

Determination of End Conditions for NURB Surface Interpolation

Anshuman Razdan
Technical Director PRISM
Arizona State University, Tempe AZ 85287-5106

Gerald Farin
Professor, Computer Sc. & Engineering
Arizona State University, Tempe AZ 85287-5406

November 21, 1996

1 Introduction

NURB (Non Uniform Rational B-spline) curves and surfaces are being used to solve curve and surface fitting problems i.e. where a sequence (or an array) of points is given and then one can find a NURB curve (or an surface) that interpolates to the given data. In general one has to provide additional information, called *end conditions* to solve the system of equations in the NURB interpolation problem. Several solutions have been proposed and used for the non rational case. Here, we present some ideas in determining *end conditions* in the rational case.

2 Motivation

We present solution to the end condition problem in the following context. We are given a NURB curve or surface that is poorly parametrized i.e. the parametrization does not correspond to the geometry. Use of this curve or surface in an application (such as grid generation for fluid dynamics analysis) may require a better representation where parametrization reflects the geometry [10]. One solution to the problem is to pick points (and therefore weights associated with the points) off the original curve or surface and solve the NURB interpolation problem [9]. Although specific to the above

context, the ideas for determining end conditions presented here may be applied to other NURB interpolation problem irrespective of the source of the interpolation points.

3 NURB Curve Interpolation

First, we briefly visit the NURB curve interpolation problem. We are given a set of data points: $\mathbf{x}_0, \dots, \mathbf{x}_L$, where $\mathbf{x}_i \in [w_i \mathbf{x}_i, w_i]$ and $\mathbf{x}_i \in \mathbb{E}^3$ and w_i are the weights. We are also given the corresponding parameter values (or knots) u_0, \dots, u_L . We want a cubic NURB curve \mathbf{c} , determined by the same knots and unknown control vertices $\mathbf{d}_{-1}, \dots, \mathbf{d}_{L+1}$ such that $\mathbf{c}(u_i) = \mathbf{x}_i$. In other words, such that \mathbf{c} *interpolates* to the data points.

We can write every rational B-spline curve as a rational Bézier curve. In that form we have:

$$\mathbf{x}_i = \mathbf{b}_{3i}; \quad i = 0, \dots, L.$$

Further algebra on the relationship of Bézier and inner Bézier points ($\mathbf{b}_{3i \pm 1}$) leads to a *tri-diagonal* linear system of equations. The system can be made *diagonally dominant* and *symmetric*. It can be solved using *LU Decomposition*. The matrix is invertible and always has a unique solution [3]. The result is a NURB curve control polygon \mathbf{d}_i , $i = -1 \dots L + 1$. The piecewise rational polynomial will interpolate to the \mathbf{x}_i . Each NURB control point in the set \mathbf{d}_i has a weight associated with it.¹ We do need to choose two Bézier points, \mathbf{b}_1 and \mathbf{b}_{3L-1} . This is known as *end condition problem*. There are several methods such as *Bessel*, *quadratic and not-a-knot* end conditions for the *non-rational* i.e. B-spline case. These fail to qualify in the rational case because of several reasons; see Section 4.

4 End Conditions for NURB Curves

The end condition amounts to finding part of the first Bézier polygon at the curve ends. Several methods exist in the non-rational case for computing end conditions, such as *Bessel*, *not-a-knot* etc. The Bessel end condition requires passing a quadratic (parabola) through the first three interpolation

¹Else, the weight of all points will be 1.0 for the non-rational piecewise polynomial case.

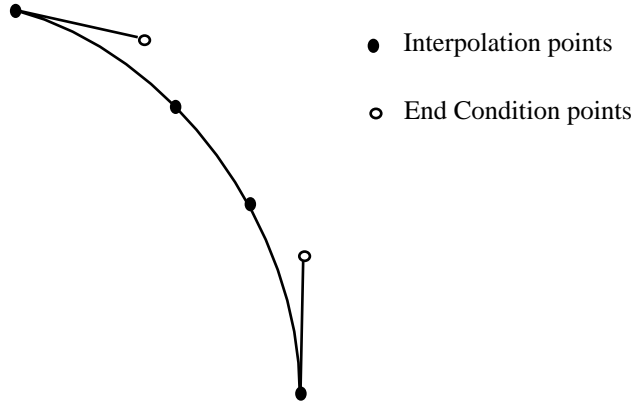


Figure 1: End Condition Problem for a Curve

points. That gives the Bézier control polygon and therefore satisfies the requirements of Section 3 described above. In most cases it works very well.

There is a problem with this approach. It arises because we are dealing with rational polynomials. In the rational case, we have to compute the location of the point in \mathbb{E}^3 and the corresponding weight. Let us look at the problem in the context of projective geometry. The interpolation

points $[w_i \mathbf{p}_i, w_i]$, $\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$, in \mathbb{E}^3 are points $\underline{\mathbf{p}}_i$, $\underline{\mathbf{p}}_i = \begin{bmatrix} w_i x_i \\ w_i y_i \\ w_i z_i \\ w_i \end{bmatrix}$, in \mathbb{P}^3 .

The points $\underline{\mathbf{p}}_i$ in \mathbb{P}^3 are the homogenized coordinates of \mathbf{p}_i or \mathbf{p}_i are the unhomogenized coordinates of $\underline{\mathbf{p}}_i$. If the *weight* of a point was considered as yet another ordinate and we applied the Bessel end condition, there is a possibility that we can end up with a negative weight. An example of the condition is as follows. If the weights of the first two points are close to unity and the weight of the third point is very high, then the parabola will dip below the 0.0 axis, as illustrated in Figure 2, resulting in a negative weight. This is undesirable.²

In our context we have additional information such as the tangent directions and curvature information at the end points to help determine the end conditions.

²The problem of negative weights for the control points can happen inside the curve also (even though the weights of interpolation points are positive) if the choice of interpolation points is poor. However, since the interpolation points are chosen based on curvature and arc length of the curve [9], the likelihood of this happening is reduced.

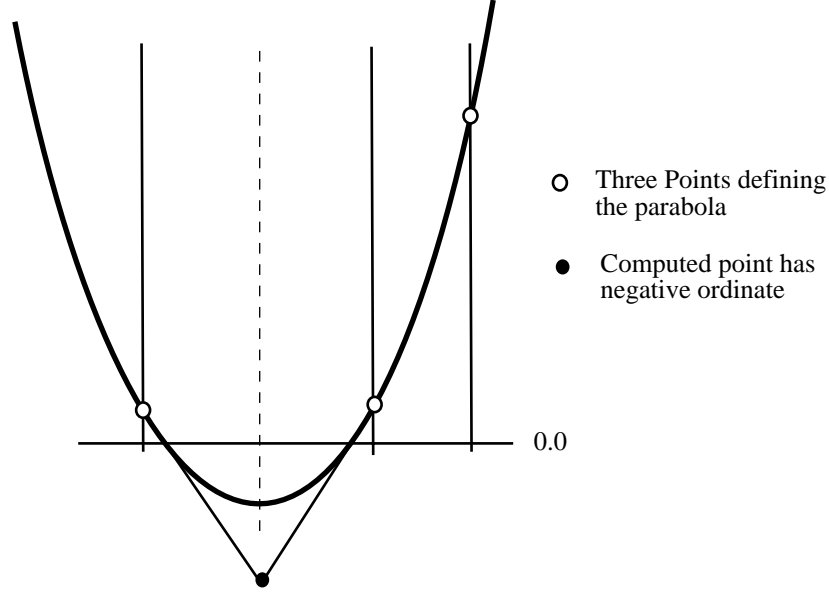


Figure 2: Bessel end condition generates negative weights

4.1 Parabola as a Rational Curve

We present a method to find the rational conic that will project to a parabola in the affine plane. This will be useful for fixing the tangents at the ends of the curve.

Given: Two points, $[w_0 \mathbf{b}_0, w_0]$ and $[w_2 \mathbf{b}_2, w_2]$, in \mathbb{E}^3 , curvature (κ_0) at \mathbf{b}_0 and the unit vector $\vec{\mathbf{v}}_0$, the tangent direction at \mathbf{b}_0 .

Wanted: A conic which projects to a parabola in the affine plane such that the projected curve passes through \mathbf{b}_0 and \mathbf{b}_2 , has curvature, κ_0 at \mathbf{b}_0 and has the tangent direction $\vec{\mathbf{v}}_0$. In other words we need to find the Bézier point $[w_1 \mathbf{b}_1]$ in \mathbb{E}^3 such that the above condition is satisfied.

Method: The equation of a quadratic rational Bézier curve can be written as follows:

$$\mathbf{b}(u) = \frac{w_0 \mathbf{b}_0 B_0^2(u) + w_1 \mathbf{b}_1 B_1^2(u) + w_2 \mathbf{b}_2 B_2^2(u)}{w_0 B_0^2(u) + w_1 B_1^2(u) + w_2 B_2^2(u)} \quad (1)$$

A rational Bézier curve (of degree 2) can be reparametrized such that:

$$\mathbf{b}(\hat{u}) = \frac{\mathbf{W}_0 \mathbf{b}_0 B_0^2(\hat{u}) + \mathbf{W}_1 \mathbf{b}_1 B_1^2(\hat{u}) + \mathbf{W}_2 \mathbf{b}_2 B_2^2(\hat{u})}{\mathbf{W}_0 B_0^2(\hat{u}) + \mathbf{W}_1 B_1^2(\hat{u}) + \mathbf{W}_2 B_2^2(\hat{u})} \quad (2)$$

where $\mathbf{W}_0 = 1.0$, $\mathbf{W}_1 = \frac{w_1}{\sqrt{w_0 w_2}}$ and $\mathbf{W}_2 = 1.0$. We can classify a conic (in the standard form) as a parabola, ellipse or a hyperbola [6] in the affine plane based on $W_1 = 1, < 1$ or > 1 . Therefore:

$$\frac{w_1}{\sqrt{w_0 w_2}} = 1.0 \quad (3)$$

or

$$w_1 = \sqrt{w_0 w_2} \quad (4)$$

Also, from [4], we have the formula for curvature κ_0 at \mathbf{b}_0 :

$$\kappa_0 = \frac{w_0 w_2 \text{ area} [\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2]}{w_1^2 \text{ dist}^3 [\mathbf{b}_0, \mathbf{b}_1]} \quad (5)$$

We know w_0, w_1 and w_2 and we need to find \mathbf{b}_1 . We can write \mathbf{b}_1 as:

$$\mathbf{b}_1 = \mathbf{b}_0 + \alpha \vec{\mathbf{v}}_0 \quad (6)$$

where α is the scaling factor and unknown. Equations 5 and 6 can be combined:

$$\alpha^2 = \frac{2 \vec{\mathbf{v}}_0 \wedge (\mathbf{b}_2 - \mathbf{b}_0)}{\kappa_0 \|\vec{\mathbf{v}}_0\|^3} \quad (7)$$

where “ \wedge ” denotes the vector cross product. Therefore, α can be determined. We only use the positive value of α for obvious reasons. Thus from

$$\mathbf{b}_1 = \mathbf{b}_0 + \alpha \vec{\mathbf{v}}_0 \quad (8)$$

we find \mathbf{b}_1 .

Next, we degree elevate the conic to get a cubic. The Bézier point $\hat{\mathbf{b}}_1$ is computed as:

$$\hat{\mathbf{b}}_1 = \frac{2}{3} \mathbf{b}_1 + \frac{1}{3} \mathbf{b}_0 \quad (9)$$

We call the above *Rational Two Point Parabolic (RTPP)* end condition.

4.2 Bessel End Condition for Rational Curves

The Bessel end condition in the nonrational polynomial case implies finding a parabola given three points (and the parameter values) that lie on it. We are given three points $[w_i \mathbf{x}_i, w_i]$, $w_i \mathbf{x}_i$ in \mathbb{E}^3 , and u_i , $i = 0, 1, 2$. We find the Bézier polygon, $\mathbf{b}_0, \mathbf{b}_1$ and \mathbf{b}_2 , such that $\mathbf{b}(u)$ traces a parabola and interpolates to \mathbf{x}_i . The solution is fairly straight-forward. However, the solution in the rational case is not as simple. We can work around the problem and obtain the solution is as follows.

Given: Three points, $[w_i \mathbf{x}_i, w_i]$, $i = 0, 1, 2$, in \mathbb{E}^3 .

Wanted: A conic which projects to a parabola in the affine plane such that the projected curve passes through Bézier points \mathbf{x}_i .

Method: If we want a parabola in the affine plane, we must set the middle weight of the rational conic (after reparametrizing it to the standard form) to 1.0. Following same steps as in Section 4.1 we can find w_1 . We also set $\mathbf{b}_0 = \mathbf{x}_0$ and $\mathbf{b}_2 = \mathbf{x}_2$ and find \mathbf{b}_1 from:

$$\mathbf{x}_1(u_1) = \frac{\mathbf{b}_0 B_0^2(u_1) + \frac{w_1}{\sqrt{w_0 w_2}} \mathbf{b}_1 B_1^2(u_1) + \mathbf{b}_2 B_2^2(u_1)}{B_0^2(u_1) + \frac{w_1}{\sqrt{w_0 w_2}} B_1^2(u_1) + B_2^2(u_1)} \quad (10)$$

or

$$\mathbf{b}_1 = \frac{\mathbf{x}_1(u_1) \left[B_0^2(u_1) + \frac{w_1}{\sqrt{w_0 w_2}} B_1^2(u_1) + B_2^2(u_1) \right] - [\mathbf{b}_0 B_0^2(u_1) + \mathbf{b}_2 B_2^2(u_1)]}{\frac{w_1}{\sqrt{w_0 w_2}} B_1^2(u_1)} \quad (11)$$

We call the above solution to the problem of *Rational Bessel End Condition*.

5 NURB Surface Interpolation

The concept of interpolation is easily extended to the surface case. In the bivariate case we are given $\underline{\mathbf{x}}_{0,0}, \dots, \underline{\mathbf{x}}_{L,M}$, where $\underline{\mathbf{x}}_{i,j} \in [w_{i,j} \mathbf{x}_{i,j}, w_{i,j}]$ and $\mathbf{x}_{i,j}$ in \mathbb{E}^3 . The result is the NURB surface control net $\underline{\mathbf{d}}_{i,j}$, $i = -1 \dots L + 1$, $j = -1 \dots M + 1$. We have the additional responsibility of choosing twist vectors at the four inner corner points and tangents along the boundary curves besides the tangents at the surface corners.

In our context (extracting data from a given surface) we have several kinds of information we can use, such as tangent directions along the boundary curves, curvatures of boundary curves at surface corners and tangent planes along the boundary curves.

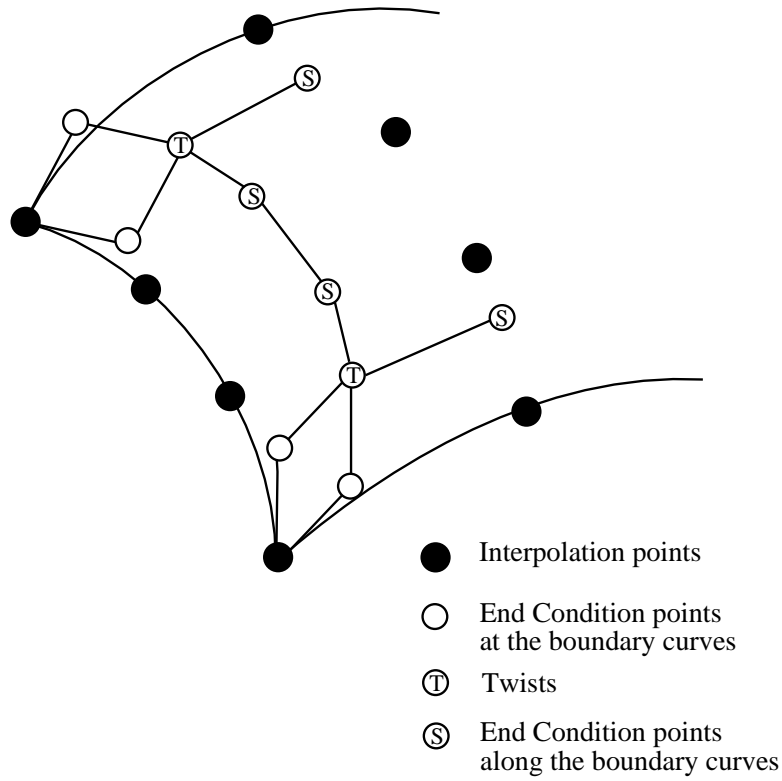


Figure 3: End condition requirements for a surface interpolation problem

6 End Conditions for NURB Surfaces

We refer to Section 5 for the statement of the NURB surface interpolation problem. We still have to fix some values before we can solve the *tri-diagonal* linear system of equations. The additional information needed is (see Figure 3):

1. tangents at the surface corners,
2. tangents along the boundary curves and
3. twist vectors at the four corner points.

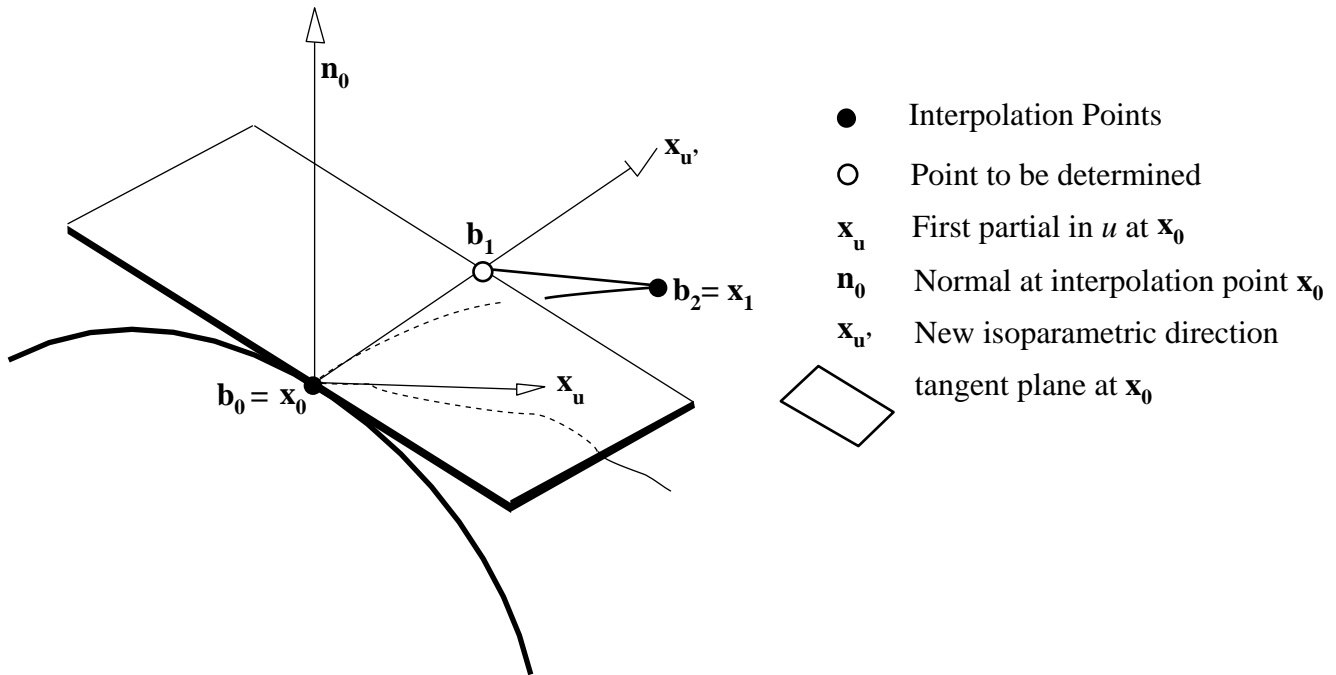


Figure 4: Finding Bézier points on the tangent strip.

6.1 Tangents at the Surface Corners

Determining tangents at the surface amounts to finding tangents at the end points in the NURB curve interpolation problem, the solution to which is already described earlier.

6.2 End Conditions along the Boundary Curves

Refer to Figure 3. The points marked “S” are the points along the boundary curves. We are limited here because unlike the four corners of the surface, we do not have prescribed tangent directions or curvatures. We do have the normals to the surface at the two neighboring interpolation points. Nielson [7] and Piper [8] have developed methods to do Point Normal interpolation to produce a cubic curve. This method could be used here except there is a heuristic value associated with determining the length of the tangent. Also, this is valid in the affine plane and not in projective space. This does not provide a way to adequately determine the weights of the Bézier control polygon. We present a way to solve this problem, however, with the

quadratic (parabola) and not a cubic. We attempt to find the parabola in the affine space (conic in projective space) that can be described under given conditions. Refer to Figure 4. We know the two interpolation points. One of the interpolation points, \mathbf{x}_0 , lies on the boundary curve and the other, \mathbf{x}_1 , on the interior of the surface as shown. Also shown are the normal, $\vec{\mathbf{n}}_0$, and the first derivatives (tangent directions) along the parametric lines, $\vec{\mathbf{x}}_u$ and $\vec{\mathbf{x}}_v$. It is clear that in general the interpolation point \mathbf{x}_1 will not lie in the direction of $\vec{\mathbf{x}}_u$. We want to preserve certain properties. There exists a tangent plane from the given surface at \mathbf{x}_0 and we do not want that modified. Let $\vec{\mathbf{x}}'_u$ be the vector containing the intersection of the tangent plane at \mathbf{x}_0 and the plane containing $\vec{\mathbf{n}}_0$, \mathbf{x}_0 and \mathbf{x}_1 . In other words, $\vec{\mathbf{x}}'_u$ is the vector $(\mathbf{x}_1 - \mathbf{x}_0)$ projected to the tangent plane at \mathbf{x}_0 . It is apparent that this vector will be the tangent vector to the parabola at point \mathbf{x}_0 and will contain point \mathbf{b}_1 . Now to uniquely define a conic $(w_0 \mathbf{b}_0, w_1 \mathbf{b}_1, w_2 \mathbf{b}_2)$, whose projection in to affine space is a parabola we can set $\mathbf{b}_0 = \mathbf{x}_0$, and $\mathbf{b}_2 = \mathbf{x}_1$ and also the corresponding weights. We need to find \mathbf{b}_1 and w_1 . However, before we can apply the **RTTP** method developed in Section 4.1, we need to find the *curvature* at \mathbf{x}_0 . Before we proceed further, we define the *normal curvature*.

Definition 1 *Normal curvature κ_n at a point \mathbf{x}_0 on a surface along a tangent direction is defined as the curvature of the curve of intersection of the surface and the plane containing the point \mathbf{x}_0 on the surface, the surface normal, \mathbf{n} , at that point and the tangent direction vector (which is also tangent to the curve of intersection at \mathbf{x}_0):*

$$\kappa_n = \frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2} \quad (12)$$

where

$$E = E(u, v) = \mathbf{x}_u \mathbf{x}_u \quad F = F(u, v) = \mathbf{x}_u \mathbf{x}_v \quad G = G(u, v) = \mathbf{x}_v \mathbf{x}_v \quad (13)$$

and

$$L = L(u, v) = \mathbf{n} \mathbf{x}_{uu} \quad M = M(u, v) = \mathbf{n} \mathbf{x}_{uv} \quad N = N(u, v) = \mathbf{n} \mathbf{x}_{vv} \quad (14)$$

The numerator of Equation 12 is called the *2nd fundamental form* and the denominator is called the *1st fundamental form* in classical differential geometry. $du : dv$ are the direction numbers of the line in the tangent plane parallel to vector $\mathbf{x}_u du + \mathbf{x}_v dv$.

We want to compute the curvature along $\vec{\mathbf{x}}'_u$. We can always write $\vec{\mathbf{x}}'_u$ as a combination of the two vectors $\vec{\mathbf{x}}_u$ and $\vec{\mathbf{x}}_v$ since they are coplanar as they all lie in the tangent plane. Therefore, we can solve for du and dv from:

$$\vec{\mathbf{x}}'_u = du \cdot \vec{\mathbf{x}}_u + dv \cdot \vec{\mathbf{x}}_v \quad (15)$$

From the given surface we can compute L, M, N, E, F and G at \mathbf{x}_0 . From Equation 12 we can therefore compute the normal curvature at \mathbf{x}_0 along $\vec{\mathbf{x}}'_u$. We now have all the entities required to compute the rational conic (parabola in the affine space) i.e. \mathbf{b}_1 and w_1 , as described in Section 6.2.

6.3 Twist Estimation

The *twist* of a tensor product surface $\mathbf{X}(u, v)$ is the mixed partial $\mathbf{X}_{uv}(u, v)$. We need twists to compute the inner Bézier points at the four corners (refer Figure 3). The twist vector of a $m \times n$ rational Bézier patch, given by:

$$\mathbf{X}(u, v) = \frac{\sum_{i=0}^m \sum_{j=0}^n w_{i,j} \mathbf{b}_{i,j} B_i^m(u) B_j^n(v)}{\sum_{i=0}^m \sum_{j=0}^n w_{i,j} B_i^m(u) B_j^n(v)} \quad (16)$$

where $B_i^m(u)$ are Bernstein polynomials and $[\mathbf{b}_{i,j}, w_{i,j}]$ are the control points. The twist is given by:

$$\mathbf{X}_{uv}(u, v) = \frac{\partial^2}{\partial u \partial v} \mathbf{X}(u, v) \quad (17)$$

If we express Equation 16 as following:

$$\mathbf{X}(u, v) = \frac{\mathbf{B}}{\mathbf{W}} \quad (18)$$

then differentiating twice by applying quotient rule

$$\frac{\partial^2}{\partial u \partial v} \mathbf{X}(u, v) = \frac{\mathbf{W}(\mathbf{B}_{uv} \mathbf{W} + \mathbf{B}_u \mathbf{W}_v - \mathbf{B}_v \mathbf{W}_u - \mathbf{B} \mathbf{W}_{uv}) - 2 \mathbf{W}_v (\mathbf{B}_u \mathbf{W} - \mathbf{B} \mathbf{W}_u)}{\mathbf{W} * \mathbf{W} * \mathbf{W}} \quad (19)$$

where $\mathbf{B}_{uv} = \frac{\partial^2}{\partial u \partial v} \mathbf{B}$ etc.

We are interested in finding the twist vector at the corners of the patch, i.e. at $(u = 0, v = 0)$, $(u = 1, v = 0)$, $(u = 0, v = 1)$ and $(u = 1, v = 1)$. We

give the formula for the first case i.e. ($u = 0, v = 0$).

$$\begin{aligned}
\mathbf{x}_{uv}(0, 0) &= \frac{m \times n}{w_{00}^2} [(w_{00}(w_{11}\mathbf{b}_{11} - w_{10}\mathbf{b}_{10} - w_{01}\mathbf{b}_{01} + w_{00}\mathbf{b}_{00}) \\
&\quad + (w_{10}\mathbf{b}_{10} - w_{00}\mathbf{b}_{00})(w_{01} - w_{00}) - \\
&\quad (w_{01}\mathbf{b}_{01} - w_{00}\mathbf{b}_{00})(w_{10} - w_{00}) \\
&\quad - w_{00}\mathbf{b}_{00}(w_{11} - w_{10} - w_{01} + w_{00})) \\
&\quad - 2(w_{01} - w_{00})((w_{10}\mathbf{b}_{10} - w_{00}\mathbf{b}_{00}) - \mathbf{b}_{00}(w_{10} - w_{00}))]
\end{aligned} \tag{20}$$

The other three cases can be similarly computed.³

Equation 20 enables us to determine the twist vector at four corners of the given NURB surface. We need to estimate the twist vector for the new surface i.e. the approximating surface. We cannot rely on the twist vector of the given surface due to its dependence on the parametrization, a characteristic that was the source of the problem we are solving. Several methods have been presented to estimate the twist, for example, [1] and [2]. Farin and Hagen [5] also give a method for local twist estimation. Although, a good method, it can produce *zero* twists under some conditions which may be undesirable. We present a new method for estimating the twist.

We compute the normal curvature along an arbitrary curve (not the boundary curve), say $du = 1, dv = 1$, on the given surface at each of its corners. We can also write the mixed partial as a function of three linearly independent factors i.e. the normal and the partial derivatives.

$$\mathbf{x}_{uv} = \alpha \mathbf{n} + \beta \mathbf{x}_u + \gamma \mathbf{x}_v \tag{21}$$

Therefore, for the given surface, we can compute α, β and γ .

Let $\hat{\cdot}$ represent the values for the new surface. Now, for the new surface we know the tangents, $\hat{\mathbf{x}}_u$ and $\hat{\mathbf{x}}_v$ (or first partials) along the boundary curves (from boundary end conditions). We solve the four linear systems for the boundary curves and therefore compute the second derivatives along the *new* boundary curves, except the mixed partial. To use the normal curvature equation (Equation 12) for the new surface we need to compute values for $\hat{d}u$ and $\hat{d}v$. These can be expressed in terms of du and dv as

$$\hat{d}u = \lambda du \tag{22}$$

³By using the same formula and changing the indices.

and

$$\hat{d}v = \mu \cdot dv \quad (23)$$

where

$$\lambda = \frac{\|\hat{\mathbf{x}}_u\|}{\|\mathbf{x}_u\|} \quad (24)$$

and

$$\mu = \frac{\|\hat{\mathbf{x}}_v\|}{\|\mathbf{x}_v\|} \quad (25)$$

The above enables computation of \hat{L} , \hat{N} , \hat{E} , \hat{F} and \hat{G} . In order to improve the quality of approximation, we want to maintain the normal curvature along the arbitrary curve in the new surface. Plugging the values in Equation 12 we get \hat{M} which by definition is:⁴

$$\hat{M} = \mathbf{n} \hat{x}_{uv} \quad (26)$$

We can also write $\hat{\mathbf{x}}_{uv}$ as a combination

$$\hat{\mathbf{x}}_{uv} = \hat{\alpha} \hat{\mathbf{n}} + \beta \hat{\mathbf{x}}_u + \gamma \hat{\mathbf{x}}_v \quad (27)$$

Dot product of Equation 27 with \mathbf{n} gives

$$\mathbf{n} \hat{\mathbf{x}}_{uv} = \hat{\alpha} \quad (28)$$

Equating Equations 26 and 28 we compute $\hat{\alpha}$. It is apparent that the effect of β and γ on the normal curvature is absorbed in the mixed partial. We set $\hat{\beta} = \beta$ and $\hat{\gamma} = \gamma$ to give an estimate of the twist. The scale of β and γ will not affect the outcome of the normal curvature of an arbitrary curve on the surface. However, to scale properly we set:

$$\hat{\beta} = \frac{\beta \hat{\alpha}}{\lambda \alpha} \quad (29)$$

and

$$\hat{\gamma} = \frac{\gamma \hat{\alpha}}{\mu \alpha} \quad (30)$$

⁴It is evident that $\mathbf{n} = \hat{\mathbf{n}}$, ignoring the scale of the normal.

We use Equation 27 to compute the mixed partial, $\hat{\mathbf{x}}_{uv}$ of the new surface.

We still need to find $[\hat{\mathbf{b}}_{11}, \hat{w}_{11}]$ etc. for the new surface. We first estimate the weight, \hat{w}_{11} . All weights except \hat{w}_{11} are known. We estimate \hat{w}_{11} by the following averaging process:

$$\hat{w}_{11a} = \hat{w}_{10} + \frac{2}{3}(\hat{w}_{01} - \hat{w}_{00}) + \frac{1}{3}(\hat{w}_{31} - \hat{w}_{30}) \quad (31)$$

$$\hat{w}_{11b} = \hat{w}_{01} + \frac{2}{3}(\hat{w}_{10} - \hat{w}_{00}) + \frac{1}{3}(\hat{w}_{13} - \hat{w}_{03}) \quad (32)$$

and

$$\hat{w}_{11} = \frac{\hat{w}_{11a} + \hat{w}_{11b}}{2} \quad (33)$$

Once we have \hat{w}_{11} we can solve for $\hat{\mathbf{b}}_{11}$ using Equation 20. The process is repeated for the other three corners.

7 Conclusion

We prefer to use the solution to the rational parabola problem in Section 4.1 for computing end condition points at the four corners of the surface. This condition is less restrictive than the rational Bessel end condition as it uses only two points to compute the parabola instead of three. We also prefer the solution described in Section 6.2 for the points along the boundary curves. As mentioned above, twists are also computed. This completes all the information required for solving the tri-diagonal system of equations. The methods mentioned above were applied to approximate a surface (Figure 5). The surface at the bottom is an approximation of the surface above in the figure. The algorithms mentioned in this paper were applied to compute end conditions. The interpolation points on the boundary curves and the interior were picked according to [9].

References

- [1] R.E. Barnhill, J. Brown, and I. Kluczewicz. A new twist in CAGD. *Computer Graphics and Image Processing*, 8:78–91, 1978.

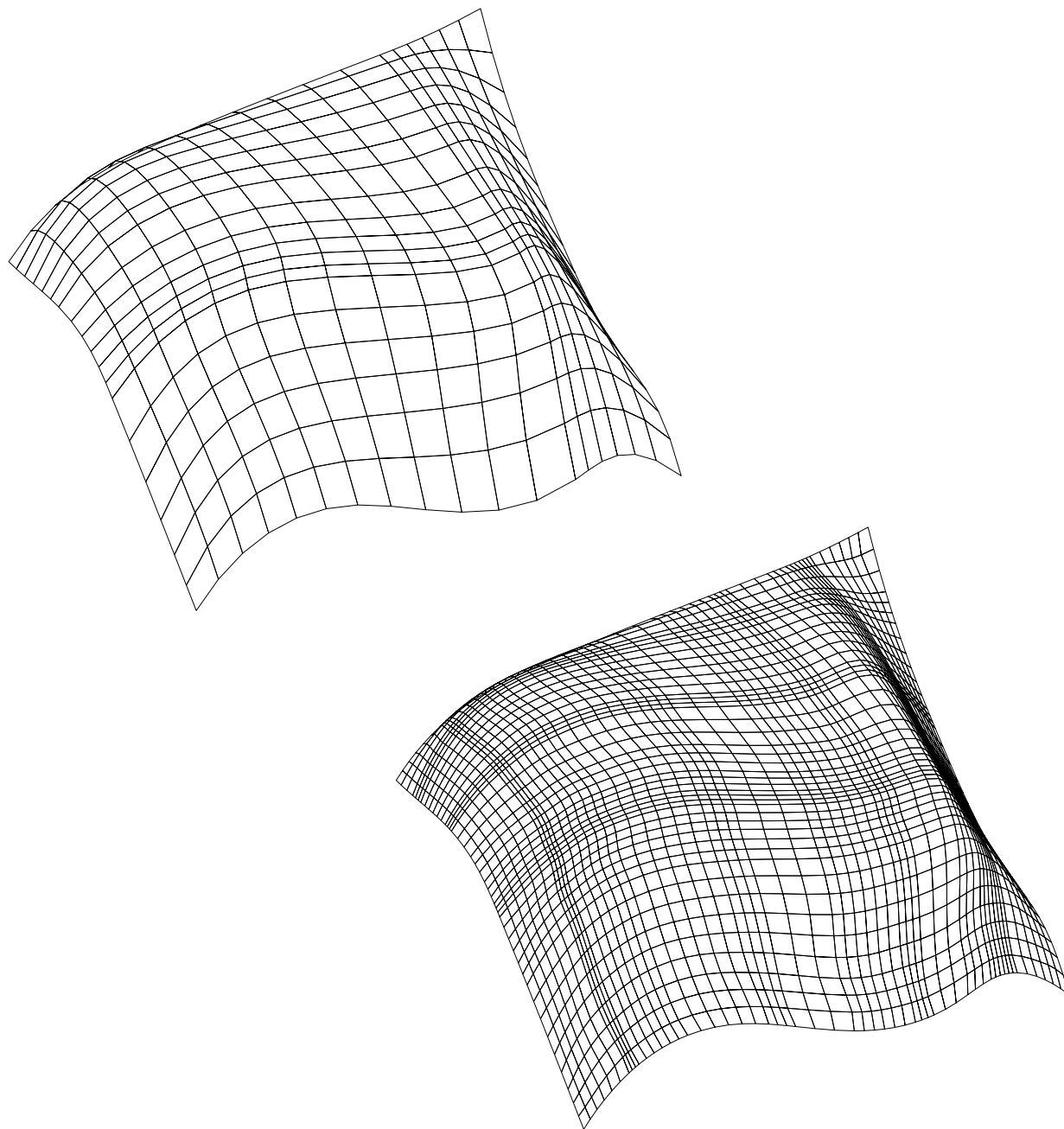


Figure 5: Surface approximation with rational end conditions.

- [2] R.E. Barnhill, G. Farin, L. Fayard, and H. Hagen. Twists, curvatures and surface interrogation. *Computer Aided Design*, 20:341–346, 1988.
- [3] C. de Boor. *A Practical Guide to Splines*. Springer, 1978.
- [4] G. Farin. *Curves and Surfaces for CAGD*. Academic Press, 1993.
- [5] G. Farin and H. Hagen. A local twist estimator. In H. Hagen, editor, *Topics in Surface Modeling*, pages 79–84. SIAM, Philadelphia, 1992.
- [6] Eugene T.Y. Lee. The rational Bézier representation for conics. In Gerald E. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 3–19. SIAM, Philadelphia, 1987.
- [7] G. Nielson. A transfinite, visually continuous, triangular interpolant. In G. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 235–246. SIAM, 1987.
- [8] B. Piper. Visually smooth interpolation with triangular Bézier patches. In G. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 221–233. SIAM, 1987.
- [9] Anshuman Razdan. *Healing NURB Surfaces*. PhD thesis, Arizona State University, 1995.
- [10] T. Yu and B.K. Soni. Application of NURBS in numerical grid generation. *Computer Aided Design*, 27:147–157, 1995.