_{0,1}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,1}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,1}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,1}\, b_1(\beta_1,\beta_2;u) \,]\, b_{-2}(\beta_1,\beta_2;v) \end{aligned}$$

$\mathbf{Q}_{2,2}$ is similarly defined by

(E30)

$$\begin{aligned} \mathbf{Q}_{2,2}(u,v) = \\ & [\, \mathbf{V}_{0,3}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,3}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,3}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,3}\, b_1(\beta_1,\beta_2;u) \,]\, b_1(\beta_1,\beta_2;v) \quad + \\ & [\, \mathbf{V}_{0,2}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,2}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,2}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,2}\, b_1(\beta_1,\beta_2;u) \,]\, b_0(\beta_1,\beta_2;v) \quad + \\ & [\, \mathbf{V}_{0,1}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,1}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,1}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,1}\, b_1(\beta_1,\beta_2;u) \,]\, b_{-1}(\beta_1,\beta_2;v) \quad + \\ & [\, \mathbf{V}_{0,0}\, b_{-2}(\beta_1,\beta_2;u) + \mathbf{V}_{1,0}\, b_{-1}(\beta_1,\beta_2;u) + \mathbf{V}_{2,0}\, b_0(\beta_1,\beta_2;u) + \mathbf{V}_{3,0}\, b_1(\beta_1,\beta_2;u) \,]\, b_{-2}(\beta_1,\beta_2;v) \end{aligned}$$

We shall discuss the behavior of these patches at their common ("horizontal") boundary, which is $\mathbf{Q}_{2,3}(u,0)$ and $\mathbf{Q}_{2,2}(u,1)$. (The argument for common "vertical" boundaries is analogous, and is therefore omitted.)

First, of course, we must verify that the curves $\mathbf{Q}_{2,3}(u,0)$ and $\mathbf{Q}_{2,2}(u,1)$ are actually identical. For any fixed u we may rewrite (E29) and (E30) as

$$\mathbf{Q}_{bot}(v) = \mathbf{W}_{-2}\, b_{-2}(v) + \mathbf{W}_{-1}\, b_{-1}(v) + \mathbf{W}_0\, b_0(v) + \mathbf{W}_1\, b_1(v) \tag{E31}$$

and

$$\mathbf{Q}_{top}(v) = \mathbf{W}_{-1}\, b_{-2}(v) + \mathbf{W}_0\, b_{-1}(v) + \mathbf{W}_1\, b_0(v) + \mathbf{W}_2\, b_1(v) \tag{E32}$$

where

$$\begin{aligned}
\mathbf{W}_2 &= \mathbf{V}_{0,4}\,b_{-2}(u) + \mathbf{V}_{1,4}\,b_{-1}(u) + \mathbf{V}_{2,4}\,b_0(u) + \mathbf{V}_{3,4}\,b_1(u) \\
\mathbf{W}_1 &= \mathbf{V}_{0,3}\,b_{-2}(u) + \mathbf{V}_{1,3}\,b_{-1}(u) + \mathbf{V}_{2,3}\,b_0(u) + \mathbf{V}_{3,3}\,b_1(u) \\
\mathbf{W}_0 &= \mathbf{V}_{0,2}\,b_{-2}(u) + \mathbf{V}_{1,2}\,b_{-1}(u) + \mathbf{V}_{2,2}\,b_0(u) + \mathbf{V}_{3,2}\,b_1(u) \\
\mathbf{W}_{-1} &= \mathbf{V}_{0,1}\,b_{-2}(u) + \mathbf{V}_{1,1}\,b_{-1}(u) + \mathbf{V}_{2,1}\,b_0(u) + \mathbf{V}_{3,1}\,b_1(u) \\
\mathbf{W}_{-2} &= \mathbf{V}_{0,0}\,b_{-2}(u) + \mathbf{V}_{1,0}\,b_{-1}(u) + \mathbf{V}_{2,0}\,b_0(u) + \mathbf{V}_{3,0}\,b_1(u)
\end{aligned} \tag{E33}$$

As we have seen, along the common border $\beta_{2,3}(u,0)$ and $\beta_{2,2}(u,1)$ both reduce to $H(\beta_{2,1},\beta_{2,2};u)$. Hence the $\beta_1$ and $\beta_2$ which appear in (E29) and (E30) are identical, so that (E31) and (E32) are simply two successive segments on a uniformly-shaped Beta-spline curve. Hence $\mathbf{Q}_{bot}(1) = \mathbf{Q}_{top}(0)$. Hence $\mathbf{Q}_{2,2}(u,1) = \mathbf{Q}_{2,3}(u,0)$, as desired.

Tangent and curvature continuity between patches follow similarly if we apply the argument used earlier. Recall that the partial derivatives of $\beta_1(u,v)$ and $\beta_2(u,v)$ with respect to v for $v = 0$ and $v = 1$ are zero. If we fully expand equations (E29) or (E30), a typical term has the form

$$\frac{c\,[\beta_1(u,v)]^m\,[\beta_2(u,v)]^n\,u^p\,v^q}{[\beta_2(u,v)] + 2\,[\beta_1(u,v)]^3 + 4\,[\beta_1(u,v)]^2 + 4\,[\beta_1(u,v)] + 2}$$

If we then compute the first partial derivative of this term with respect to v we find, after repeated application of the product, quotient and chain rules, that the only resulting term which does not contain a product with at least one of

$$\frac{d}{dv}\,\beta_1(u,v) \quad \text{and} \quad \frac{d}{dv}\,\beta_2(u,v) \quad ,$$

both of which are zero, is

$$\frac{c\,q\,[\beta_1(u,v)]^m\,[\beta_2(u,v)]^n\,u^p\,v^{q-1}}{[\beta_2(u,v)] + 2\,[\beta_1(u,v)]^3 + 4\,[\beta_1(u,v)]^2 + 4\,[\beta_1(u,v)] + 2}$$

This is exactly the derivative which would have been obtained if $\beta_1$ and $\beta_2$ had not been functions of v. Therefore the first partial derivative of (E29) with respect to v, for any u and $v = 0$, is exactly

$$\mathbf{Q}^{(1)}_{top}(0) = \mathbf{W}_{-1}\,b^{(1)}_{-2}(0) + \mathbf{W}_0\,b^{(1)}_{-1}(0) + \mathbf{W}_1\,b^{(1)}_0(0) + \mathbf{W}_2\,b^{(1)}_1(0)$$

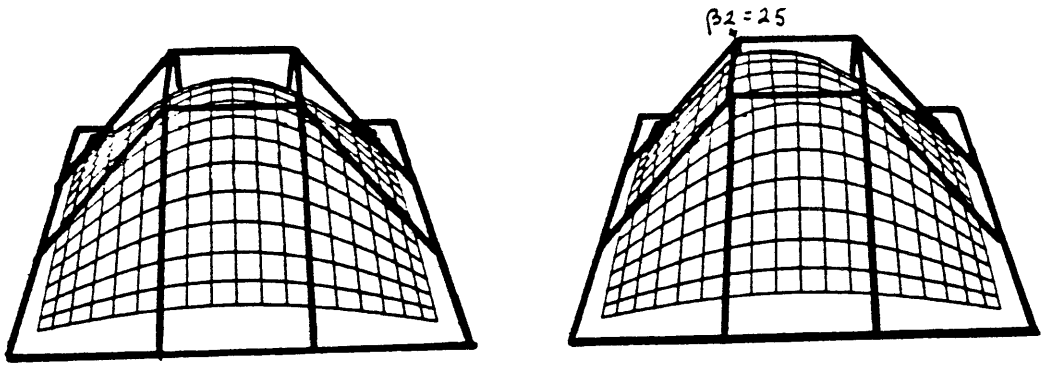
and the first partial derivative of (E30) with respect to v, for any u and $v = 1$, is exactly

$$\mathbf{Q}^{(1)}_{bot}(1) = \mathbf{W}_{-2}\,b^{(1)}_{-2}(1) + \mathbf{W}_{-1}\,b^{(1)}_{-1}(1) + \mathbf{W}_0\,b^{(1)}_0(1) + \mathbf{W}_1\,b^{(1)}_1(1)$$

These are simply the derivatives of two successive segments of a uniformly-shaped Beta-spline curves for particular values of $\beta_1$ and $\beta_2$, and we already know that such a curve has tangent continuity at its joints. Hence our surface has tangent continuity along its "horizontal" boundaries. The same argument works, *mutatis mutandis,* for the "vertical" boundaries as well, and generalizes to arbitrary patch boundaries, so that our surface is everywhere $G^1$ continuous.

An analogous argument suffices to establish curvature continuity.

$G^2$ continuity can also be directly verified using Vaxima by evaluating the Beta-spline constraint equations if (E28) is used to compute the values of $\beta_1$ and $\beta_2$. The algebra involved is, however, rather extensive...




Continuously shaped surfaces: $\beta_2$ is 0 except where indicated.

## 6. An Obvious Simpler Alternative - Which Doesn't Work

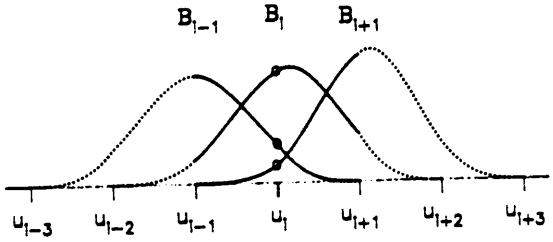
While the interpolated Beta-splines have many of the nice properties of the uniformly-shaped Beta-splines, the parametric functions X(u) and Y(u) which result are quotients of $18^{th}$ and $15^{th}$ degree polynomials. An obvious question is whether $G^2$ continuity, with local control of shape, can be obtained with lower degree polynomials. There is an obvious way to achieve this; unfortunately, the curves which result have some undesirable properties.

Suppose that we associate distinct $\alpha 1_i$ and $\alpha 2_i$ with each joint by directly inserting them into equations (E16); we obtain the following equations for the basis function $B_i(u)$ which is centered at $u = i$.

$$\begin{aligned}
0 &= b_1(0) & 0 &= b_1^{\{1\}}(0)\\
b_1(1) &= b_0(0) & \alpha 1_{i-1}\, b_1^{\{1\}}(1) &= b_0^{\{1\}}(0)\\
b_0(1) &= b_{-1}(0) & \alpha 1_i\, b_0^{\{1\}}(1) &= b_{-1}^{\{1\}}(0)\\
b_{-1}(1) &= b_{-2}(0) & \alpha 1_{i+1}\, b_{-1}^{\{1\}}(1) &= b_{-2}^{\{1\}}(0)\\
b_{-2}(1) &= 0 & \alpha 1_{i+2}\, b_{-2}^{\{1\}}(1) &= 0
\end{aligned} \qquad \text{(E34)}$$

$$\begin{aligned}
& & 0 &= b_1^{\{2\}}(0)\\
\alpha 1_{i-1}^2\, b_1^{\{2\}}(1) + & & \alpha 2_{i-1}\, b_1^{\{1\}}(1) &= b_0^{\{2\}}(0)\\
\alpha 1_i^2\, b_0^{\{2\}}(1) + & & \alpha 2_i\, b_0^{\{1\}}(1) &= b_{-1}^{\{2\}}(0)\\
\alpha 1_{i+1}^2\, b_{-1}^{\{2\}}(1) + & & \alpha 2_{i+1}\, b_{-1}^{\{1\}}(1) &= b_{-2}^{\{2\}}(0)\\
\alpha 1_{i+2}^2\, b_{-2}^{\{2\}}(1) + & & \alpha 2_{i+2}\, b_{-2}^{\{1\}}(1) &= 0
\end{aligned}$$

This is straightforward. The normalizing equation, however, is a problem. We might hope to obtain basis functions possessing the convex hull property, as we did for the uniformly-shaped Beta-splines, by asking that the sum of the basis segments at each joint be 1. But consider an arbitrary joint:




$b_{i+1,0}(u_i)$, $b_{i,-1}(u_i)$ and $b_{i-1,-2}(u_i)$ are circled

If the basis functions which are nonzero at the knot $u_i$ sum to 1 then, much as before, we have

$$b_{i+1,0}(0) + b_{i,-1}(0) + b_{i-1,-2}(0) = 1$$

In the case of uniformly-shaped Beta-splines, since the basis functions are all translates of one another, $b_{i+1,0}(u) = b_{i,0}(u)$ and $b_{i-1,-2}(u) = b_{i,-2}(u)$ so that we could write

$$b_{i,0}(0) + b_{i,-1}(0) + b_{i,-2}(0) = 1 \qquad \text{(E36)}$$

and, in fact, there was no need to identify the basis segments with a particular basis function since they were all identical. Since we are now associating distinct $\beta$ values with each joint we no longer expect that the basis functions will be translates of one another, and this substitution no longer makes sense. Indeed, there is a distinct version of equation (E35) for each knot $u_i$, so that all of the basis functions corresponding to a sequence of $\alpha 1_i$ and $\alpha 2_i$ values are defined by an interlinked system of equations. While these can, of course, be solved each time we altered an $\alpha_i$, the resulting basis functions would not change locally in response to a change in the value of a particular $\alpha$, and this is undesirable.

We might as well, then, continue to use equation (E36) because the basis segments which result will satisfy the convex hull property for a curve segment defined by four successive basis functions having the same $\alpha$ values.

Program P3 may be used to solve the representation of equations (E34) (created by program P6) and (E36). The resulting basis functions have the following form.

$$b_{i,1}(u) = \frac{1}{\delta}\left\{ [2t_1]u^3 \right\} \qquad \text{(E37)}$$

$$b_{i,0}(u) = \frac{1}{\delta}\left\{ [2t_1] + [6\alpha 1_{i-1}t_1]u + [3(\alpha 2_{i-1} + 2\alpha 1_{i-1}^2)t_1]u^2 + [t_3 - 2(\alpha 2_{i-1} + 2\alpha 1_{i-1}^2 + \alpha 1_{i-1})t_1]u^3 \right\}$$

$$b_{i,-1}(u) = \frac{1}{\delta}\left\{ [(\alpha 2_{i+1} + 2\alpha 1_{i+1}^3 + 4\alpha 1_{i+1}^2 + 2\alpha 1_{i+1})t_2 - \alpha 1_i t_3] + [3\alpha 1_i t_3]u - [3(\alpha 2_{i+1} + 2\alpha 1_{i+1}^2 + 2\alpha 1_{i+1})t_2 + 3\alpha 1_i t_3]u^2 + [2(\alpha 2_{i+1} + \alpha 1_{i+1}^2 + 2\alpha 1_{i+1})t_2 + \alpha 1_i t_3]u^3 \right\}$$

$$b_{i,-2}(u) = \frac{1}{\delta}\left\{ [2\alpha 1_{i+1}^3 t_2] - [6\alpha 1_{i+1}^3 t_2]u + [6\alpha 1_{i+1}^3 t_2]u^2 - [2\alpha 1_{i+1}^3 t_2]u^3 \right\}$$

where

$$t_1 = \alpha 2_{i+1}[2\alpha 2_i + 4\alpha 1_i^2 + 8\alpha 1_i + 4] + \alpha 1_{i+1}^3[4\alpha 2_i + 8\alpha 1_i^2 + 8\alpha 1_i] + \alpha 1_{i+1}^2[8\alpha 2_i + 16\alpha 1_i^2 + 24\alpha 1_i + 8] + \alpha 1_{i+1}[4\alpha 2_i + 8\alpha 1_i^2 + 16\alpha 1_i + 8]$$

$$t_2 = \alpha 2_{i-1}[2\alpha 2_i + 4\alpha 1_i^3 + 8\alpha 1_i^2 + 4\alpha 1_i] + \alpha 1_{i-1}^2[4\alpha 2_i + 8\alpha 1_i^3 + 16\alpha 1_i^2 + 8\alpha 1_i] + \alpha 1_{i-1}[8\alpha 2_i + 8\alpha 1_i^3 + 24\alpha 1_i^2 + 16\alpha 1_i] + [4\alpha 2_i + 8\alpha 1_i^2 + 8\alpha 1_i]$$

$$t_3 = \frac{[\alpha 2_{i+1} + 2\alpha 1_{i+1}(\alpha 1_{i+1} + 1)^2]t_2 - [\alpha 2_{i-1} + 2(\alpha 1_{i-1} + 1)^2]t_1}{\alpha 1_i + 1}$$

and $\delta$ is simply the sum of the constant terms of $b_0$, $b_{-1}$ and $b_{-2}$ (as is apparent from the normalizing equation).

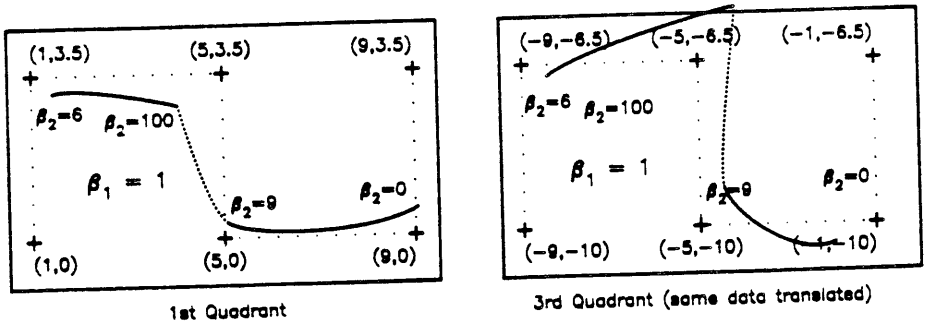
Although these equations are somewhat complex, they do define cubic polynomials, so that forward differencing can be used to evaluate them efficiently. Unfortunately the curves they define have a rather fatal flaw.

As one would guess from the preceding discussion, this technique does not yield basis functions which uniformly sum to one. To see that this is so, recall that any particular segment is given by

$$\begin{aligned} \mathbf{Q}_i(u) = {} & \mathbf{V}_{i-2} b_{-2}(\ \alpha 2_{i-1}, \alpha 1_{i-1},\ \alpha 2_i, \alpha 1_i,\ \alpha 2_{i+1}, \alpha 1_{i+1},\ u\ ) \\ & + \mathbf{V}_{i-1} b_{-1}(\ \alpha 2_i, \alpha 1_i,\ \alpha 2_{i+1}, \alpha 1_{i+1},\ \alpha 2_{i+2}, \alpha 1_{i+2},\ u\ ) \\ & + \mathbf{V}_i b_0(\ \alpha 2_{i+1}, \alpha 1_{i+1},\ \alpha 2_{i+2}, \alpha 1_{i+2},\ \alpha 2_{i+3}, \alpha 1_{i+3},\ u\ ) \\ & + \mathbf{V}_{i+1} b_1(\ \alpha 2_{i+2}, \alpha 1_{i+2},\ \alpha 2_{i+3}, \alpha 1_{i+3},\ \alpha 2_{i+4}, \alpha 1_{i+4},\ u\ ) \end{aligned}$$

where we have explicitly indicated the dependencies of each basis segment on $\alpha$ values. If for some particular set of $\alpha$ values the basis segments sum to one, then by altering any one of $\alpha 2_{i-1}$, $\alpha 1_{i-1}$, $\alpha 2_{i+4}$ or $\alpha 1_{i+4}$ we may alter one of the basis segments without altering any of the others, and thereby cause the sum to differ from one.

Hence the curve segments defined by this technique need not lie within the convex hull of their control vertices. A consequence of this is that the curves defined in the usual way by equations (E9) and (E37) are not translation independent, and the effect of altering an $\alpha$ value depends on where the joint in question is located. The following figure illustrates this fact.

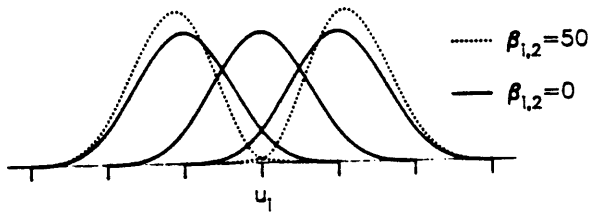



1st Quadrant

3rd Quadrant (same data translated)

The easiest way to understand this phenomenon is to examine the effect of increasing $\beta 2$ on the basis

functions.




It happens, for example, that if the $\alpha$ values at joints $u_{i-1}$ and $u_{i+1}$ are identical then the basis function $B_i(u)$ is completely independent of $\alpha 2_i$; increasing $\alpha 2_i$ in such a situation "pushes" $B_{i-1}(u)$ and $B_{i+1}(u)$ left and right, respectively, as in the above figure, and the sum of the basis functions at $u_i$ is less than one. When the basis functions are weighted by vertex coordinates, increasing $\beta 2$ then reduces the weight given to $\mathbf{V}_{i-1}$ and $\mathbf{V}_{i+1}$, "pulling" the joint toward the origin. Since translating the control vertices generally changes the relative direction of the origin, the result obtained by altering $\beta 2$ depends on the position of the joint in the coordinate plane - a highly objectionable characteristic which probably makes these curves impractical.

## 7. Conclusions

Though the continuously shaped Beta-splines are more expensive to compute than their uniformly-shaped brethren, they provide the first means of locally controlling the bias and tension in a cubic polynomial spline. This is an important feature in computer aided design applications. Although the continuously-varying Beta-splines are, in principle, rather high degree polynomials, they are naturally factored into tractable pieces. Indeed, they are interesting exactly because they provide a useful and convenient way of controlling higher than cubic piecewise polynomial curves.

## Appendix

### Program P1 - CreateSegs

```
/*
    Create cubic polynomial representations of the four segments forming
    a (de Boor) basis function.  Then compute their first and second
    derivatives as well.  The segments are numbered from right to left
    because in summing to form a curve the rightmost segment is weighted
    by the leftmost control point.
*/

poly(a,b,c,d) := a + b*u + c*u^2 + d*u^3;

seg0d0(u) := ''( poly(k00,k10,k20,k30) );
seg0d1(u) := ''( diff( seg0d0(u), u ) );
seg0d2(u) := ''( diff( seg0d1(u), u ) );

seg1d0(u) := ''( poly(k01,k11,k21,k31) );
seg1d1(u) := ''( diff( seg1d0(u), u ) );
seg1d2(u) := ''( diff( seg1d1(u), u ) );

seg2d0(u) := ''( poly(k02,k12,k22,k32) );
seg2d1(u) := ''( diff( seg2d0(u), u ) );
seg2d2(u) := ''( diff( seg2d1(u), u ) );

seg3d0(u) := ''( poly(k03,k13,k23,k33) );
seg3d1(u) := ''( diff( seg3d0(u), u ) );
seg3d2(u) := ''( diff( seg3d1(u), u ) );
```

## Program P2 - CreateEQ for Constant $\beta_1$ and $\beta_2$

```
/*
    Create the constraint equations appropriate for a constant value of beta1
    and beta2 throughout the curve.  It is assumed that CreateSegs has already
    been invoked to create the polynomial segidj.
*/

eq[ 1] : seg0d0(1) = 0;
eq[ 2] : seg1d0(1) = seg0d0(0);
eq[ 3] : seg2d0(1) = seg1d0(0);
eq[ 4] : seg3d0(1) = seg2d0(0);
eq[ 5] : seg3d0(0) = 0;

eq[ 6] : beta1 * seg0d1(1) = 0;
eq[ 7] : beta1 * seg1d1(1) = seg0d1(0);
eq[ 8] : beta1 * seg2d1(1) = seg1d1(0);
eq[ 9] : beta1 * seg3d1(1) = seg2d1(0);
eq[10] : beta1 *         0 = seg3d1(0) ;

eq[11] : beta1^2 * seg0d2(1) + beta2 * seg0d1(1) = 0;
eq[12] : beta1^2 * seg1d2(1) + beta2 * seg1d1(1) = seg0d2(0);
eq[13] : beta1^2 * seg2d2(1) + beta2 * seg2d1(1) = seg1d2(0);
eq[14] : beta1^2 * seg3d2(1) + beta2 * seg3d1(1) = seg2d2(0);
eq[15] : beta1^2 *         0 + beta2 *         0 = seg3d2(0) ;

eq[16] : seg0d0(0) + seg1d0(0) + seg2d0(0) + seg3d0(0) = 1;

for i:1 step 1 thru 16 do eq[i] : ratsimp( lhs(eq[i]) - rhs(eq[i]) = 0 );

save(equations,eq);
```

## Program P3 - SolveEQ

```
/*
    Solve the 16 equations placed in eqlist for the 16 unknowns in xlist using
    the standard linear equation solver. A transcript of the computation is
    written into the file "transcript". It is assumed that CreateSegs and
    CreateEQ have been executed previously in order to create the equations in
    eqlist. A distinct version of CreateEQ is used for differing constraint
    equations.
*/

writefile(transcript);
showtime:true;
eqlist:
   [  eq[ 1],eq[ 2],eq[ 3],eq[ 4],eq[ 5],eq[ 6],eq[ 7],eq[ 8],
      eq[ 9],eq[10],eq[11],eq[12],eq[13],eq[14],eq[15],eq[16]  ];
xlist: [ k00,k10,k20,k30, k01,k11,k21,k31, k02,k12,k22,k32, k03,k13,k23,k33 ];
answer:linsolve(eqlist,xlist);
closefile();
```

## Program P4 - Verify P3 for Interpolated $\beta$'s

```
/*
    Use the Beta-spline basis segments which are derived for constant beta1
    and beta2, but use quintic Hermite interpolation to vary beta1 and beta2
    between alpha1 and alpha2 values at each joint.  The first derivatives of
    beta1 and beta2 wrt u at the left and right of each segment are set to 0.
    Verify that the resulting curves satisfy the Beta-spline constraint
    equations, and that the basis segments sum to one for all u.
*/

Hermite(p1,p2,u) := p1 + 10*(p2-p1)*u^3 - 15*(p2-p1)*u^4 + 6*(p2-p1)*u^5;

batch("DefineSegs.v");

Curve( V0,V1,V2,V3, a1s,a1f, a2s,a2f, u ) :=
''(
      V0 * Seg0( a1s,a1f, a2s,a2f )
    + V1 * Seg1( a1s,a1f, a2s,a2f )
    + V2 * Seg2( a1s,a1f, a2s,a2f )
    + V3 * Seg3( a1s,a1f, a2s,a2f )
);

d1Curve( V0,V1,V2,V3, a1s,a1f, a2s,a2f, u ) :=
    ''( diff( Curve( V0,V1,V2,V3, a1s,a1f, a2s,a2f, u ), u ) );

d2Curve( V0,V1,V2,V3, a1s,a1f, a2s,a2f, u ) :=
    ''( diff( d1Curve( V0,V1,V2,V3, a1s,a1f, a2s,a2f, u ), u ) );
    );

/*
    Verify that the curve which results when interpolating beta1(u) and beta2(u)
    does, in fact, yield continuous unit tangent and curvature vectors.  If so
    each of the next three expressions should evaluate to 0.
*/
ratsimp(
      Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
    - Curve(    V1,V2,V3,VR,      a1cc,a1p1,      a2cc,a2p1, 0 )
);
ratsimp(
      a1cc*d1Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
         - d1Curve(    V1,V2,V3,VR,      a1cc,a1p1,      a2cc,a2p1, 0 )
);
ratsimp(
      a1cc*a1cc*d2Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
         + a2cc*d1Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
    -            d2Curve(    V1,V2,V3,VR,      a1cc,a1p1,      a2cc,a2p1, 0 )
);

/*  Verify that the basis segments sum to one for all values of u */
ratsimp(
    Seg0(a1lft,a1rgh,a2lft,a2rgh,u) +
    Seg1(a1lft,a1rgh,a2lft,a2rgh,u) +
    Seg2(a1lft,a1rgh,a2lft,a2rgh,u) +
    Seg3(a1lft,a1rgh,a2lft,a2rgh,u)
);
```

## Program P4 - Verify P3 for Interpolated $\beta$'s

```
/*
    Use the Beta-spline basis segments which are derived for constant beta1
    and beta2, but use quintic Hermite interpolation to vary beta1 and beta2
    between alpha1 and alpha2 values at each joint.  The first derivatives of
    beta1 and beta2 wrt u at the left and right of each segment are set to 0.
    Verify that the resulting curves satisfy the Beta-spline constraint
    equations, and that the basis segments sum to one for all u.
*/

Hermite(p1,p2,u) := p1 + 10*(p2-p1)*u^3 - 15*(p2-p1)*u^4 + 6*(p2-p1)*u^5;

batch("DefineSegs.v");

Curve( V0,V1,V2,V3, als,alf, a2s,a2f, u ) :=
''(
        V0 * Seg0( als,alf, a2s,a2f )
    +   V1 * Seg1( als,alf, a2s,a2f )
    +   V2 * Seg2( als,alf, a2s,a2f )
    +   V3 * Seg3( als,alf, a2s,a2f )
);

d1Curve( V0,V1,V2,V3, als,alf, a2s,a2f, u ) :=
    ''( diff( Curve( V0,V1,V2,V3, als,alf, a2s,a2f, u ), u ) );

d2Curve( V0,V1,V2,V3, als,alf, a2s,a2f, u ) :=
    ''( diff( d1Curve( V0,V1,V2,V3, als,alf, a2s,a2f, u ), u ) );
    );

/*
    Verify that the curve which results when interpolating beta1(u) and beta2(u)
    does, in fact, yield continuous unit tangent and curvature vectors.  If so
    each of the next three expressions should evaluate to 0.
*/
ratsimp(
      Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
    - Curve(    V1,V2,V3,VR,      a1cc,a1p1,      a2cc,a2p1, 0 )
);
ratsimp(
      a1cc*d1Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
         - d1Curve(    V1,V2,V3,VR,      a1cc,a1p1,      a2cc,a2p1, 0 )
);
ratsimp(
      a1cc*a1cc*d2Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
         + a2cc*d1Curve( VL,V1,V2,V3,    a1m1,a1cc,      a2m1,a2cc,      1 )
    -          d2Curve(    V1,V2,V3,VR,      a1cc,a1p1,      a2cc,a2p1, 0 )
);

/*  Verify that the basis segments sum to one for all values of u */
ratsimp(
    Seg0(a1lft,a1rgh,a2lft,a2rgh,u) +
    Seg1(a1lft,a1rgh,a2lft,a2rgh,u) +
    Seg2(a1lft,a1rgh,a2lft,a2rgh,u) +
    Seg3(a1lft,a1rgh,a2lft,a2rgh,u)
);
```

**Program P5** - $C^2$ Continuity of b1(u) and b2(u) does not imply $G^2$ continuity

```
/*
   Use the Beta-spline basis segments which are derived for constant beta1
   and beta2, but with quintic Hermite interpolation to vary beta1 and beta2
   between alpha1 and alpha2 values at each joint.  The first and second
   derivatives of beta1 and beta2 wrt u at the left and right of each segment
   are set to d1u and d2u. This computation then demonstrates that for some
   particular alpha and coordinate values the left and right first derivative
   vectors at a joint are not colinear.
*/

writefile("out10");
ttyoff:true;

Hermite(p1,p2,u) :=
                                        p1
                                        d1u * u
    +                                   d2u * u^2 / 2
    +
    -   ( 10*(p1-p2) + 10*d1u + d2u ) * u^3
    +   ( 30*(p1-p2) + 30*d1u + d2u ) * u^4 / 2
    -   (  6*(p1-p2) +  6*d1u       ) * u^5    ;

batch("DefineSegs.v");

Curve( v0,v1,v2,v3, a1s,a1f, a2s,a2f, d1u,d2u, u ) :=
''(
        v0 * Seg0( a1s,a1f, a2s,a2f )
    +   v1 * Seg1( a1s,a1f, a2s,a2f )
    +   v2 * Seg2( a1s,a1f, a2s,a2f )
    +   v3 * Seg3( a1s,a1f, a2s,a2f )
);

d1Curve( v0,v1,v2,v3, a1s,a1f, a2s,a2f, d1u,d2u, u ) :=
    ''( diff( Curve( v0,v1,v2,v3, a1s,a1f, a2s,a2f, d1u,d2u, u ), u ) );

/* Declarations... */
array( lsd, 2 );
array( rsd, 2 );
array( ld, float, 2 );
array( rd, float, 2 );

ttyoff : false;
typeset : true;

/* Compute the left and right derivatives at a joint - NOTE that a1cc = 5 */
d1lft(v1,v2,v3) :=
    ''(ratsimp( d1Curve( v0,v1,v2,v3, a1m1,5, a2m1,2, 5,3, 1 ) ));
d1rgh(v1,v2,v3) :=
    ''(ratsimp( d1Curve( v1,v2,v3,v4, 5,a1p1, 2,a2p1, 5,3, 0 ) ));

/* Evaluate the 1st derivative at this joint for the particular points
** (-2,4), (-3,2), and (1,1) */
lsd[1] : ratsimp( d1lft( -2, -3,  1 ) );
lsd[2] : ratsimp( d1lft(  4,  2,  1 ) );
rsd[1] : ratsimp( d1rgh( -2, -3,  1 ) );
rsd[2] : ratsimp( d1rgh(  4,  2,  1 ) );

/* Turn these into genuine floating point values and normalize the results to
** yield unit tangent vectors (which are not identical...).  */
ld[1] : float( lsd[1] );
ld[2] : float( lsd[2] );
rd[1] : float( rsd[1] );
rd[2] : float( rsd[2] );
lbot  : sqrt( ld[1]^2 + ld[2]^2 );
ld[1] : ld[1] / lbot;
ld[2] : ld[2] / lbot;
rbot  : sqrt( rd[1]^2 + rd[2]^2 );
rd[1] : rd[1] / rbot;
rd[2] : rd[2] / rbot;
```

```
/*****************************************************************************/

/*
    Perform the same computation, except use zeros for d1u and d2u.
*/
d1lft(v1,v2,v3) :=
    ''(ratsimp( d1Curve( v0,v1,v2,v3, a1m1,5, a2m1,2, 0,0, 1 ) ));
d1rgh(v1,v2,v3) :=
    ''(ratsimp( d1Curve( v1,v2,v3,v4, 5,a1p1, 2,a2p1, 0,0, 0 ) ));

/* Evalutate the 1st derivative at this joint for the particular points
** (-2,4), (-3,2), and (1,1) */
lsd[1] : ratsimp( d1lft( -2, -3,  1 ) );
lsd[2] : ratsimp( d1lft(  4,  2,  1 ) );
rsd[1] : ratsimp( d1rgh( -2, -3,  1 ) );
rsd[2] : ratsimp( d1rgh(  4,  2,  1 ) );

/* Turn these into genuine floating point values and normalize the results to
** yield unit tangent vectors (which ARE identical...). */
ld[1] : float( lsd[1] );
ld[2] : float( lsd[2] );
rd[1] : float( rsd[1] );
rd[2] : float( rsd[2] );
lbot  : sqrt( ld[1]^2 + ld[2]^2 );
ld[1] : ld[1] / lbot;
ld[2] : ld[2] / lbot;
rbot  : sqrt( rd[1]^2 + rd[2]^2 );
rd[1] : rd[1] / rbot;
rd[2] : rd[2] / rbot;

/*****************************************************************************/

/* Finally, check the Hermite interpolating formula for correctness. */
ratsimp(Hermite(a,b,0));
ratsimp(Hermite(a,b,1));
d1Hermite(a,b,u) := ''( diff(Hermite(a,b,u),u) );
ratsimp(d1Hermite(a,b,0));
ratsimp(d1Hermite(a,b,1));
d2Hermite(a,b,u) := ''( diff(d1Hermite(a,b,u),u) );
ratsimp(d2Hermite(a,b,0));
ratsimp(d2Hermite(a,b,1));

closefile("out10");
typeset : false;
```

**Partial output from Program P5**

```
(c22) /* Compute the left and right derivatives at a joint - NOTE that alcc = 5 */
d1lft(v1,v2,v3) :=
   "(ratsimp( d1Curve( v0,v1,v2,v3, a1m1,5, a2m1,2, 5,3, 1 ) ));
```

$$\text{d1lft}(v1,v2,v3) := \frac{147v3 + 9528v2 - 9675v1}{69938} \tag{d22}$$

```
(c23) d1rgh(v1,v2,v3) :=
   "(ratsimp( d1Curve( v1,v2,v3,v4, 5,a1p1, 2,a2p1, 5,3, 0 ) ));
```

$$\text{d1rgh}(v1,v2,v3) := \frac{4635v3 + 117240v2 - 121875v1}{69938} \tag{d23}$$

```
(c24) /* Evaluate the 1st derivative at this joint for the particular points
** (-2,4), (-3,2), and (1,1) */
* * * (omitted material) * * *
(c32) lbot : sqrt( ld[1]^2 + ld[2]^2 );
```

$$0.3075669231657496 \tag{d32}$$

```
(c33) ld[1] : ld[1] / lbot;
```

$$-0.4224425847634948 \tag{d33}$$

```
(c34) ld[2] : ld[2] / lbot;
```

$$-0.9063896858296312 \tag{d34}$$

```
(c35) rbot : sqrt( rd[1]^2 + rd[2]^2 );
```

$$3.846588867538818 \tag{d35}$$

```
(c36) rd[1] : rd[1] / rbot;
```

$$-0.3841125214514973 \tag{d36}$$

```
(c37) rd[2] : rd[2] / rbot;
```

$$-0.9232862886798293 \tag{d37}$$

```
(c38) /*************************************************************************/
/*
   Perform the same computation, except use zeros for d1u and d2u.
*/
d1lft(v1,v2,v3) :=
   "(ratsimp( d1Curve( v0,v1,v2,v3, a1m1,5, a2m1,2, 0,0, 1 ) ));
```

$$\text{d1lft}(v1,v2,v3) := \frac{3v3 + 72v2 - 75v1}{187} \tag{d38}$$

```
(c39) d1rgh(v1,v2,v3) :=
   "(ratsimp( d1Curve( v1,v2,v3,v4, 5,a1p1, 2,a2p1, 0,0, 0 ) ));
```

$$\text{d1rgh}(v1,v2,v3) := \frac{15v3 + 360v2 - 375v1}{187} \tag{d39}$$

```
* * * (omitted material) * * *
(c48) lbot : sqrt( ld[1]^2 + ld[2]^2 );
                                    0.8848288364045568                    (d48)

(c49) ld[1] : ld[1] / lbot;
                                    -0.3807498052542948                   (d49)

(c50) ld[2] : ld[2] / lbot;
                                    -0.924678098474716                    (d50)

(c51) rbot : sqrt( rd[1]^2 + rd[2]^2 );
                                    4.424144032022784                     (d51)

(c52) rd[1] : rd[1] / rbot;
                                    -0.3807498052542948                   (d52)

(c53) rd[2] : rd[2] / rbot;
                                    -0.924678098474716                    (d53)
```

## Program P8 - CreateEQ for Varying $\beta_1$ and $\beta_2$

```
/*
    Create the constraint equations for varying beta-splines. That is,
    distinct values of beta1 and beta2 are allowed at each joint. The
    naming scheme is as follows. The peak/center of the (de Boor) basis
    function  is represented by the suffix "cc"; the joint one knot to
    its left  is represented by the suffix "m1"; the joint one knot to
    its right is represented by the suffix "p1"; and so on. "a1" stands
    for alpha1 and a2 stands for alpha2. By convention the term "alphai"
    is used to represent a value of betai at some fixed value of the
    parameter u.
*/

eq[ 1] : seg0d0(1) = 0;
eq[ 2] : seg1d0(1) = seg0d0(0);
eq[ 3] : seg2d0(1) = seg1d0(0);
eq[ 4] : seg3d0(1) = seg2d0(0);
eq[ 5] : seg3d0(0) = 0;

eq[ 6] : a1p2 * seg0d1(1) = 0;
eq[ 7] : a1p1 * seg1d1(1) = seg0d1(0);
eq[ 8] : a1cc * seg2d1(1) = seg1d1(0);
eq[ 9] : a1m1 * seg3d1(1) = seg2d1(0);
eq[10] : a1m2 *         0 = seg3d1(0) ;

eq[11] : a1p2^2 * seg0d2(1) + a2p2 * seg0d1(1) = 0;
eq[12] : a1p1^2 * seg1d2(1) + a2p1 * seg1d1(1) = seg0d2(0);
eq[13] : a1cc^2 * seg2d2(1) + a2cc * seg2d1(1) = seg1d2(0);
eq[14] : a1m1^2 * seg3d2(1) + a2m1 * seg3d1(1) = seg2d2(0);
eq[15] : a1m2^2 *         0 + a2m2 *         0 = seg3d2(0) ;

eq[16] : seg0d0(0) + seg1d0(0) + seg2d0(0) + seg3d0(0) = 1;

for i:1 step 1 thru 16 do eq[i] : ratsimp( lhs(eq[i]) - rhs(eq[i]) = 0 );

save(equations,eq);
```

## 8. References


[Barsky81a] Brian A. Barsky (1981), The Beta-spline: A Local Representation Based On Shape Parameters and Fundamental Geometric Measures, PhD dissertation, Department of Computer Science, University of Utah, December.

[Barsky82a] Brian A. Barsky (1982), Exponential and Polynomial Methods for Applying Tension to an Interpolating Spline Curve [submitted for publication].

[Barsky82b] Brian A. Barsky (1982), A Study of the Parametric Uniform B-spline Curve and Surface Representations [in preparation].

[Barsky82c] Brian A. Barsky (1982), The Beta-spline: A Curve and Surface Representation for Computer Graphics and Computer Aided Geometric Design [submitted for publication].

[Barsky82d] Brian A. Barsky (1982), Algorithms for the Evaluation and Perturbation of Beta-splines [submitted for publication].

[Bézier70a] Pierre E. Bézier (1970), *Emploi des machines à commande numérique*, Masson et Cie., Paris [Translated by A. Robin Forrest and Anne F. Pankhurst as *Numerical Control -- Mathematics and Applications*, John Wiley and Sons, Ltd., London, 1972].

[Bézier77a] Pierre E. Bézier (1977), Essai de definition numérique des courbes et des surfaces expérimentales, PhD dissertation, l'Université Pierre et Marie Curie, Paris, February.

[Bogen77a] Richard Bogen, Jeffrey Golden, Michael Genesereth, and Alexander Doohovskoy (1977), *MACSYMA Reference Manual*, Massachusetts Institute of Technology [version nine].

[deBoor72a] Carl de Boor (1972), On Calculating with B-splines, *Journal of Approximation Theory* 6(1), July, 50-62.

[deBoor78a] Carl de Boor (1978), *A Practical Guide to Splines*, Applied Mathematical Sciences Volume 27, Springer-Verlag.

[Cline74a] Alan K. Cline (1974), Scalar- and Planar-Valued Curve Fitting Using Splines Under Tension, *Communications of the ACM* 17(4), April, 218-220 [also in *Atmos. Tech.*, No. 3, 1973, pp. 60-65].

[Coons64a] Steven A. Coons (1964), Surfaces for Computer Aided Design, Design Division, Mech. Engin. Dept., M.I.T., Cambridge, Massachusetts.

[Coons67a] Steven A. Coons (1967), Surfaces for Computer-Aided Design of Space Forms, MAC-TR-41, Project MAC, M.I.T., Cambridge, Massachusetts, June [Available as AD-663 504 from NTIS, Springfield, Virginia.].

[Cox72a] Morris G. Cox (1972), The Numerial Evaluation of B-splines, *J. Inst. Maths. Applics.* 10, 134-149.

[Fateman82a] Richard J. Fateman (1982), Addendum to the MACSYMA Reference Manual for the VAX, University of California, Berkeley.

[Gordon74a] William J. Gordon and Richard F. Riesenfeld (1974), B-spline Curves and Surfaces, *Computer Aided Geometric Design*, Robert E. Barnhill & Richard F. Riesenfeld (eds.), Academic Press, 95-126.

[Lane77a] Jeffrey M. Lane (1977), Shape Operators for Computer Aided Geometric Design, PhD dissertation, University of Utah, Salt Lake City, Utah, June.

[Nielson74a] Gregory M. Nielson (1974), Some Piecewise Polynomial Alternatives to Splines under Tension, *Computer Aided Geometric Design*, Robert E. Barnhill & Richard F. Riesenfeld (eds.), Academic Press, New York, 209-235.

[Nielson74b] Gregory M. Nielson, William M. Newman, and Robert F. Sproull (1973), *Principles of Interactive Computer Graphics*, McGraw-Hill.

[Pilcher73a] David T. Pilcher (1973), Smooth Approximation of Parametric Curves and Surfaces, PhD dissertation, University of Utah, Salt Lake City, Utah, August.

[Riesenfeld73a] Richard F. Riesenfeld (1973), Applications of B-spline Approximation to Geometric Problems of Computer-Aided Design, PhD dissertation, Department of Systems and Information Science, Syracuse University, May.

[Schweikert66a] Daniel G. Schweikert (1966), An Interpolation Curve Using a Spline in Tension, *Journal of Mathematics and Physics* 45, 312-317.