Computer Aided Geometric Design 7 (1990) 499–516
North-Holland

Local cubic and bicubic C^1 surface interpolation with linearly varying boundary normal *

Jörg PETERS

Center of Mathematical Sciences and Computer Sciences Department, University of Wisconsin–Madison, Madison, WI 53706, USA

Received June 1988
Revised March 1989

Abstract. An algorithm for the interpolation of a 'mesh of points' in 3-space by a C^1 surface is developed. At each point, the surface normal can be specified. The surface is constructed locally and consists of cubic and bicubic patches to match the underlying mesh facets. The surface construction is special in that it generates a piecewise parametric surface such that the normal along patch boundaries varies linearly.

Keywords. Bernstein–Bézier form, C^1 surface, interpolation, cusping, convexity.

1. Introduction

Fitting a smooth parametric piecewise polynomial surface to given data points is a well studied problem. Because of its nonlinear nature and the advantages of a local construction, different approaches have been centered around selecting geometrically meaningful variables that can be fixed a priori (input or derived from data) so as to arrive at a sufficient and consistent set of linear constraints on the remaining variables. 'Blending' approaches, for example, prescribe a mesh of boundary curves and their transversal derivatives, while 'splitting' approaches expect at least the tangent vectors at the data points and sometimes the complete boundary to be given. The algorithm presented here differs in that it constrains the normal direction along the patch boundaries rather than the boundaries themselves. As such it is most akin to a (nonlinear) method by Sabin [Sabin '68]. Our choice of *boundary normals as weighted linear blends* of the normals at the data points leads to a *linear local* solution with patches of low degree. This basic idea is independent of the number of sides of the patches and thus naturally combines triangular and rectangular patches. While the simple character of the surface normal is, on one hand, desirable to control shape, it imposes, on the other hand, restrictions on the nature and distribution of the data. Employing only one patch per facet, the method additionally encounters a version of the 'compatibility problem'. The corresponding restrictions on the data are less severe than those for blending methods, but still amount to one additional constraint for each point that has an even number of neighbors. Since the reference [Peters '89] shows how the compatibility problem is resolved by splitting facets, this paper limits itself to the plain approach with *one patch per facet*.

* This research was supported by NSF DMS-8701275.

We use the Bernstein–Bézier form (*BB form*) to represent the polynomial patches, because this form gives easy access to value and derivative information along the boundaries of a patch, as well as geometric meaning to its coefficients (cf. [Farin '86], [de Boor '87]). To construct the surface, we determine the coefficients. The coefficients are 3-vectors, i.e. the polynomial pieces map from the unit square or triangle to 3-space. In our derivation and analysis, it will suffice to look at derivatives evaluated at a boundary. Since our main algebraic work consists of multiplying these univariate polynomials, it is advantageous to work with the modified BB-form

$$p : t \to \sum_{j=0}^{d} t^j (1-t)^{d-j} b^j \tag{$*$}$$

and to write

$$p \sim (b^0, \ldots, b^d)$$

to indicate that p is given by $(*)$. Thus $p \sim (a^0, \ldots, \binom{d}{j} a^j, \ldots, a^d)$ in terms of its control points a^j. For example, raising the degree of the quadratic polynomial $(a^0, 2a^1, a^2)$ is expressed as

$$(1, 1)(a^0, 2a^1, a^2) = (a^0, 2a^1 + a^0, 2a^1 + a^2, a^2).$$

If nothing else, this avoids writing fractions. Abbreviating the scalar product of a and b as ab keeps the notation simple, e.g.

$$(a^0, a^1)(n^0, n^1) = (a^0 n^0, a^0 n^1 + a^1 n^0, a^1 n^1).$$

We follow the rule that superscripts distinguish vectors along a boundary, while subscripts count the neighbors viewed from a given point. For example, the linear normal along the boundary from P to its ith neighbor, P_i, is (N^0, N_i^1). Super- and subscripts are dropped, whenever this is unambiguous. The counting is cyclic.

We can now be more precise. The data for the algorithm consist of a 'mesh of points' and their normals. That is, each data point P knows its neighbors P_1 through P_k, but no connecting curve mesh is prescribed. The algorithm first constructs a *cubic* curve between any two neighboring points in accordance with their normals and then fills in the interior of the patches. Most of the effort, however, is spent in computing the tangent vectors at the data points, and it is here where the constraints on the data come into play. Our basic assumption is that the data allow for a linearly varying normal along the boundaries. This is explained in more detail in Section 2. A second assumption is that the mesh facets are either three or four sided, so that it can be covered by a cubic or bicubic patch. Finally, we make an assumption on points with an even number of neighbors (which we call *even-points* for short). This insures compatibility and, in the case of Assumption 1.3(a), guarantees that the surface is cusp-free at the data points. For Assumption 1.3(a), we need the following definition, where the k_i are scalar (curvature) values and the ξ_i are mutually perpendicular vectors (principal directions). We say that *the data at P are consistent with the second fundamental form $Q := k_1 \xi_1 \xi_1' + k_2 \xi_2 \xi_2'$* if

$$N_i \xi_1 = k_1, \qquad N_i \xi_2 = k_2, \qquad N_{i+1}(P_{i+1} - P) = N_i(P_i - P) \quad \text{for } i \in \{1, \ldots, k\}.$$

Assumption 1.3(a) is a strong condition on the data and may only hold after splitting facets or additional sampling. We list the three assumptions right here for easy reference.

Assumption 1.1 [Linear boundary normal]. *Let P^i and P^j be neighbor data points with normals N^i and N^j. Then*

$$\frac{N^i(P^i - P^j)}{N^j(P^j - P^i)} \geqslant 0.$$

Assumption 1.2 [Mesh structure]. (a) *All facets of the surface have either three or four sides.*

(b) *The data are well-distributed, i.e. at each mesh point the projections into the tangent plane of any two edges belonging to the same facet span an angle of less than* π.

Assumption 1.3 [Compatibility]. *Either*

(a) *the data at each even-point are consistent with some second fundamental form; or*

(b) (1) *if* $D_i := \det(N, N_i, N_{i+1}) \neq 0$ *for* $i \in \{1, \ldots, k\}$ *and P is an even-point, then* $\det(N, N_i, N_{i+2}) = 0$ *for* $i \in \{1, \ldots, k\}$; *else*

(2) *if* $D_i = 0$ *for some* $i \in \{1, \ldots, k\}$, *then the data are either 'cylindrical' or 'planar'* (see Theorem 4.5).

Sections and content. Section 2 reviews the conditions for a C^1 match between two piecewise polynomial patches. We specialize the conditions, which are different from [Farin '83], to the BB setup and explain how our approach localizes and linearizes. Section 3 states the algorithm and Section 4 discusses solvability and convexity. Section 5 analyzes the cusping problem and leads to the use of 'curvature weights' for the linear blend of the normals. Section 6 gives some examples.

2. The C^1 conditions

We now review the C^1 conditions and specialize them to the BB setup. Consider two smooth patches, p and q, that meet along a common boundary curve as depicted in Fig. 2.1. Patch p has parameters u and v and its neighbor patch q has parameters u and w. We define

$$p_u := \frac{\partial}{\partial u} p(u, 0), \qquad p_v := \frac{\partial}{\partial v} p(u, 0) \quad \text{and} \quad q_w := \frac{\partial}{\partial w} q(u, 0).$$

Then p and q regularly parametrize a C^1 surface if and only if the surface normal is well-defined for both patches and agrees at every point of the boundary:

$$\frac{p_v \times p_u}{\| p_v \times p_u \|} = \frac{p_u \times q_w}{\| p_u \times q_w \|} \qquad [\text{matching normal}], \tag{1}$$

$$p_v \times p_u \neq 0, \qquad [\text{non-vanishing normal}]. \tag{2}$$

Note that we were careful with the order in the vector products, that $p_u = q_u$ and that p_u, p_v and q_w are univariate vector-valued polynomials. For our purposes, we rephrase the conditions to remove the denominator.

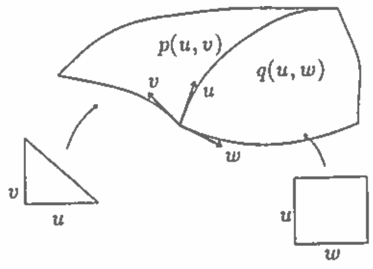

Fig. 2.1. Parametrization of abutting patches.

Lemma 2.1 [C^1 matching conditions]. *A match between two polynomial patches $p(u, v)$ and $q(u,w)$ across a common boundary parametrized by u is C^1 if and only if*

$$(p_v \times p_u) q_w = 0 \qquad \text{[common tangent plane]} \tag{E}$$

and

$$((p_v \times p_u) \times p_u) q_w > 0 \qquad \text{[proper orientation].} \tag{I}$$

Proof. We will use the identities $(a \times b)(c \times d) = \det((a \times b), c, d) = ((a \times b) \times c)d$. First we show that (2) and (1) imply (I) and (E). To see that the equality (E) holds, we need only multiply (1) by $q_w \| p_v \times p_u \|$. The inequality (I) is implied by $p_v \times p_u \neq 0$ together with (1) since

$$0 < (p_v \times p_u)(p_u \times q_w) = ((p_v \times p_u) \times p_u) q_w.$$

To show the converse, we observe that

$$0 < ((p_v \times p_u) \times p_u) q_w = \det((p_v \times p_u), p_u, q_w) = \det(p_v, p_u, (p_u \times q_w)).$$

Thus $p_v \times p_u \neq 0$ and $p_u \times q_w \neq 0$. Furthermore, (E) shows that $q_w \perp (p_v \times p_u)$. Hence $(p_v \times p_u)$ and $(p_u \times q_w)$ are collinear. Finally, since $0 < ((p_v \times p_u) \times p_u) q_w = (p_v \times p_u)(p_u \times q_w)$, we may conclude that the normalized vectors are the same. $\square$

The inequality (I) is a regularity condition. It implies the existence of a unique normal at each point of the patch boundary. The equality (E) is a coplanarity condition. It expresses perpendicularity of the normal of patch p to the tangent plane of q at the common boundary. Together the two constraints imply uniqueness of the normal across the boundary. Note that the order of the terms in (I) is important since the direction of the inequality distinguishes the C^1 match from a cusping match.

2.1. Facet separation

We now consider the C^1 conditions as they apply to cubic and bicubic patches in BB form. For mnemonic efficiency, we will denote the BB coefficients of p_u and p_v by u^j and v^j. We count the neighbors of a data point P clockwise: coefficients of the ith curve emanating from P are subscripted by i, e.g. u_i^0 is the ith tangent vector at P. We define for *triangular* patches

$$(p_u)_i \sim (u_i^0, 2\bar{u}_i, u_i^1), \qquad (p_v)_i \sim (v_i^0, 2\bar{v}_i, v_i^1), \qquad (q_w)_i \sim (w_i^0, 2\bar{w}_i, w_i^1)$$

and for *rectangular* patches

$$(p_u)_i \sim (u_i^0, 2\bar{u}_i, u_i^1), \qquad (p_v)_i \sim (v_i^0, 3v_i^{01}, 3v_i^{10}, v_i^1),$$
$$(q_w)_i \sim (w_i^0, 3w_i^{01}, 3w_i^{10}, w_i^1).$$

This is illustrated in Fig. 2.2.

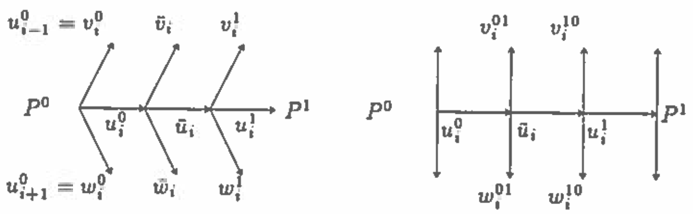

Fig. 2.2. Naming conventions for difference vectors.

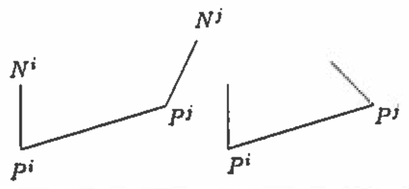

Fig. 2.3. Left data imply an inflection, the right data do not.

We focus on the equality constraints, $(p_v \times p_u)q_w = 0$. These constraints involve interior Bézier coefficients both along the patch boundaries and around the data points thus linking the corresponding system of equations globally. To reduce it to local blocks we *impose additional constraints where the polynomial pieces meet*. In particular, we prescribe the normal direction, n, along each boundary to be the simplest fit to the data, i.e. $n \sim (N^0, N^{-1})$. With this, the equality constraints (E) become

$$n(u)\, p_u = 0, \tag{E$_u$}$$

$$n(u)\, p_v = 0, \tag{E$_v$}$$

$$n(u)\, q_w = 0. \tag{E$_w$}$$

This *linearizes and symmetrically separates the constraints* along the boundaries, while keeping the degree of np_u, np_v and nq_w as low as possible. However, it also imposes restrictions on the nature and distribution of the data: the normal direction can only be linear if $p(u, 0)$ need not have a point of inflection in [0, 1], i.e. if

$$\text{shape}(i,\ j) := \frac{N^i(P^i - P^j)}{N^j(P^j - P^i)} \geqslant 0. \tag{3}$$

If shape$(i,\ j) < 0$, then n must be at least quadratic (see Fig. 2.3). Interpreting $N^i(P^i - P^j) \leqslant 0$ as 'concavity of the data' at P^i and $N^i(P^i - P^j) \geqslant 0$ as 'convexity', shape consistency at both endpoints of a boundary, implies that the space curve $p(u, 0)$ can stay strictly on one side of its osculating plane. Note that, for $\Delta^{ij} := P^i - P^j$, $\nu := \Delta^{ij}(N^i \times N^j)$ and $\Delta^{ij}_* := \Delta^{ij} - \nu(N^i \times N^j)$, $N^i\Delta^{ij} = N^i\Delta^{ij}_*$. That is, Δ^{ij} and the projection of Δ^{ij} onto the plane spanned by N^i and N^j form the same angle with N^i. Hence the planar sketches of Fig. 2.3 are justified.

2.2. *Compatibility complication*

We are almost ready to rewrite the equality constraints (E$_u$), (E$_v$) and (E$_w$) explicitly in terms of the Bézier coefficients. From experience, however, we are led to consider the 'cross derivative compatibility problem' which occurs when smooth patches are pieced together at a vertex. For this, we concentrate on the boundaries connecting the data point P with its neighbors P_i and P_{i-1} as shown in Fig. 2.4. Let u be the parameter of the ith boundary and v

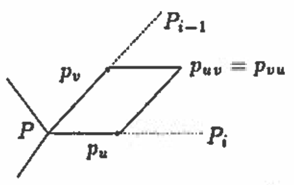

Fig. 2.4. Cross derivative compatibility

of the $i-1$st. Define $n_u := (\partial/\partial u)n$. Then, at P,

$$N_i p_v = (n_u)_i p_v = -Np_{vu} = -Np_{uv} = (n_v)_{i-1} p_u = N_{i-1} p_u \tag{C}$$

Compatibility, (C), imposes an additional constraint on the boundary coefficients of each patch and connects the equations (E_u) at the ith and (E_v) at the $i+1$st boundary, thus upsetting all our separation efforts. But we still have a 'trick' to offer, namely a transformation of (E_u) that takes advantage of the cubic nature of the boundary polynomial. This results in a *system of constraints localized around each data point* as opposed to localized to facets.

2.3. Vertex separation

To be more precise, we list (E_u) in terms of the Bézier coefficients:

$$(N^0, N^1)(u^0, 2u, u^1) = 0. \tag{E_u'}$$

We leave off subscripts for now since the discussion is independent of any ordering around data points. To obtain the constraints, we set the coefficients with respect to the BB basis of the scalar product polynomial to zero:

$$0 = N^0 u^0, \tag{E_u^1}$$

$$0 = 2N^0 \bar{u} + N^1 u^0, \tag{E_u^2}$$

$$0 = 2N^1 \bar{u} + N^0 u^1, \tag{E_u^3}$$

$$0 = N^1 u^1. \tag{E_u^4}$$

Our 'trick' is then contained in the following claim.

Claim 2.2. *The collection of constraints* (E_u^1), (E_u^2), (E_u^3) *and* (E_u^4) *is equivalent to* (E_u^1), (E_u^4) *and*

$$0 = \tfrac{2}{3}(N^0 + 2N^1)(P^1 - P^0) - N^1 u^0, \tag{$E_u^{2\prime}$}$$

$$0 = \tfrac{2}{3}(N^1 + 2N^0)(P^1 - P^0) - N^0 u^1. \tag{$E_u^{3\prime}$}$$

Proof. We use the fact that $\bar{u} = (P^1 - P^0) - u^0 - u^1$, and hence

$$2N^0\bar{u} + N^1 u^0 = 2N^0\big((P^1 - P^0) - u^1\big) + N^1 u^0,$$

$$2N^1\bar{u} + N^0 u^1 = 2N^1\big((P^1 - P^0) - u^0\big) + N^0 u^1.$$

By adding two times (E_u^2) to (E_u^3) and two times (E_u^3) to (E_u^2) we obtain

$$0 = (4N^0 + 2N^1)(P^1 - P^0) - 3N^0 u^1,$$

$$0 = (4N^1 + 2N^0)(P^1 - P^0) - 3N^1 u^0.$$

We divide by 3 (and exchange the second and the third equation) to get the result. □

Thus we have a separate set of conditions for the tangent vectors at each data point.

2.4. Collecting constraints

We proceed by collecting all the constraints at the data point P with normal N and neighbors P_i, $i \in \{1, \ldots, k\}$. We parametrize the boundary curves starting at P. For simplicity, we leave out the subscript i on the v_i^j terms, i.e. $v^j := v_i^j$. The derivative in the v direction evaluated at $v = 0$, i.e. along the boundary, is a univariate cubic. Hence, for *cubic* patches, we have

$$Nv^0 = 0, \qquad (\mathrm{E}_v^1)$$

$$2N\bar{v} + N_i v^0 = 0, \qquad (\mathrm{E}_{v3}^2)$$

$$2N_i\bar{v} + Nv^1 = 0, \qquad (\mathrm{E}_{v3}^4)$$

$$N_i v^1 = 0 \qquad (\mathrm{E}_v^5)$$

and for *bicubic* patches

$$Nv^0 = 0, \qquad (\mathrm{E}_v^1)$$

$$3Nv^{01} + N_i v^0 = 0, \qquad (\mathrm{E}_{v4}^2)$$

$$3Nv^{10} + 3N_i v^{01} = 0, \qquad (\mathrm{E}_{v4}^3)$$

$$3N_i v^{10} + Nv^1 = 0, \qquad (\mathrm{E}_{v4}^4)$$

$$N_i v^1 = 0. \qquad (\mathrm{E}_v^5)$$

The equations for (E_w) are obtained by replacing v by w in the above, and the equations for (E_u) have already been listed and replaced by (E_u'). Thus we only need to state the compatibility conditions:

$$N_i v_i^0 = N_{i-1} u_i^0. \qquad (\mathrm{C})$$

2.5. Tangent constraints

A first glance at the constraints seems to indicate that the tangent vectors are overdetermined. However, since $v_i^0 = u_{i-1}^0$ and $w_i^0 = u_{i+1}^0$, enforcing (E_u^1) for all boundaries implies that (E_u^4), (E_v^1), (E_v^5), (E_w^1) and (E_w^5) hold. In the end, the constraints for the ith tangent vector at the data point P are merely:

$$Nu_i^0 = 0 \qquad [\text{vertex coplanarity}], \qquad (\mathrm{E}_u^1)$$

$$N_i u_i^0 = \tfrac{2}{3}(P^i - P)(N + 2N_i) \qquad [\text{boundary coplanarity}], \qquad (\mathrm{E}_u^{2\prime})$$

$$N_{i+1} u_i^0 - N_i u_{i+1}^0 = 0 \qquad [\text{compatibility}]. \qquad (\mathrm{C})$$

We call this set of constraints the *tangent constraints*. The tangent constraints form a linear system that is almost block diagonal and almost circulant. Section 4 has more on this.

2.6. Interior constraints

In many other surface interpolation schemes, the conditions on the inner Bézier coefficients closest to the data points, sometimes called 'twist coefficients', are computationally the most expensive (cf. e.g. [Sarraga '86]). Here, however, this work is comparatively small, the major part being the enforcement of the tangent constraints. We observe that, by symmetry, (E_v) at one endpoint of a boundary is equivalent to (E_w) at the other. Thus we need only enforce (E_{v3}^2) and (E_{v3}^4) for cubic patches, and (E_{v4}^2), (E_{v4}^3), and (E_{v4}^4) for bicubic patches. Again the center coefficient of the cubic seems overdetermined since there are two constraints imposed on it from each of the three boundaries. The compatibility conditions turn out to help.

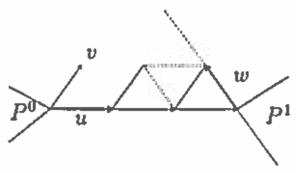

Fig. 2.5. Notation for Claim 2.3.

Claim 2.3. *If the tangent constraints are enforced, then* (E_{vi}^2) *at all data points implies* (E_{vi}^4), *where* $i \in \{3, 4\}$.

Proof. Consider the setup of Fig. 2.5. Since np_u is a cubic scalar-valued polynomial in u and (E_v^1), (E_v^5) and (E_{v3}^2), resp. (E_{v4}^2), correspond to $np_v(0) = np_v(1) = 0$ and $(np_v)_u(0) = 0$, the constraint (E_{vi}^4) is equivalent to $(np_v)_u(1) = 0$. The tangent constraints further imply $(np_u)_u(1) = 0$, and hence $(np_v)_u(1) = -(np_w)_{-u}(1)$, where $w := v - u$. Finally, by compatibility, $(np_w)_u(1) = (np_u)_w(1) = 0$, i.e. (E_{vi}^4) follows from (E_{vi}^2) for the boundary parametrized by w and emanating from P^1. □

All in all, we derived the *interior constraints*:

$$2N^0\bar{v} + N^1v^0 = 0 \qquad [\text{cubic}], \qquad (\mathrm{E}_{v3}^2)$$

$$3N^0v^{01} + N^1v^0 = 0 \qquad [\text{bicubic}], \qquad (\mathrm{E}_{v4}^2)$$

$$3N^0v^{10} + 3N^1v^{01} = 0 \qquad [\text{bicubic}]. \qquad (\mathrm{E}_{v4}^3)$$

The interior constraints and the tangent constraints together imply the C^1 equality constraints (E).

3. The algorithm

The algorithm proceeds in two stages. The first stage determines the tangent vectors from the tangent constraints. The second stage computes the interior coefficients from the interior constraints. Solvability of the equations is discussed in Section 4 and shown to hold under the assumptions listed in Section 1. In Section 5, we point out a slight but important improvement of the basic algorithm presented below: by scaling the normals at the endpoints appropriately, the surface construction is more resistant to cusping. For simplicity, we state the algorithm without 'curvature weights'—the scaling is easily accounted for by replacing N_i by $\omega_i N_i$ in the algorithm below.

Algorithm LINNOR

Step 1 [Tangent constraints]. Assemble the following system of equations $\mathrm{N}U = R$ at each data point P with normal N and neighbors P_1 through P_k:

$$\begin{pmatrix} N & 0 & 0 & \dots & 0 \\ N_1 & 0 & 0 & \dots & 0 \\ N_2 & -N_1 & 0 & \dots & 0 \\ 0 & N & 0 & \dots & 0 \\ 0 & N_2 & 0 & \dots & 0 \\ 0 & N_3 & -N_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & -N_{k-1} \\ 0 & 0 & 0 & \dots & N \\ 0 & 0 & 0 & \dots & N_k \\ -N_k & 0 & 0 & \dots & N_1 \end{pmatrix} \begin{pmatrix} u_1^0 \\ \vdots \\ u_k^0 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{2}{3}(P_1 - P)(N + 2N_1) \\ 0 \\ 0 \\ \frac{2}{3}(P_2 - P)(N + 2N_2) \\ 0 \\ \vdots \\ 0 \\ 0 \\ \frac{2}{3}(P_k - P)(N + 2N_k) \\ 0 \end{pmatrix}. \qquad (4)$$

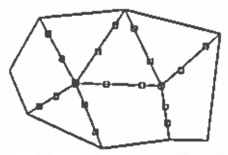
Fig. 3.1. Coefficients determined after Step 1.

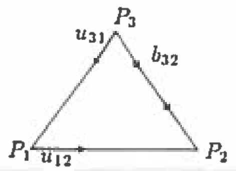

Fig. 3.2. Naming conventions for Step 2.

Solve

$$\min_{\mathbf{N}U=R} \| U - U^* \|^2 \tag{5}$$

where $(U^*)_i := u_i^* := \frac{1}{3}((P_i - P) - \nu N)$ and $\nu := (P_i - P)N$.

Remarks. (a) The diagonal of $\mathbf{N}$ is lined with $k-1$ blocks of 3×3 matrices of the form $(N, N_i, N_{i+1})^{\mathrm{T}}$. The kth block is 'wrapped around'.

(b) The analysis of Section 4 will show that $\mathbf{N}$ is of full rank exactly if the diagonal blocks are all invertible and k is odd. Otherwise the data have to meet additional constraints and (4) is underdetermined. We can use the additional degrees of freedom by solving (5), i.e. minimizing the distance of the tangents to a set of 'desirable' vectors. In particular, we choose u_i^* to be a certain fraction of the projection of $P_i - P$ into the tangent plane at P, e.g. $u_i^* := \frac{1}{3}((P_i - P) - \nu N)$. Another use of (5) is to fix the undetermined boundary coefficients of open surfaces as in the 'saddle' of Fig. 6.2.

(c) Theorem 5.1 gives the weighted tangent constraints explicitly.

At the end of Step 1, we have the situation illustrated by Fig. 3.1.

Step 2 [Inner constraints]. Denote the tangent vector at P_i on the boundary to P_j by u_{ij} and the Bézier coefficient closest to P_i on the boundary to P_j by b_{ij} as shown in Fig. 3.2. The interior ('twist') coefficients are denoted t_i and $T := (t_i)_{i=1,\dots,k}$.

(a) **Triangular** patches: Solve the following 3×3 system for the center coefficient t:

$$\begin{pmatrix} N_1 \\ N_2 \\ N_3 \end{pmatrix} t = \begin{pmatrix} N_1 P_1 - \frac{1}{2} N_2 u_{13} \\ N_2 P_2 - \frac{1}{2} N_3 u_{21} \\ N_3 P_3 - \frac{1}{2} N_1 u_{32} \end{pmatrix}. \tag{6}$$

(b) **Rectangular** patches: Solve

$$\min_{\mathbf{M}T=r} \| T - T^* \|^2, \tag{7}$$

where the components t_i^* of T^* are derived as the linear blend $t_i^* := P + u_{ii-1}^0 + u_{0ii+1}$,

$$\mathbf{M} := \begin{pmatrix} N_1 & 0 & 0 & 0 \\ N_2 & N_1 & 0 & 0 \\ 0 & N_2 & 0 & 0 \\ 0 & N_3 & N_2 & 0 \\ 0 & 0 & N_3 & 0 \\ 0 & 0 & N_4 & N_3 \\ 0 & 0 & 0 & N_4 \\ N_4 & 0 & 0 & N_1 \end{pmatrix} \quad \text{and} \quad r := \begin{pmatrix} N_1 P_1 - \frac{1}{3} N_2 u_{14} \\ N_2 b_{12} + N_1 b_{21} \\ N_2 P_2 - \frac{1}{3} N_3 u_{21} \\ N_3 b_{23} + N_2 b_{32} \\ N_3 P_3 - \frac{1}{3} N_4 u_{32} \\ N_4 b_{34} + N_3 b_{43} \\ N_4 P_4 - \frac{1}{3} N_1 u_{43} \\ N_1 b_{41} + N_4 b_{14} \end{pmatrix}.$$

Remark. Bicubic patches have 4 center coefficients to enforce 8 conditions. We can alternatively break the resulting 8×12 system into 4 pieces of 3 equations by choosing constants c_i:

$$\begin{pmatrix} N_i \\ N_i + 1 \\ N_{i-1} \end{pmatrix} t_i = \begin{pmatrix} N_i P_i - \frac{1}{3} N_{i+1} u_{ii-1} \\ N_{i+1} b_{ii+1} + c_i \\ N_{i-1} b_{ii-1} + c_{i-1} \end{pmatrix} \quad \text{for } i \in \{1, 2, 3, 4\}. \tag{7'}$$

4. Solvability analysis of the algorithm

In this 'technical' section, we verify that the algorithm will succeed under the assumptions of Section 1. We first look at the system (4), i.e. $\mathbf{N}U = R$, and then at (6) and (7). Our main effort will go into the analysis of (4).

4.1. Solvability of the tangent constraints

To simplify the analysis, we transform the coordinate system rigidly, preserving orientation and angle, so that P is mapped into the origin, and N into (0, 0, 1). This allows us to restrict attention to the tangent plane. In particular, we can drop the constraints $Nu_i^0 = 0$ and assume that the third component of the tangent vectors is zero. Since the first two components determine u_i^0 uniquely, we identify the u_i^0 with the 2-vectors that solve (4′) below. Let n_i denote the tangential components of N_i. Then the following $2k \times 2k$ system is equivalent to (4):

$$\begin{pmatrix} n_1 & 0 & 0 & \dots & 0 \\ n_2 & -n_1 & 0 & \dots & 0 \\ 0 & n_2 & 0 & \dots & 0 \\ 0 & n_3 & -n_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & -n_{k-1} \\ 0 & 0 & 0 & \dots & n_k \\ -n_k & 0 & 0 & \dots & n_1 \end{pmatrix} \begin{pmatrix} u_1^0 \\ \vdots \\ u_k^0 \end{pmatrix} = \begin{pmatrix} r_1 \\ 0 \\ r_2 \\ 0 \\ \vdots \\ 0 \\ r_k \\ 0 \end{pmatrix}, \tag{4'}$$

where

$$r_i := \tfrac{2}{3}(P_i - P)(N + 2N^i).$$

The further analysis depends on the invertibility of the blocks on the diagonal. We say that a triple of normals, $\{N, N_i, N_{i+1}\}$, is *face-sharing* if the corresponding data points lie in the same facet and observe that such a triple is linearly independent if and only if $D_i := \det(n_i n_{i+1}) \neq 0$.

4.1.a. Linearly independent normals

We first test $\mathbf{N}$ for invertibility.

Theorem 4.1 [No problem at odd-points]. *Let k be the number of neighbors of P. If $\det(N, N_i, N_{i+1}) \neq 0$ for $i = 1, \dots, k$, then (4) is of full rank if and only if k is odd.*

Proof. The independence assumption implies that we can solve blockwise, i.e. there exist unique scalars

$$\gamma_i := -\frac{D_{i-1}}{D_i} \neq 0 \quad \text{and} \quad \beta_i := \frac{B_i}{D_i} := \frac{\det(n_{i-1} n_{i+1})}{D_i}$$

such that

$$n_{i-1} = \beta_i n_i + \gamma_i n_{i+1} \quad \text{for } i \in \{1, \ldots, k\}. \tag{8}$$

Using (8), we can apply block elimination, with multipliers β_i and γ_i, $k-1$ times to (4′). Thus, in the jth step, the last entry in the $j+1$st column becomes $-\gamma_1 \cdots \gamma_j n_j$, and hence, using the identity $\gamma_1 \cdots \gamma_i = (-)^i D_k / D_i$, the final entry in the bottom row of the $2k$th column is

$$m := -\gamma_1 \cdots \gamma_{k-1} n_{k-1} + n_1 = \frac{(-)^{k-1}}{\gamma_k} n_{k-1} + n_1.$$

The elimination leaves (4′) unchanged except for the lower right corner and the accumulated contributions to the right-hand side which we denote by s:

$$\begin{pmatrix} n_1 & 0 & 0 & \ldots & 0 \\ n_2 & -n_1 & 0 & \ldots & 0 \\ 0 & n_2 & 0 & \ldots & 0 \\ 0 & n_3 & -n_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & -n_{k-1} \\ 0 & 0 & 0 & \ldots & n_k \\ 0 & 0 & 0 & \ldots & m \end{pmatrix} \begin{pmatrix} u_1^0 \\ \vdots \\ u_k^0 \end{pmatrix} = \begin{pmatrix} r_1 \\ 0 \\ r_2 \\ 0 \\ \vdots \\ 0 \\ r_k \\ s \end{pmatrix}. \tag{4''}$$

Since the transformation from (4) to (4″) is rank preserving, we need only test the bottom 2×2 subsystem of (4″) for invertibility:

$$M_2 u_k^0 := \begin{pmatrix} n_k \\ \dfrac{(-)^{k-1}}{\gamma_k} n_{k-1} + n_1 \end{pmatrix} u_k^0 = \begin{pmatrix} r_k \\ s \end{pmatrix}. \tag{9}$$

M_2 is singular if and only if its rows are dependent, i.e. $(-)^{k-1} n_{k-1}/\gamma_k + n_1 = \nu n_k$ for some constant ν or, equivalently,

$$n_{k-1} = \nu(-)^{k-1} \gamma_k n_k + (-)^k \gamma_k n_1. \tag{10}$$

We compare this to the recurrence (8) with $i = k$. Since ν is arbitrary we see that (4″) and thus (4) is of full rank if and only if

$$(-)^k \gamma_k \neq \gamma_k, \tag{11}$$

that is, if k is odd. □

If P has an *even* number of neighbors, we have to focus on the right-hand side terms r_k and s of (4″). We define

$$\kappa_i := \frac{B_i}{D_{i-1} D_i} := \frac{\det(N, N_{i-1}, N_{i+1})}{\det(N, N_{i-1}, N_i) \det(N, N_i, N_{i+i})}$$

since this term will frequently appear in the subsequent analysis and is related to the curvature of the surface.

Theorem 4.2 [Compatibility at even-points]. *Let k be the number of neighbors of P. If* $\det(N, N_i, N_{i+1}) \neq 0$ *for* $i = 1, \ldots, k$ *and k is even, then* (4) *has* (*a* 1-*dimensional set of*) *solutions if and only if*

$$S^k := -\sum_{j=1}^{k} (-)^i \kappa_i r_i = 0. \tag{12}$$

Proof. Recall the elimination process of Theorem 4.1. At each step the bottom slot of the right-hand side accumulates an additional term of the form $\beta_j\gamma_1 \cdots \gamma_{j-1}r_j$, hence

$$s = \beta_1 r_1 + \beta_2\gamma_1 r_2 + \cdots + \beta_{k-1}\gamma_1 \cdots \gamma_{k-2}r_{k-1},$$

or, since $\gamma_1 \cdots \gamma_i = (-)^i D_k/D_i$,

$$s = -D_k \sum_{i=1}^{k-1} (-)^i \kappa_i r_i.$$

Theorem 4.1 implies that the next elimination step creates a zero bottom row in the constraint matrix. Hence the right-hand side must be zero, too, if (4) is to have a solution. Since k is even, (10) becomes $n_{k-1} = -\nu\gamma_k n_k + \gamma_k n_1$. Comparing this with the recurrence relation (8), we find that ν, and thus the multiplier of the last elimination step, is $-\beta_k/\gamma_k$. Hence, the final bottom right-hand side entry is

$$s + \frac{\beta_k}{\gamma_k} r_k = -D_k \sum_{j=1}^{k-1} (-)^j \kappa_j r_j - \frac{D_k B_k}{D_{k-1} D_k} r_k = D_k S_k.$$

Since $D_k \neq 0$, (12) follows. □

Corollary 4.3. *If* $\kappa_i = 0$ *for all* $i \in \{1, \ldots, k\}$ *and* k *is even, then* (4) *has a* 1*-dimensional set of solutions.*

If the projeced normals of all odd-numbered neighbors are pairwise linearly dependent and the projected normals of all even numbered neighbors are pairwise linearly dependent, than all κ_i are zero. Here is an example.

Example 4.4. Consider six points equally distributed over a sphere with the normals of the sphere. That is, the piecewise linear interpolant forms an octahedron. Then each of the 8 systems (4) has a 1-dimensional set of solutions. (See Fig. 4.1.) Example 4.4 is also a special case of Theorem 5.1, which asserts solvability for the improved algorithm if the data are in some sense C^2.

4.1.b. Linearly dependent normals

We now look at *dependent* triples $\{N^i, N^j, N^k\}$ of face-sharing normals. There are two types of dependencies:

(a) collinearity of a pair $\{N^\alpha, N^\beta\} \subset \{N^i, N^j, N^k\}$ of normals connected by an edge and

(b) coplanarity of the triple $\{N^i, N^j, N^k\}$ of normals without collinearity in any of its pairs.

We will refer to the system of type (4) at the data point P^l as (4^l) and say that a system (4^l) is *associated* with the triple $\{N^i, N^j, N^k\}$ if either the full triple of normals or two collinear

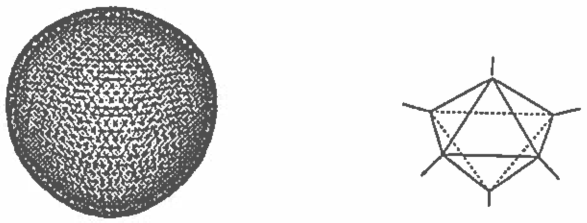
Fig. 4.1. The octahedron: a compatible even-point configuration.

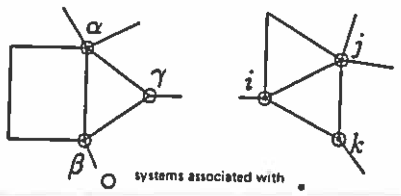

Fig. 4.2. Associated equations.

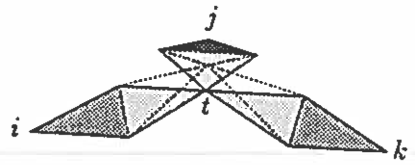

Fig. 4.3. A partially convex BB-net.

normals of that triple appear in the system. Thus, if the dependency is of type (b), then exactly (4^i), (4^j), and (4^k) are associated with $\{N^i, N^j, N^k\}$. If the dependency is of type (a), then (4^α) and (4^β) are associated with $\{N^i, N^j, N^k\}$ as is any (4^l) such that both α and β are neighbors of l (e.g. γ in Fig. 4.2). With this definition, we can capture all cases not treated in Theorem 4.1 and Theorem 4.2.

Theorem 4.5 [Cylindrical and planar data]. *Let $\{N^i, N^j, N^k\}$ be a triple of dependent face-sharing normals. Let n^i and n^j be the projections of N^i and N^j into the tangent plane at P^k, and let $\eta^{ij} \neq 0$ be defined by $n^i =: \eta^{ij} n^j$. Then the systems associated with $\{N^i, N^j, N^k\}$ are solvable if and only if, for any pair $\{N^\alpha, N^\beta\} \subset \{N^i, N^j, N^k\}$ with $N^\alpha = N^\beta$,*

$$(P^\alpha - P^\beta) N^\alpha = 0 \qquad \text{[planar data]} \tag{13}$$

and for any triple with $\det(N^i, N^j, N^k) = 0$ *and* $n^i = \eta^{ij} n^j$

$$(\eta^{ij})^2 (P^i - P^k)(N^k + 2N^i) = (P^j - P^k)(N^k + 2N^j) \qquad \text{[cylindrical data]}. \tag{14}$$

Proof. We choose without loss of generality $\{N, N_1, N_2\} := \{N^i, N^j, N^k\}$ and $\{N, N_1\} := \{N^\alpha, N^\beta\}$. First, we look at dependencies of type (a). If N and N_1 are linearly dependent, then (E_u^1) implies $N_1 u_1^0 = 0$ so that ($E_u^{2\prime}$) is equivalent to (13). Note that $u_1^0 = 0$ is a legitimate choice enforcing (C) at P for pairs $P_k P_1$ and $P_1 P_2$ since then, by symmetry, $\bar{u}_1 = (P_1 - P) \perp N$. We now look at dependencies of type (b) with $n_1 = \eta_{12} n_2$. By ($E_u^{2\prime}$), for the boundaries indexed 1 and 2, the compatibility condition, $\eta_{12} n_1 u_1^0 - 1/(\eta_{12} n_2 u_2^0) = 0$, becomes (14).

Conversely, once (14) or (13) holds, we may drop the redundant constraint. Moving, for example, (C) for the boundaries k and 1 into its place, leaves (4′) with a band of $k-1$ matrices of size 2×2 along the diagonal, a 2-vector in the $2k-1$st row and no other terms in the lower part of the matrix. Thus we can choose one tangent component freely and backsolve by virtue of either (13), (14) or the invertibility of the 2×2 matrices on the diagonal. □

Example 4.6. Consider patches on a cylinder without lids. On such a cylinder, any 3 edge-adjacent normals lie in a plane. If we choose, for example, a triangular patch such that two data points lie on one of the generating circles and the third is at equal distance from the two points on a different generating circle, then (14) holds.

4.2. Solvability of the inner constraints

The analysis for the inner constraints is simple. The corresponding systems are solvable if the tangent constraints can be enforced. We omit the cumbersome analysis for cylindrical data.

Theorem 4.7. *If* $\det(N_1, N_2, N_3) \neq 0$, *then* (7), *for rectangular patches, is solvable and* (6), *for triangular patches, is uniquely solvable. If $N^i = N^j$ for $i \neq j$, $i, j \in \{1, 2, 3\}$, then* (7) *and* (6) *are solvable if and only if the data are planar, i.e. if* (13) *holds.*

Proof. If $\det(N_1, N_2, N_3) \neq 0$, then the corresponding system (6) or (7′) is invertible for arbitrary c_i. If $N_1 = N_2$, then, for triangular patches,

$$N_1 b = N_1 P_1 \quad \text{and} \quad N_1 b = N_1 P_2 - \tfrac{1}{2} N_3 u_{21}. \tag{15}$$

Using the compatibility relation and the definition of a tangent vector $N_3 u_{21} = N_1 u_{23} = N_2 u_{23} = 0$, (15) is solvable exactly when the data are planar, i.e. $N_1(P_2 - P_1) = 0$. The reverse argument shows that planar data yield a 1- or even 2-dimensional set of solutions. With the choice $c_i := N_{i+1}P_{i+1} - N_{i+1}b_{ii+1} - \frac{1}{3}N_{i+2}(u_{i+1i})$, (7′) differs from (6) merely by a factor. If $N_1 = N_2$, then $c_i := N_{i+1}P_{i+1}$ and the analysis for the triangular patch applies. □

4.3. Convexity considerations

As Section 6 illustrates, the surface patches generated by LINNOR have few, if any, inflections. Along the boundaries this is not surprising since we restrict the boundary normal. However, with the notation of Fig. 3.2, if each tangent vector u_{ij} does not deviate too much from its 'desirable' value u_{ij}^*, i.e. from the projection of $P^j - P^i$ into the tangent plane at P^i, then we obtain additional 'convexity': all twist coefficients corresponding to P^i lie on the same side of the tangent plane as the curves emanating from P^i. Put differently, the second layer of the BB-net lies below the tangent plane as we traverse the BB-net from any data point to the interior (Fig. 4.3). This is significant since a polynomial piece is convex if the corresponding BB-net is.

Claim 4.8. *Consider the notation of Fig.* 3.2 *and a patch with vertices P_0, P_1 and P_2. Let the data be convex, i.e. $N_i(P_j - P_i) \leqslant 0$ and $N_j(P_i - P_j) \leqslant 0$ for $\{i, j, k\} := \{0, 1, 2\}$. If $u_{ij}(P_j - P_i) \geqslant 0$, then the boundary curves are convex. If additionally $u_{ik}(P_i - P_j) \geqslant 0$, the second layer of the BB-net lies below the tangent plane as the BB-net is traversed from any data point to the interior.*

Proof. The first assumption, $u_{ij}(P_j - P_i) \geqslant 0$, then implies $N_j u_{ij} \geqslant 0$ and hence the first claim follows from (E_u^2): since $N_i \bar{u} = -\frac{1}{2} N_j u_{ij} \leqslant 0$, the boundary curve has to lie 'below' the tangent plane at P^i. Similarly, the second assumption implies $N_k u_{ij} \geqslant 0$ and hence the second claim follows from (E_{v3}^2), resp. (E_{v4}^2). □

Thus the choice of U^* and T^* in Section 3 improves convexity. In fact, if $U = U^*$, then the first claim holds and the second claim holds, provided neighboring tangent vectors form angles less than $\pi/2$.

5. An improvement: curvature weights

Like most other methods for local surface interpolation, LINNOR constructs surface patches such that the versal and transversal derivatives are coplanar. This does not imply proper orientation of the surface normal, however. As the following example illustrates, cusping is an important issue.

Example 5.1 [Example 4.4 contd.]. We orient the octahedron so that one point lies at (0, 0, 1) and another at ($\pi/4$, $\pi/4$, 0). We bend the normal at (0, 0, 1) towards (0, 1, 0) and thus generate three sets of data that illustrate how a cusp 'develops'. Figs. 5.1–5.3 show the BB-net of the lower front facet as it reacts to the change. In Fig. 5.3 the order of the tangent vectors is reversed. That is, the surface has two cusps.

The precise statement of the cusping conditions for the algorithm is possible but somewhat tedious and the general result unappealing. Only special cases, e.g. data equally distributed over a sphere, provide a simple characterization. We refer to [Peters '88] for a detailed discussion of the cusping problem and the fact that cusping at a data point P is characterized by k inequalities made up from the location and normal of the data point and all its neighbors. Hence we cannot hope to remove cusps by changing one or several normals since each change affects all the systems at the neighbor points. Instead we improve our choice of n. Since we only have to interpolate to the normal direction and not its magnitude, we can choose

$$n(u) \sim (N^0, \omega N^1),$$

where ω is a positive weight. Changes in ω affect only the systems at the end points of the boundary. The weight, ω, prescribes the rate at which the boundary normal turns from N^0 to N^1. While it is possible to fine-tune the surface with the help of curvature weights, we stick with the simple *a priori* choice $\omega^{ij} := \text{shape}(i, j)$. Example 5.2 demonstrates that this choice is adequate.

Example 5.2 [Example 4.4 contd.]. The top of the regular octahedron of Fig. 4.1 is depressed, so that the point (0, 0, 1) moves to (0, 0, 0.5). The normals remain $N_i = P_i / \| P_i \|$.
Fig. 5.4 shows the cusping match without curvature weights, Fig. 5.5 the C^1 match with $\omega^{ij} = \text{shape}(i, j)$. The surface remains C^1 until the data are no longer convex, i.e. until the top is pushed below the plane spanned by its four neighbors.

The choice $\omega^{ij} = \text{shape}(i, j)$ has some justification as a 'mean value' between the extremes $\omega^{ij} = \text{shape}(i, j)*2$ and $\omega^{ij} = \text{shape}(i, j)/2$ which generate a cusp-free [bi]linear interpolant and as a value that reduces the boundary curve to a quadratic (cf. [Peters '88]). Another justification is its role in Theorem 5.1 below.

Theorem 5.1 [C^2 data]. *If the data at* P *are consistent with some second fundamental form and* $\omega^{ij} = \text{shape}(i, j)$, *then* (4) *has a solution and the resulting surface is cusp-free at* P.

Proof. Let $\Delta_i := P_i - P$. We first observe that $(E_u^{2\prime})$ simplifies to

$$N_i u_i^0 = \tfrac{2}{3} \Delta_i N \tag{$E_u^{2\omega}$}$$

for $\omega^{ij} = \text{shape}(i, j)$. Furthermore, (C) becomes

$$N_{i+1} u_i^0 \frac{N\Delta_{i+1}}{N_{i+1}\Delta_{i+1}} = N_i u_{i+1}^0 \frac{N\Delta_i}{N_i \Delta_i}. \tag{C^ω}$$

Since $N_i \xi_1 = k_1$ and $N_i \xi_2 = k_2$ for all i, we may replace N_i by N_{i+1} in $(E_u^{2\omega})$ to obtain

$$\tfrac{2}{3} N\Delta_i \frac{N\Delta_{i+1}}{N_{i+1}\Delta_{i+1}} = \tfrac{2}{3} N\Delta_{i+1} \frac{N\Delta_i}{N_i \Delta_i}$$

which in turn simplifies to $N_{i+1}\Delta_{i+1} = N_i \Delta_i$, the third condition on the data. Hence, we may set $u_i^0 =: \delta_i T_i$ for some vector T_i in the tangent plane and enforce the constraint $(E_u^{2\omega})$ by proper choice of δ_i. In particular, we are free to choose T_i as the projection of Δ_i into the tangent

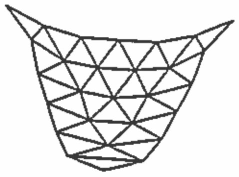
Fig. 5.1. Early stage.

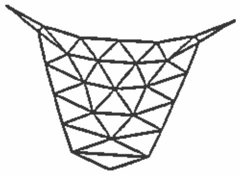
Fig. 5.2. Almost a cusp.

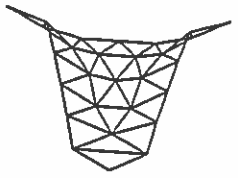
Fig. 5.3. Two cusps.

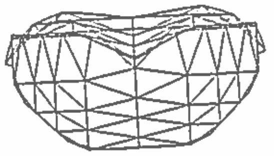
Fig. 5.4. 'Squashed' octahedron without curvature weights.

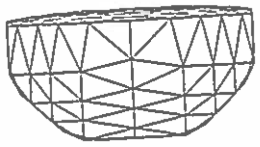
Fig. 5.5. 'Squashed' octahedron with curvature weights.

plane. The tangent vectors, u_i^0, are then ordered like the neighbor points, and, by Assumption 1.2, satisfy

$$\nu((p_v \times p_u) \times p_u) q_w(0, 0) = ((u_{i-1}^0 \times u_i^0) \times u_i^0) u_{i+1}^0 > 0$$

for some positive constant ν. □

Table 1
Approximating the unit sphere

points	volume	max. curvature
4	1.558	1.323
6	1.102	1.030
8	1.161	1.152
12	1.043	1.059

6. Examples

This section shows some surfaces generated by LINNOR. Table 1 summerizes the results for approximating the unit sphere. The data points are evenly spaced and the volume is measured relative to the volume of the unit sphere.

Fig. 6.1 shows the 12-point interpolant. The checker pattern of the *left* view emphasizes isoparametric lines and patch distribution. The isoclines on the *right* give the change in the

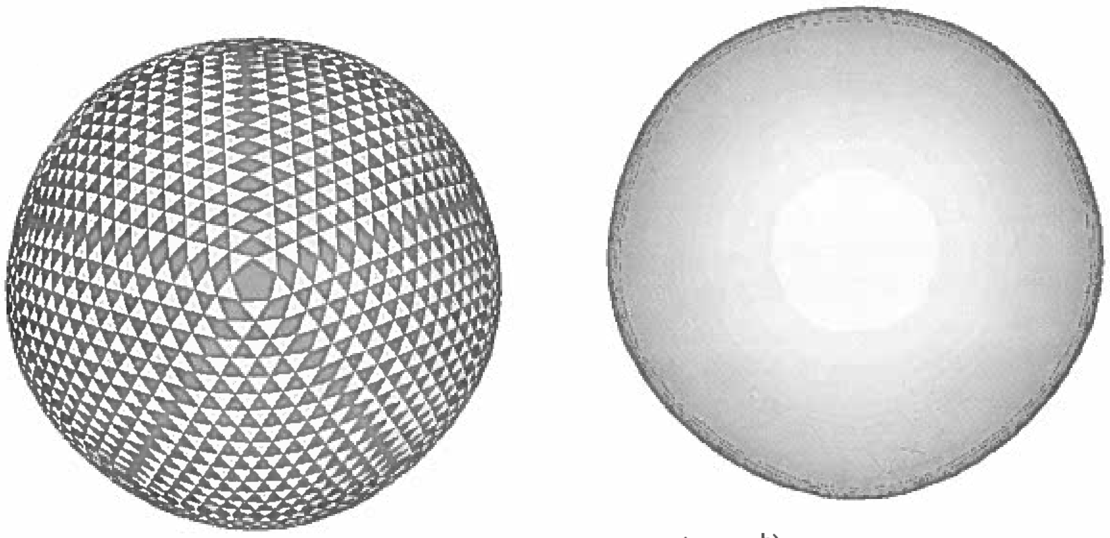
Fig. 6.1. 12-point interpolant (ikosahedron mesh).

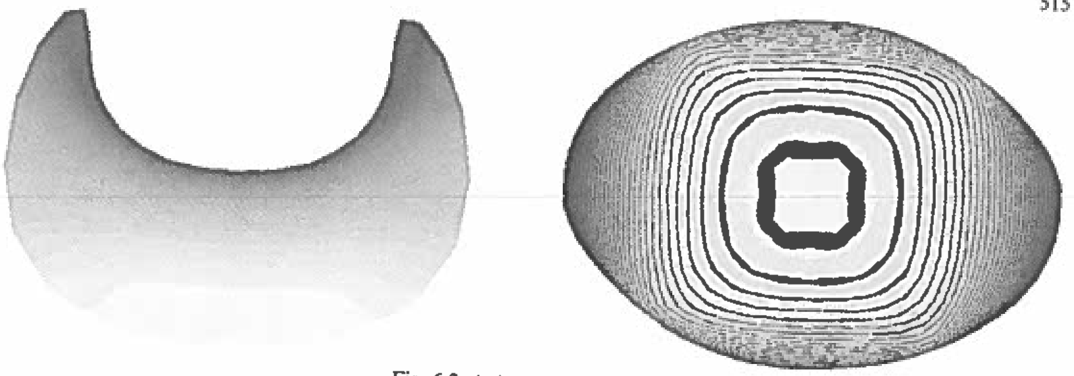

Fig. 6.2. A 4-patch saddle.

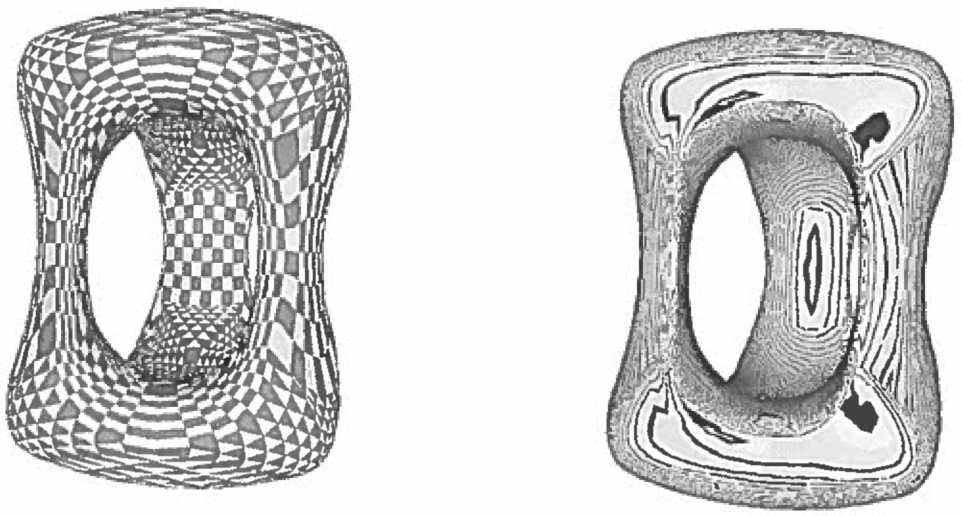

Fig. 6.3. A smooth surface of genus 2.

normal. Fig. 6.2 gives a side view and a top view with isoclines of a four-patch saddle. The last set of 50 data points (Fig. 6.3) was suggested by M.A. Sabin: three 'handles' meet smoothly at $2\pi/3$.

Acknowledgements

I thank Carl de Boor and the referees for their comments.

References

de Boor, C. (1987), B-form basics, in: G. Farin, ed., *Geometric Modeling*, SIAM, Philadelphia, PA.

Farin, G. (1983), Smooth interpolation to scattered 3D-data, in: R.E. Barnhill and W. Boehm, eds., *Surfaces in CAGD*, North-Holland, Amsterdam.

Farin, G. (1986), Triangular Bernstein–Bézier patches, Computer Aided Geometric Design 3, 83–127.

Peters, J. (1988), Local piecewise cubic C^1 surface interpolants for rectangular and triangular tessellations, CMS Tech. Rep. 89-10 UW-Madison.

Peters, J. (1989), Local interpolation of a cubic curve mesh by a piecewise [bi]quartic C^1 surface without splitting, CMS Tech. Rep. 89-25 UW-Madison to appear in Constr. Approx.

Sabin, M.A. (1968), Conditions for continuity of surface normal between adjacent parametric surfaces, Tech. Rep., British Aircraft Corporation Ltd.

Sarraga, R.F. (1986), G^1 interpolation of generally unrestricted cubic Bézier curves, Computer Aided Geometric Design 4, 23–40.