Robust Feature-Preserving Mesh Denoising Based on Consistent Subneighborhoods

Hanqi Fan, Yizhou Yu, and Qunsheng Peng

Abstract—In this paper, we introduce a feature-preserving denoising algorithm. It is built on the premise that the underlying surface of a noisy mesh is piecewise smooth, and a sharp feature lies on the intersection of multiple smooth surface regions. A vertex close to a sharp feature is likely to have a neighborhood that includes distinct smooth segments. By defining the consistent subneighborhood as the segment whose geometry and normal orientation most consistent with those of the vertex, we can completely remove the influence from neighbors lying on other segments during denoising. Our method identifies piecewise smooth subneighborhoods using a robust density-based clustering algorithm based on shared nearest neighbors. In our method, we obtain an initial estimate of vertex normals and curvature tensors by robustly fitting a local quadric model. An anisotropic filter based on optimal estimation theory is further applied to smooth the normal field and the curvature tensor field. This is followed by second-order bilateral filtering, which better preserves curvature details and alleviates volume shrinkage during denoising. The support of these filters is defined by the consistent subneighborhood of a vertex. We have applied this algorithm to both generic and CAD models, and sharp features, such as edges and corners, are very well preserved.

Index Terms—Denoising, features, clustering, shared nearest neighbors, normals, curvature tensors, quadrics, bilateral filtering.

1 Introduction

Digital scanning devices are widely used to acquire high-resolution 3D models in recent years. It has become commonplace to model detailed 3D shapes by scanning real physical models. However, the acquired data inevitably have noise from various sources [1], [2]; therefore, smoothing algorithms are required to improve the quality of the reconstructed meshes for further processing. One of the major obstacles is that sharp features, including creases and corners that are very important for real models, are often corrupted by noises, and most existing techniques tend to more or less blur sharp features since they typically make use of the entire neighborhood of a vertex when performing noise removal even though different neighbors may receive different weights.

As we know, a surface with sharp features is actually piecewise smooth and a feature lies on the intersection of multiple smooth surface segments. When we perform noise removal for a point lying on one of the surface segments, it is much desired for that point to receive influence from neighboring points lying on the same surface segment only. Nevertheless, there exist challenging technical problems we need to solve before realizing this goal. First, how can we classify noisy vertices as features and nonfeatures? Second, in the presence of noise, how can we identify piecewise smooth surface segments in the neighborhood of a vertex? Third, how should we effectively perform noise removal for a curved surface region while preserving curvature and volume?

In this work, we introduce a piecewise surface denoising method that can effectively preserve sharp features and volume (Fig. 1). Because the surface normal varies continuously within a smooth region while changes abruptly across sharp features, we attempt to exploit normals as the primary cue for identifying piecewise smooth surface segments. Nevertheless, normal estimations are unreliable when vertices are corrupted with noise. This approach of using normals to assist the identification of discontinuities has been described as a "chicken and egg" problem [3] , since a normal is well defined only when local smoothness is assumed, but in the presence of noise, this computation is unreliable, even worse near a discontinuity.

We tackle this problem with a few effective techniques. First, a relatively reliable estimate of normals is obtained by robustly fitting a quadric approximation of the local surface geometry. Second, to identify piecewise smooth surface segments, we explicitly divide the normals within a vertex's neighborhood into multiple groups using a class of noise-resistant clustering techniques in data mining, called density-based clustering techniques. Specifically, we adopt a robust clustering algorithm based on shared nearest neighbors (SNN), which aims to find clusters of widely differing sizes, shapes, and densities in noisy data [4]. Each resulting cluster forms a subneighborhood. Third, before actual noise removal for vertex positions, we perform noise removal for normals and curvature tensors using a filtering algorithm based on the optimal estimation theory [5], which is on combining measurements from multiple noisy sources to provide an optimal estimate of a target. Such filtering happens within the consistent subneighborhood of every vertex. Our algorithm effectively filters normals, principal curvatures, and principal directions in a cascaded manner.

- *H. Fan and Q. Peng are with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310027, China.*
- *Y. Yu is with the Department of Computer Science, University of Illinois at Urbana-Champaign, 201 North Goodwin Ave., Urbana, IL 61801. E-mail: yyz@uiuc.edu.*

Manuscript received 24 Oct. 2008; revised 21 Apr. 2009; accepted 4 June 2009; published online 17 June 2009.
Recommended for acceptance by W. Wang.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2008-10-0175.
Digital Object Identifier no. 10.1109/TVCG.2009.70.

Fig. 1. (a) An initial mesh extracted from a noisy volume. (b) The denoised model from our method.

The final denoising procedure applied to a vertex's position is chosen according to the category of the vertex. For a nonfeature vertex, we apply second-order bilateral filtering within the vertex's consistent subneighborhood using an estimated local quadric surface. Being a more accurate approximation of the local geometry, higher order bilateral filters can better preserve local surface curvatures, and therefore, alleviate volume shrinkage. To preserve the sharpness of a feature vertex, its denoised position is the point along its normal vector that is closest to the tangent planes of its neighbors.

2 Related Work

There has been much work in the area of surface denoising in recent years [6], [7], [8], [9], [10], [11], [12], [13], [14]. Techniques for curvature-minimizing geometric fairing can also be found in [15], [16], [17]. Taubin [6] pioneered a signal processing approach to triangle mesh smoothing. Mesh fairing with mean curvature flow was introduced by Desbrun et al. [7] for smoothing irregular meshes. Ohtake et al. [18] further improved this work by integrating it with parameterization regularization. Such work has been extended to perform feature-preserving surface denoising [8], [19] by making diffusion nonlinear or anisotropic.

Recently, Fleishman et al. [11] and Jones et al. [12] successfully extended the bilateral filter from image denoising [20] to mesh denoising, which has a close relationship with anisotropic diffusion [21] and can be seen as a robust statistical technique [22]. These two papers define a fast, feature-preserving anisotropic filter. In [11], a vertex and its normal define a parameterization plane, over which a bilateral filter is applied to the neighborhood of that vertex. However, a vertex on a sharp edge that these algorithms aim to preserve is defined by the intersection of two planes rather than one. Bilateral filters cannot handle such situations well even though they automatically assign much smaller weights to outliers. Fleishman et al. [11] have also pointed out that their method gives rise to a certain amount of volume shrinking. Takeda et al. [23] extended the original bilateral filter [20] in image processing by using higher order local approximations of the signal. Duguet et al. [14] proposed a similar idea for mesh denoising that fits a quadric surface at each vertex to extend the method of Jones et al. [12]. Bilateral filters are applied to normals and curvature tensors in two passes, and a quality parameter based on local quadric approximation errors was introduced to further control the weights of the bilateral filters. These local quadric surfaces play the same role as tangent planes in Jones et al. [12]. This second-order technique achieves better results than Jones et al. [12]. However, it is a drifting method that does not explicitly determine whether there are multiple piecewise smooth regions near a vertex. It alters the vertex density over a mesh surface by making vertices near a sharp feature move toward that feature.

Hildebrandt and Polthier [24] use a prescribed mean curvature (PMC) flow to remove noise while preserving and sharpening geometric features and preventing boundary shrinkage. This approach works perfectly at low noise levels. When the noise became reasonably large; however, it would create features that are not present in the original mesh. Some other two-stage feature-preserving mesh denoising methods [25], [26], [27], [28] have also been proposed, which first filter face normals and then update vertex positions under the guidance of the filtered normals. These methods produce fairly good results. However, most of them cannot preserve small-scale discontinuities very well, especially when the noise level is reasonably high.

The assumption of an underlying piecewise smooth surface has been previously utilized in surface reconstruction from point clouds. Such reconstructions can leave sharp features in the original point cloud intact. Based on implicit modeling of a distance field, Hoppe et al. [29] introduce a method for reconstructing piecewise smooth surfaces in multiple passes, and sharp features are defined by two polygons with a crease angle that is larger than a threshold. Fleishman et al. [3] improved the moving least-squares (MLSs) technique by introducing a robust statistical method for reconstructing piecewise smooth point-set surfaces [30]. Their approach uses a forward search technique for outlier detection, which divides a point cloud into multiple smooth regions. Sharp features are defined as intersections of these smooth regions. However, since this approach computes the subneighborhoods independently for each point, it creates jagged edges due to the fact that nearby points may construct different neighborhoods [31].

3 Overview

We extend the piecewise smoothness assumption previously used for point-set surfaces [29], [3], [31] to noisy meshes, and devise a series of algorithms to achieve robust mesh denoising that preserves sharp features. That is, we assume that a mesh surface consists of smooth regions as well as sharp features in between these regions. A smooth region can be either planar or curved. A sharp feature can be an edge, where two smooth regions intersect, or a corner, where more than two smooth regions meet.

Our method begins with a classification of vertices into three categories, *sharp features*, *interior points* of a smooth region, and points close to the boundary of a smooth region. This is made possible by checking whether the neighbor-

hood of a vertex can be sufficiently modeled by a quadric surface (Section 4). When the quadric fitting error is sufficiently large, the vertex is considered a feature. An interior point is a nonfeature vertex that does not have any sharp features in its neighborhood. The remaining vertices belong to the third category, *transitional points*.

After obtaining an initial estimate of vertex normals from Section 4, we perform density-based normal clustering inside the neighborhood of every feature or transitional vertex to identify piecewise smooth regions (Section 5). Each resulting cluster forms a subneighborhood, and the *consistent* subneighborhood of a transitional vertex is defined as the one whose geometry and normal orientation are most consistent with those of the central vertex. For an interior vertex, its entire neighborhood is defined as a consistent neighborhood. Since our mesh denoising algorithm relies on normal and curvature estimations which can be noisy themselves, we improve their estimation using a filtering algorithm based on the optimal estimation theory (Section 6.1). Such filtering is performed within the consistent subneighborhood of every interior or transitional vertex.

The actual denoising procedure applied to a vertex is chosen according to the category of the vertex. For an interior or transitional vertex, we directly apply second-order bilateral filtering within its consistent subneighborhood (Section 6). If the vertex is a feature point, we first estimate a smooth surface, a tangent plane in our implementation, for each vertex in its entire neighborhood, and then its denoised position is simply the point along its normal vector that is closest to these estimated surfaces.

Since we frequently make use of quadric surfaces and principal curvatures, the following definition will be useful. Let $\mathbf{p}$ be a point on a differentiable surface region S. A local frame at $\mathbf{p}$ is defined as follows: Its origin is at $\mathbf{p}$. One of its axes is aligned with the surface normal at $\mathbf{p}, \mathbf{n}_p$. The other two axes are aligned with the principal directions at $\mathbf{p}$. Let $\mathbf{e}_1$ and $\mathbf{e}_2$ be orthogonal unit vectors aligned with the principal directions for which the normal curvature at $\mathbf{p}$ takes on maximum and minimum values κ_{max} and κ_{min}. Then, $\Delta_p = (\mathbf{p}, \mathbf{e}_1, \mathbf{e}_2, \mathbf{n}_p, \kappa_{max}, \kappa_{min})$ is often collectively referred to as the augmented Darboux frame at $\mathbf{p}$ [32].

4 Robust Quadric Estimation

The local geometry around a surface point $\mathbf{p}$ can be viewed as a height field over its tangent plane. Because a second-order polynomial can faithfully represent this height field up to its second-order derivatives, which represent essential information characterizing local geometric properties, it is sufficient to fit second-order surfaces to estimate the normal and principal curvatures at $\mathbf{p}$.

Estimating all the parameters of a parabolic quadric at $\mathbf{p}$ is separated into two subproblems [33]: estimating the surface normal and estimating the principal directions. We first construct a local frame from current normal at $\mathbf{p}$ and transform its nearest neighbors into this frame. The quadric surface model has the following form:

$$f_\beta(\mathbf{x}) = \beta \cdot \mathbf{x}, \tag{1}$$

where $\beta = [a, b, c, d, e, f]$ is a coefficient vector and $\mathbf{x} = [x^2, xy, y^2, x, y, 1]^T$ is the second-order polynomial basis vector. Because there exists noise, in order to minimize the sum of residues along the local z-axis,

$$\arg\min_\beta \sum_i |f_\beta(\mathbf{x}_i) - z_i|, \tag{2}$$

we adopt a robust statistics technique, iteratively reweighted least squares (IRLSs) [34], to reduce the influences of the outliers, including noises and discontinuities. The iterative minimization thus becomes

$$\arg\min_\beta \sum_i w(\mathbf{x}_i)|f_\beta(\mathbf{x}_i) - z_i|^2, \tag{3}$$

where the weight $w(\mathbf{x}_i)$ is computed by applying the Gaussian kernel to the residue at $\mathbf{x}_i$ from the previous iteration.

However, IRLS does not take spatial locality into account, which are especially important when the center point is on or near a sharp feature. In order to make the estimated quadric model more local and precise, we extend this single Gaussian kernel to a double Gaussian kernel as follows:

$$\begin{aligned} w(\mathbf{x}_i) &= w_d(\mathbf{x}_i) \cdot w_s(\mathbf{x}_i), \\ w_d(\mathbf{x}_i) &= \exp\left(-\|\mathbf{x}_i - \mathbf{x}\|^2 / 2\sigma_d^2\right), \\ w_s(\mathbf{x}_i) &= \exp\left(-\|f_\beta(\mathbf{x}_i) - z_i\|^2 / 2\sigma_s^2\right), \end{aligned} \tag{4}$$

where w_d controls the locality influence and σ_d is the largest distance between $\mathbf{x}$ and its k nearest neighbors, while w_s controls the residual influence and σ_s is the standard deviation of the residuals. Once the coefficient vector β has been determined, we update the local frame according to the new normal, $\mathbf{n} = [-d, -e, 1]^T / (d^2 + e^2 + 1)^{1/2}$, and then, repeat these IRLS and normal estimation steps. This process stops when the incremental change in the direction of the normal falls below a tolerance level. Principal curvatures and directions can then be derived from the last estimated quadric. Specifically, the principal curvatures are

$$a + c + \sqrt{(a-c)^2 + b^2}$$

and

$$a + c - \sqrt{(a-c)^2 + b^2},$$

respectively. Since the initial normal is obtained from a rough estimation, the normal is refined during the iterative process, which gives rise to more precise curvature estimation.

4.1 Sharp Feature Detection

The estimation of a local quadric surface entails a minimization of a weighted sum of squared distances as in (3). Note that this fitting error is not likely to completely vanish. We use the error from the best fit to indicate the quality of surface fitting. Local quadratic surface approximation is assumed to be accurate if the corresponding fitting error is low. Let χ_i be the fitting error at vertex i. The quality coefficient at this vertex is defined as follows [14]:

$$q_i = exp\big(-\chi_i^2 / \sigma_q^2\big), \tag{5}$$

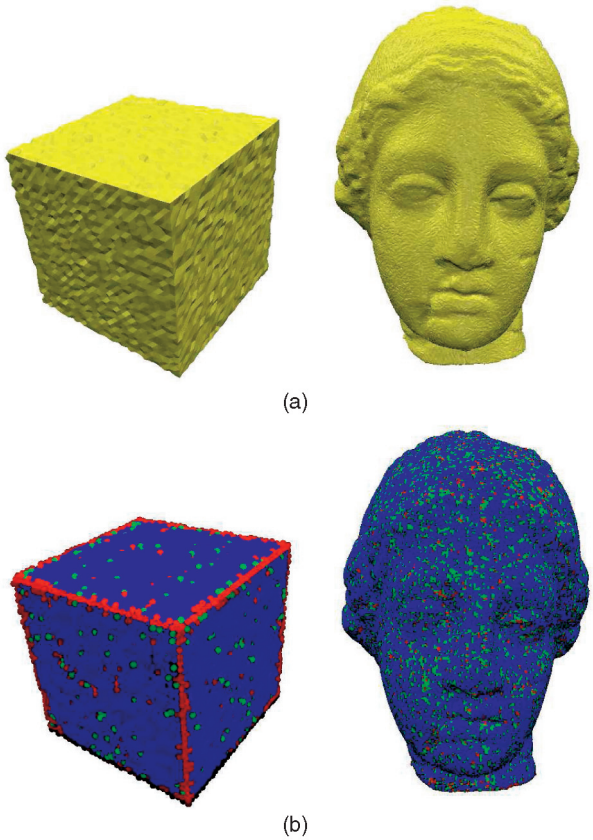


Fig. 2. Illustration of quality coefficients. (a) The noise corrupted models. (b) Red points are classified as feature points ($q_i \leq 0.1$), blue ones have best fitting quality ($q_i \geq 0.4$), and green ones have intermediate fitting quality.

where σ_q is the standard deviation of the fitting errors. Thus, $q_i \in [0, 1]$, and we choose vertices with low-quality coefficients as sharp features. As illustrated in Fig. 2, the quality coefficient becomes worse around sharp edges or in the presence of noise.

From the estimated quality coefficients, vertices are classified into three categories. 1) Feature points, with $q_i < \epsilon$. A large ϵ classifies more points as features than necessary, while a small one may miss some of the real features. Based on histograms of the quality coefficients computed for multiple meshes (Fig. 3), we chose $\epsilon = 0.1$ that worked very well in all our experiments. However, points satisfying this condition may not be real features, but noises.

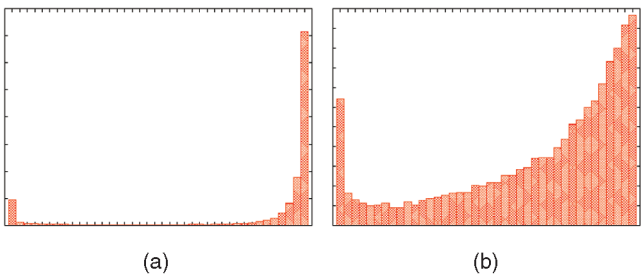


Fig. 3. Two typical histograms of the quality coefficients on the corresponding noisy models used in our paper. Each histogram has 40 bins which uniformly cover the range [0, 1]. It is clear that the lowest 10 percent of the bins forms a local peak. (a) Fandisk. (b) Bunny.

Fig. 4. Since normals in the red region have larger variations than those in the yellow region, DBSCAN is able to separate vertices in these regions into different clusters, and therefore, can achieve denoising without blurring the sharp edge between these regions.

This would not impose a problem, and the techniques presented in Sections 5 and 6 can deal with such misclassifications in a graceful way. 2) Interior points, without any feature points in its predefined neighborhood. 3) Transitional points, nonfeature points with feature points in its neighborhood. A true interior point may be misclassified as a transitional point if some of its neighbors have been misclassified as feature points. As discussed in Section 5.1 and Fig. 6d, our method can still function correctly.

5 Density-Based Normal Clustering

As discussed earlier, we would like to identify piecewise smooth regions in the neighborhood of feature and transitional vertices before applying any specifically tailored denoising procedure. Because the surface normal varies continuously within the same smooth region but changes abruptly across region boundaries, the normal vectors within a piecewise smooth neighborhood form multiple clusters on the Gaussian sphere. Normal discontinuities at region boundaries give rise to noticeable gaps in between these clusters, each of which may have a different density due to the curvature of the corresponding surface region. Based on these observations, we perform normal clustering within the neighborhood of every vertex (Fig. 4). Since our estimated surface normals are corrupted with noises as well, we adopt a density-based clustering algorithm frequently used in data mining. This algorithm has a strong capability in detecting clusters corrupted with outliers.

Density-based clustering algorithms, such as DBSCAN [35], locate transition boundaries among multiple clusters each of which has a relatively more uniform internal density. To make such algorithms less sensitive to density

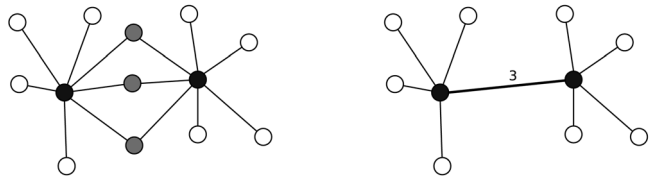


Fig. 5. Computation of SNN similarity between two points. The weight for the link between the two black points is set to the number of neighbors they share.

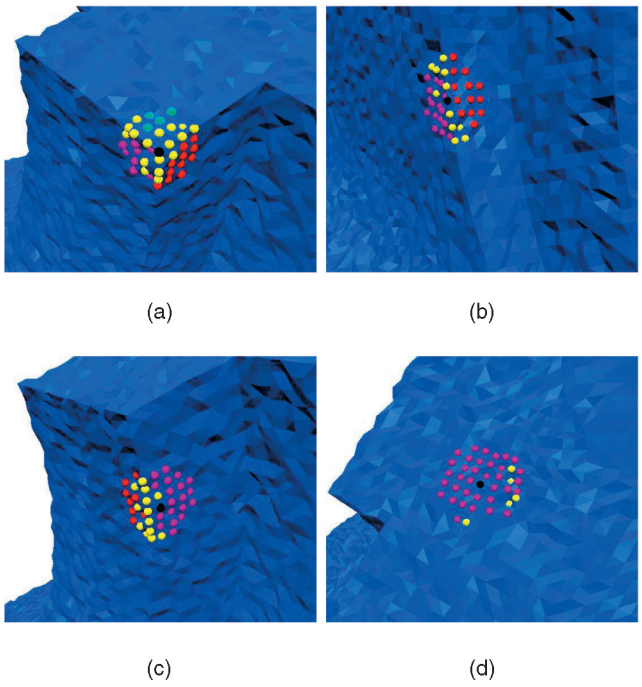


Fig. 6. Four typical clustering scenarios. The black dot represents the central vertex of the neighborhood, and the yellow ones have been classified as features, which do not participate in the clustering process. (a) The central point is a corner vertex. The entire neighborhood is divided into at least three clusters (red, cyan, and pink clusters). (b) The central point is on an edge. The neighborhood is divided into two clusters (red and pink clusters). (c) The central point is near an edge. The pink points form the consistent subneighborhood, while the red ones do not since their locations and normals are further away from those of the central point. (d) The central point lies on a flat surface region and some of its noise-corrupted neighbors are misclassified as features (yellow). Most nonfeature vertices in the neighborhood form a single cluster (pink), which also forms the consistent subneighborhood.

variations within the same cluster, Erröz et al. [4] improved the original DBSCAN algorithm with shared nearest neighbor (SNN) density, which has succeeded in finding climate indices [36]. This powerful improved algorithm first finds the nearest neighbors of each data point, and then, defines the similarity (distance) between pairs of points in terms of how many nearest neighbors they share. By this definition of similarity, this algorithm identifies core points, and then, builds clusters around these points.

The same as Jarvis-Patrick's approach [37], an SNN similarity graph needs to be constructed first for normals of vertices within the neighborhood of a mesh vertex. There is a node in the graph for each normal. For each pair of normals $\mathbf{n}_p$ and $\mathbf{n}_q$, a link is created between their corresponding graph nodes, if and only if the two normals are on each other's k nearest normal lists, and the strength/weight of the link between $\mathbf{n}_p$ and $\mathbf{n}_q$, that is, their similarity, is defined as

$$\text{similarity}(\mathbf{n}_p, \mathbf{n}_q) = \text{size}(NN(\mathbf{n}_p) \cap NN(\mathbf{n}_q)), \qquad (6)$$

where $NN(\mathbf{n}_p)$ and $NN(\mathbf{n}_q)$ are the nearest normal lists of $\mathbf{n}_p$ and $\mathbf{n}_q$, respectively.

Since SNN similarity is a robust measure of similarity (distance), we can use it in the nearest neighbor approach to density estimation. In detail, SNN density measures the degree to which a node is surrounded by similar nodes in the similarity graph, which means the number of neighbors with a link strength equal to or greater than *Eps*. Nodes in areas of either a high or low density typically have a relatively high SNN density, while nodes in areas where there is a transition from a low to high density, i.e., nodes that are between clusters, tend to have a low SNN density. For our problem, vertex normals on a local surface region with sharp features usually have such a character. There are low SNN density normal sets between the smooth subregions.

The nodes that have an SNN density greater than *MinPts* are further defined as *core points*. These points are in the interior of a cluster, which includes those core points that are within a similarity (distance) of *Eps* of each other. All noncore points that are not within a radius of *Eps* of a core point are eliminated. These points are regarded as outliers, and such elimination can make the filtering of the normal field more effective and robust. A border point, which is defined as a noncore and nonnoise point, is assigned to the cluster where its nearest core point belongs.

In conclusion, the main steps of the SNN density-based clustering algorithm are as follows:

1. Compute the SNN similarity graph for normals of vertices in a neighborhood on the mesh and the SNN density of each graph node.
2. Label the core points.
3. Form clusters from the core points.
4. Eliminate noisy outliers.
5. Assign each border point to the nearest cluster.

The most important parameter is the size of the nearest normal lists, k, which determines the granularity of the clusters. If k is too large, the algorithm tends to find only a few large clusters of nodes, since small local variations in similarity do not have an impact. On the other hand, if k is too small, even a uniform cluster would be broken up into many tiny pieces due to local variations in the similarity, and the algorithm tends to find many small clusters. Another parameter, *MinPts*, should be less than the size of the nearest normal lists since we only consider k most similar normals of a given normal. A larger Eps can better resist noise during clustering. But an overly large Eps would give rise to very small clusters since Eps also determines when to merge core points into the same cluster. To identify core points reliably, we should set $MinPts$ larger than Eps. However, an overly large $MinPts$ would leave few nodes in each cluster and render the clustering result useless. Altogether we choose a relatively small Eps to detect very loosely connected nodes as noisy points and a relatively large $MinPts$ to reliably identify core points.

In comparison with other popular clustering techniques, including K-means clustering and mean shift clustering, density-based clustering is better suited for outlier detection and neighborhood clustering where the number of clusters is unknown. K-means clustering has two inherent limitations: it does not pay attention to natural cluster boundaries due to the adoption of euclidean distance, and the number of clusters has to be known a priori. Mean shift clustering is not suitable either because it does not detect outliers and always assign every normal to a cluster even when the normal is corrupted with extreme noise.

5.1 Consistent Subneighborhoods

Once the improved DBSCAN algorithm divides a local neighborhood into piecewise smooth surface segments by means of normal clustering, we still need to determine the consistent subneighborhood of every transitional point. Note that every normal cluster has a corresponding vertex cluster in the neighborhood since there is a vertex associated with each normal. For a vertex cluster C in the neighborhood of a transitional point $\mathbf{p}$, its center $\mathbf{c}$ and correlation matrix $\mathbf{M}$ are defined as

$$\mathbf{c} = \frac{\sum_{\mathbf{q}\in C} \mathbf{q}}{\|C\|}, \tag{7}$$

$$\mathbf{M} = \frac{\sum_{\mathbf{q}\in C} (\mathbf{q}-\mathbf{c})(\mathbf{q}-\mathbf{c})^T}{\|C\|}. \tag{8}$$

The eigenvectors $\{\mathbf{v_0}, \mathbf{v_1}, \mathbf{v_2}\}$ of the correlation matrix together with the corresponding eigenvalues $\{\lambda_0, \lambda_1, \lambda_2\}$, where $\lambda_0 \leq \lambda_1 \leq \lambda_2$, define the correlation ellipsoid that indicates the configuration of the vertices in C. If $\lambda_0 \ll \lambda_1 \approx \lambda_2$ holds, which means that the ellipsoid is nearly flat, the vertices in C approximately lie on a planar surface segment [38]. Furthermore, $\mathbf{v_0}$ is an approximation of the surface normal. $\mathbf{c}$ and $\mathbf{v_0}$ together serve as the least-squares solution of fitting a plane to the vertices within C. To further judge whether vertex $\mathbf{p}$ lies on this surface segment, we check the distance between the vertex normal and the surface normal as well as the distance from the vertex $\mathbf{p}$ to the plane. The consistent subneighborhood is chosen to be the one with the smallest discrepancies in both position and normal orientation. In another word, the cluster with the smallest product of these two distances is chosen as the consistent subneighborhood. All the vertices in the consistent subneighborhood are recorded as $\mathbf{p}$'s true neighbors, and other neighbors are simply ignored in subsequent filtering steps.

As shown in Fig. 6, in the event that a true interior point were misclassified as a transitional point because some of its noise-corrupted neighbors had been misclassified as feature points, density-based clustering would still be able to assign most of its neighbors to a single large cluster, which then forms its consistent subneighborhood. All subsequent filtering steps discussed in Section 6 can still function correctly using this subneighborhood and produce very good denoising results.

6 Second-Order Mesh Denoising

Fleishman et al. [11] and Jones et al. [12] extend bilateral filters [20] from images to meshes. The idea of Fleishman et al. [11] is to take the tangent plane at each vertex as a local parameterization plane. The heights of the vertices over this plane are synonymous with the gray-scale values of an image. They formulate the bilateral filter for mesh denoising as:

$$d = \frac{\sum_{p_i\in N} W_c(\|\mathbf{p}_i - \mathbf{p}\|) W_s(h_i) h_i}{\sum_{p_i\in N} W_c(\|\mathbf{p}_i - \mathbf{p}\|) W_s(h_i)}, \tag{9}$$

where d is a signed distance with respect to the tangent plane, N is the neighborhood of $\mathbf{p}$, $\mathbf{n_p}$ is the normal at $\mathbf{p}$, and $h_i = <\mathbf{n_p}, \mathbf{p}_i - \mathbf{p}>$. The spatial weighting function is defined as $W_c(x) = \exp(-x^2/2\sigma_c^2)$, and the feature-preserving weighting function is defined as $W_s(x) = \exp(-x^2/2\sigma_s^2)$. σ_c is simply the largest distance between $\mathbf{p}$ and the k nearest neighbors, while σ_s is the standard deviation of h_is. The position of vertex $\mathbf{p}$ is updated as follows:

$$\hat{\mathbf{p}} = \mathbf{p} + d \cdot \mathbf{n_p}. \tag{10}$$

This method is simple and effective, and achieves good results. However, in high-curvature regions, using a single parameterization plane could smooth sharp features relatively fast even though bilateral filters can discount the influence of outliers to a certain extent.

We apply different denoising procedures for feature and nonfeature vertices. For nonfeature vertices, we make use of an estimated local quadric surface, such as the one obtained in Section 4, to perform second-order bilateral filtering. In comparison to [11], we define the height h_i of neighbor $\mathbf{p}_i$ over the quadric surface as the distance from $\mathbf{p}_i$ to this quadric along the normal $\mathbf{n_p}$. Otherwise, our second-order bilateral filter follows the same formula as in (9). The filtered vertex position can be obtained from (10). The reason to use a quadric surface instead of a plane is that the quadric is a higher order and more accurate approximation of the local geometry. It can better preserve volume (Table 3) and local surface curvature. Note that the neighborhood N used in our bilateral filter includes only neighbors in the consistent subneighborhood determined in Section 5.1. By excluding outliers and vertices in other subneighborhoods from the filtering process, we achieve more accurate and feature-preserving results.

For vertices classified as sharp features, because they are defined as the intersection of multiple local surface segments, we obtain their denoised positions by minimizing the quadric error metric defined in Garland and Heckbert [39] which was originally used for triangle mesh simplification. One deviation from the minimization in [39] is that we optimize this metric along the normal direction, which can achieve better results (Fig. 7).

In detail, in the neighborhood of a feature vertex $\mathbf{p}$, we first fit a plane to each cluster resulted from density-based clustering, and then, reset the normal at $\mathbf{p}$ to be the average normal of these planes. For example, in Fig. 5, the normal of $\mathbf{p}$ is set to the average of three plane normals each of which is defined by a cluster of cyan, red, and pink vertices, respectively. Once the normal at $\mathbf{p}$ has been refined, we apply a nondrifting scheme to optimize the position of $\mathbf{p}$. The optimal position is defined to be the point on the normal that minimizes the following weighted quadric error:

$$\arg\min_t \frac{\sum_{\mathbf{p}_i\in N} w_i \left\|\mathbf{n}_i^T(\mathbf{p} + t\mathbf{n}_p) + d_i\right\|^2}{\sum_{\mathbf{p}_i\in N} w_i}, \tag{11}$$

where

$$w_i = \exp\left(-\left\|\mathbf{n}_i^T(\mathbf{p} + t\mathbf{n}_p) + d_i\right\|^2/\sigma^2\right), \tag{12}$$

N is the neighborhood around $\mathbf{p}$, $\mathbf{n}_p$ is the normal at $\mathbf{p}$, and $\mathbf{n}_i$ and d_i are, respectively, the normal and offset of the tangent plane at a neighbor $\mathbf{p}_i$. This optimization is actually a minimization of the weighted average distance between $\mathbf{p}$ and the tangent planes at all neighbors, and σ is the standard variation of these distances. It has a quadric

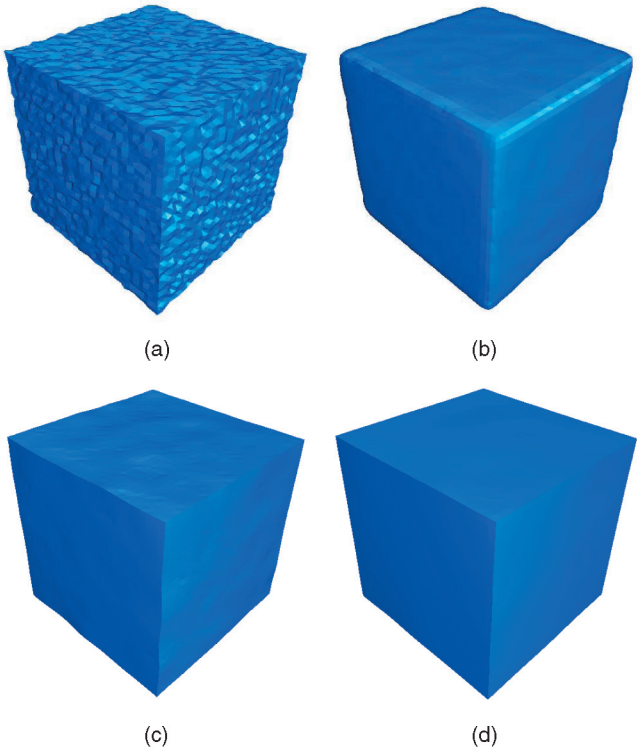


Fig. 7. (a) The input mesh is corrupted by Gaussian noise. (b) Sharp features could not be preserved without neighborhood clustering. (c) Directly minimizing the quadric error metric in [39] gives rise to a drifting method that produces wavy edges. (d) In contrast, our nondrifting method performs best, and can perfectly preserve sharp edges and corners.

energy function, and therefore, has a closed-form solution. Note that this energy-minimizing technique can still function very well in the event that a noise-corrupted vertex is misclassified as a feature point.

6.1 Optimal Estimation of Darboux Frames

Up to now, normals and curvatures are estimated from noisy vertex positions. Normals are first-order differential properties, while curvatures and principle directions are second-order differential properties. Noises amplify their influences on such estimated differential properties. It is desired to remove such ill influences before second-order bilateral filtering is performed using normals and quadrics. This is achieved using the optimal estimation theory.

An optimal estimator utilizes all measurement data plus prior knowledge about a system. It relies on the variance of the past prediction errors of a noisy source to determine its contribution in predicting the behavior of a target in the next iteration [5]. The curvature consistency framework in [40] adopts this idea and conducts an iterative process to obtain an optimal estimation of the augmented Darboux frame at every vertex. Consider a vertex $\mathbf{p}$ and its neighborhood $\{\mathbf{q}_j\}_{j\in N}$. Denote the Darboux frame estimated during iteration k for vertex $\mathbf{p}$ as Δ_p^k. At each iteration k, we intersect the normal vector at $\mathbf{p}$ with the quadric at $\mathbf{q}_j$ and further derive a local Darboux frame $\Delta_{q_j\frown p}^k$ on this quadric at the intersection using $\Delta_{q_j}^{k-1}$, which is the Darboux frame estimated during iteration $k-1$ for vertex $\mathbf{q}_j$. The detailed derivation can be found in [41]. This Darboux frame serves as the predicted version of the

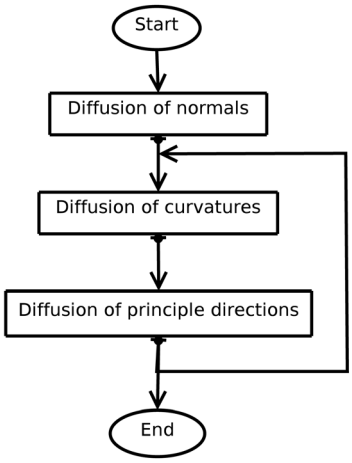


Fig. 8. The flow chart of our cascaded iterative method for curvature tensor filtering.

Darboux frame at $\mathbf{p}$ using the Darboux frame estimated in the previous iteration at $\mathbf{q}_j$ since the quadric at $\mathbf{q}_j$ is an accurate local second-order model and can provide a reasonable estimation of the second-order properties at $\mathbf{p}$.

In summary, the Darboux frames are iteratively updated as follows:

$$\Delta_{\mathbf{p}}^k = \sum_{j\in N} \lambda_j(k)\Delta_{q_j\frown p}^k, \tag{13}$$

where

$$\lambda_j(k) = \frac{\exp\left(-\frac{E_j^2(k-1)}{2\sigma_r^2(k-1)}\right)}{\sum_{l\in N}\exp\left(-\frac{E_l^2(k-1)}{2\sigma_r^2(k-1)}\right)}, \tag{14}$$

$$E_j^2(k) = E_j^2(k-1) + \epsilon_j^2(k), \tag{15}$$

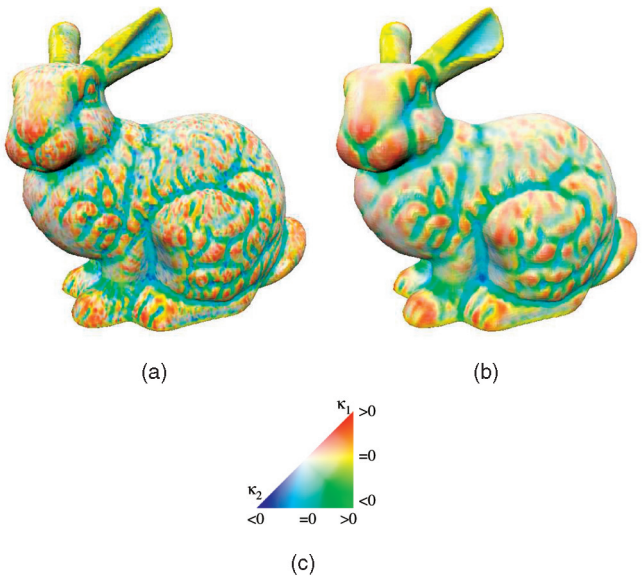
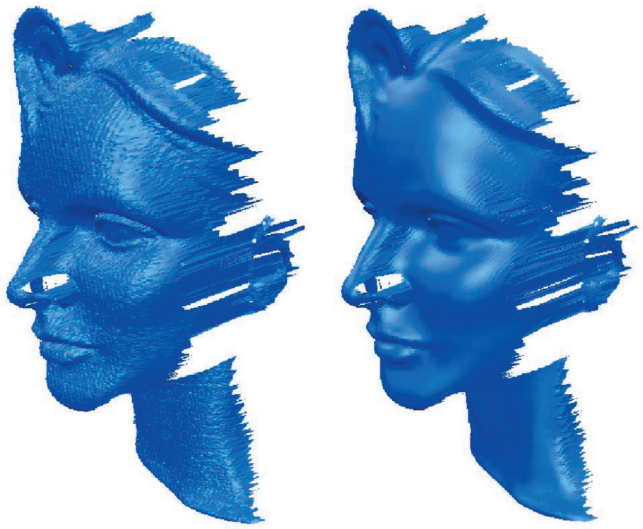


Fig. 9. Results from our improved curvature consistency algorithm. (a) Original curvatures. (b) Filtered curvatures after four iterations. (c) The color map of the curvature visualizations in (a) and (b).



(a) (b)

Fig. 10. Denoising result for the FEMME model. (a) The original model contaminated with real noise. (b) The denoised model generated from our method.

$$\epsilon_j(k) = \Delta^k_{\mathbf{p}} - \Delta^k_{q_j \frown p}, \tag{16}$$

$$\sigma_r^2(k) = \frac{2}{|N|} \sum_{j \in N} E_j^2(k). \tag{17}$$

Note that in the first iteration, we need to set $\lambda_j(1) = \frac{1}{|N|}$ and $E_j^2(0) = 0$.

One problem with this iterative estimation method is that it updates all components of an augmented Darboux frame simultaneously using the same weighting function and the metric for estimation errors also fuses the errors of all components together even though the range of individual components may be widely different. For example, the principal curvatures may approach infinity, while the vector components always have unit length. Thus, the iterations may be primarily driven by the curvature components without sufficient attention to the vector components [42]. We solve this problem by developing a cascaded iterative method to update various components in a sequential order. In our revised method, each iteration consists of three phases, which respectively improves the estimation of normals, two principal curvatures, and the principal direction $\mathbf{e}_1$. A result from our algorithm is shown in Fig. 9.

In the first phase, only the normal vectors are updated. Thus, (13) becomes

$$\mathbf{n}_p^k = \sum_{j \in N} \lambda_j^1(k) \mathbf{n}^k_{q_j \frown p}, \tag{18}$$

and the error metric is defined to be the angular distance between estimated normals:

$$\epsilon_j(k) = 1 - \frac{\mathbf{n}_p^k \cdot \mathbf{n}^k_{q_j \frown p}}{\left| \mathbf{n}_p^k \cdot \mathbf{n}^k_{q_j \frown p} \right|}. \tag{19}$$

In the second phase, two principal curvatures $\kappa = (\kappa_{max}, \kappa_{min})$ are updated simultaneously as follows, while the normal and principal directions are held constant:

$$\kappa_p^k = \sum_{j \in N} \lambda_j^2(k) \kappa^k_{q_j \frown p}. \tag{20}$$

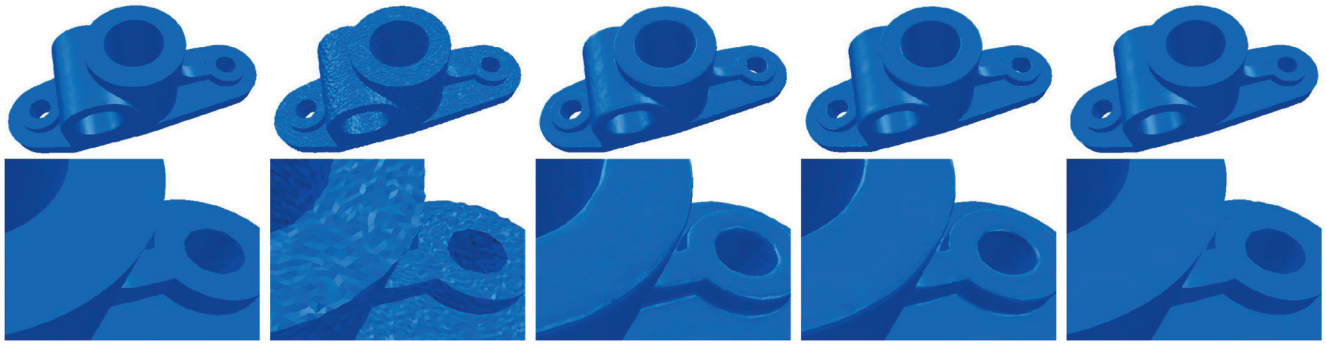
And the error metric is defined as follows:

$$\epsilon_j(k) = \left\| \mathbf{n}_p^k - \mathbf{n}^k_{q_j \frown p} \right\|^2 + \left\| \mathbf{e}_1^k - \mathbf{e}^k_{1_j} \right\|^2 + \left\| \kappa_p^k - \kappa^k_{q_j \frown p} \right\|^2. \tag{21}$$

In the third phase, only the principal direction $\mathbf{e}_1$ is updated as follows, while the other components are held constant:

$$\mathbf{e}_1^k = \sum_{j \in N} \lambda_j^3(k) \mathbf{e}^k_{1_j}. \tag{22}$$

The error metric is still the same as (21). In order to preserve the orthogonality of the Frenet frame, $\mathbf{e}_1$ is projected to the tangent plane defined by $\mathbf{n_p}$. As shown in Fig. 8, we alternate the second and third phases two to five iterations and each phase is executed around three times within each iteration. The specific choice depends on the noise level. Larger noise needs more iterations.



(a) (b) (c) (d) (e)

Fig. 11. A comparison of mesh denoising results among our method, bilateral filtering [11], and higher order bilateral filtering [14]. The original model is corrupted with Gaussian noises with $\sigma = 15$ percent of the mean edge length. The second row shows zoomed views. As presented, Fleishman et al. [11] removed noises, but at the same time, blurs the sharp features inevitably, while Duguet et al. [14] did slightly better. In contrast, the result of our method keeps most of the original sharp features. (a) Original model. (b) Noise-corrupted model. (c) Bilateral filtering. (d) High-order filtering. (e) Our method.

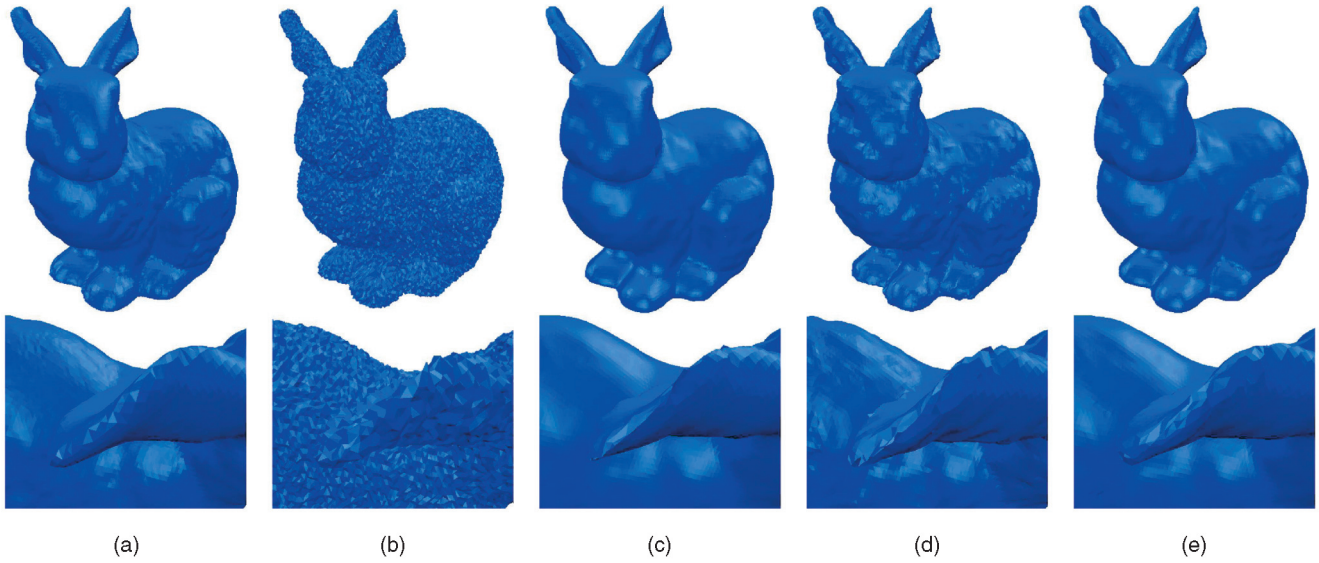


Fig. 12. A comparison of mesh denoising results from our method, bilateral filtering [11], and high-order bilateral filtering [14]. The original model is corrupted with large-scale Gaussian noises with $\sigma = 50$ percent of mean edge length. The second row shows a zoomed view of an ear. As presented, Fleishman et al. [11] removed noises, but at the same time, noticeably shrunk the volume in areas such as the ears, while Duguet et al. [14] could not completely remove noises. The result of our method is most similar to the original model. (a) Original model. (b) Noise-corrupted model. (c) Bilateral filtering. (d) High-order filtering. (e) Our method.

Liu et al. [42] proposed a related method that defines the error between principal curvatures using two parts. The first part is the euclidean distance between the vectors of principal curvatures, and the second one is based on the angular distance between corresponding principal directions. That is,

$$\epsilon_j^1(k) = \left\| \kappa_p^k - \kappa_{q_j \frown p}^k \right\| \quad \text{and} \quad \epsilon_j^2(k) = sin\theta_e^k. \tag{23}$$

The weighting function $\lambda_j(k) = \lambda_j^1(k) \cdot \lambda_j^2(k)$. This scheme encounters problems in the vicinity of umbilical points because the estimated principal directions are unstable around umbilical points, which, in turn, leads to unstable weighting functions. Our method alleviates this problem because none of our error metrics considers principal directions only. Liu et al. [42] also proposed another error metric which is defined as the determinant of the difference of two curvature tensors. This error metric vanishes when the difference tensor becomes singular. The method by Duguet et al. [14] adopts the same error metric and has the same problem too.

7 Experimental Results and Discussion

We have implemented the techniques presented in previous sections and successfully tested them on the following three

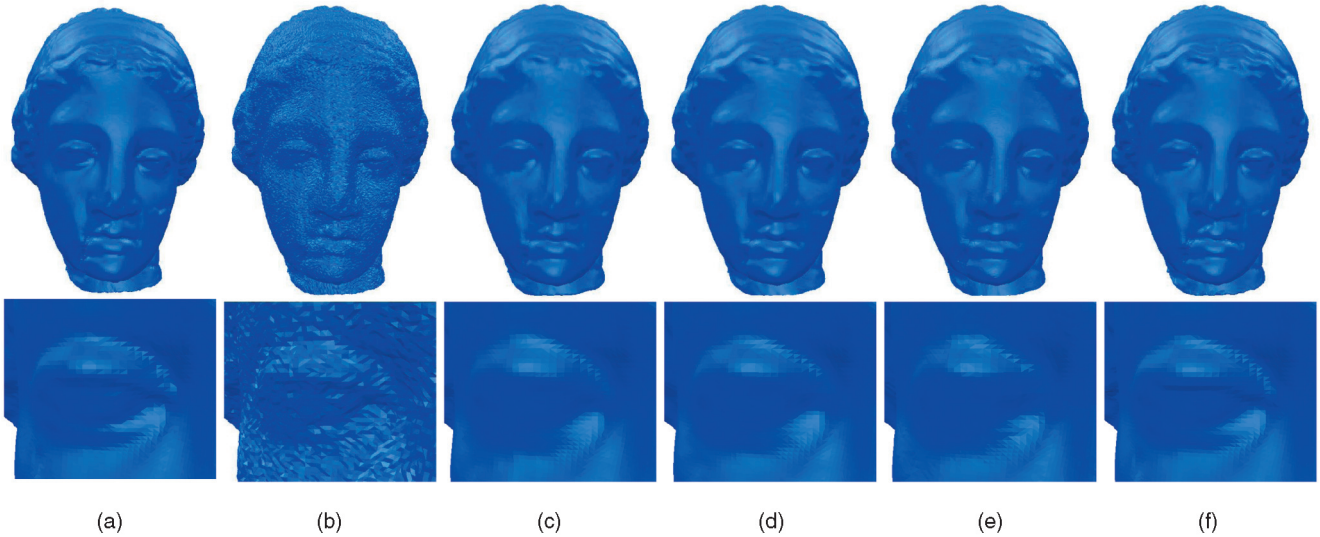


Fig. 13. Comparison between our method and other feature-preserving methods on the IGEA model with Gaussian noise ($\sigma = 10$ percent of mean edge length) added. The second row shows zoomed views of the left eye of the model. Our method preserves small-scale features best. (a) Original model. (b) Noisy model. (c) PMC [24]. (d) Sun et al. [27]. (e) Sun et al. [28]. (f) Our method.

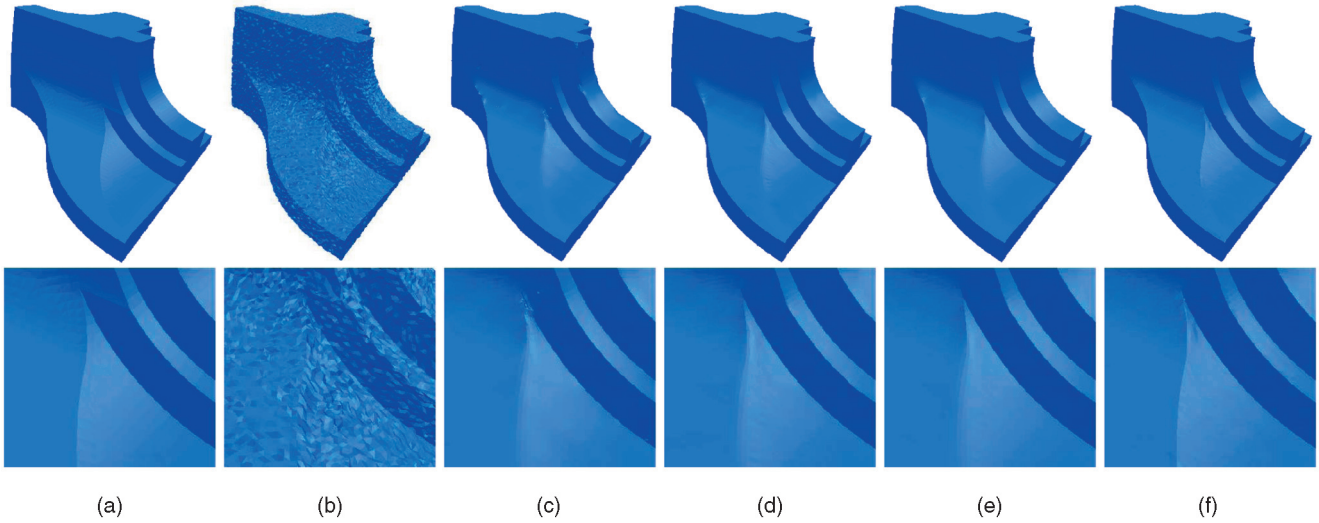


Fig. 14. Comparison between our method and other feature-preserving methods on the FANDISK model with Gaussian noise ($\sigma = 20$ percent of mean edge length) added. The second row shows zoomed views of a shallow feature on the model. Our method perfectly preserves this feature. (a) Original model. (b) Noisy model. (c) PMC [24]. (d) Sun et al. [27]. (e) Sun et al. [28]. (f) Our method.

classes of meshes, generic meshes with real (Figs. 10 and 15) or synthetic noise (Figs. 13 and 12), CAD models with synthetic noise (Figs. 14 and 11), and a mesh extracted from volume data with nonuniform volume noise (Fig. 1). As discussed in previous sections, our method involves only a small number of parameters which were held constant on all experimental results reported in this section. Since we detect consistent subneighborhoods, our method is insensitive to the original size of the neighborhood. We always use a neighborhood size of 30. When density-based clustering is performed, we set $k = 8, Eps = 2$, and $MinPts = 6$.

In Fig. 1, we show denoising results of a twirl that was extracted from volume data with nonuniform noise. This model is not uniformly sampled with low-curvature parts undersampled. Nonuniform sampling makes it harder to preserve sharp features while removing noise. However, our method achieved an almost perfect result.

Fig. 11 shows a comparison between our method and existing bilateral filtering methods, including Fleishman et al. [11] and Duguet et al. [14]. This comparison demonstrates the effectiveness of our method for both denoising and feature preservation. We can observe that the other methods have more or less blurred the sharp features.

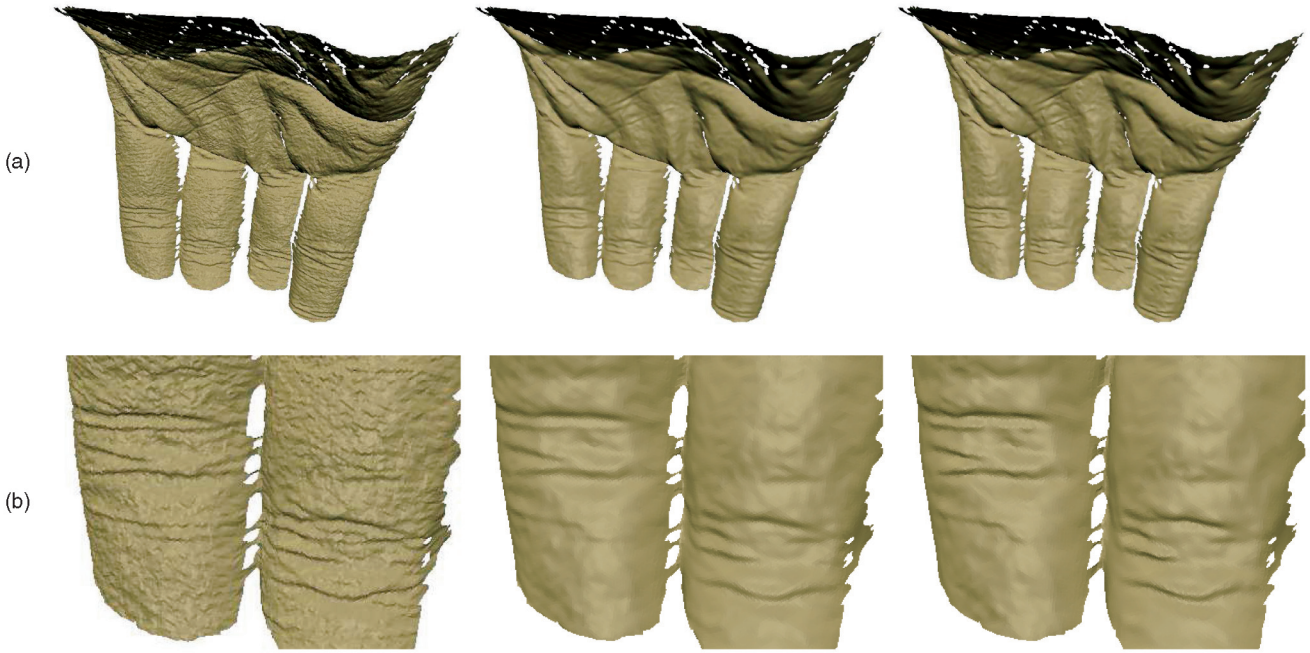


Fig. 15. Denoising a real scan. (a) From left to right are the input noisy model, the result from Fleishman et al. [11], and our result. (b) We show zoomed views of the fingers in the scanned model. The image in the middle shows the result from Fleishman et al., while our result on the right demonstrates that our method can better preserve small wrinkles.

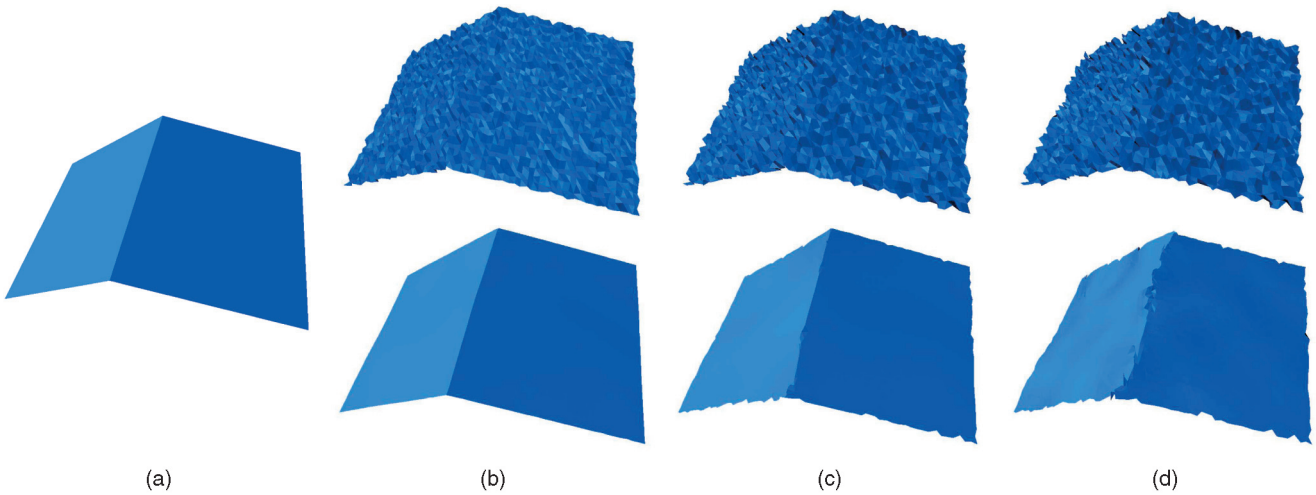


Fig. 16. Denoising with an increasing noise level. (a) Original Wedge model. (b) Denoised result of a model with additive noise of 40 percent of the mean edge length. (c) Denoised result of a model with additive noise of 80 percent of the mean edge length. (d) With additive noise of 100 percent of the mean edge length, features in the denoised result become rougher and planar regions start to undulate.

We have also compared these methods on the Stanford bunny model (Fig. 12). With the presence of large noise, Fleishman et al. [11] noticeably shrink volume and create minor self-intersections around the tip of an ear (Fig. 12c). As for the technique in Duguet et al. [14], Fig. 12d shows that it cannot smooth the surface very well, even worse on the ears.

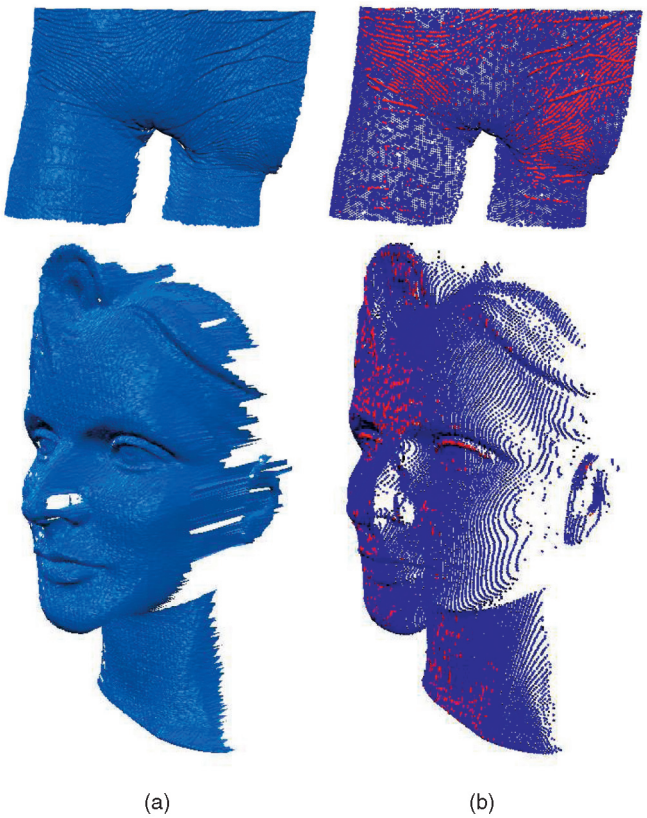


Fig. 17. Feature detection on models with a nonstationary sampling density. (a) Two noisy models acquired from real objects. (b) Detected feature points (red) over the models in (a) (only the point clouds are rendered here to indicate the original sampling density).

We have compared our method with a few recent feature-preserving denoising methods. Figs. 13 and 14 clearly show that our method can better preserve small-scale features and shallow discontinuities than other methods while removing noises.

Fig. 16 shows the results from our method on a model with an increasing level of noise. If the noise level is not high enough to make the sharp features indistinguishable, the presented algorithm recovers features perfectly (Fig. 16b). As the noise level increases, a small number of feature points are classified as transitional points since the information carried by these features are not distinguishable from smooth regions. Blurring occurs on the unrecognized features (Fig. 16c). As the noise level becomes even larger, more and more features mix with noise and blurring happens more frequently (Fig. 16d).

The correct detection of feature points relies on the sampling density over the mesh surface. The Nyquist-Shannon sampling theorem states that signal reconstruction is possible only when the sampling frequency is at least twice as high as the highest frequency in the original signal. In our algorithm, we try to reconstruct the original signal using second-degree polynomial fitting. When the sampling density is not too low on the features, our algorithm is able to detect them. Fig. 17 shows that on the nonuniformly sampled models, our algorithm successfully finds out feature points in regions with a reasonable sampling density, while points in regions with a low sampling density are unlikely to be classified as features even when they are real features. As mentioned earlier, the performance of our algorithm would not degrade severely even when it cannot detect feature points accurately. We have applied our method to meshes with nonuniform sampling density and real noise. Femme is a model acquired with 3D photography [43], and Palm is digitized with laser scanning. Figs. 10 and 15 show that our method successfully removed noise on these models while preserving features as much as possible.

Table 1 reports the running time of our algorithm on a few meshes using a PC with a 2.8 GHz Intel Pentium D

TABLE 1
Timing Report on a Few Meshes Using Our Algorithm

Model	Figure	# Vertices	Time (seconds)
Twirl	1	28,876	3.141
Femme	10	40,928	4.018
Block	11	45,540	4.596
Bunny	12	34,834	5.339
Palm	15	62,566	6.122

TABLE 2
Timing Comparison with Other Methods
(All Timings are Reported in Seconds)

Model	Fandisk	Igea
Figure	14	13
# Vertices	45,430	134,345
PMC ([24])	8.179	24.929
Sun et al. ([27])	0.661	2.117
Sun et al. ([28])	0.685	2.228
Ours	4.557	13.404

TABLE 3
Volume Ratios (Denoised/Original) of the Closed Models

Model	Igea	Fandisk	Block
Figure	13	14	11
# Vertices	134,345	45,430	45,540
Bilateral ([11])	0.998	0.991	0.991
PMC ([24])	0.997	0.989	0.989
Sun et al. ([27])	0.998	0.993	0.990
Sun et al. ([28])	0.998	0.993	0.990
Ours	0.999	0.998	0.993

processor and 1 GB RAM. Our code was optimized using Intel's SIMD instructions [44] and compiled with GCC-4.3.2. This table shows that the running time grows with the number of vertices. It takes a longer time to smooth Bunny than Block although Bunny has less vertices. This is because more iterations are applied to remove the large noise on Bunny. Table 2 gives a comparison of running times among a few methods. It can be seen that our method can achieve high-quality results in a reasonable amount of time. We can further significantly improve the performance of our code with multithreading on multicore CPUs. Table 3 shows that our method better preserves volume than other methods.

8 Conclusions

We have presented a feature-preserving mesh denoising algorithm. It is built on the premise that the underlying surface of a noisy mesh is piecewise smooth. By defining the consistent subneighborhood of a vertex and applying various filtering processes within the consistent subneighborhood, we can completely remove the influence from neighbors in other subneighborhoods during denoising. Our method identifies piecewise smooth subneighborhoods using a robust density-based clustering algorithm based on shared nearest neighbors. We adopt second-order bilateral filters for vertex positions and a filter based on the optimal estimation theory for normals and curvature tensors. Experiments show that our method can effectively preserve volume as well as sharp features.

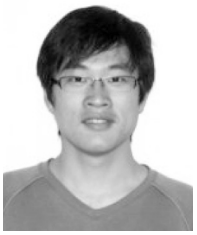
Acknowledgments

The authors would like to thank the reviewers for their valuable comments. This work was partially supported by National Natural Science Foundation of China (60728204/F020404).

References

[1] M. Levoy et al., "The Digital Michelangelo Project 3d Scanning of Large Statues," *Proc. 27th Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 131-144, 2000.
[2] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy, "Real-Time 3d Model Acquisition," *Proc. 29th Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 438-446, 2002.
[3] S. Fleishman, D. Cohen-Or, and C. Silva, "Robust Moving Least-Squares Fitting with Sharp Features," *Proc. ACM SIGGRAPH '05*, vol. 24, no. 3, pp. 544-552, 2005.
[4] L. Ertöz, M. Steinbach, and V. Kumar, "Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data," *Proc. 2003 SIAM Int'l Conf. Data Mining*, 2003.
[5] A. Gelb, *Applied Optimal Estimation*. MIT Press, 1974.
[6] G. Taubin, "A Signal Processing Approach to Fair Surface Design," *Proc. 22nd Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 351-358, 1995.
[7] M. Desbrun, M. Meyer, P. Schröder, and A. Barr, "Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow," *Proc. 26th Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 317-324, 1999.
[8] U. Clarenz, U. Diewald, and M. Rumpf, "Anisotropic Geometric Diffusion in Surface Processing," *Proc. Conf. Visualization '00*, pp. 397-405, 2000.
[9] G. Taubin, "Linear Anisotropic Mesh Filtering," IBM Research Report RC2213, 2001.
[10] C. Bajaj and G. Xu, "Anisotropic Diffusion on Surfaces and Functions on Surfaces," *ACM Trans. Graphics*, vol. 22, no. 1, pp. 4-32, 2003.
[11] S. Fleishman, I. Drori, and D. Cohen-Or, "Bilateral Mesh Denoising," *Proc. Int'l Conf. Computer Graphics and Interactive Techniques*, pp. 950-953, 2003.
[12] T. Jones, F. Durand, and M. Desbrun, "Non-Iterative, Feature-Preserving Mesh Smoothing," *ACM Trans. Graphics*, vol. 22, no. 3, pp. 943-949, 2003.
[13] H. Yagou, Y. Ohtake, and A. Belyaev, "Mesh Denoising via Iterative Alpha-Trimming and Nonlinear Diffusion of Normals with Automatic Thresholding," *Proc. Computer Graphics Int'l*, 2003.
[14] F. Duguet, F. Durand, and G. Drettakis, "Robust Higher-Order Filtering of Points," Technical Report RR-5165, INRIA, 2004.
[15] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel, "Interactive Multi-Resolution Modeling on Arbitrary Meshes," *Proc. ACM SIGGRAPH '98*, pp. 105-114, 1998.
[16] R. Schneider and L. Kobbelt, "Geometric Fairing of Irregular Meshes for Free-Form Surface Design," *Computer Aided Geometric Design*, vol. 18, pp. 359-379, 2001.
[17] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher, "Geometric Surface Processing via Normal Maps," *ACM Trans. Graphics*, vol. 22, no. 4, pp. 1012-1033, 2003.
[18] Y. Ohtake, A. Belyaev, and I. Bogaevski, "Polyhedral Surface Smoothing with Simultaneous Mesh Regularization," *Proc. Conf. Geometric Modeling and Processing '00 (Theory and Applications)*, pp. 229-237, 2000.
[19] M. Desbrun, M. Meyer, P. Schroder, and A. Barr, "Anisotropic Feature-Preserving Denoising of Height Fields and Bivariate Data," *Graphics Interface*, vol. 11, no. 10, pp. 145-152, 2000.
[20] C. Tomasi and R. Manduchi, "Bilateral Filtering for Gray and Color Images," *Proc. Sixth Int'l Conf. Computer Vision*, p. 839, 1998.
[21] F. Durand and J. Dorsey, "Fast Bilateral Filtering for the Display of High-Dynamic-Range Images," *Proc. 29th Ann. Conf. Computer Graphics and Interactive Techniques*, pp. 257-266, 2002.
[22] D. Barash, "Fundamental Relationship between Bilateral Filtering, Adaptivesmoothing, and the Nonlinear Diffusion Equation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 844-847, June 2002.
[23] H. Takeda, S. Farsiu, and P. Milanfar, "Higher Order Bilateral Filters and Their Properties," *Proc. SPIE Conf. Computational Imaging V*, vol. 6498, p. 64, 2007.

[24] K. Hildebrandt and K. Polthier, "Anisotropic Filtering of Non-Linear Surface Features," *Computer Graphics Forum,* vol. 23, no. 3, pp. 391-400, 2004.
[25] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, and H.-Y. Shum, "Mesh Editing with Poisson-Based Gradient Field Manipulation," *ACM Trans. Graphics,* special issue for *ACM SIGGRAPH 2004,* vol. 23, no. 3, pp. 641-648, 2004.
[26] K. Lee and W. Wang, "Feature-Preserving Mesh Denoising via Bilateral Normal Filtering," *Proc. Ninth Int'l Conf. Computer Aided Design and Computer Graphics 2005,* p. 6, 2005.
[27] X. Sun, P. Rosin, R. Martin, and F. Langbein, "Fast and Effective Feature-Preserving Mesh Denoising," *IEEE Trans. Visualization and Computer Graphics,* vol. 13, no. 5, pp. 925-938, Sept./Oct. 2007.
[28] X. Sun, P. Rosin, R. Martin, and F. Langbein, "Random Walks for Feature-Preserving Mesh Denoising," *Computer Aided Geometric Design,* vol. 25, no. 7, pp. 437-456, 2008.
[29] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle, "Piecewise Smooth Surface Reconstruction," *Proc. 21st Ann. Conf. Computer Graphics and Interactive Techniques,* pp. 295-302, 1994.
[30] N. Amenta and Y. Kil, "Defining Point-Set Surfaces," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 264-270, 2004.
[31] J.I. Daniels, L.K. Ha, T. Ochotta, and C.T. Silva, "Robust Smooth Feature Extraction from Point Clouds," *Proc. IEEE Int'l Conf. Shape Modeling and Applications (SMI '07),* pp. 123-136, 2007.
[32] M. do Carmo, *Differential Geometry of Curves and Surfaces.* Prentice-Hall, 1976.
[33] R. Garimella and B. Swartz, "Curvature Estimation for Unstructured Triangulations of Surfaces," Technical Report LA-UR-03-8240, Los Alamos Nat'l Lab, 2003.
[34] P. Huber, *Robust Statistics.* Wiley Interscience, 2004.
[35] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining.* Addison-Wesley Longman Publishing Co., Inc., 2005.
[36] M. Steinbach, P. Tan, V. Kumar, S. Klooster, and C. Potter, "Discovery of Climate Indices Using Clustering," *Proc. Ninth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 446-455, 2003.
[37] R. Jarvis and E. Patrick, "Clustering Using a Similarity Measure Based on Shared Near Neighbors," *IEEE Trans. Computers,* vol. C-22, no. 11, pp. 1025-1034, Nov. 1973.
[38] S. Gumhold, X. Wang, and R. MacLeod, "Feature Extraction from Point Clouds," *Proc. 10th Int'l Meshing Roundtable,* pp. 293-305, 2001.
[39] M. Garland and P. Heckbert, "Surface Simplification Using Quadric Error Metrics," *Proc. ACM SIGGRAPH,* vol. 97, no. 31, pp. 209-216, 1997.
[40] S. Mathur and F. Ferrie, "Edge Localization in Surface Reconstruction Using Optimal Estimation Theory," *Proc. 1997 Conf. Computer Vision and Pattern Recognition (CVPR '97),* 1997.
[41] F. Ferrie, J. Lagarde, and P. Whaite, "Darboux Frames, Snakes, and Super-Quadrics: Geometry from the Bottom Up," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 15, no. 8, pp. 771-784, Aug. 1993.
[42] M. Liu, Y. Liu, and K. Ramani, "Anisotropic Filtering on Normal Field and Curvature Tensor Field Using Optimal Estimation Theory," *Proc. IEEE Int'l Conf. Shape Modeling and Applications 2007,* pp. 169-178, 2007.
[43] J. Bouguet and P. Perona, "3D Photography on Your Desk," *Proc. Sixth Int'l Conf. Computer Vision '98,* pp. 43-50, 1998.
[44] Intel, *Intel®64 and IA-32 Architectures Optimization Reference Manual,* http://download.intel.com/design/processor/manuals/248966.pdf, 2009.



Hanqi Fan received the BS degree in computer science and technology in 2004 from the State Key Laboratory of CAD&CG, Zhejiang University, China, where he is currently working toward the PhD degree majoring in computer science. His research interests include computer animation, 3D modeling, reconstruction, and denoising algorithms.

Yizhou Yu received the PhD degree in computer science from the University of California at Berkeley in 2000. He is currently an associate professor in the Department of Computer Science, University of Illinois at Urbana-Champaign and an adjunct professor at Zhejiang University, China. He is a recipient of the Best Paper Award at 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2002 National Science Foundation CAREER Award, and 1998 Microsoft Graduate Fellowship. He is on the editorial board of the *Visual Computer Journal* and *International Journal of Software and Informatics*. His current research interests include computer animation, geometry processing, texture analysis, and visual analytics.

Qunsheng Peng graduated from Beijing Mechanical College in 1970, and received the PhD degree from the Department of Computing Studies, University of East Anglia, UK, in 1983. He is a professor of computer graphics at Zhejiang University. His research interests include realistic image synthesis, computer animation, scientific data visualization, virtual reality, and biomolecule modeling. He serves currently as a member of the editorial boards of several international and Chinese journals.