Interactive Shape Control and Rapid Display of A-patches*

Chandrajit L. Bajaj Jindon Chen Guoliang Xu

Department of Computer Sciences
Purdue University
West Lafayette, Indiana

{bajaj,jdc,xuguo}@cs.purdue.edu

Abstract

A-patches are implicit surfaces in Bernstein-Bézier(BB) form that are smooth and single-sheeted. In this paper, we present algorithms to utilize the extra degrees of freedom of each A-patch for local shape control. A *ray shooting* scheme is also given to rapidly generate polygonal approximations of A-patches for graphic display. A distributed implementation of this scheme gives nice "real time" performance on rendering the A-patches to support interactive shape control.

Keywords: algebraic surface, free form surface, interactive techniques, smoothing, spline and piecewise surface, rendering, polygonization, computer-aided geometric design, solid modeling.

1 Introduction

The A-patch is a smooth and single-sheeted zero-contour patch of a trivariate polynomial in Bernstein-Bézier(BB) form defined within a tetrahedron[BCX94a], where the "A" stands for algebraic. Solutions to the problem of constructing a C^1 mesh of implicit algebraic patches based on polyhedron $\mathcal{P}$ have been given by [Dah89, BCX94a, BCX94b, DTS93, Guo91, Guo93, BI92]. While papers [BI92, Dah89, DTS93, Guo91, Guo93] provide heuristics based on monotonicity and least square approximation to circumvent the multiple sheeted and singularity problems of implicit patches, [BCX94a] introduces new sufficiency conditions for the BB form of trivariate polynomials within a tetrahedron, such that the zero contour of the polynomial is a single sheeted non-singular surface within the tetrahedron (the A-patch) and guarantees that its cubic-mesh complex for $\mathcal{P}$ is both nonsingular and single sheeted.

The geometry of implicit surfaces has been proven to be more difficult to specify, interactively control, or polygonize than those of their parametric counterpart. Literature that concerns these issues includes [Bli82, WMW86, Pra87, Blo88, BW90, Hal90, BIW93, WH94]

*This work was supported in part by NSF grants CCR 92-22467, DMS 91-01424, AFOSR grants F49620-93-10138, F49620-94-1-0080, NASA grant NAG-1-1473 and a gift from AT&T.

In the A-patch scheme proposed in [BCX94a], several degrees of freedom remain to modify the resulting C^1 piecewise surface. In this paper, we utilize these weights for local shape control. We also present a rapid *ray shooting* algorithm, to show how to polygonize A-patches in a distributed fashion.

The rest of this paper is as follows. Section 2 gives some preliminary facts about Bernstein-Bézier(BB) representations, A-patches and a simplicial hull. Section 3 discuss the techniques for shape controls of A-patches. Section 4 describes the ray shooting algorithm for polygonization and graphics display.

2 Notation and Preliminary Details

2.1 Bernstein-BézierRepresentation

Let $\{p_1, \dots p_j\} \in I\!R^3$. Then the *convex hull* of these points is defined by $[p_1p_2...p_j] = \{p \in I\!R^3 : p = \sum_{i=1}^{j} \alpha_i p_i, \alpha_i \geq 0, \sum_{i=1}^{j} \alpha_i = 1\}$. Let $p_1,\ p_2,\ p_3,\ p_4 \in I\!R^3$ be affine independent. Then the tetrahedron(or three dimensional simplex) with vertices $p_1,\ p_2,\ p_3$, and p_4, is $V = [p_1p_2p_3p_4]$. For any $p = \sum_{i=1}^{4} \alpha_i p_i \in I\!R^3$, $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T$ is the barycentric coordinate of p. Let $p = (x, y, z)^T$, $p_i = (x_i, y_i, z_i)^T$.

Any polynomial $f(p)$ of degree m can be expressed in Bernstein-Bézier(BB) form over V as $f(p) = \sum_{|\lambda|=m} b_\lambda\ B_\lambda^m(\alpha), \quad \lambda \in \mathcal{Z}_+^4$ where $B_\lambda^m(\alpha) = \frac{m!}{\lambda_1!\lambda_2!\lambda_3!\lambda_4!}\ \alpha_1^{\lambda_1}\alpha_2^{\lambda_2}\alpha_3^{\lambda_3}\alpha_4^{\lambda_4}$ are the trivariate Bernstein polynomials for $|\lambda| = \sum_{i=1}^{4} \lambda_i$ with $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)^T$. Also $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T$ is the barycentric coordinate of p, $b_\lambda = b_{\lambda_1\lambda_2\lambda_3\lambda_4}$(as a subscript, we simply write λ as $\lambda_1\lambda_2\lambda_3\lambda_4$) are called control points, and $\mathcal{Z}_+^4$ stands for the set of all four dimensional vectors with nonnegative integer components. Let

$$F(\alpha) = \sum_{|\lambda|=m} b_\lambda B_\lambda^m(\alpha),\ |\alpha| = 1, \tag{2.1}$$

be a given polynomial of degree m on the tetrahedron $S = \{(\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T \in I\!R^4\ :\ \sum_{i=1}^{4} \alpha_i = 1,\quad \alpha_i \geq 0\}$. The surface patch within the tetrahedron is defined by $S_f \subset S\ :\ F(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = 0$.

2.2 A-Patch

Definition 2.1 *Three-sided patch.*

Let the surface patch S_F be smooth on the boundary of the tetrahedron S. If any open line segment (e_j, α^*) with $\alpha^* \in S_j = \{(\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T\ :\ \alpha_j = 0, \alpha_i > 0,\ \sum_{i \neq j} \alpha_i = 1\}$ intersects S_F at most once(counting multiplicities), then we call S_F a three-sided j-patch (see Figure 2.1).

Definition 2.2 *Four-sided patch.*

Let the surface patch S_F be smooth on the boundary of the tetrahedron S. Let (i, j, k, ℓ) be a permutation of (1, 2, 3, 4). If any open line segment (α^*, β^*) with $\alpha^* \in (e_ie_j)$ and $\beta^* \in (e_ke_\ell)$ intersects S_F at most once(counting multiplicities), then we call S_F a four-sided ij-$k\ell$-patch (see Figure 2.1).

It is easy to see that if S_F is a four-sided ij-$k\ell$-patch, it is then also a ji-ℓk-patch, a ℓk-ji-patch, and so on.

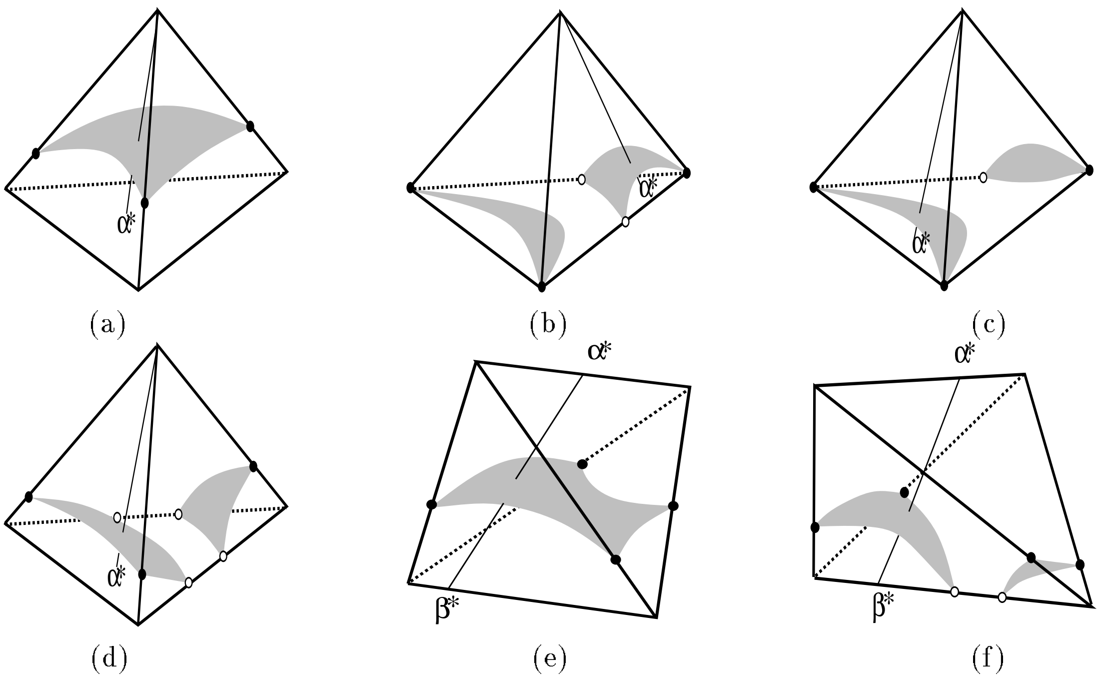

Figure 2.1: Three-sided patches (a)-(d) and four-sided patches (e)(f) The filled vertices mark the boundary patch. Note that these disconnected patches are still single sheeted.

Lemma 2.1 *([BCX94a]) The three-sided j-patch and the four-sided ij-$k\ell$-patch are smooth (non-singular) and single-sheeted.*

Theorem 2.1 *([BCX94a]) (i) Let $F(\alpha) = \sum_{|\lambda|=n} b_\lambda B_\lambda^n(\alpha)$ satisfy the smooth vertex and smooth edge conditions and $j(1 \le j \le 4)$ be a given integer. If there exists an integer $k(0 \le k < n)$ such that*

$$b_{\lambda_1\lambda_2\lambda_3\lambda_4} \geq 0, \qquad \lambda_j = 0, 1, \ldots, k-1, \tag{2.2}$$

$$b_{\lambda_1\lambda_2\lambda_3\lambda_4} \leq 0, \qquad \lambda_j = k+1, \ldots, n \tag{2.3}$$

and $\sum_{\substack{|\lambda|=n \\ \lambda_j=0}} b_\lambda > 0$ if $k > 0$, $\sum_{\substack{|\lambda|=n \\ \lambda_j=m}} b_\lambda < 0$ for at least one $m(k < m \le n)$, then S_F is a three-sided j-patch.
(ii) Let $F(\alpha) = \sum_{|\lambda|=n} b_\lambda B_\lambda^n(\alpha)$ satisfy the smooth vertex and smooth edge conditions and (i,j,k,ℓ) be a permutation of (1, 2, 3, 4). If there exists an integer $k(0 \le k < n)$ such that

$$b_{\lambda_1\lambda_2\lambda_3\lambda_4} \geq 0; \qquad \lambda_i + \lambda_j = 0, 1, \ldots, k-1, \tag{2.4}$$

$$b_{\lambda_1\lambda_2\lambda_3\lambda_4} \leq 0; \qquad \lambda_i + \lambda_j = k+1, \ldots, n \tag{2.5}$$

and $\sum_{\substack{|\lambda|=n \\ \lambda_i+\lambda_j=0}} b_\lambda > 0$ if $k > 0$, $\sum_{\substack{|\lambda|=n \\ \lambda_i+\lambda_j=m}} b_\lambda < 0$ for at least one $m(k < m \le n)$, then S_F is four-sided ij-$k\ell$-patch.

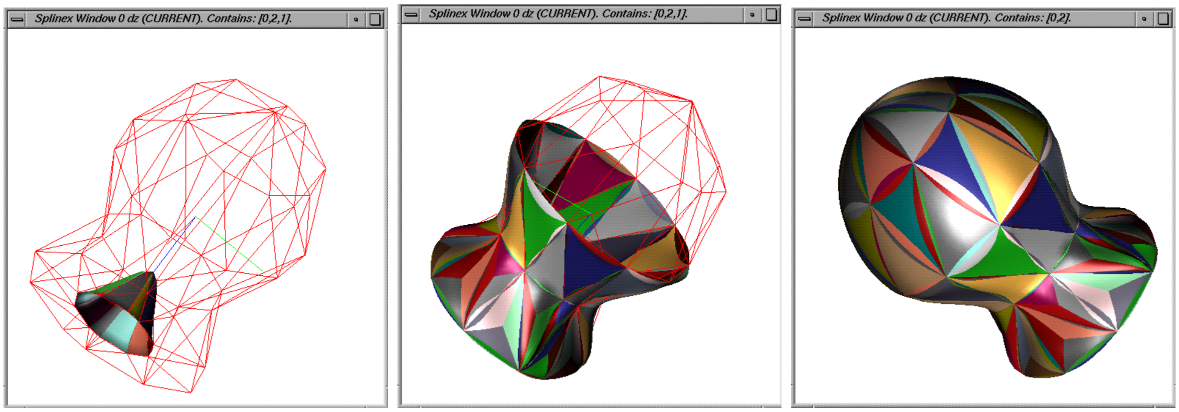

Figure 2.2: Interpolatory scheme: smoothing a "jar"

2.3 Two useful properties

a. For a three-sided j-patch, any point $P \in S_F$ can be mapped to a barycentric triple $(\alpha_i, \alpha_k, \alpha_\ell)$, $\alpha_i + \alpha_k + \alpha_\ell = 1$, $\alpha_i, \alpha_k, \alpha_\ell \geq 0$ or a point $\alpha^* \in S_j = \{(\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T : \alpha_j = 0$, Furthermore, there exists a one to one mapping between S_F and $S'_j = \{\alpha^* : \alpha^* \in S_j, F(e_j) \cdot F(a*) \leq 0\}$.

b. For a four-sided ij-$k\ell$-patch, any point $P \in S_F$ can be mapped to a tuple (α_i, α_k), $0 \leq \alpha_i \leq 1$, $0 \leq \alpha_k \leq 1$, or two points $\alpha^* \in (e_i e_j) = \{(\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T : \alpha_k = \alpha_\ell = 0\}$ and $\beta^* \in (e_k e_\ell) = \{(\alpha_1, \alpha_2, \alpha_3, \alpha_4)^T : \alpha_i = \alpha_j = 0\}$. Furthermore, there exists a one to one mapping between S_F and $\{(\alpha_i, \beta_k)^T : F(\alpha^*) \cdot F(\beta^*) \leq 0\}$. If $F(\alpha^*) = 0$, S_F is degenerate and all the points with the same α_k collapse into one point.

2.4 Simplicial Hull

The simplicial hull packing scheme that we use here is detailed in [BCX94a]. For a triangulation $\mathcal{T}$, a pair of *face tetrahedra*, $[p_1p_2p_3p_4]$ and $[p_1p_2p_3q_4]$ are built on each face $[p_1p_2p_3] \in \mathcal{T}$, one on each side. Two face tetrahedra $[p_1p_2p_3p_4]$ and $[p'_1p_2p_3p'_4]$ on adjacent faces do not intersect with each other (other than share a common edge). Two *edge tetrahedra* $[p''_1p_2p_3p_4]$ and $[p''_1p_2p_3p'_4]$ are "filled" into and gap in between, where $p_4p''_1p'_4$ are collinear (see Figure 2.5). Each of the tetrahedra is a local domain that defines an A-patch.

In a C^1 scheme, the face tetrahedra need to be high enough so that for any vertex of the triangulation, there is a tangent plane that intersects all the face tetrahedra pair built on its incident faces. We call this the *tangent containment* condition. Face tetrahedra that do not intersect with any tangent plane can be deleted as they contain no A-patches. A three-sided patch that passes through p_1, p_2, p_3 is defined in each face tetrahedron $[p_1p_2p_3p_4]$, while a four-sided patch passes through p_2 and p_3 is defined in each edge tetrahedron.

2.5 The C^1 and single-sheeted scheme

See Figure 2.5. Let $V_1 = [p_1p_2p_3p_4]$ and $V'_1 = [p_1p_2p_3q_4]$ be a pair of face tetrahedra, $V_2 = [p'_1p_2p_3p'_4]$ and $V'_2 = [p'_1p_2p_3q'_4]$ be the neighboring pair that share edge $[p_2p_3]$, $W_1 = [p''_1p_2p_3p_4]$, $W_2 = [p''_1p_2p_3p'_4]$, $W'_1 = [q''_1p_2p_3p_4]$, $W'_2 = [q''_1p_2p_3p'_4]$. be the edge tetrahedra between them, and

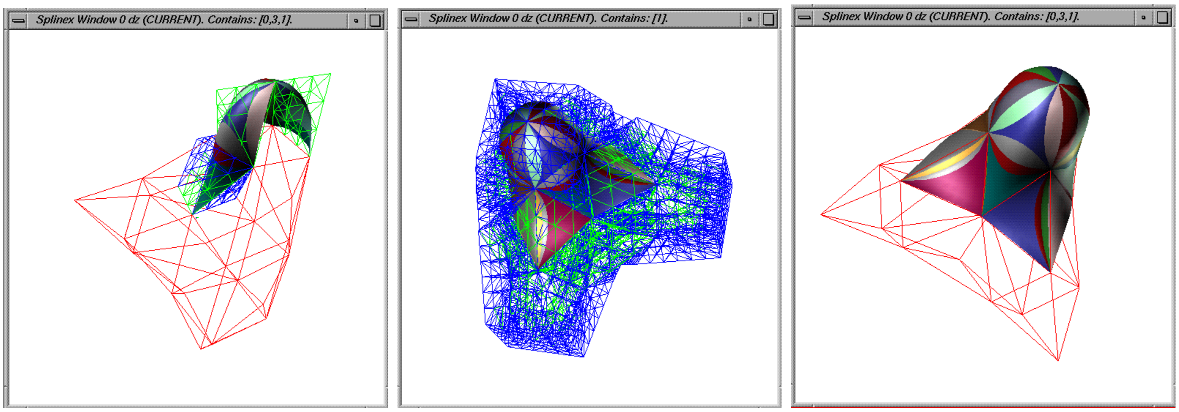

Figure 2.3: Interpolatory scheme: smoothing a "three finger" free form object

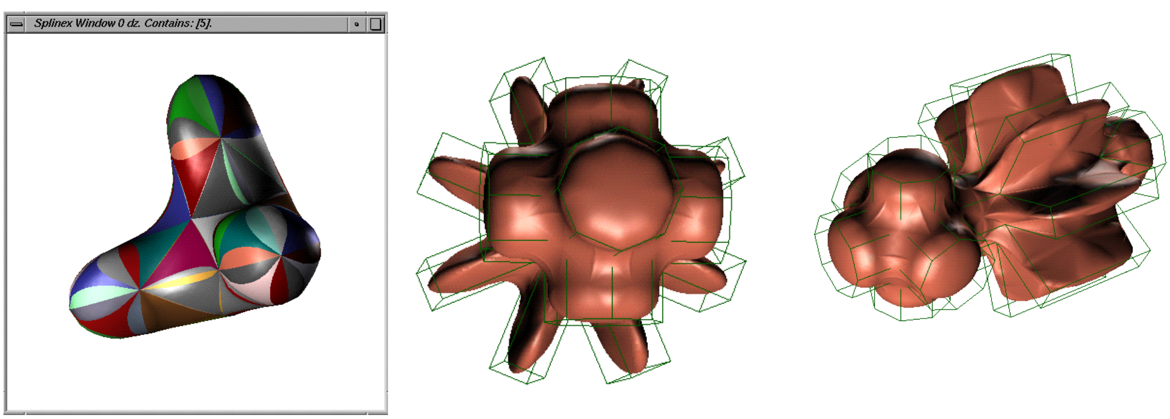

Figure 2.4: Left: a complete "three finger". Center and Right: Corner cutting scheme, smoothing a "satellite"

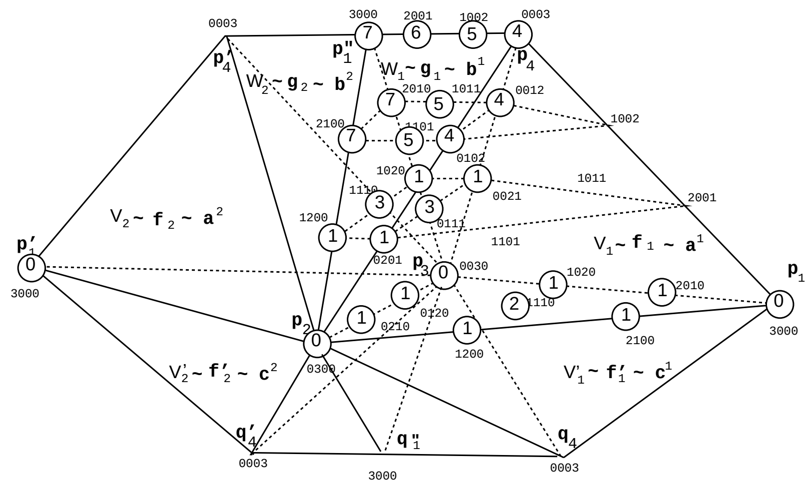

Figure 2.5: Adjacent Tetrahedra, Cubic Functions and Control Points for two Non-Convex Adjacent Faces

the polynomials f_i over V_i, g_i over W_i, f'_i over V'_i and g'_i over W'_i be expressed in Bernstein-Bézier form with coefficients a^i_λ, b^i_λ and c^i_λ, respectively.

In the following algorithms, we mostly describes the setting between V_1 and W_1 and between W_1 and W_2, as the others are trivially symmetric.

Algorithm 1 C^1 cubic scheme

(1) *(At vertices. Interpolating the positions and normals of the vertices)*
For face tetrahedron V_1*, set* $a^1_{3000} = a^1_{0300} = a^1_{0030} = 0.0$*, and* $(a^1_{2100}, a^1_{2010}, a^1_{2001})$*,* $(a^1_{1200}, a^1_{0210}, a^1_{0201})$*, and* $(a^1_{1020}, a^1_{0120}, a^1_{0021})$ *according to the normal at the vertices. For edge tetrahedron* W_1*, set* $b^1_{0300} = b^1_{0030} = 0.0$ *and set* $(b^1_{1200}, b^1_{0210}, b^1_{0201})$ *and* $(b^1_{1020}, b^1_{0120}, b^1_{0021})$ *accordint to the normal.*

(2) *(Third level)*
(i) When $[p_1p_2p_3p'_1]$ *is not coplanar,* $\alpha_4 \neq 0$*, solve a linear equation (*C^1 *condition between* V_1 *and* V_2 *arount edge* $[p_2p_3]$*) for* a^1_{0111}*, then solve compute* b^1_{0111} *accordingly. Similarly,* a^1_{1011} *and* a^1_{1101} *are set.*

(ii) A degenerated case is when $[p_1p_2p_3p'_1]$ *are coplanar, namely, the base faces of two neighboring tetrahedra are coplanar, in which case* $\alpha_4 = 0$*,* a^1_{0111} *cannot be expressed in terms of other weights in* b_1*. In this case, a Clough-Tocker split is needed.*

(3) *(Second level)*
Setting b_{1101}*,* b_{1011} *of the neighboring tetrahedra by* C^1 *condition.*

(4) *(Top level)*
Setting b_{2100} *in the neighboring edge tetrahedra by* C^1 *condition.*

(5) *(*C^1 *condition between edge patches)*
Setting b_{2100}*,* b_{2010} *and* b_{3000} *according to the* C^1 *condition between* W_1W_2*.*

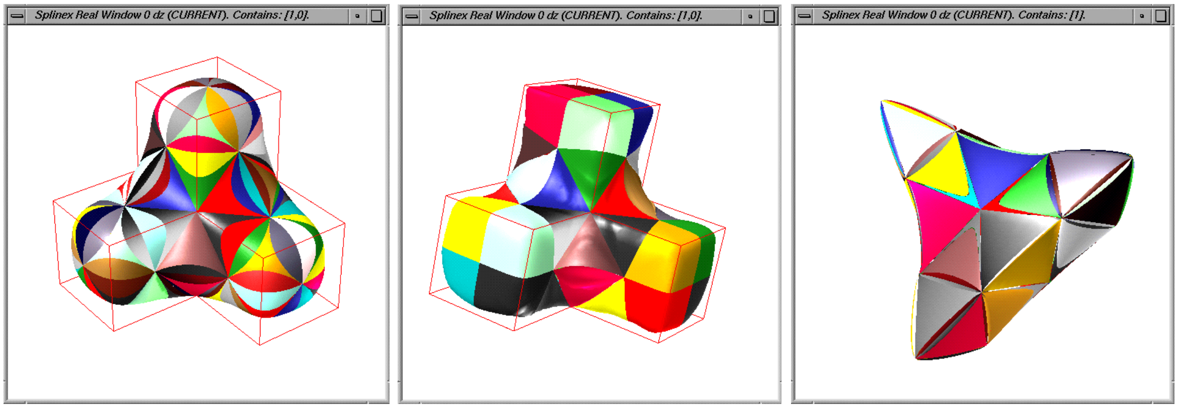

Figure 2.6: Corner cutting scheme: smoothing the same tricube with different configurations.

In Algorithm 1, a^1_{1110}, a^1_{1002}, a^1_{0102}, a^1_{0012} and a^1_{0003} in f_i and b^1_{2001} in g_i are free. In the following algorithm, Algorithm 2, we adjust them to enforce single-sheetedness. The description of Algorithm 2 is actually an add-on of Algorithm 1. Step n' is to be inserted before step n of Algorithm 1.

Algorithm 2 Single-sheeted

(2') *If* $[p_1p_2p_3]$ *is not coplanar to the base face of any its neighboring face tetrahedra,* a^1_{1110} *is left free from the* C^1 *condition.*
(i) If all $a^1_{\lambda_1\lambda_2\lambda_3 0}$ *except* a^1_{1110} *are of the same sign, adjust* a^1_{1110} *to make sure the BB polynomial defined over* $[p_1p_2p_3]$ *does not intersect with face* $[p_1p_2p_3]$*. A simple solution is just setting* a^1_{1110} *to be of the same sign.*
(ii) If one of the three edges $[p_1p_2]$, $[p_2p_3]$, *or* $[p_3p_2]$ *is nonconvex (for example, on* $[p_2p_3]$, a^1_{0210} *and* a^1_{0120} *are of different signs), then the weights at the third level,* $a^1_{\lambda_1\lambda_2\lambda_3 1}$ *have to be of the same sign as that of the top level weight. If needed, a Clough-Tocher split is performed to ensure this sign restriction. In the case that a Clough-Tocher split is need,* $V_1 \Rightarrow (V_{11}, V_{12}, V_{13})$ *and* $V_2 \Rightarrow (V_{21}, V_{22}, V_{23})$*, Adjust* a^{11}_{1110}, a^{21}_{1110} *to ensure that* a^1_{0111}, a^2_{0111}, *to be positive and* c^1_{0111}, c^2_{0111} *to be negative.*

(3') *(Single-sheeted condition/Second level)*
Adjust a_{2001}, a_{0201} *and* a_{0021} *so that they are big enough in absolute value to ensure that* b_{1101}, b_{1011} *in the neighboring edge tetrahedra are of the same sign as they are related in* C^1 *condition.*

(4') *(Single-sheeted condition/Top level)*
Adjust a_{0003} *so that it is big enough in absolute value to ensure that* b_{2100} *in the neighboring edge tetrahedra are of the same sign when setting the* C^1 *condition between them.*

(5') *(Single-sheeted condition/Edge patch) Set edge weight* b_{1200} *to be of the same sign as* b_{0003}.

3 Shape Control

In the C^1 and single-sheeted scheme, weights a^1_{1110}, a^1_{1002}, a^1_{0102}, a^1_{0012} and a^1_{0003} of V_1 and b^1_{2001} of W_1 are adjustable within some ranges. This freedom, on the one hand, allows us to locally change the shape of the surface, while on the other hand, burdens us with extra work to remove bumpy defects.

3.1 Default weights

3.1.1 Approximating lower patches

One commonly used method is to keep the surface patch close to a lower degree patch([Baj92, DTS93]), which, in our case, is quadric patch. Specifically, we determine a quadric that is least square approximating the known weights of the cubic and then set the unknown weights of the cubic from the quadric by a degree raising formula.

However, the least square optimization is subject to the C^1 and single-sheeted linear constraints. Hence this is typically a non-linear programming problem with quadratic object function and linear constraints. We employ some simple heuristic to obtain an approximation of the optimal solution should the solution of the unconstrained counterpart falls outside the constraints.

3.1.2 Edge patch first

The previous method fails to consider the shape of neighboring face patches beyond the fact that they share common normals. Such neglect could lead to unwanted variation in the edge patch between them. The next scheme sets the ideal weights of the edge patches first. Then the weights of the face patches "honor" the choice of the edge patches by making sure the ideal edge patch weights are changed the least when the C^1 conditions are set.

Considering edge patch W_1 and W_2. Let ℓ be the intersection of the two tangent plane at p_2 and p_3. The weights around the two vertices are set by the interpolatory and normal conditions. We set the rest of the weights so that $g_1 = 0$ is actually a swept surface parallel to ℓ and the cross section curve approximate a quadric in least square sense. This can be done by a few basis change of the polynomials. Set g_2 to be the same as g_1.

By C^1 conditions, the edge patch weights propose values for the neighboring face patches. A face patch, however, take weighted averages of the proposed values from different edge patches around it and set them as default values. In taking the weighted average, smaller edge patch weighs more as for the same BB-represented surfaces, smaller tetrahedra yields larger curvature and larger curvature change for the same amount of change in the weights.

The heuristic methods we discussed above are "cheap" for they do not require any massive computations. "Expensive" global optimization methods can also be used to improve the surface by minimizing the "energy", or some smoothness criteria, of the surface.

3.2 Interactive Shape Control

At a vertex, if the normal becomes longer then the surface becomes flatter around this vertex. The change of normal length is equivalent to a scaling of all the weights around the vertex by the same ratio. Altering the direction of a normal also changes the shape around the verttex. However, when we desire a direction change we need to check that it does not violate the tangent containment constraints.

In a face tetrahedra V, we can raise or lower the weights a_{0003}, a_{1002}, a_{0102} and a_{0012} defining the surface patch without altering its variation (see Figure 2.6); Weight a_{1110} alters the variation of the surface. Please note that by C^1 conditions, a_{1110} is related to a_{0111}, a_{1011} and a_{1101}, and also

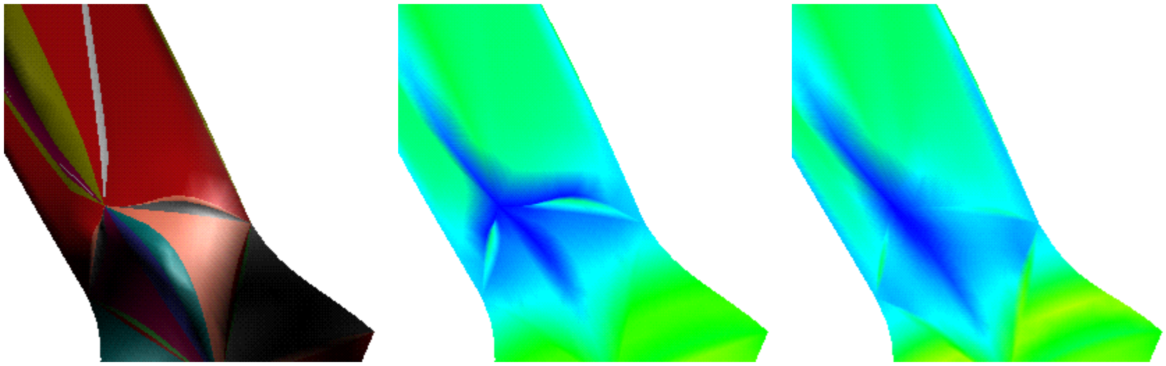

Figure 3.7: Setting the edge patches first to reduce unwanted variations. Left: piecewise surface in multipatch shading mode. Center and Right: shown in an RGB model where the normal vectors become the color RGB vector. Center: face patches are set first. Right: edge patch are set first. One sees that the right picture exibits a fairer surface

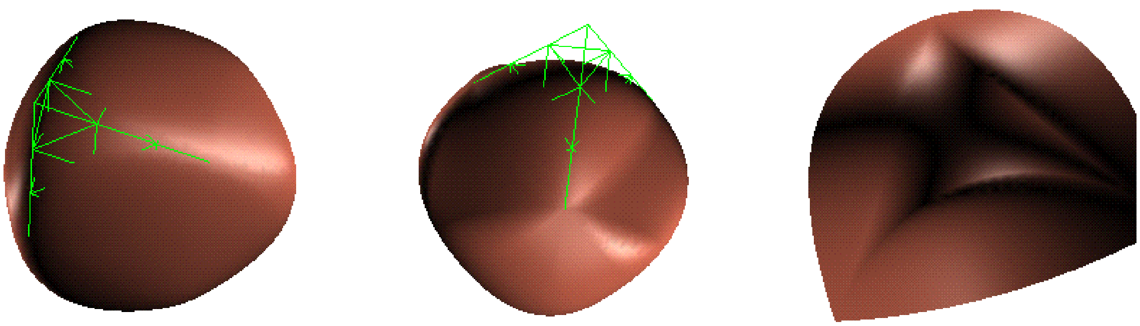

Figure 3.8: Changing a_{1110} on a face patch affecting other face patches. From left, $a_{1110} = -0.5, -1.0, 0.5$. The corner is popped up in the center picture while sucked in in the right one

there are other weights that could alter the surface geometry (see Figure 3.8). Hence the effect of changing a_{1110} in changing variation could be exaggerated or reduced, depending on the geometric relationship between the tetrahedra. Also, as a_{1110} are related to neighboring patches in a linear equation, change of this weight propagates further into to neighboring patches, while that of a_{1002}, a_{0102} and a_{0012} do not affect neighboring face patches and that of a_{0003} affects only adjacent edge patches.

In general, a desirable modification involves collaboration of several adjustable weights rather than a single one. Hence an alternative way is to specify some additional data points in the tetrahedra, and then approximate these points in the least square sense.

4 Rapid Display Scheme

Algorithms to generate polygonal approximations of a three-sided or four-sided patch are suggested by properties (d) and (e) of A-patches in [BCX94a].

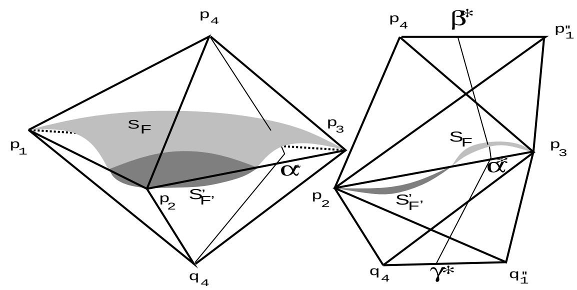

Figure 3.9: Ray shooting A-patches

For two based sharing 3-sided 4-patch $[p_1p_2p_3p_4]$ and $[p_1p_2p_3q_4]$, (see Figure 2.5) any point V on the surface defined in them is in a one to one mapping with a point α^* on face $[p_1p_2p_3]$. This is because polyline $p_4\alpha^*q_4$ intersects with the union of two patches exactly once (see Figure 3.9). Hence the barycentric coordinate of points on $[p_1p_2p_3]$ can be used as a parameterization of the union of the two three-side apatches. Similarly, for two edge sharing four-sided 14-patches $[p_1''p_2p_3p_4]$ and $[q_1''p_2p_3q_4]$(see Figure 2.5), the union of the two 4-sided A-patches can be parameterized into a quadtilateral domain (s,t), where $(1-s,s)$ is the barycentric coordinate of both a point $\beta^* \in [p_1''p_4]$ and a point $\gamma^* \in [q_1''q_4]$. Let $(1-t,t)$ be the coordinate of a point $\alpha^* \in [p_2p_3]$ as polyline $\beta^*\alpha^*\gamma^*$ intersection the union of A-patches once(see Figure 3.9). We call such a pair of three-sided or four-sided patches as a *double* A-patch and call the parametrizations *ray coordinate*.

A simple polygonization algorithm can be described as follows, for both three-sided and four-sided A-patches. The algorithm works in an adaptive fashion. Beginning with some initial triangles, we keep subdividing them until the shape is desirable to some criterion.

Algorithm 3 Ray-shooting

(1) Initialization. (i) three-sided double A-patches*: Compute vertices A, B and C whose ray coordinates are* $(0,0)$*,* $(1,0)$ *and* $(0,1)$*, respectively. Enter $[ABC]$ as the first cell in the polygon list.*

(ii) four-sided double A-patch*: Compute vertices A, B, C and D whose ray coordinates are* $(0,0)$*,* $(0,1)$*,* $(1,1)$ *and* $(1,0)$*, respectively. Enter $[ABC]$ and $[ACD]$ as the first two cells in the polygon list.*

(2) For each edge $[A=(s_0,t_0), B=(s_1,t_1)]$, if it is too long, Compute

$$C = \begin{pmatrix} s \\ t \end{pmatrix} = (1-\alpha)\begin{pmatrix} s_0 \\ t_0 \end{pmatrix} + \alpha \begin{pmatrix} s_1 \\ t_1 \end{pmatrix}$$

for some $\alpha \in [0,1]$, weighted by the normals at A and B. Replace $[AB]$ by $[AC]$ and $[CB]$. Exit if no edge is broken; go to (3) otherwise.

(3) Triangulate every cell that has broken edges due to step 2. Go to (2).

Please note that a four-sided double patch may have some special points where the surface passes through edge p_2p_3. At those points, s, the first component of its ray coordinate, is undefined. For all the points with the same t value coincide at one point.

We observe that, in a simplicial hull, a large portion of edge tetrahedra are thin compared to their neighboring face tetrahedra. If we ray-shoot each double patch separately, the polygonal mesh of an edge tetrahedron could be rather skew and dense compared to that of its neighboring face, which is not desirable for display or further processing of the polygonal representation. To obtain a more uniform polygonal mesh, we instead ray-shoot a group of A-patches collectively, namely, a double face patch and the double edge patches adjacent to it. The algorithm is essentially the same. except

The two algorithms can be speed up dramatically by distributing the computation to different machines or parallelizing into different processors. However, we need to make sure that the boundary curves of two neighboring partition are approximated by the same polyline. This can be achieved by enforcing, on both side of the common boundary, the same deterministic criterion for breaking up edges. We have implemented this distributed scheme in SplineX[BCE93]. With servers running on 9 networked Sun Workstation, it takes less than 2 minutes to polygonize the satellite object(Figure 2.4) containing over 1392 A-patches. Each apatch is polygonized into about 40 triangles,

References

[Baj92] C. Bajaj. Surface fitting with implicit algebraic surface patches. In H. Hagen, editor, *Topics in Surface Modeling*, pages 23 – 52. SIAM Publications, 1992.

[BCE93] C. Bajaj, J. Chen, and S. Evans. Distributed Modeling and Rendering of Splines Using Ganith and SplineX. Technical Report CSD-TR-93-006, CAPO Report CER-93-04, Computer Sciences Department, Purdue University, January 1993.

[BCX94a] C. Bajaj, J. Chen, and G. Xu. *Modeling with Cubic A-Patches. ACM Transactions on Graphics*, To appear:x–y, 1994.

[BCX94b] C. Bajaj, J. Chen, and G. Xu. *Smooth Low Degree Approximations of Polyhedra*. Computer Science Technical Report, CSD-TR-94-002, Purdue University, 1994.

[BI92] C. Bajaj and I. Ihm. C^1 Smoothing of Polyhedra with Implicit Algebraic Splines. *SIGGRAPH'92, Computer Graphics*, 26(2):79–88, 1992.

[BIW93] C. Bajaj, I. Ihm, and J. Warren. Higher order interpolation and least squares approximation using implicit algebraic surfaces. *ACM Trans. on Graphics*, 12(4):327–347, Oct. 1993.

[Bli82] James F. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256, July 1982.

[Blo88] J. Bloomenthal. Polygonization of Implicit Surfaces. Computer Aided Geometric Design, 5(00):341–355, 1988.

[BW90] J. Bloomenthal and B. Wyvill. Interactive techniques for implict modeling. *Proc. of the 1990 Symposium on Interactive 3D Graphics*, pages 109–116, 1990.

[Dah89] W. Dahmen. Smooth piecewise quadratic surfaces. In T. Lyche and L. Schumaker, editors, *Mathematical Methods in Computer Aided Geometric Design*, pages 181–193. Academic Press, Boston, 1989.

[DTS93] W. Dahmen and T-M. Thamm-Schaar. Cubicoids: modeling and visualization. *Computer Aided Geometric Design*, 10:93–108, 1993.

[Guo91] B. Guo. Surface generation using implicit cubics. In N.M. Patrikalakis, editor, *Scientific Visualizaton of Physical Phenomena*, pages 485–530. Springer-Verlag,Tokyo, 1991.

[Guo93] B. Guo. Non-splitting Macro Patches for Implicit Cubic Spline Surfaces. *Computer Graphics Forum*, 12(3):434 – 445, 1993.

[Hal90] Hall, M., and Warren, J. Adaptive Polygonalization of Implicitly Defined Surfaces. *IEEE Computer Graphics and Applications*, pages 33–42, 1990.

[Pra87] V. Pratt. Direct Least Squares Fitting of Algebraic Surfaces. *Computer Graphics*, 21(00):145–152, 1987.

[WH94] A. Witkin and P. Heckbert. Using Particles to Sample and Control Implicit Surfaces. In *SIGGRAPH 94, Orlando, Florida*, pages 269–277, July 24-29 1994.

[WMW86] Brian Wyvill, Craig McPheeters, and Geoff Wyvill. Animating soft objects. *The Visual Computer*, 2(4):235–242, 1986.