Smooth Shell Construction with Mixed Prism Fat Surfaces

Chandrajit L. Bajaj [*]

Department of Computer Science,
University of Texas, Austin, TX 78712

Guoliang Xu [†]

State Key Laboratory of Scientific and Engineering Computing,
ICMSEC, Chinese Academy of Sciences, Beijing

Abstract

Several naturally occurring as well as manufactured objects have shell like structures, that is their boundaries consist of surfaces with thickness. In an earlier paper, we have provided a reconstruction algorithm for such shell structures using smooth fat surfaces within three-sided prisms. In this paper, we extend the approach to a scaffolding consisting of three and four-sided prisms. Within each prism the constructed function is converted to a spline representation. In addition to the adaptive feature of our earlier scheme, the new scheme has the following extensions: (a) four sided fat patches are employed; (b) the size of individual fat patches are bigger; (c) fairing techniques are combined to obtain nicely shaped fat surfaces.

Key words. Shell, Geometric Modeling, Curves and Surfaces, Splines

1 Introduction

Many human manufactured and several naturally occurring objects have shell like structures, that is the object bodies consist of surfaces with thickness. Such surfaces are called *fat surfaces* in [2]. The problem of constructing smooth approximations to fat surface objects arises in creating geometric models such as airfoils, tin cans, shell canisters, engineering castings, sea shells, the earth's outer crust, the human skin, and so forth.

Problem Description. *As input we are given a matched triangulation pair $\mathcal{T} = (\mathcal{T}^{(0)}, \mathcal{T}^{(1)})$ (also called a fat triangulation) with attached normals at each vertex which presents a linearization of the inner and outer boundary surfaces of a shell domain. The goal is to reconstruct smooth fat surface whose bounding surfaces provide approximations of $\mathcal{T}^{(0)}$ and $\mathcal{T}^{(1)}$, respectively. Additionally mid-surfaces between the boundary surfaces are also provided.*

[*]Research supported in part by NSF grants CCR 9732306, KDI-DMS-9873326 and ACI-9982297
[†]Project 19671081 supported by NSFC

The matched pair of surface triangulation with normals could be obtained via several inputs, such as nearby iso-contours of volume data, point clouds, single surfaces (see methods in [2]).

Needless to say, one could solve this geometric modeling problem by classical or existing methods (see, e.g. [7, 8, 9]) of surface splines construction to construct individual boundary surfaces as well as mid-surfaces of the fat boundaries. However, besides the added space complexity of individually modeling the primary bounding surfaces and mid-surfaces, post local and/or global interactive surface modification would require extremely cumbersome surface-surface interference checks to be performed to preserve geometric model consistency.

The implicit method was shown effective for solving such a problem which was proposed in [2], in which the fat surface is defined by the contours of a single trivariate function F. The function is piecewise, and defined on a collection of triangular prisms in $\mathbb{R}^3$, such that it is C^1 and its contour $F(x, y, z) = \alpha$ for any $\alpha \in (-1, 1)$ provides a smooth mid-surface with $F(x, y, z) = -1$ and $F(x, y, z) = 1$ as the inner and outer boundaries of the shell structure. It should be pointed out that the simplicial hull scheme for constructing A-patches on tetrahedra (see [1, 5]) cannot serve to our purpose, since the simplicial hull, over which a trivariate function F is defined, has no thickness at each vertex.

In this paper, we extend the construction of the function F in [2] by incorporating quadrilateral patches, spline functions and fairing techniques, so that the size of several individual fat surface patches is bigger, the number of patches is fewer, and the "shape" of the fat surfaces is better.

2 Algorithm and Notations

This section gives the algorithm outline (see Fig 2.1). Notations used are also introduced here.

2.1 Outline of the algorithm

Step 1. *Decimation.*

This step reduces the number of fat triangles and maintains features. We use the curvature adaptive decimation scheme of [2].

Step 2. *Merge triangles into quadrilaterals.*

Merge certain adjacent triangles into quadrilaterals to further reduce the number of patches. Details of this step appear in Section 3.

Step 3. *Construct C^1 trivariate function approximations.*

Construct a C^1 piecewise trivariate function $F^{(\sigma)}$, over a collection of 3-prisms and 4-prisms defined on the fat triangles and quadrilaterals, so that $S_\alpha^{(\sigma)} = \left\{p : F^{(\sigma)}(p) = \alpha,\ \alpha \in [-1, 1]\right\}$ are smooth surfaces and $S_{-1}^{(\sigma)}$ and $S_1^{(\sigma)}$ are approximation of $\mathcal{T}^{(0)}$ and $\mathcal{T}^{(1)}$, respectively. Here σ is a given integer related to the freedom of the spline function used. This step is detailed in section 4.

Step 4. *Fairing.* Fairing by spline functions. Details are again in section 4.

Step 5 (optional). *Capturing Sharp Features* is detailed in section 5.

Step 6. *Display the fat surface.* Details are given in section 6.

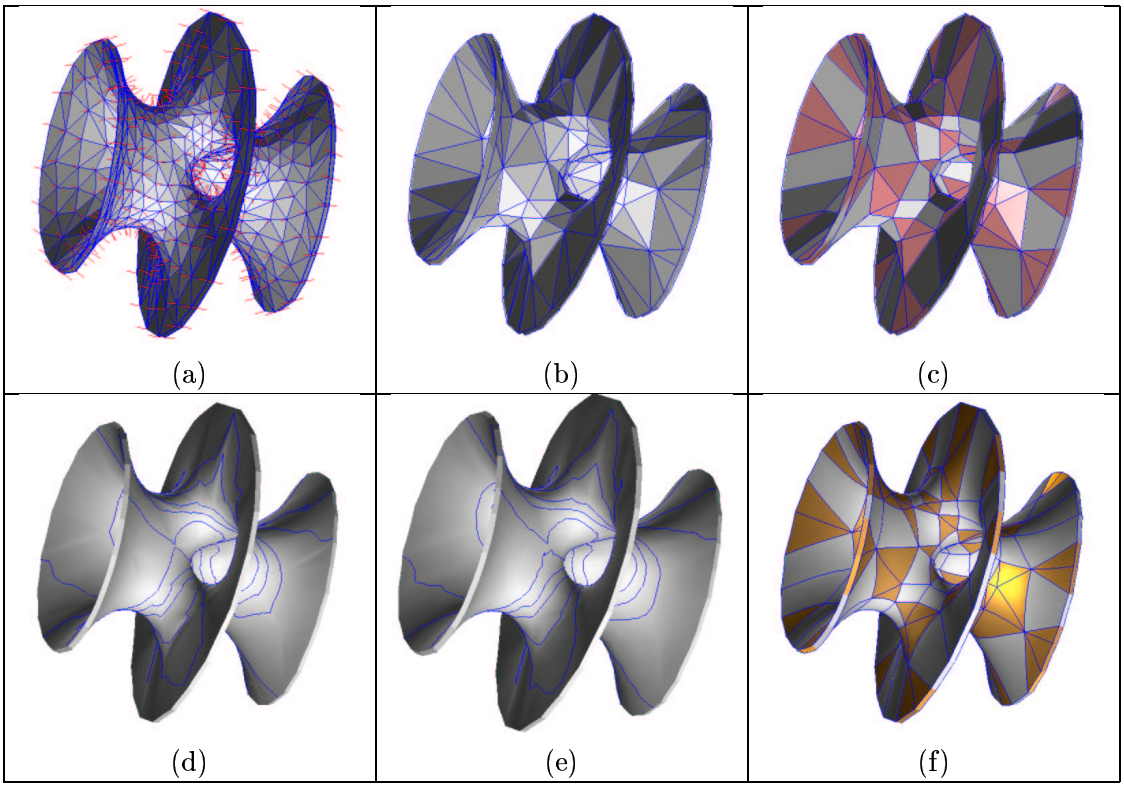

Fig 2.1: The algorithm steps: Figure (a) is the input triangulation pair (917 fat triangles) with normals at vertices. (b) is the decimated result (265 fat triangles). (c) is the output (119 fat triangles and 73 fat quadrilaterals) of the merging step. (d) is a C^1 function construction without using splines. (e) is the fairing result using splines. The curves on the surfaces (d) and (e) are isophote lines. (f) is a display showing the mixed patch nature.

2.2 Notations

Our trivariate function $F^{(\sigma)}$ is piecewise defined on a collection of 3-prisms and 4-prisms. To define these prisms, we denote the i-th fat vertex (vertex pair) as $V_i = (V_i^{(0)}, V_i^{(1)}) \in I\!R^6$. Let $[V_iV_jV_k]$ be a fat triangle. Then the 3-prism D_{ijk} is a volume in $I\!R^3$ enclosed by the surfaces H_{ij}, H_{jk}, and H_{ki} (see Fig 2.2), where H_{lm} is a ruled surface defined by V_l and V_m:

$$H_{lm} = \{p : p = b_1 v_l(\lambda) + b_2 v_m(\lambda), \quad b_1 + b_2 = 1, \quad \lambda \in I\!R\}$$

with $v_i(\lambda) = V_i^{(0)} + \lambda N_i$, $N_i = V_i^{(1)} - V_i^{(0)}$. For any point $p = b_1 v_l(\lambda) + b_2 v_m(\lambda)$ with $b_1 + b_2 = 1$, (b_1, b_2, λ) will be called H_{lm}–coordinate of p. The 3-prism D_{ijk}, for $[V_iV_jV_k]$, is a volume which is represented explicitly as

$$D_{ijk} = \{p : p = b_1 v_i(\lambda) + b_2 v_j(\lambda) + b_3 v_k(\lambda), \quad b_1 + b_2 + b_3 = 1, \quad b_l \geq 0, \quad \lambda \in I\!R\}.$$

We call (b_1, b_2, b_3, λ) as the D_{ijk}–coordinate of p. For each λ, $P_{ijk}(\lambda) := \{p : p = b_1 v_i(\lambda) + b_2 v_j(\lambda) + b_3 v_k(\lambda), \quad b_1 + b_2 + b_3 = 1, b_l \geq 0\}$ defines a triangle. Let G_{ijk} be the point set that is

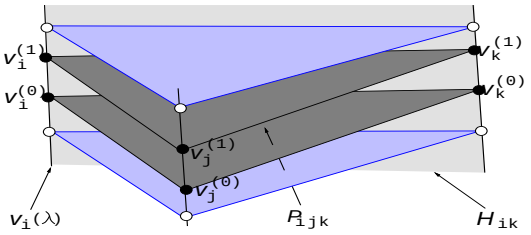

Fig 2.2: The volume prism cell D_{ijk} and a face $H_{jk}(t,\lambda)$ defined by a fat triangle $[V_iV_jV_k]$

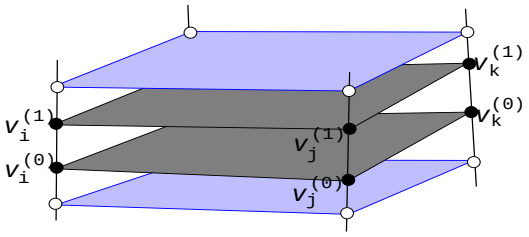

Fig 2.3: The volume prism cell D_{ijkl} defined by a fat quadrilateral $[V_iV_jV_kV_l]$

the grouping of points into prism D_{ijk} in the decimation step.

Let $[V_iV_jV_kV_l]$ be a fat quadrilateral. The 4-prism D_{ijkl} for $[V_iV_jV_kV_l]$ is defined by (sse Fig. 2.3)

$$\begin{aligned} D_{ijkl} = \{p : p = {} & B_{00}(u,v)v_i(\lambda) + B_{10}(u,v)v_j(\lambda) \\ & + B_{01}(u,v)v_l(\lambda) + B_{11}(u,v)v_k(\lambda), \quad u,v \in [0,1], \lambda \in I\!R\}, \end{aligned}$$

where $B_{00} = (1-u)(1-v)$, $B_{10} = u(1-v)$, $B_{01} = (1-u)v$, $B_{11} = uv$. We shall call (u,v,λ) as the D_{ijkl}-coordinate of p. The equation

$$p = B_{00}(u,v)v_i(\lambda) + B_{10}(u,v)v_j(\lambda) + B_{01}(u,v)v_l(\lambda) + B_{11}(u,v)v_k(\lambda) \tag{2.1}$$

defines a transform between (u,v,λ) and (x,y,z).

3 Merging Fat Triangles

Let $[V_iV_jV_k]$ and $[V_jV_kV_l]$ be two adjacent fat triangles of the decimated mesh. They could be merged to form a quadrilateral if the following condition is satisfied:

$$N_s^T[B_{00}(u,v)N_i + B_{01}(u,v)N_j + B_{10}(u,v)N_l + B_{11}N_k] > 0 \quad \text{for} \quad p_s \in G_{ijkl}, \tag{3.1}$$

where $G_{ijkl} = G_{ijk} \cup G_{jkl}$, (u,v,λ) is the D_{ijkl}-coordinate of p_s, N_s is the normal at p_s and the term in the square brackets is the average of the normals at four vertices. Condition (3.1) implies that the angle between N_s and the averaging normal is less than $\pi/2$. We only need to consider the merging of one of $\mathcal{T}^{(0)}$ and $\mathcal{T}^{(1)}$, the other is correspondingly merged.

In [6], M. Eck and H. Hoppe also merge triangles into quadrilaterals where they attempt to pair up all the triangles by a graph matching. Since we allow a hybrid of triangular and rectangular patches (e.g., to keep sharp features (see §5), some of the edges are not removable), and since our implementation and the tests show that the shape of quadrilateral surface patches become bad if the quadrilateral is too narrow, we do not seek to merge all the triangles into quadrilaterals. Instead, we grade each edge by the deviation from a rectangle of the quadrilateral formed by merging the two adjacent triangles. An edge is removed (that is its two adjacent triangles are merged) if condition (3.1) is satisfied and if the grade of this edge is less than a

given threshold value and is less than its four neighbor edge grades. To grade an edge, for each vertex of the quadrilateral that is formed by merging the two adjacent triangles of the edge, compute the absolute value of the difference of the angle formed by the two incident edges and $\pi/2$, then choose the maximal value of the four absolute values, for the four vertices, as the grade of the edge. If a quadrilateral is a rectangle, then its grade is zero. The worst case is where its grade is close to $3\pi/2$, in which the angle at one vertex is close to 2π.

We notice that most of the CAGD models or some parts of the models come from curvilinear partition of objects. The triangulation is then formed by subdividing quadrilaterals, obtained from the curve partition, into triangles. Our triangle merging policy has the property that it recovers the original curve partition in most of the cases. Fig 3.1 shows such an example for a teapot.

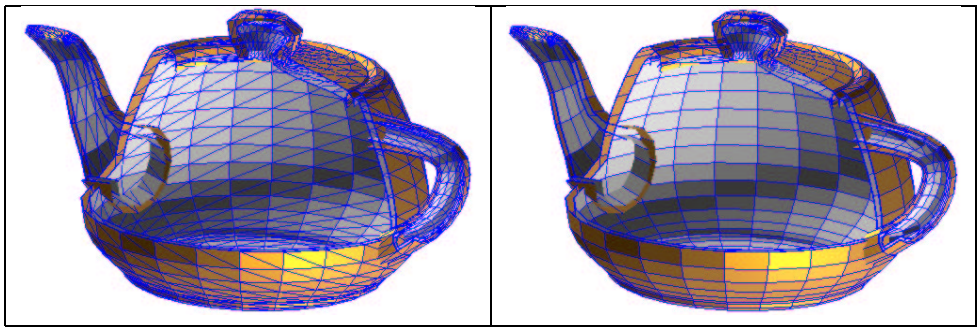

Fig 3.1: Left: the input triangulation pair approximate of a teapot that has 1428 fat triangles. Right: the merging result that has 294 fat triangles and 567 fat quadrilaterals. The threshold value that controls the merging is taken as $\pi/4$.

4 Construct C^1 Trivariate Function Approximations

After step 2, we have a mesh consisting of fat triangles and fat quadrilaterals. For each triangle $[V_iV_jV_k]$ and quadrilateral $[V_iV_jV_kV_l]$ we have volumes D_{ijk} and D_{ijkl} with the grouped point sets G_{ijk} and G_{ijkl}, respectively. In this section, we construct a C^1 trivariate piecewise function $F = F^{(\sigma)}$ ($\sigma \geq 0$ fixed) over the collection of these volumes, so that it is the required approximation. This function is constructed stepwise. First, the function is defined on the edges of the volumes (see §4.2), then on the faces (see §4.3) and finally in the volumes (see §4.4–§4.5).

4.1 Spline Functions

To achieve better approximation and better shape, spline functions defined on triangles and rectangles are utilized in the construction of F. On a triangular domain with a regular partition (see Fig 4.1), C^1 cubic splines defined in BB form were given by Sabin, 1976 (see [10]). Fig 4.2 gives the BB–form coefficients of a typical base function defined on 13 sub-triangles. Note that, these splines in general are not linearly independent (see Bőhm, Farin and Kahmann [4]).

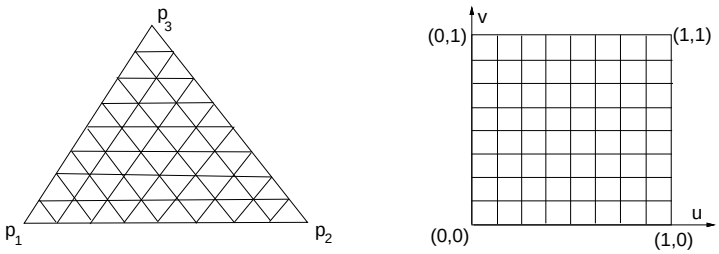

Fig 4.1: Regular partition of triangular and rectangular domains with resolution 2^σ for $\sigma = 3$.

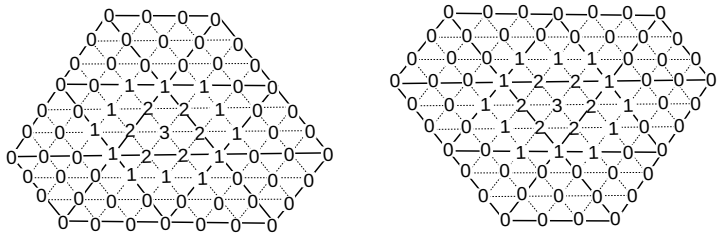

Fig 4.2: Bézier coefficients for two C^1 cubic spline basis functions. Each is defined on the union of 13 sub-triangles, which forms the support of the function.

But the collection we use are indeed linearly independent. For a regular partition of a triangle, say T, we shall associate a base function to each sub-triangle of the partition. To give proper indices for these bases, we label the sub-triangles as T_{ijk} for $(i,j,k) \in J^\sigma = J_1^\sigma \cup J_2^\sigma$, J_1^σ and J_2^σ are defined as follows:

$$J_1^\sigma = \{(i,j,k) : \quad i,j,k \in \{1,2,3,...,2^\sigma\}; \qquad i+j+k = 2^\sigma + 2\},$$

$$J_2^\sigma = \{(i,j,k) : \quad i,j,k \in \{1,2,3,...,2^\sigma - 1\}; \quad i+j+k = 2^\sigma + 1\},$$

where 2^σ is the resolution of the partition. Fig 4.3 gives J_1 and J_2 for $\sigma = 2$. Now we denote the base function defined by Fig 4.2 with center triangle T_{ijk} as N_{ijk}^σ.

Using N_{ijk}^σ, a C^1 cubic spline function on a regularly partitioned triangle is expressed as $\sum_{(i,j,k)\in J^\sigma} b_{ijk} N_{ijk}^\sigma$. On a rectangle, we use tensor-product B-splines $\sum_i \sum_j b_{ij} N_{i3}^\sigma(u) N_{j3}^\sigma(v)$, where $\{N_{i3}^\sigma(t)\}_{i=-1}^{2^\sigma+1}$ are C^2 cubic B-spline bases defined on the uniform knots $t_i = \frac{i}{2^\sigma}$, $i = 0, 1, \cdots, 2^\sigma$. Here we have shifted N_{i3}^σ so that t_i is the center of the support $\mathrm{supp}(N_{i3}^\sigma) = ((i-2)/2^\sigma, (i+2)/2^\sigma)$.

4.2 F and ∇F on the edge of the volume

The function value F and the gradient ∇F on the edge $v_i(\lambda)$ of a volume are defined by

$$F(v_i(\lambda)) = 2\lambda - 1, \quad \nabla F(v_i(\lambda)) = (1-\lambda) N_i^{(0)} + \lambda N_i^{(1)}.$$

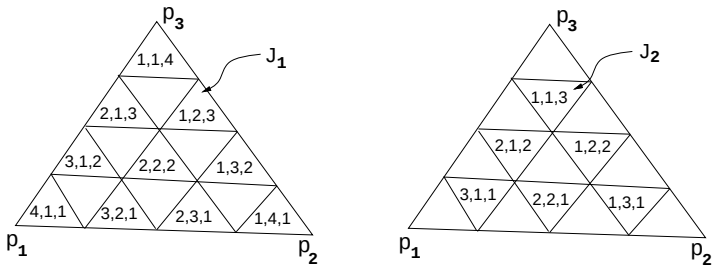

Fig 4.3: For the regular partition of a triangle with resolution 2^σ, the index set J^σ of the sub-triangles are divided into J_1^σ and J_2^σ. This figure shows them for $\sigma = 2$.

We normalize the normals such that $N_i^T N_i^{(0)} = N_i^T N_i^{(1)} = 2$, and so that the definitions of $F(v_i(\lambda))$ and $\nabla F(v_i(\lambda))$ remain consistent. Here N_i^T is the transpose of $N_i = V_i^{(1)} - V_i^{(0)}$. Note $D_{N_i} F = N_i^T \nabla F$ holds on the edge, where $D_{N_i} F$ denotes the directional derivative of F in the direction N_i.

4.3 F and ∇F on the face of the volume

The function and gradient on a face of a volume determine the position and tangent of the constructed surface at the face. Since the C^1 construction of F in the volume requires the function and gradient to be C^2 and C^1 on the face, respectively (see (4.6) and (4.8)), we construct them in the following steps:

a. *Construct a C^1 function and a C^0 gradient by averaging pre-constructed volume functions.*
b. *Using the result of step* **a**, *construct the C^2 function and the C^1 gradient.*

4.3.1 C^1 function and C^0 gradient on the face

Let H_{ij} be the face for the edge $[V_i V_j]$. The C^1 function $\tilde{F}_{ij}$ and the C^0 gradient $\nabla \tilde{F}_{ij}$ on H_{ij} are defined by averaging the C^1 functions and C^0 gradients on the two adjacent volumes. Hence the tasks of this sub-section are to construct the volume functions and then do averaging. The volume function constructed here is not our final result of F, since they are not smoothly and even not continuously join at the boundaries of the volumes. However, their average on the common face (regarded as a 2D function) are C^1 and C^0, respectively.

Let $[V_i V_j V_k]$ and $[V_i V_j V_l]$ be the two neighbor fat triangles of the edge $[V_i V_j]$. The case of one or two neighbors are quadrilaterals is similar. On the volume D_{ijk}, we construct a function of the following form

$$F_{ijk} = B_{ijk} + S^\sigma_{ijk}$$

where B_{ijk} is of cubic BB-form and S^σ_{ijk} is of spline form:

$$B_{ijk}(b_1, b_2, b_3, \lambda) = \sum_{i_1+i_2+i_3=3} b_{i_1 i_2 i_3}(\lambda) B^3_{i_1 i_2 i_3}(b_1, b_2, b_3),$$

$$S^{\sigma}_{ijk}(b_1,b_2,b_3,\lambda) = \sum_{(i_1,i_2,i_3)\in \tilde{J}^{\sigma}} (a_{i_1i_2i_3} + w_{i_1i_2i_3}\lambda) N^{\sigma}_{i_1i_2i_3}(b_1,b_2,b_3),$$

where $\tilde{J}^{\sigma} = J^{\sigma} - \{(2^{\sigma},1,1),(1,2^{\sigma},1),(1,1,2^{\sigma})\}$ and $B^3_{i_1i_2i_3}(b_1,b_2,b_3) = \frac{3!}{i_1!i_2!i_3!} b_1^{i_1} b_2^{i_2} b_3^{i_3}$. The BB-form part B_{ijk} is used to interpolate function values and gradients on the three edges of the volume. The spline part S^{σ}_{ijk} is a modification of B_{ijk} for achieving a better approximation in the volume.

The coefficients of B_{ijk} are defined by interpolating the data on the edge of the volume:

$$b_{300}(\lambda) = F(v_i(\lambda)), \quad b_{030}(\lambda) = F(v_j(\lambda)), \quad b_{003}(\lambda) = F(v_k(\lambda)),$$
$$b_{210}(\lambda) = F(v_i(\lambda)) + \frac{1}{3}[v_j(\lambda) - v_i(\lambda)]^T \nabla F(v_i(\lambda)),$$

b_{201}, b_{120}, b_{021}, b_{102} and b_{012} are similarly defined. Also, b_{111} is defined by making the cubic B_{ijk} approximate a quadratic: $b_{111} = \frac{1}{4}(b_{210}+b_{120}+b_{021}+b_{012}+b_{102}+b_{201}) - \frac{1}{6}(b_{300}+b_{030}+b_{003})$.

S^{σ}_{ijk} is determined by fitting the points inside the volume D_{ijk} and fairing. Let $\{q_1^{(0)},\cdots,q_{\mu_0}^{(0)}\} \subset \mathcal{T}^{(0)} \cap G_{ijk}$ be the vertex list. Similarly, let $\{q_1^{(1)},\cdots,q_{\mu_1}^{(1)}\} \subset \mathcal{T}^{(1)} \cap G_{ijk}$. We compute the coefficients of the splines by the following equations:

$$\begin{cases} w_0\left[F_{ijk}(b_{1s}^{(t)},b_{2s}^{(t)},b_{3s}^{(t)},\lambda_s^{(t)}) - (-1)^{t+1}\right] = 0, \quad s=1,2,\cdots,\mu_t, \quad t=0,1, \\ w_1 n_{si}^T \dfrac{\partial S(v_{si})}{\partial b_1} = 0, \quad s=0,1,2; \quad i=1,\cdots,2^{\sigma}-1, \\ w_1 n_{si}^T \dfrac{\partial S(v_{si})}{\partial b_2} = 0, \quad s=0,1,2; \quad i=1,\cdots,2^{\sigma}-1, \\ \int\int \left(\dfrac{\partial^2 \tilde{S}}{\partial x^2}\right)^2 + 2\mu \dfrac{\partial^2 \tilde{S}}{\partial x^2}\dfrac{\partial^2 \tilde{S}}{\partial y^2} + \left(\dfrac{\partial^2 \tilde{S}}{\partial y^2}\right)^2 + 2(1-\mu)\left(\dfrac{\partial^2 \tilde{S}}{\partial x \partial y}\right)^2 = min. \end{cases} \tag{4.1}$$

where $(b_{1s}^{(t)},b_{2s}^{(t)},b_{3s}^{(t)},\lambda_s^{(t)})$ are the D_{ijk}–coordinates of $q_s^{(t)}$, n_{si}, $s=0,1,2; i=1,\cdots,2^{\sigma}-1$, are the given normals on the three boundaries of the mid-surface. These normals are computed by averaging the mid-surface normals defined by $B_{ijk}=0$. v_{si} are points on boundaries of mid-surface. $S(b_1,b_2)$ is the mid-surface defined by $F_{ijk}=0$. $\tilde{S}(x,y) = S(b_1(x,y),b_2(x,y))$ with (x,y) to be a local Descartes coordinate. We choose $\frac{1}{2}(V_k^{(0)}+V_k^{(1)})$ as the origin of this system, $\frac{1}{2}[(V_i^{(0)}+V_i^{(1)}) - (V_k^{(0)}+V_k^{(1)})]$ as the x-direction. The y-direction is chosen to be perpendicular to x-direction and point to the side on which $\frac{1}{2}(V_j^{(0)}+V_j^{(1)})$ lies. Note that we do not use a (b_1,b_2) coordinate system directly, because the energy defined in this system is not rotation invariant. System (4.1) is solved in the least square sense. The first set of equations forces the surface interpolating the points in the volume. The second and third sets of equations force the mid-surface to have the given normals on the boundaries. The last minimization forces the surface to have minimal strain energy. Also w_0 and w_1 are weights balancing the three sets of constrains. The minimization leads to a nonlinear system of equations. The integrations in the system are computed by a 6-point numerical quadrature rule (see [3], page 35) on each sub-triangle. We solve the entire system by Newton iteration. Since the system behaves linearly, it converges fast. It needs, in general, 2 or 3 iterations to achieve a single word-length precision.

After F_{ijk} and F_{ijl} have been defined, then we are ready to define

$$\tilde{F}_{ij} = \frac{1}{2}\left(F_{ijk}|_{H_{ij}} + F_{ijl}|_{H_{ij}}\right), \quad \nabla\tilde{F}_{ij} = \frac{1}{2}\left(\nabla F_{ijk}|_{H_{ij}} + \nabla F_{ijl}|_{H_{ij}}\right).$$

If a four-sided polygon, say $[V_iV_jV_kV_l]$, is a neighbor of $[V_iV_j]$, then we define

$$F_{ijkl} = B_{ijkl} + S^{\sigma}_{ijkl}$$

with

$$B_{ijkl}(u,v,\lambda) = \sum_{i_1=0}^{3}\sum_{i_2=0}^{3} b_{i_1i_2}(\lambda)B^3_{i_1}(u)B^3_{i_2}(v),$$
$$S^{\sigma}_{ijkl}(u,v,\lambda) = \sum_{i_1=0}^{2^\sigma}\sum_{i_2=0}^{2^\sigma} (a_{i_1i_2} + w_{i_1i_2}\lambda)N^{\sigma}_{i_1 3}(u)N^{\sigma}_{i_2 3}(v).$$

The coefficients $B_{ijkl}(u,v,\lambda)$ are determined as follows:

$$b_{00} = F(v_i(\lambda)), \quad b_{30} = F(v_j(\lambda)), \quad b_{33} = F(v_k(\lambda)), \quad b_{03} = F(v_l(\lambda)),$$
$$b_{10} = v_i(\lambda) + \frac{1}{3}(v_j(\lambda) - v_i(\lambda))^T\nabla F(v_i(\lambda)).$$

The other coefficients on the edge are similarly defined. Define

$$b_{11} = \tfrac{1}{3}(b_{10}+b_{01}) + \tfrac{1}{6}(b_{31}+b_{13}), \quad b_{22} = \tfrac{1}{3}(b_{32}+b_{23}) + \tfrac{1}{6}(b_{20}+b_{02}),$$
$$b_{21} = \tfrac{1}{3}(b_{20}+b_{31}) + \tfrac{1}{6}(b_{23}+b_{01}), \quad b_{12} = \tfrac{1}{3}(b_{02}+b_{13}) + \tfrac{1}{6}(b_{32}+b_{10}).$$

The coefficients of S^{σ}_{ijkl} are determined as that of S^{σ}_{ijk} by fitting the data inside the volume D_{ijkl} and fairing.

4.3.2 C^2 function and C^1 gradient on the face

Now let us define the C^2 function F_{lm} and C^1 gradient ∇F_{lm}. Let

$$\begin{aligned} F_{lm}(t,\lambda) = {} & F(v_l(\lambda))H^3_0(t) + [v_m(\lambda) - v_l(\lambda)]^T\nabla F(v_l(\lambda))H^3_1(t) \\ & + F(v_m(\lambda))H^3_2(t) + [v_m(\lambda) - v_l(\lambda)]^T\nabla F(v_m(\lambda))H^3_3(t) \\ & + \phi^{\sigma}_{lm}(t) + \psi^{\sigma}_{lm}(t)\lambda, \end{aligned} \tag{4.2}$$

with

$$H^3_0(t) = 1 - 3t^2 + 2t^3, \qquad H^3_1(t) = t - 2t^2 + t^3,$$
$$H^3_2(t) = 3t^2 - 2t^3, \qquad H^3_3(t) = -t^2 + t^3,$$
$$\phi^{\sigma}_{lm}(t) = \textstyle\sum_{i=2}^{2^\sigma-2}\phi_i N^{\sigma}_{i3}(t), \quad \psi^{\sigma}_{lm}(t) = \sum_{i=2}^{2^\sigma-2}\psi_i N^{\sigma}_{i3}(t).$$

From the construction of $\tilde{F}_{lm}$, we know that it has the same form as F_{lm} defined by (4.2) but with different ϕ^{σ}_{lm} and ψ^{σ}_{lm} that are C^1 cubic splines. We denote them as $\tilde{\phi}^{\sigma}_{lm}$ and $\tilde{\psi}^{\sigma}_{lm}$. Now we determine ϕ^{σ}_{lm} and ψ^{σ}_{lm} by approximating $\tilde{\phi}^{\sigma}_{lm}$ and $\tilde{\psi}^{\sigma}_{lm}$ in the least square sense:

$$\int_0^1 \left[\phi^{\sigma}_{lm}(t) - \tilde{\phi}^{\sigma}_{lm}(t)\right]^2 dt = min, \quad \int_0^1 \left[\psi^{\sigma}_{lm}(t) - \tilde{\psi}^{\sigma}_{lm}(t)\right]^2 dt = min. \tag{4.3}$$

Each of them leads to a system of linear equations. The integrations in these systems are computed by Gauss-Legendre quadrature rule on each sub-interval and then summed up. Let

$$d_1(\lambda) = v_m(\lambda) - v_l(\lambda), \quad d_2(t) = (1-t)N_l + tN_m, \quad d_3(t,\lambda) = d_1 \times d_2.$$

Then we define $\nabla F_{lm}(t,\lambda)$ by the following conditions:

$$\begin{cases} d_1^T \nabla F_{lm}(t,\lambda) = \dfrac{\partial F_{lm}(t,\lambda)}{\partial t}, \\ d_2^T \nabla F_{lm}(t,\lambda) = \dfrac{\partial F_{lm}(t,\lambda)}{\partial \lambda}, \\ d_3^T \nabla F_{lm}(t,\lambda) = d_3^T \breve{\nabla F}_{lm}(t,\lambda), \end{cases} \tag{4.4}$$

where $\breve{\nabla F}_{lm}(t,\lambda)$ is a C^1 approximation of $\tilde{\nabla F}_{lm}(t,\lambda)$:

$$\breve{\nabla F}_{lm}(t,\lambda) = (1-t)\nabla F(v_l(\lambda)) + t\nabla F(v_m(\lambda)) + \breve{\phi}^\sigma_{lm}(t) + \breve{\psi}^\sigma_{lm}(t)\lambda,$$

with $\breve{\phi}^\sigma_{lm}(t) = \sum_{i=2}^{2^\sigma-2} \phi_i N^\sigma_{i3}(t)$, $\breve{\psi}^\sigma_{lm}(t) = \sum_{i=2}^{2^\sigma-2} \psi_i N^\sigma_{i3}(t)$. $\breve{\phi}^\sigma_{lm}$ and $\breve{\psi}^\sigma_{lm}$ are determined by

$$\int_0^1 \|\breve{\nabla F}_{lm}(t,\lambda_i) - \tilde{\nabla F}_{lm}(t,\lambda_i)\|^2 dt = min, \quad \text{for } \lambda_0 = 0,\ \lambda_1 = 1.$$

It might be necessary to point out now that $\breve{\nabla F}_{lm}$ cannot be used as ∇F_{lm} even though it is C^1, since it may not satisfy the first two conditions of (4.4). It is clear that, these two conditions must be satisfied because F_{lm} is previously defined. Though the right-handed side of the third equation of (4.4), which is a directional derivative, can be any value, it is reasonable to choose this value by approximating the existing information about ∇F_{lm}. Hence we use $\breve{\nabla F}_{lm}$ to compute this directional derivative.

Since $\|d_3\|^2[d_1,d_2,d_3]^{-1} = \left[d_1\|d_2\|^2 - d_2(d_1^T d_2), d_2\|d_1\|^2 - d_1(d_1^T d_2), d_3\right]^T$, (4.4) implies

$$\nabla F_{lm}(t,\lambda) = \frac{1}{\|d_3\|^2}\left\{\left[d_1\|d_2\|^2 - d_2(d_1^T d_2)\right]P + \left[d_2\|d_1\|^2 - d_1(d_1^T d_2)\right]Q + d_3 R\right\}, \tag{4.5}$$

where $P(t,\lambda) = \frac{\partial F_{lm}(t,\lambda)}{\partial t}$, $Q(t,\lambda) = \frac{\partial F_{lm}(t,\lambda)}{\partial \lambda}$, $R(t,\lambda) = d_3^T \breve{\nabla F}_{lm}(t,\lambda)$.

4.4 F on the volume D_{ijkl}

Now we are ready to define F within the volumes. Let $[V_1V_2V_3V_4]$ be a typical fat quadrilateral. Let F_u and F_v be defined by the cubic Hermite interpolation in the u and v directions, respectively:

$$\begin{aligned} F_u(u,v,\lambda) &= H_0^3(u)F_{14}(v,\lambda) + H_1^3(u)d_u(v,\lambda)^T\nabla F_{14}(v,\lambda) \\ &+ H_2^3(u)F_{23}(v,\lambda) + H_3^3(u)d_u(v,\lambda)^T\nabla F_{23}(v,\lambda), \\ F_v(u,v,\lambda) &= H_0^3(v)F_{12}(u,\lambda) + H_1^3(v)d_v(u,\lambda)^T\nabla F_{12}(u,\lambda) \\ &+ H_2^3(u)F_{43}(u,\lambda) + H_3^3(v)d_v(u,\lambda)^T\nabla F_{43}(u,\lambda), \end{aligned}$$

where $d_u(v,\lambda) = H_{23}(v,\lambda) - H_{14}(v,\lambda)$, $d_v(u,\lambda) = H_{43}(u,\lambda) - H_{12}(u,\lambda)$. Then we define

$$F^{(\sigma)}(u,v,\lambda) = \frac{w_u F_u(u,v,\lambda) + w_v F_v(u,v,\lambda)}{w_u + w_v} + R^{\sigma}(u,v,\lambda) \tag{4.6}$$

with

$$R^{\sigma}(u,v,\lambda) = \sum_{i_1=2}^{2^{\sigma}-2} \sum_{i_2=2}^{2^{\sigma}-2} (a_{i_1 i_2} + w_{i_1 i_2}\lambda) N^{\sigma}_{i_1 3}(u) N^{\sigma}_{i_2 3}(v),$$

where $w_u = [(1-v)v]^2$, $w_v = [(1-u)u]^2$. The last term $R^{\sigma}(u,v,\lambda)$ in (4.6) is referred as *correction term*, which is used to fit the data in the volume and it does not change the surface on the face of the volume. Let $\{V_s^{(\tau)}\} \subset G_{1234} \cap \mathcal{T}^{(\tau)} (\tau = 0 \text{ or } 1)$, and $(u_s^{(\tau)}, v_s^{(\tau)}, \lambda_s^{(\tau)})$ be the D_{1234}–coordinate of $V_s^{(\tau)}$. Thus $a_{i_1 i_2}$ and $w_{i_1 i_2}$ are defined by

$$\begin{cases} w\left[F^{(\sigma)}(u_s^{(\tau)}, v_s^{(\tau)}, \lambda_s^{(\tau)}) - (-1)^{\tau+1}\right] = 0, \\ \int\int \left(\dfrac{\partial^2 S}{\partial u^2}\right)^2 + 2\mu \dfrac{\partial^2 S}{\partial u^2}\dfrac{\partial^2 S}{\partial v^2} + \left(\dfrac{\partial^2 S}{\partial v^2}\right)^2 + 2(1-\mu)\left(\dfrac{\partial^2 S}{\partial u \partial v}\right)^2 = min, \end{cases} \tag{4.7}$$

where $S(u,v)$ is the mid-surface defined by $F^{(\sigma)}(u,v,\lambda) = 0$. The first equality is in the least square sense. The weight w is put to address the importance of interpolating the points in the volume.

4.5 F on Volume D_{ijk}

Let $[V_1 V_2 V_3]$ be a typical fat triangle. we Define

$$F^{(\sigma)}(b_1,b_2,b_3,\lambda) = \sum_{i=1}^{3} w_i D_i(b_1,b_2,b_3,\lambda) + T^{\sigma}(b_1,b_2,b_3,\lambda) \tag{4.8}$$

with

$$T^{\sigma}(b_1,b_2,b_3,\lambda) = \sum_{(i_1,i_2,i_3)\in J_3^{\sigma}} (a_{i_1 i_2 i_3} + w_{i_1 i_2 i_3}\lambda) N^{\sigma}_{i_1 i_2 i_3}(b_1,b_2,b_3)$$

where $w_i = \dfrac{\prod_{j\neq i} b_j^2}{\sum_{k=1}^{3}\prod_{j\neq k} b_j^2}$, $J_3^{\sigma} = \{(i,j,k) \in J^{\sigma} : i > 1, j > 1, k > 1\}$,

$$\begin{aligned} D_i(b_1,b_2,b_3,\lambda) &= F(v_i(\lambda)) H_2^3(b_i) + F(p_i(b_1,b_2,b_3,\lambda)) H_0^3(b_i) \\ &\quad + d_i(b_1,b_2,b_3,\lambda)^T \nabla F(v_i(\lambda)) H_3^3(b_i) \\ &\quad + d_i(b_1,b_2,b_3,\lambda)^T \nabla F(p_i(b_1,b_2,b_3,\lambda)) H_1^3(b_i), \\ p_i(b_1,b_2,b_3,\lambda) &:= \frac{b_j}{1-b_i} v_j(\lambda) + \frac{b_k}{1-b_i} v_k(\lambda), \\ d_i(b_1,b_2,b_3,\lambda) &:= -\frac{b_j}{1-b_i} e_{ji}(\lambda) - \frac{b_k}{1-b_i} e_{ki}(\lambda), \\ e_{ji}(\lambda) &= v_j(\lambda) - v_i(\lambda),\ e_{ki}(\lambda) = v_k(\lambda) - v_i(\lambda), \end{aligned}$$

and $(i,j,k) \in \{(1,2,3),(2,3,1),(3,1,2)\}$. Again, the last term in (4.8) is called the *correction term*. The parameters $a_{i_1 i_2 i_3}$ and $w_{i_1 i_2 i_3}$ are similarly defined as $a_{i_1 i_2}$ and $w_{i_1 i_2}$ in (4.6) by fitting and fairing.

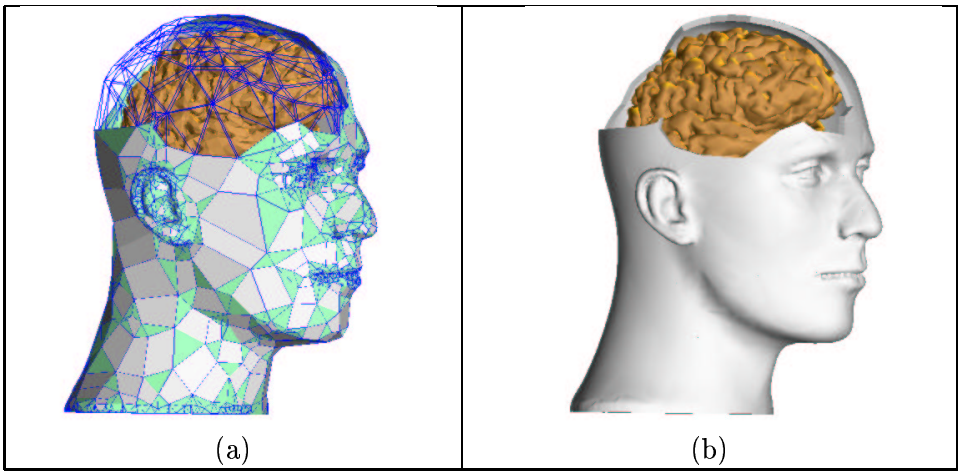

Fig 4.4: Smooth fat surface construction: Figure (b) is the smoothing of Figure (a). Polygon (a), that has 3296 fat triangles and 389 fat quadrilaterals, is the decimated and merged result of a mesh that has fat 25552 triangles. Note the adaptive nature: More fat triangles are used at the parts of the ears, eyes and mouth. To capture sharp features, fat triangles are not merged at the parts of the neck, eyes and mouth. The brain model consists of 40884 fat triangles.

4.6 Basic Results

Theorem 4.1 *The function $F^{(\sigma)}$ constructed is C^1 on $\sum D_{ijk} \cup \sum D_{ijkl}$.*

Proof. First note that the function F^σ is C^1 within each of the volumes, since the gradient on the faces of the volumes is C^1 and the correction terms is C^1 in the volume. Second note that the function values and gradients of the correction term R^σ in (4.6) and the term T^σ in (4.8) vanish on the boundary of the corresponding volume. Hence these terms do not influence the continuity of the function F^σ. On each edge of the volumes, the C^1 continuity of F^σ can be similarly proved as Theorem 4.1 of [2]. Hence, a fact, that needs to be proved, is that the function values and gradients of F^σ on the boundary of the volumes coincide with function values and gradients defined in section 4.3. This fact will guarantee that the function is C^1 on the boundary faces. For the 3-prisms, this fact could be proved similar to the proof of Theorem 4.1 of [2]. Hence the remaining is to prove the fact for the 4-prisms. Consider the function value and gradient of F^σ on the edge $u = 0$ for a typical fat quadrilateral $[V_1V_2V_3V_4]$. It follows from (4.6) that

$$F^\sigma(0,v,\lambda) = F_u(0,v,\lambda) = F_{14}(v,\lambda).$$

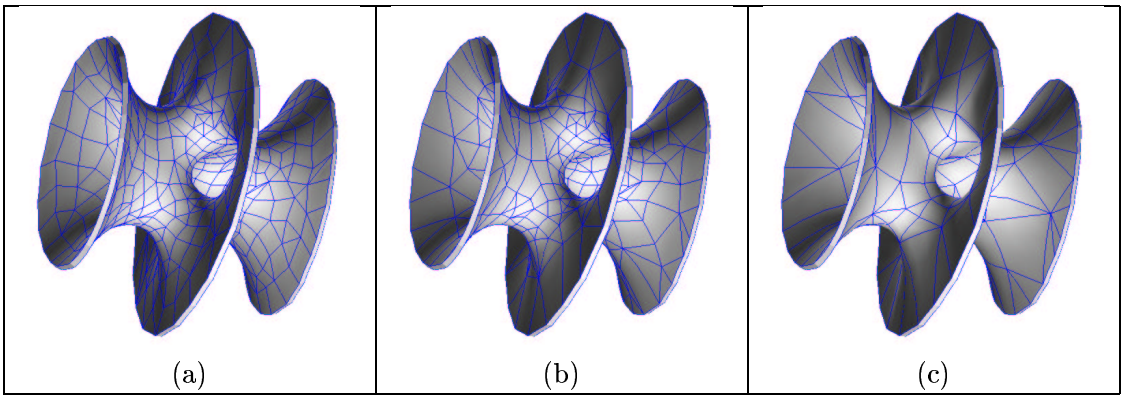

Fig 4.5: Different resolution constructions of smooth fat surfaces. Three mesh levels (h direction) with fixed $\sigma = 3$ are shown. From the left to the right, they have 249, 213 and 95 fat triangles and have 334, 206 and 64 fat quadrilaterals, respectively.

Hence the function value is what we require. Computing partial derivatives of F^σ and using the relation (4.4), we have

$$\begin{aligned}
\frac{\partial F^\sigma}{\partial u} &= d_u(v,\lambda)^T \nabla F_{14}(v,\lambda), \\
\frac{\partial F^\sigma}{\partial v} &= \frac{\partial F_{14}}{\partial v} = d_1(\lambda)^T \nabla F_{14}(v,\lambda), \\
\frac{\partial F^\sigma}{\partial \lambda} &= \frac{\partial F_{14}}{\partial \lambda} = d_2(v)^T \nabla F_{14}(v,\lambda).
\end{aligned}$$

Differentiating (2.1) about x, y and z, we have

$$\begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial \lambda}{\partial x} \\ \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial \lambda}{\partial y} \\ \frac{\partial u}{\partial z} & \frac{\partial v}{\partial z} & \frac{\partial \lambda}{\partial z} \end{bmatrix} = \left\{ [d_u(v,\lambda), d_1(\lambda), d_2(v)]^T \right\}^{-1}.$$

Computing partial derivatives of F^σ with respect to x, y and z and combining the two sets of equations above, we obtain $\nabla F^\sigma = \nabla F_{14}(v,\lambda)$. Hence, F^σ is C^1. $\diamond$

To show the smoothness of function F^σ, Fig 4.4 shows a construction example. Figure (b) is the fat surface construction of the input in figure (a). Fig 4.5 gives a multiresolution construction of the hypersheet surface in the mesh direction (h direction). Fig. 2.1 (d) and (e) show two resolutions in the spline levels (p direction) with $\sigma = 1, 3$, respectively.

5 Sharp Features of the Constructed Surfaces

To capture sharp features, we need to mark certain edges as sharp. To this end, we compute dihedral angle $\theta = \pi - \theta_1$ for the two incident faces, for each edge of the triangulation $T^{(0)}$ and $T^{(1)}$. If $\theta < \alpha$, then this edge is marked as a sharp edge. Here θ_1 is the angle between the two

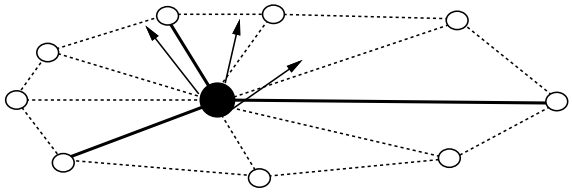

Fig 5.1: Grouping the triangles by the sharp edges (thick lines) and assigning one normal for each group.

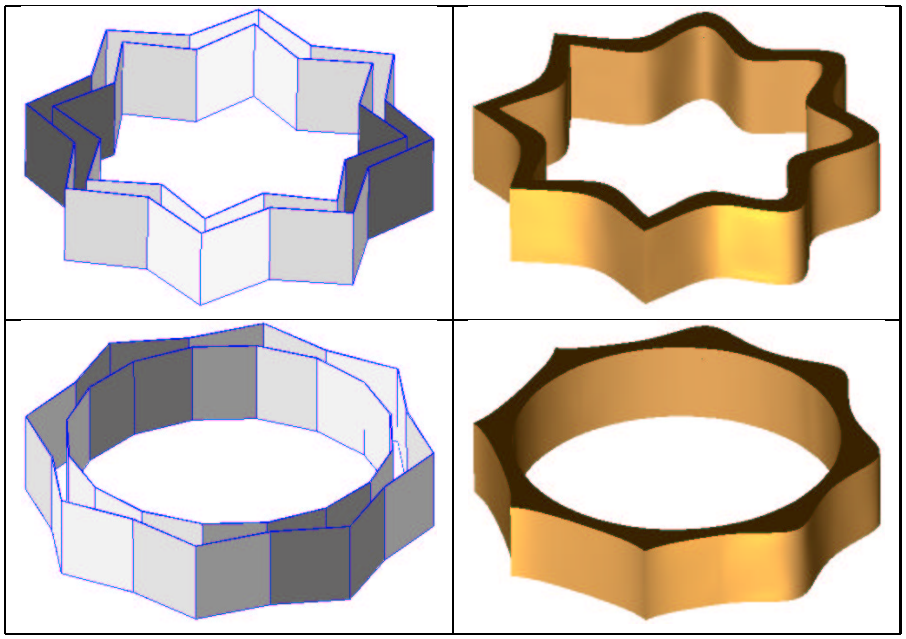

Fig 5.2: Left: the input polygons with some edges are marked as sharp. Right: the constructed fat surfaces with sharp features. There are four fat edgs (inner and outer) on the top polygon are marked as sharp. On the bottom polygon, only four outer edgs are marked as sharp.

normals of the two triangles and α is a threshold value for controlling the sharp feature. After marking the edges, the vertices also need to be marked. If there exist sharp edges incident to a vertex, then we say this vertex is sharp, otherwise, it is non-sharp. For a sharp vertex, the normal that has been assigned before needs to be re-computed. Now the triangles around a sharp vertex are divided into some groups by the sharp edges (see Figure 5.1). For each group, we assign a single normal for the vertex. This normal could be computed by the weighted average of face normals. The weight is chosen to be the angle of the edges that are incident to this vertex. In the construction of surface patch for one triangle, there is only one normal is used for one vertex of the triangle. This normal is the vertex normal if the vertex is non-sharp, otherwise the normal is the group's normal.

Two examples are shown in Fig 5.2. The left two are input polygons, the right shell bodies are the corresponding output. In the star-like polygon on the top-left, the left four inner and

outer peak edges are selectively marked as sharp. The fat surface on the top-right exhibit the sharp feature. For the bottom-left polygon, the left four peak edges of the outer polygon are marked as sharp, no edge is marked for inner polygon. The figure on the bottom-right presents the outer-sharp, inner-smooth nature. Another example that has sharp feature is shown in Fig 5.3.

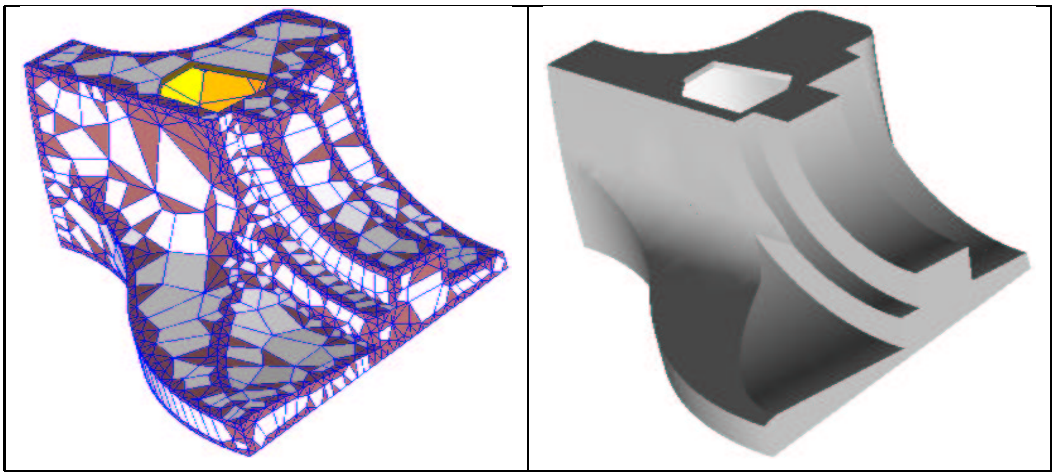

Fig 5.3: Left: the input polygon, that has 1914 fat triangles and 836 fat quadrilaterals, with some edges are marked as sharp. Right: the constructed fat surface with sharp features. To show the fat nature, the shell that is closed is cutaway on the top to show the interior.

6 Display of the Fat Surfaces

Often we wish to evaluate the surface $F = \alpha$ for a given $\alpha \in [-1, 1]$. Let $[V_i V_j V_k]$ be any fat triangle. Then for each (b_1, b_2, b_3), $b_i \geq 0$, $\sum b_i = 1$, determine $\lambda_{min}^{(\alpha)} = \lambda_{min}^{(\alpha)}(b_1, b_2, b_3)$ such that

$$\begin{aligned} &F_{ijk}(b_1, b_2, b_3, \lambda_{min}^{(\alpha)}) = \alpha, \\ &\left|\lambda_{min}^{(\alpha)} - \tfrac{1}{2}\right| = \min\left\{\left|\lambda - \tfrac{1}{2}\right| : \; F_{ijk}(b_1, b_2, b_3, \lambda) = \alpha\right\}. \end{aligned}$$

The surface point is defined by $p = p(b_1, b_2, b_3, \lambda_{min}^{(\alpha)})$. The main task here is to compute $\lambda_{min}^{(\alpha)}$ for each (b_1, b_2, b_3). It follows from (4.8) that $D_i(b_1, b_2, b_3, \lambda)$ is a rational function of λ. It is of the form

$$F_0 + F_1\lambda + F_2\lambda^2 + \frac{N_0 + N_1\lambda + N_2\lambda^2 + N_3\lambda^3 + N_4\lambda^4}{D_0 + D_1\lambda + D_2\lambda^2}. \tag{6.1}$$

Hence $\phi(\lambda) := F_{ijk}(b_1, b_2, b_3, \lambda)$ is a rational function of λ. The nearest zero to $\frac{1}{2}$ of $\phi(\lambda) - \alpha$ is the required $\lambda_{min}^{(\alpha)}$.

Although $\phi(\lambda) - \alpha = 0$ is a nonlinear algebraic equation, $\phi(\lambda) - \alpha$ can be approximated by a polynomial of degree at most 2, since the rational term in (6.1) is small compared with the

polynomial part. Hence, taking the root of the polynomial part as an initial value, and then using Newton iteration, we obtain the required solution.

For the four sided polygon $[V_iV_jV_kV_l]$, the surface point $F_{ijkl}(u, v, \lambda) = \alpha$ is evaluated similarly.

7 Conclusions

Using Bézier, triangular form and tensor form trivariate spline functions, we construct a C^1 function F^σ on a collection of 3-prisms and 4-prisms, such that the contours $F^\sigma = -1$ and $F^\sigma = 1$ approximate the given input triangulation pair, which represent the inner and outer boundaries of a shell body. Apart from fitting the data clouds, the spline functions also serve to fair the shape of the constructed surface. The implementation and test examples show that the proposed method for fat surface construction is correct and fulfills our initial goals.

References

[1] C. Bajaj, J. Chen, and G. Xu. Modeling with Cubic A-Patches. *ACM Transactions on Graphics*, 14(2):103–133, 1995.

[2] C. Bajaj and G. Xu. *Smooth Adaptive Reconstruction and Deformation of Free-Form Fat Surfaces.* TICAM REPORT 99-08, March, 1999, Texas Institute for Computational and Applied Mathematics, The University of Texas at Austin, 1999.

[3] M. Bernadou and J. M. Boisserie. *The Finite Element Method in Thin Shell Theory: Application to Arch Dam Simulatoins.* Birkhäuser, 1982.

[4] W. Bőhm, G. Farin, and J. Kahmann. A survey of curve and surface methods in CAGD. *Computer Aided Geometric Design*, 1:1–60, 1984.

[5] W. Dahmen and T-M. Thamm-Schaar. Cubicoids: modeling and visualization. *Computer Aided Geometric Design*, 10:89–108, 1993.

[6] M. Eck and H. Hoppe. *Automatic Reconstruction of B-Spline Surfaces of Arbitrary Topological Type.* In *Computer Graphics Proceedings, Annual Conference series, ACM SIGGRAPH96*, pages 325–334, 1996.

[7] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide, Second Edition.* Academic Press Inc., 1990.

[8] J. Hoschek and D. Lasser. *Fundamentals of Computer Aided Geometric Design.* A. K. Peters, 1993.

[9] L. Piegl and W. Tiller. *The NURBS Book.* Springer, 1997.

[10] M. Sabin. *The use of piecewise form of numerical representation of shape.* PhD thesis, Hungarian Academy of Science, Budapest, 1976.