Eurographics Symposium on Geometry Processing 2008
Pierre Alliez and Szymon Rusinkiewicz
(Guest Editors)

Volume 27 (*2008*), *Number 5*

Fitting Sharp Features with Loop Subdivision Surfaces

Ruotian Ling, Wenping Wang and Dongming Yan

Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong, China
rtling@cs.hku.hk, wenping@cs.hku.hk, dmyan@cs.hku.hk

Abstract

Various methods have been proposed for fitting subdivision surfaces to different forms of shape data (e.g., dense meshes or point clouds), but none of these methods effectively deals with shapes with sharp features, that is, creases, darts and corners. We present an effective method for fitting a Loop subdivision surface to a dense triangle mesh with sharp features. Our contribution is a new exact evaluation scheme for the Loop subdivision with all types of sharp features, which enables us to compute a fitting Loop subdivision surface for shapes with sharp features in an optimization framework. With an initial control mesh obtained from simplifying the input dense mesh using QEM, our fitting algorithm employs an iterative method to solve a nonlinear least squares problem based on the squared distances from the input mesh vertices to the fitting subdivision surface. This optimization framework depends critically on the ability to express these distances as quadratic functions of control mesh vertices using our exact evaluation scheme near sharp features. Experimental results are presented to demonstrate the effectiveness of the method.

Categories and Subject Descriptors (according to ACM CCS): I.3.1 [Computer Graphics]: Surface Modeling

1. Introduction

Data reduction in computer graphics and CAD calls for fitting smooth parametric or implicit surfaces to mesh surfaces or 3D data points generated by laser range scanning systems. Subdivision surfaces have been used in shape fitting due to several favorable properties, such as smoothness, arbitrary control mesh connectivity, and intuitive shape control. However, the research on subdivision surface fitting has so far focused mainly on accuracy and efficiency issues with general smooth shapes [CWQ*07, MK05], without adequate emphasis on preservation of sharp shape features, that is, creases, darts, and corners. Normally, a dense set of control vertices are used to approximate a region of high curvature. This is clearly inefficient when it comes to faithful representation of sharp features. Fitting subdivision surfaces with features is first considered in [HDD*94], but Hoppe et al.'s algorithm uses piecewise linear approximation to represent a smooth surface, making exact error evaluation impossible.

We present a new method for fitting a Loop subdivision surface to a dense triangle mesh with sharp features. The fitting surface we compute faithfully captures the sharp features of the input shape without increasing the density of control vertices near them. Our method follows the same optimization framework of [MK05, WPL06] in shape fitting that solves a nonlinear least squares problem defined in terms of the squared distances from the input mesh vertices to the fitting subdivision surface. A key requirement of this optimization framework is to express the closest point (also called *foot point*) of an input mesh vertex on the limit subdivision surface as a linear combination of the control vertices through the basis functions of the subdivision surface. While this task is straightforward for B-spline surfaces, which are well parameterized everywhere, it becomes a difficult problem when one has to deal with a subdivision surface in the neighborhood of sharp features, because of the special subdivision rules used for feature generation. This is called the *foot point finding problem*.

Our contribution is an exact evaluation scheme of the Loop subdivision surface for various types of control vertex configurations near sharp features – an exact evaluation scheme means a method that can be used to exactly compute the point on the limit subdivision surface corresponding to any parameter values (s,t). This allows the parametrization of the fitting Loop subdivision surface near sharp features,

therefore enables us to solve the foot point finding problem for Loop surfaces near sharp features. Based on this result, we develop a subdivision surface fitting method that faithfully reconstructs sharp features. Our work can be regarded as an extension of [MK05] to include shapes possessing sharp features.

2. Related work

Several algorithms for fitting subdivision surfaces to data points have been presented in recent years [CWQ*04, CWQ*07, MK05]. Following [PLH02], Cheng et al. [CWQ*04] presented a method for smooth subdivision surfaces using the second order approximation of the squared distance from sampled points on the fitting surface to a target shape defined by a noise-free point cloud. This method cannot deal with data points with noise or sharp features properly.

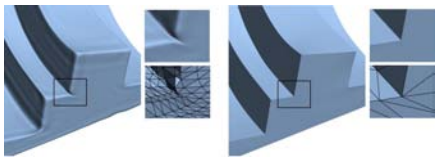

Figure 1: *Comparison between subdivision surfaces generated by a fitting method without using subdivision rules for features (e.g., [MK05]) and our method proposed in this paper. Close-up views of the subdivision surfaces and their control meshes are shown on the right.*

Marinov et al. [MK05] developed a subdivision surface fitting method based on *parameter correction* to achieve better error measurement. For each given data point, the closest point (or *foot point*) on the limit subdivision surface is found and this foot point needs to be expressed as a linear function of the control mesh vertices via the basis functions. An objective function, which is a function of the control points, is then defined in a least squares sense as the summation of the squared distances between the data points and their foot points. This objective function is minimized iteratively with repeated computation of the foot points to update the control points. The method yields a final fitting subdivision surface upon convergence. In curve or surface fitting, computing the foot points amounts to *data parametrization*, and updating the foot points of the data points due to the change of the control points is called *parameter correction* [Hos88, WPL06].

Only smooth fitting surfaces are generated by the algorithms above (i.e., [CWQ*04, MK05]). Given an input model with sharp features, these algorithms fail to accurately capture these features, since they do not use special subdivision rules for sharp features. One possible but inefficient remedy is local refinement that uses a set of dense control vertices – this handles regions of high curvature very well but cannot resolve the case of sharp features satisfactorily, as shown in Fig. 1. Aiming to preserve the sharp features in the fitting surface, Ma et al. [MMTP04] proposed an algorithm using special subdivision rules for features. However, in this method the fitting errors are only measured from a small number of sampled points on the fitting surface to a fixed subset of data points, which are quite different from true geometric errors, especially near sharp features. While this circumvents the need to accurately compute the foot points of all data points, it prevents the method from properly measuring and reducing the fitting error around sharp features, thus leading to poor fitting quality, as also discussed in [MK05].

Fitting subdivision surfaces to a general shape, including sharp features, has been considered by Hoppe et al. [HDD*94]. Their algorithm uses a piecewise linear approximation of the subdivision surface for fitting error evaluation, leaving much room for improvement in efficiency and accuracy. This issue has been resolved by Marinov and Kobbelt [MK05] for fitting subdivision surface without sharp features, based on Stam's exact evaluation scheme [Sta99].

We present an algorithm for fitting a Loop subdivision surface to a shape represented as a triangle mesh with sharp features. It consists of two main phases:

1. **Phase 1 (Initialization)**: An initial control mesh $\mathcal{M}_1$ is obtained by simplifying the input dense mesh $\mathcal{M}_0$ using quadric error metric (QEM) based method [GH97]. Sharp features are detected and labeled on the control mesh $\mathcal{M}_1$. Then a feature-sensitive edge flip operation is performed on $\mathcal{M}_1$ to reduce the number of its extraordinary control points.
2. **Phase 2 (Iterative optimization)**: Gradient-based optimization is run iteratively by performing the following two steps alternatively until convergence: (*i*) finding the foot points of the data points (i.e., input mesh vertices) to compute fitting errors; and (*ii*) updating the control mesh points to further reduce the fitting errors by solving a linear system of equations. (The convergence analysis of this type of iterative procedures is similar to that for B-spline curve fitting [WPL06].)

Our contribution is a new evaluation scheme for the Loop subdivision surface near sharp features, covering all configurations of mesh connectivity with respect to different types of sharp features. The main idea is to enumerate all these configurations and map them into the regular cases using a series of geometric transformations, so that Stam's exact evaluation scheme for the smooth case can be applied. This evaluation scheme is the key to enabling us to accurately compute the foot points on a Loop surface near sharp features, as required in Phase 2 of our method. Note that another evaluation method for subdivision surfaces at sharp features has also been presented by Zorin et al. [ZK02], which generates piecewise smooth surfaces different from what generated in the present paper using the scheme in [HDD*94].

3. Generation of initial control mesh

The initial control mesh used in our algorithm is obtained by simplifying the input triangle mesh in three steps: a) *simpli-*

fication; b) *feature detection*; and (c) *regularization of vertex valence*.

Simplification The input shape is represented by a dense triangle mesh $\mathcal{M}_0$, whose vertices will be called *data points*. The mesh $\mathcal{M}_0$ is first normalized by uniform scaling to fit all of its data points in the cube $[0,1]^3$. Then we use a QEM-based method [GH97] with edge length aspect ratio control to simplify $\mathcal{M}_0$ to obtain a coarse mesh $\mathcal{M}_1$.

Feature detection Sharp features on a surface can be corners or creases. An edge of a mesh can be *smooth edge* or *crease edge*, depending on whether it lies on a crease. All boundary edges of an open mesh are classified as crease edges. A mesh vertex is a *smooth vertex*, *dart vertex*, *crease vertex*, or *corner vertex* if it is incident to exactly 0, 1 and 2, or more than 2 crease edges, respectively.

For each edge e of the simplified mesh $\mathcal{M}_1$, we use a threshold θ_c, the angle between the normals of the two faces incident to e, to detect if e is a candidate crease edge. Although QEM simplification tends to preserve sharp features, there is certain loss of details. Therefore candidate crease edges detected solely based on the coarse mesh $\mathcal{M}_1$ need to be validated. For this purpose, we also detect feature vertices (i.e., dart vertices, crease vertices or corner vertices) on the input mesh $\mathcal{M}_0$ using a similar threshold approach, but with an appropriate threshold value θ_d that is larger than θ_c. Then a candidate crease edge on the coarse mesh $\mathcal{M}_1$ will be validated as a true crease edge if all m uniformly sampled points on the edge are close to some feature vertices on the dense mesh $\mathcal{M}_0$. We use the value $m = 5$ in our implementation. Note that once a mesh vertex of $\mathcal{M}_1$ is labeled as a feature vertex, its identity as a feature vertex will remain when its position is updated by subsequent optimization.

Note that detecting feature edges and vertices is a well studied topic in geometric processing. Here we have applied only the simplest intuitive approach to this problem and found that it produced acceptable results to be used subsequently in the fitting stage. In this sense, other possibly superior sharp feature detection methods (e.g., [HG01]) can also be used to provide the initial control mesh with labeled sharp features. Our contribution lies only in the optimization method that uses this initial control mesh as an input.

Feature-sensitive edge flip Since the Loop subdivision surface has only C^1 continuity at an irregular vertex (i.e., the valence is not 6), we perform feature-sensitive edge flip following the method of [SG03] to reduce the number of irregular mesh vertices. Specifically, we define the irregularity of a smooth edge e as

$$Irreg(e) = |val(v_0) - opt(v_0)| + |val(v_1) - opt(v_1)|, \quad (1)$$

where v_0 and v_1 are the two endpoints of the edge e, $val(v)$ is the valence of the vertex v, $opt(v) = 6$ for a smooth vertex and $opt(v) = 4$ for a crease vertex. To make the operation preserve crease edges, we define $Irreg(e) = 0$ for any crease edge e. Then we perform edge flip [SG03] to reduce the vertex irregularity of the mesh $\mathcal{M}_1$ by minimizing the function

$$R(\mathcal{M}) = \sum_{e \in \mathcal{M}} Irreg(e). \quad (2)$$

4. Evaluation of subdivision surface near features

In our optimization method, we need to repeatedly compute accurately the foot point of a data point (that is, a vertex of the input mesh $\mathcal{M}_0$) on the fitting subdivision surface and express the foot point as a linear combination of the control points via the basis functions of the subdivision surface. This calls for a method for exact evaluation of the subdivision surface $P(s,t)$ for any given parameter values (s,t); in other words, it is necessary to have the parametrization of the subdivision surface, especially near sharp features.

4.1. Subdivision rules

A face of a triangle mesh is called a *smooth face* if all of its three vertices are non-feature vertices; otherwise, it is a *feature face*, also called a *non-smooth face*. If all the three vertices of a feature face have regular valences (i.e., 6 for smooth or dart vertices, 4 for crease vertices, and 2 for corner vertices), then it is a *regular feature face*; otherwise, it is an *irregular feature face*. We will focus on subdivision rules involving *irregular feature faces*.

The Loop's subdivision rule for a smooth face uses a 1-4 splitting operator [Loo87], which updates every existing vertex and adds a new vertex associated with each edge, as illustrated in Fig. 2. Here the weight $\beta = \frac{1}{n}\left[\frac{5}{6} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right]$, where n is the vertex valence.

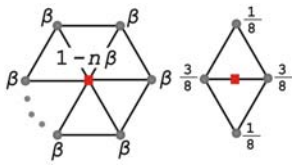

Figure 2: *Loop subdivision masks: (left) for an updated position of a vertex; (right) for a newly vertex on an edge.*

The masks for a Loop subdivision surface at corner vertices and creases edges are given in [HDD*94] and shown in Fig. 3.

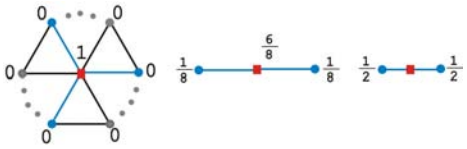

Figure 3: *Loop subdivision marks for features: (left) for a corner vertex; (middle) for a crease vertex; (right) for a newly inserted vertex on a crease edge.*

4.2. Mirror vertices for regular feature faces

The evaluation technique for internal smooth faces is presented in [Sta99]. The method for computing the exact limit position of a feature vertex is proposed in [Sch96] and used in [MMTP04]. These techniques are applied in our method, but we will skip the details due to the space limitation.

For an internal smooth face, Stam's technique for exact evaluation on a subdivision surface needs a mask of 12 vertices, as shown in Fig. 6. However, these rules cannot be applied directly to the case where the face is incident to a surface boundary or a sharp feature (that is, dart, corner or crease), due to missing vertices or the special subdivision rules for features. Our solution to this problem adopts a two-step strategy: 1) Resolve the case of *regular feature faces* using *mirror vertices* (see below); 2) resolve the case of *irregular feature faces* by reducing it to the case of regular feature faces.

The concept of mirror vertices is introduced in [Sch96] for analyzing regular feature faces. The idea is as follows. Referring to Fig. 4, to provide the missing vertices required for evaluating a regular face with one crease edge, the existing vertices p_2^k and p_3^k are reflected in the crease edge $p_1^k p_4^k$ to obtain the *mirror vertices* $\tilde{p}_6^k$ and $\tilde{p}_5^k$ which are expressed as $\tilde{p}_6^k = p_0^k + p_1^k - p_2^k$, $\tilde{p}_5^k = p_0^k + p_4^k - p_3^k$.

In Fig. 4, p_i^{k+1}s are new vertices in the next level of subdivision generated by special rules for a regular feature face in Fig. 4 (left) and rules for a smooth face in Fig. 4 (right). The region bounded by the dash lines is called the *mirror region*, and the region bounded by the solid lines is called the *non-mirror region*. It is shown in [Sch96] that the upper half of the smooth subdivision surface generated by smooth scheme using the complete mask in Fig. 4 (right) is the same as the limit surface that is generated using the subdivision rules for features using the "half mask" in Fig. 4 (left). This observation allows us to perform exact evaluation on a regular feature face containing a crease edge.

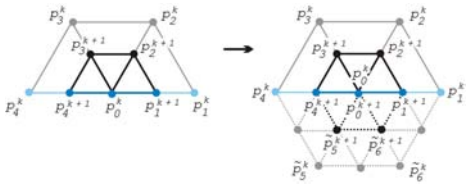

Figure 4: *(Left) the smallest invariant stencil for regular feature faces containing crease edges; (right) after adding mirror vertices.*

Extending this idea of using suitable reflections to obtain mirror vertices to make up for the missing vertices, we can also construct a complete mask for parameterizing a regular corner face as shown in Fig. 5. Here the mirror vertices are given by $\tilde{p}_3^k = p_0^k + p_2^k - p_1^k$, $\tilde{p}_4^k = 2p_0^k - p_1^k$, $\tilde{p}_5^k = 2p_0^k - p_2^k$, $\tilde{p}_6^k = p_0^k + p_1^k - p_2^k$.

By exhaustive enumeration it can be shown that, besides the case of a regular smooth face shown in Fig. 6, there are in total four cases of regular feature faces and eight cases of irregular feature faces. For reference, the basis functions for evaluating a regular smooth face based on the mask in Fig. 6 are listed in Appendix A. We will enumerate all the cases of feature faces in the remaining subsections and explain how these cases are handled.

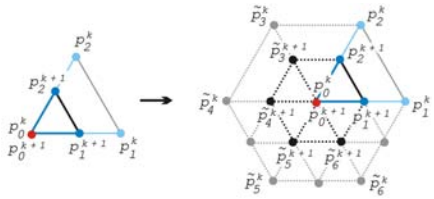

Figure 5: *(Left) the smallest invariant stencil for a regular feature face containing a corner vertex; (right) after adding mirror vertices.*

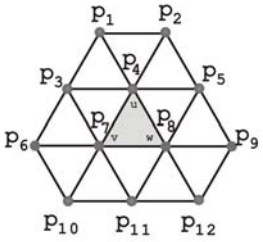

Figure 6: *The mask for evaluating a smooth internal face.*

4.3. Exact evaluation for regular feature faces

In the following we will list the four cases of regular feature faces and show how the mirror vertices can be generated to provide the full mask for exact evaluation.

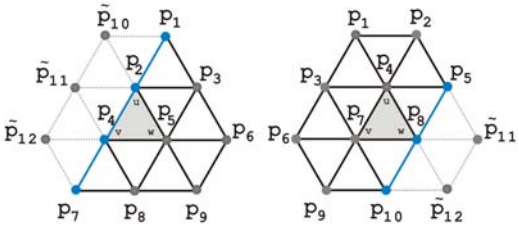

Figure 7: *(Left) Case 1: a regular feature face having one crease edge; (right) Case 2: a regular feature face having one crease vertex.*

Case 1: (See Fig. 7 (left)): The triangular face has a crease edge and two smooth edges. The mirror vertices are given by $\tilde{p}_{10}^k = p_1^k + p_2^k - p_3^k$, $\tilde{p}_{11}^k = p_2^k + p_4^k - p_5^k$, and $\tilde{p}_{12}^k = p_4^k + p_7^k - p_8^k$.

Case 2: (See Fig. 7 (right)): The triangular face has three smooth edges and one crease vertex. The mirror vertices are given by $\tilde{p}_{11}^k = p_5^k + p_8^k - p_4^k$, $\tilde{p}_{12}^k = p_8^k + p_{10}^k - p_7^k$.

Case 3: (See Fig. 8 (left)): The triangular face has two crease edges, two crease vertices and one corner vertex. The

mirror vertices are given by $\tilde{p}_7^k = p_1^k + p_2^k - p_3^k$, $\tilde{p}_8^k = p_2^k + p_4^k - p_5^k$, $\tilde{p}_9^k = 2p_4^k - p_5^k$, $\tilde{p}_{10}^k = 2p_4^k - p_2^k$, $\tilde{p}_{11}^k = p_4^k + p_5^k - p_2^k$, $\tilde{p}_{12}^k = p_5^k + p_6^k - p_3^k$.

Case 4: (See Fig. 8 (right)): The triangular face has three smooth edges and two crease vertices. The mirror vertices are given by $\tilde{p}_9^k = p_3^k + p_6^k - p_4^k$, $\tilde{p}_{10}^k = p_6^k + p_8^k - p_7^k$, $\tilde{p}_{11}^k = p_7^k + p_8^k - p_6^k$, $\tilde{p}_{12}^k = p_5^k + p_7^k - p_4^k$. Note that the corner vertex (i.e. p_8^k)in this mask can be replaced by a crease vertex and the computation of the mirror vertices remains the same.

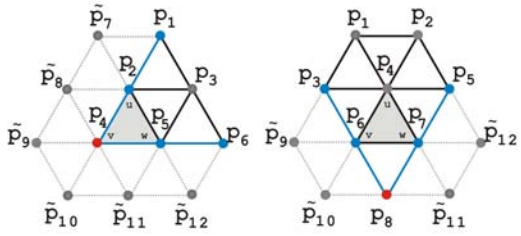

Figure 8: *(Left) Case 3: a regular feature face having one corner vertex; (right) Case 4: a regular feature face having two crease vertices but no crease edge.*

4.4. Exact evaluation of irregular feature faces

There are in total eight cases of irregular feature faces, which are faces that contain at least one irregular feature vertex, due to its incidence to either a crease edge or a dart/crease/corner vertex. We will only explain in detail one of these cases and list the other seven cases in the Appendix B.

Consider a face that has one crease edge, two smooth edges, one irregular crease vertex, one regular crease vertex and one smooth regular vertex, as shown in Fig. 9. Here the face $\triangle p_1 p_2 p_3$ is the patch which we are going to parameterize. Since the irregular crease vertex has valence $N \neq 4$, there are in total $J = N + 5$ vertices in the mask of this case. We store the initial J control vertices in a $J \times 3$ matrix

$$C_0^T = (p_{0,1}, ..., p_{0,J}). \tag{3}$$

After a step of subdivision, a new set of $M = N + 10$ control vertices are generated. The newly generated vertices provide us enough vertices to evaluate three-quarters of the triangular patch, as done in [Sta99]. The new set of control vertices are defined by

$$\begin{cases} C_1^T = (p_{1,1}, ..., p_{1,J}), \\ \bar{C}_1^T = (p_{1,1}, ..., p_{1,J}, p_{1,J+1}, ..., p_{1,M}). \end{cases} \tag{4}$$

This step of subdivision can be represented in a matrix form: $C_1 = AC_0$ and $\bar{C}_1 = \bar{A}C_0$. If we repeat the subdivision step, we generate an infinite sequence of control vertices

$$\bar{C}_k = \bar{A}C_{k-1} = \bar{A}A^{k-1}C_0, \; k \geq 1. \tag{5}$$

For each $k \geq 1$, the subset of vertices from $\bar{C}$ form the control vertices of a regular (feature) triangular patch. Note that this step is different from Stam's technique, because we select a different number of vertices from $\bar{C}$ for different triangular patches – in this case, the subset contains 9 vertices for patch 1, 10 vertices for patch 2, and 12 vertices for patch 3.

Let us denote the subsets of control vertices by matrices $B_{k,i}$ with $i = 1, 2, 3$. Then it follows that

$$B_{k,i} = P_i \bar{C}_k, i = 1, 2, 3. \tag{6}$$

The matrix size is $9 \times M$ for the pickup matrix P_1, $10 \times M$ for P_2, and $12 \times M$ for P_3. Each sub triangular patch is then defined as

$$s_{k,i}(v, w) = B_{k,i}^T b_h(v, w) = \bar{C}_k^T P_i^T b_h(v, w), \tag{7}$$

where b_h is given in Appendix A. Here $h = II$ when $i = 1$, $h = III$ when $i = 2$ and $h = I$ if $i = 3$, since different basis functions are given for different types of faces.

When $k \geq 1$ and $i = 1, 2, 3$, the subdomains are defined as used in Stam's technique:

$$B_{k,i} = P_i \bar{C}_k, i = 1, 2, 3. \tag{8}$$

With the modified pickup matrix and basis functions, we can find a parametrization $s(v, w)$ for all $(v, w) \in \Omega$. The parameter domain is partitioned into an infinite set of tiles Ω_i^k, with $k \geq 1$ and $i = 1, 2, 3$. The subdomains are defined by:

$$\begin{cases} \Omega_1^k = \{(v, w) | v \in [2^{-k}, 2^{-k+1}], w \in [0, 2^{-k+1} - v]\}; \\ \Omega_2^k = \{(v, w) | v \in [0, 2^{-k}], w \in [0, v]\}; \\ \Omega_3^k = \{(v, w) | v \in [0, 2^{-k}], w \in [2^{-k}, 2^{-k+1} - v]\}. \end{cases} \tag{9}$$

The surface patch is then defined as

$$s(v, w)|_{\Omega_i^k} = s_{k,i}(t_{k,i}(v, w)) = C_0^T (P_i \bar{A} A^{k-1})^T b_h(t_{k,i}(v, w)), \tag{10}$$

where

$$\begin{cases} t_{k,1}(v, w) = (2^k v - 1, 2^k w), \\ t_{k,2}(v, w) = (1 - 2^k v, 1 - 2^k w), \\ t_{k,3}(v, w) = (2^k v, 2^k w - 1). \end{cases} \tag{11}$$

The parametrization of the triangular face with one crease edge is defined by Eq.(10).

The parametrization for the seven other types of irregular feature faces is similar to the case discussed above, so we skip the derivation here. The masks for those seven types of feature faces are listed in Appendix B. If two irregular crease/dart/corner vertices are connected directly, we virtually subdivide the control mesh once and then evaluate on the underlying control mesh. This is similar to the case of two directly connected irregular smooth vertices, as addressed in [MK05, CWQ*04].

5. Optimization of control mesh

Let x_i, $i = 1, 2, 3, ..., n$, denote input data points, i.e., the vertices of the dense input mesh $\mathcal{M}_0$. We use the squared distance of each x_i to the limit subdivision surface to define the objective function

$$F_{dist} = \sum_{i=1}^{n} \|x_i - D(s_i, t_i)\|_2^2 \tag{12}$$

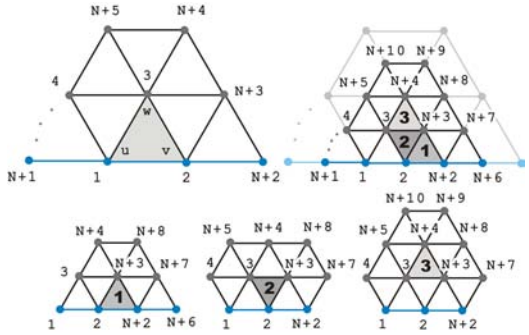

Figure 9: *Top left: An irregular feature face defined by $J = N + 5$ control vertices. Top right: after one step of Loop subdivision. $M = N + 10$ vertices in the new mesh. Bottom: three regular meshes corresponding to the three shaded patches. The labeling system of the control vertices defines the picking matrices.*

where (s_i, t_i) is the parameter values assigned to x_i such that $D(s_i, t_i)$ is the foot point of x_i on the subdivision surface $D(s,t)$.

We compute the foot points using the same Newton iteration scheme as used in [MK05], with necessary modifications when the foot point is found to be located in a crease edge. Here the exact evaluation rules discussed in the preceding sections are used within this Newton iteration to compute the exact surface point corresponding to updated parameter values, converging to the true foot point.

Following [LKE00], we use the energy term F_s to control the smoothness of the control mesh and discourage self-intersection.

$$F_s = \sum_{i=1}^{m} V(p_i)^T V(p_i), \tag{13}$$

where $\{p_i\}, i = 1,2,..,m$, are the smooth or dart vertices of the control mesh and $V(\cdot)$ is a discrete version of Laplacian. The crease and corner vertices are not included in this smoothness term, for otherwise crease edges and corners would be smoothed out.

Let $\{x_{ft,i}\}$ denote the set of feature vertices on the dense input mesh $\mathcal{M}_0$. To ensure that the sharp features are well fitted, we incorporate one more energy term defined as

$$F_{ft} = \sum_{i=0}^{l} \|p_{ft,i} - s_{ft,i}\|_2^2 \tag{14}$$

where $\{p_{ft,i}\}, i = 1,2,...,l$, are the uniformly sampled positions on the crease edges of the control mesh and $x_{ft,i}$ is the nearest point corresponding to $p_{ft,i}$ in $\{x_{ft,i}\}$.

The next energy term is used for smoothing along creases,

$$F_{s'} = \sum_{i=1}^{g} V(p_{cr,i})^T V(p_{cr,i}), \tag{15}$$

where $\{p_{cr,i}\}, i = 1,2,...,g$, are the crease vertices of the control mesh. Here $V(\cdot)$ is a discrete version of Laplacian operator. Only consecutive crease vertices are considered to be neighbors of $p_{cr,i}$ in this case.

The energy function for fitting a subdivision surface with features is finally given by

$$F = F_{dist} + \alpha F_s + \beta F_{ft} + \gamma F_{s'} \tag{16}$$

where α, β, γ are constants. Our tests show that α in the range of $[0.1, 0.5]$ leads to good results. We use $\beta = 1.0$ in all our experiments. The selection of γ will be discussed in next section. Since a foot point $\{D(s_i, t_i)\}$ and sampled points on crease edges $\{p_{ft,i}\}$ are linear combinations of the control points $\{p_i\}$, the updated control points $\{p_i\}$ can be computed by solving a linear system of equations.

Because only a small number of control points contribute to $D(s_i, t_i)$ or $p_{ft,i}$, the matrix for the linear system of equations is sparse. For efficiency, the conjugate gradient method is used to exploit the sparsity of the coefficient matrix to solve the linear system of equations. The conjugate gradient solver is terminated if the relative error improvement is less than a specific small value or the number of iterations exceeds a pre-specified number.

If the fitting result is not satisfied in a region due to too few control vertices or local minima in that region, we use local refinement operator to add new vertices. This strategy is fully discussed in [CWQ*04, MK05].

6. Results

In this section, we present some results to demonstrate the effectiveness of our fitting method. All experiments were conducted on a PC with Intel Duo Core2 2.8 GHz CPU and 2GB RAM. The fitting errors are obtained after normalizing the input dense mesh in the cube $[0,1]^3$.

Fig. 11 shows a mechanical model, Fandisk. We use $\gamma = 1.0$ for Eq. 16. Fig. 11 (bottom left) shows the initial error distribution. We see that most of large initial errors lie in smooth regions with higher curvature. The average L^2 fitting error (i.e., $(F_{dist}/n)^{1/2}$) decreases quickly in the first 3 iterations of optimization, beginning to decrease only slowly afterwards. The ratio between the number of vertices in the final control mesh and the initial dense mesh is 1.42% and the average L^2 fitting error is 5.81×10^{-4}. The total time consumed in the 22 iterations of optimization is 141 seconds.

Fig. 12 shows a model twisted in two directions. Since the feature curves have larger curvature and torsion, extra effort on smoothing the feature curves is needed by setting $\gamma = 5.0$ for this model. Fig. 13 shows another mechanical part. The optimized control mesh is generated using $\gamma = 1.0$. From the close-up views of the results, we see that the sharp features are well reconstructed. The total time of the optimization is

Table 1: *Statistics of the results*

	Fandisk	D.twist	Mechpart
Vert# of dense mesh	24322	32002	46076
Initial avg L^2 err	5.64*e*-3	5.99*e*-3	1.02*e*-2
Iteration#	22	7	9
Total time (sec)	141	165	127
Vert# of final mesh	346	1202	520
Fitting avg L^2 err	5.81*e*-4	1.07*e*-3	6.17*e*-4
Max fitting err	1.35*e*-2	1.49*e*-2	1.32*e*-2
Min fitting err	4.68*e*-6	1.29*e*-5	7.52*e*-6

165 and 127 seconds, respectively. The ratios between the vertex counts of the input dense mesh and the final control mesh are 3.75% and 1.13%, respectively.

Appendix A: Basis functions of regular cases

The basis functions [Sta99] for smooth internal face (Fig. 6), where $b_I = (b_{I,1}, \ldots b_{I,12})^T$:

$$\begin{cases}
b_{I,1} = (u^4 + 2u^3v)/12 \\
b_{I,2} = (u^4 + 2u^3w)/12 \\
b_{I,3} = (u^4 + 2u^3w + 6u^3v + 6u^2vw + 12u^2v^2 + 6uv^2w + \\
\qquad 6uv^3 + 2v^3w + v^4)/12 \\
b_{I,4} = (6u^4 + 24u^3w + 24u^2w^2 + 8uw^3 + w^4 + 24u^3v + \\
\qquad 60u^2vw + 36uvw^2 + 6vw^3 + 24u^2v^2 + 36uv^2w + \\
\qquad 12v^2w^2 + 8uv^3 + 6v^3w + v^4)/12 \\
b_{I,5} = (u^4 + 6u^3w + 12u^2w^2 + 6uw^3 + w^4 + 2u^3v + \\
\qquad 6u^2vw + 6uvw^2 + 2vw^3)/12 \\
b_{I,6} = (2uv^3 + v^4)/12 \\
b_{I,7} = (u^4 + 6u^3w + 12u^2w^2 + 6uw^3 + w^4 + 8u^3v + \\
\qquad 36u^2vw + 36uvw^2 + 8vw^3 + 24u^2v^2 + 60uv^2w + \\
\qquad 24v^2w^2 + 24uv^3 + 24v^3w + 6v^4)/12 \\
b_{I,8} = (u^4 + 8u^3w + 24u^2w^2 + 24uw^3 + 6w^4 + 6u^3v + \\
\qquad 36u^2vw + 60uvw^2 + 24vw^3 + 12u^2v^2 + 36uv^2w + \\
\qquad 24v^2w^2 + 6uv^3 + 8v^3w + v^4)/12 \\
b_{I,9} = (2uw^3 + w^4)/12 \\
b_{I,10} = (2v^3w + v^4)/12 \\
b_{I,11} = (2uw^3 + w^4 + 6uvw^2 + 6vw^3 + 6uv^2w + 12v^2w^2 \\
\qquad + 2uv^3 + 6v^3w + v^4)/12 \\
b_{I,12} = (w^4 + 2vw^3)/12
\end{cases} \tag{17}$$

The basis functions for regular feature case 1 (Fig. 7 (left)), where $b_{II} = (b_{II,1}, \ldots b_{II,9})^T$:

$$\begin{cases}
b_{II,1} = b_{I,1} + b_{I,2} & b_{II,2} = b_{I,1} + b_{I,3} + b_{I,4} \\
b_{II,3} = b_{I,5} - b_{I,1} & b_{II,4} = b_{I,3} + b_{I,6} + b_{I,7} \\
b_{II,5} = b_{I,8} - b_{I,3} & b_{II,6} = b_{I,9} \\
b_{II,7} = b_{I,6} + b_{I,10} & b_{II,8} = b_{I,11} - b_{I,6} \\
b_{II,9} = b_{I,12}
\end{cases} \tag{18}$$

The basis functions for regular feature case 2 (Fig. 7 (right)), where $b_{III} = (b_{III,1}, \ldots b_{III,10})^T$:

$$\begin{cases}
b_{III,1} = b_{I,1} & b_{III,2} = b_{I,2} \\
b_{III,3} = b_{I,3} & b_{III,4} = b_{I,4} - b_{I,9} \\
b_{III,5} = b_{I,5} + b_{I,9} & b_{III,6} = b_{I,6} \\
b_{III,7} = b_{I,7} - b_{I,12} & b_{III,8} = b_{I,8} + b_{I,9} + b_{I,12} \\
b_{III,9} = b_{I,10} & b_{III,10} = b_{I,11} + b_{I,12}
\end{cases} \tag{19}$$

The basis functions for regular feature case 3 (Fig. 6 (left)), where $b_{IV} = (b_{IV,1}, \ldots b_{IV,6})^T$:

$$\begin{cases}
b_{IV,1} = b_{I,1} + b_{I,2} \\
b_{IV,2} = b_{I,1} + b_{I,3} + b_{I,4} - b_{I,10} - b_{I,11} \\
b_{IV,3} = b_{I,5} - b_{I,1} - b_{I,12} \\
b_{IV,4} = b_{I,3} + b_{I,7} + 2b_{I,6} + 2b_{I,10} + b_{I,11} \\
b_{IV,5} = b_{I,8} + b_{I,11} + b_{I,12} - b_{I,3} - b_{I,6} \\
b_{IV,6} = b_{I,9} + b_{I,12}
\end{cases} \tag{20}$$

The basis functions for regular feature case 4 (Fig. 6 (right)), where $b_V = (b_{V,1}, \ldots b_{V,8})^T$:

$$\begin{cases}
b_{V,1} = b_{I,1} & b_{V,2} = b_{I,2} & b_{V,3} = b_{I,3} + b_{I,6} \\
b_{V,4} = b_{I,4} - b_{I,6} - b_{I,9} & & b_{V,5} = b_{I,5} + b_{I,9} \\
b_{V,6} = b_{I,6} + b_{I,7} + b_{I,10} - b_{I,12} \\
b_{V,7} = b_{I,8} + b_{I,9} + b_{I,12} - b_{I,10} \\
b_{V,8} = b_{I,10} + b_{I,11} + b_{I,12}
\end{cases} \tag{21}$$

Appendix B: Types of irregular feature faces

Besides the case shown in Fig. 9, Fig. 10 lists all other feature types.

(a) *Vertices*: one crease vertex with valence N and two regular smooth vertex. *Edges*: three smooth edges.

(b) *Vertices*: one crease vertex with valence 2 and two regular crease vertices. *Edges*: two crease edges and one smooth edge.

(c) *Vertices*: one dart vertex with valence N, one regular crease vertex and one regular smooth vertex. Edges: one crease edge and two smooth edges.

(d) *Vertices*: one dart vertex with valence N and two regular smooth vertices. *Edges*: three smooth edges.

(e) *Vertices*: one corner vertex with valence N and two regular smooth vertices. *Edges*: three smooth edges.

(f) *Vertices*: two regular crease vertices and one regular smooth vertices. *Edges*: three smooth edges. The orange vertex is either a corner vertex or a crease vertex.

(g) *Vertices*: one corner vertex with valence N, one regular crease vertex and one regular smooth vertex. *Edges*: one crease edge and two smooth edges.

References

[CWQ*04] CHENG K., WANG W., QIN H., WONG K. Y., YANG H. P., LIU Y.: Fitting subdivision surfaces

to unorganized point data using sdm. In *Proc. Pacific Graphics 2004* (2004), pp. 16–24.

[CWQ*07] CHENG K., WANG W., QIN H., WONG K. Y., YANG H. P., LIU Y.: Design and analysis of optimization methods for subdivision surface fitting. *IEEE Transactions on Visualization and Computer Graphics 13*, 5 (2007), 878–890.

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. In *Proc. SIGGRAPH '97* (1997), pp. 209–216.

[HDD*94] HOPPE H., DEROSE T., DUCHAMP T., HALSTEAD M., JIN H., MCDONALD J., SCHWEITZER J., STUETZLE W.: Piecewise smooth surface reconstruction. In *Proc. SIGGRAPH '94* (1994), pp. 295–302.

[HG01] HUBELI A., GROSS M.: Multiresolution feature extraction for unstructured meshes. In *Proc. Visualization, 2001* (2001), pp. 287–294.

[Hos88] HOSCHEK J.: Intrinsic parameterization for approximation. *Computer Aided Geometric Design 5*, 1 (1988), 27–31.

[LKE00] LÜRIG C., KOBBELT L., ERTL T.: Hierarchical solutions for the deformable surface problem in visualization. *Graphical Models 62*, 1 (2000), 2–18.

[Loo87] LOOP C.: *Smooth subdivision surfaces based on triangles*. Master's thesis, Universtity of Utah, 1987.

[MK05] MARINOV M., KOBBELT L.: Optimization methods for scattered data approximation with subdivision surfaces. *Graphical Models 67*, 5 (2005), 452–473.

[MMTP04] MA W., MA X., TSO S. K., PAN Z.: A direct approach for subdivision surface fitting from a dense triangle mesh. *Computer-Aided Design 36*, 6 (2004), 525–536.

[PLH02] POTTMANN H., LEOPOLDSEDER S., HOFER M.: Approximation with active b-spline curves and surfaces. In *Proc. Pacific Graphics 2002* (2002), pp. 8–25.

[Sch96] SCHWEITZER J. E.: *Analysis and Application of Subdivision Surface*. PhD thesis, University of Washington Seattle, Washington, 1996.

[SG03] SURAZHSKY V., GOTSMAN C.: Explicit surface remeshing. In *Proc. of Eurographics Symposium on Geometry Processing 2003* (2003), pp. 17–28.

[Sta99] STAM J.: Evalution of loop subdivision surfaces. In *Proc. SIGGRAPH '99 Course Notes* (1999).

[WPL06] WANG W., POTTMANN H., LIU Y.: Fitting b-spline curves to point clouds by curvature-based squared distance minimization. *ACM TOG 25*, 2 (2006), 214–238.

[ZK02] ZORIN D., KRISTJANSSON D.: Evaluation of piecewise smooth subdivision surfaces. *The Visual Computer 18*, 5 (2002), 299–315.

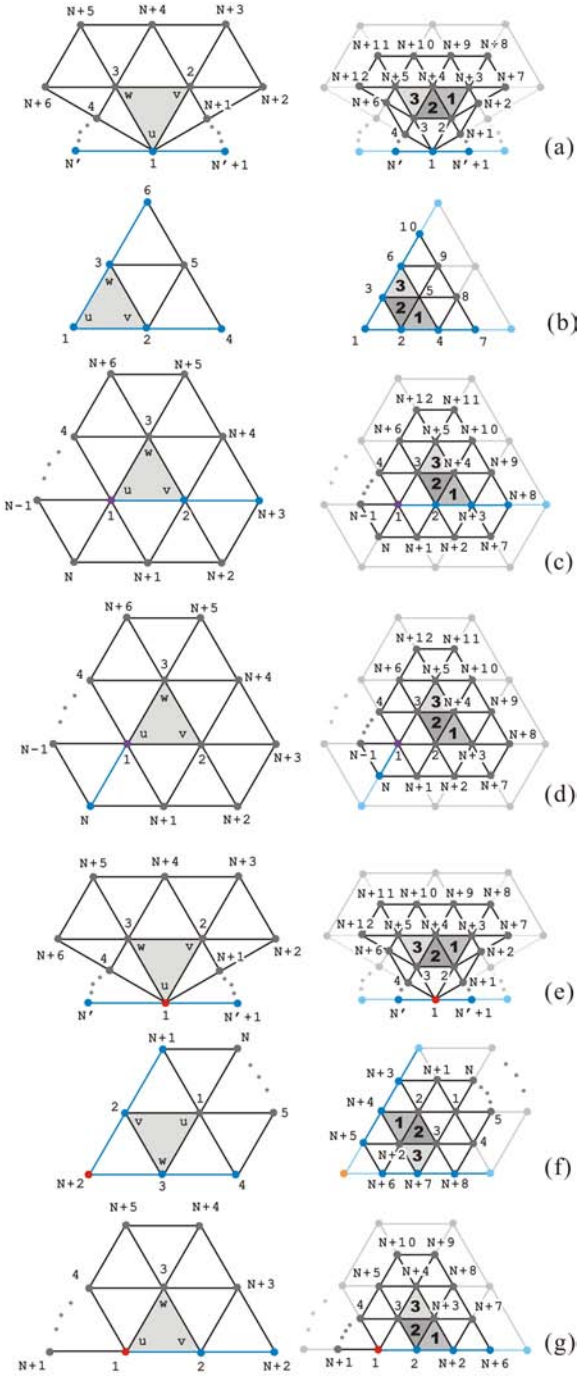

Figure 10: *The other seven cases of irregular feature faces besides Fig. 9, where blue edges are crease edges; purple, blue, red vertices are dart, crease, corner vertices, respectively.*

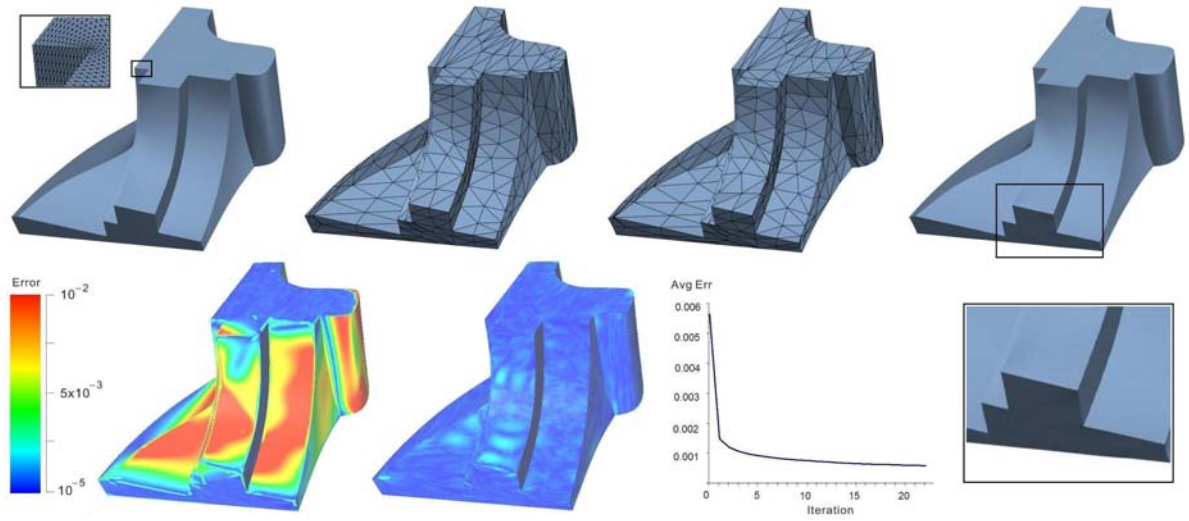

Figure 11: *Fan disk. Top row: input dense mesh; initial control mesh; optimized control mesh; subdivision surface. Bottom row: initial error distribution; final error distribution; average L^2 error curve; close-up view of the subdivision surface.*

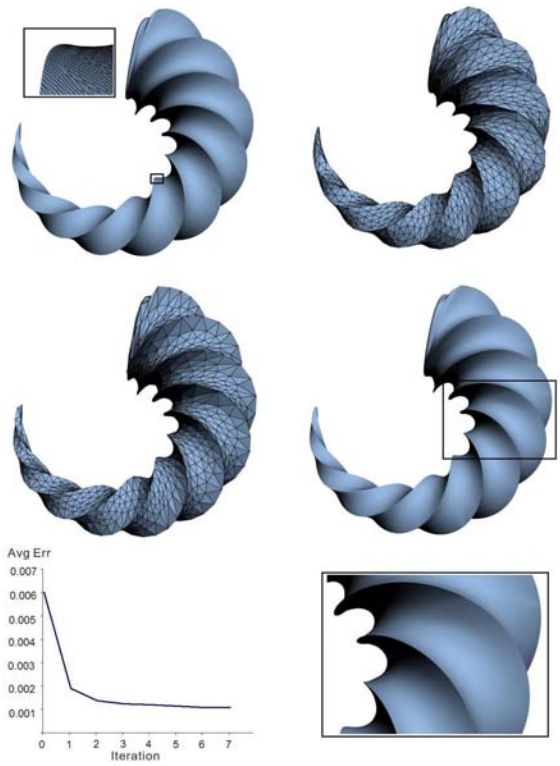

Figure 12: *Double twist. Top row: input dense mesh; initial control mesh. Second row: optimized control mesh; subdivision surface. Bottom row: average L^2 error curve; close-up view of the subdivision surface.*

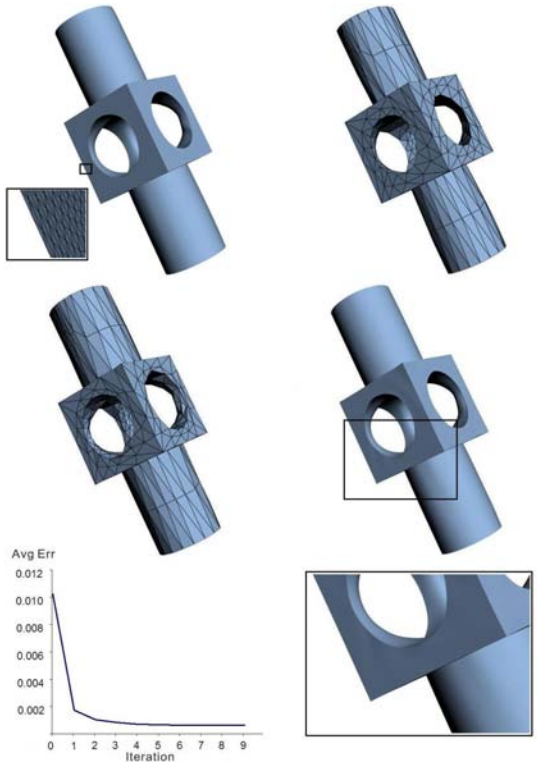

Figure 13: *Mechanical part. Top row: input dense mesh; initial control mesh. Second row: optimized control mesh; subdivision surface. Bottom row: average L^2 error curve; close-up view of the subdivision surface.*