Bilateral Recovering of Sharp Edges on Feature-insensitive Sampled Meshes

Charlie C. L. Wang, *Member*, *IEEE*

Abstract—A variety of computer graphics applications sample surfaces of 3D shapes in a regular grid without making the sampling rate adaptive to the surface curvature or sharp features. Triangular meshes that interpolate or approximate these samples usually exhibit relative big error around the insensitive sampled sharp features. This paper presents a robust general approach conducting bilateral filters to recover sharp edges on such insensitive sampled triangular meshes. Motivated by the impressive results of bilateral filtering for mesh smoothing and denoising, we adopt it to govern the sharpening of triangular meshes. After recognizing the regions that embed sharp features, we recover the sharpness geometry through bilateral filtering, followed by iteratively modifying the given mesh's connectivity to form singlewide sharp edges that can be easily detected by their dihedral angles. We show that the proposed method can robustly reconstruct sharp edges on feature-insensitive sampled meshes.

Index Terms—Boundary representations, Geometric algorithms, languages, and systems.

1 Introduction

More and more computer graphics applications employ volumetric representation to reconstruct and modify the shape of three-dimensional surfaces. Many of such applications conduct the transformation between volumetric and boundary representation in the manner of: surface-volume conversion, processing, and volume-surface conversion. For instance, the voxel-based mesh simplification [15], the voxel-based three-dimensional metamorphosis [9], and some algorithms fixing topological errors (ref. [21] and [32]), they all use the volumetric representation as intermediate. Besides, in computer-aided engineering, there are many approaches (see [1], [4], and [45]) using implicit representation to evolve the shape and topology of a model to provide the optimal mechanical property.

In above applications, the simplest method to determine the volumetric or implicit representation is to sample given models on regular grids. However, as been mentioned in [2], sharp edges and corners on the original surface are removed by the sampling process. Over-sampling could somewhat reduce the aliasing error by taking the cost of increasing storage memory. Furthermore, as be observed by Kobbelt et al. in [23], even if an over-sampling is applied, the associated aliasing error will not be absolutely eliminated since the surface normals in the reconstructed model usually do not converge to the normal field of the original model. Therefore, the technique of recovering sharp edges on feature-insensitive sampled models is desired. Some of currently existed approaches (e.g., [22], [23], [33] and [35]) encode the original surface normals during sampling, so that a Hermite dataset is generated to reconstruct sharp features. However if the volumetric data is not converted from an explicit mesh representation, e.g. the Heterogeneous object modeling in [1], no normal could be accurately given, we thus need to recover sharp features only from the given mesh surfaces.

The purpose of the approach presented in this paper is akin to [2], to recover singlewide sharp edges on triangular meshes. However, the filters introduced in [2] can only sharpen the "chamfered" edges that are generated by the Marching-Cubes algorithm [27] or its variants (ref. [5], [26], and [31]). For the insensitive sampled edges that are either rounded (e.g., the model in Fig. 1a) or with noises corrupted (e.g., the model in Fig. 1b), the algorithm of [2] fails. The Gaussian noises on Fig. 1b and some of the later shown models are added for the purpose to demonstrate the robustness of our approach. In generality, noises are only in the form of slight normal disturbances or the smoothly degenerated sharp regions, where the rounded edges are usually generated by dynamic surface extraction algorithms – e.g., ShrinkWrap [36] or Skin [28]. Besides, objects with a non-manifold structure (e.g., the model in Fig. 2) are not addressed in [2].

1.1 Previous Work

The work proposed in this paper relates to previous researches conducted in the areas of: feature sensitive sampling, anisotropic mesh smoothing, sharp feature detection and recovering, and remeshing techniques, which are consecutively reviewed below.

Feature Sensitive Sampling

A variety of feature sensitive sampling techniques are adopted for isosurface reconstruction and surface remeshing. The authors in [13] generated adaptively sampled distance fields (ADFs) to increase the accuracy of sampling around sharp features. However, as mentioned above, the normals on a reconstructed model do not converge to the normal field of the original model. Thus, two improved isosurface extraction algorithm preserving sharp features

The author is with the Department of Automation and Computer-Aided Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, P. R. China
E-mail: cwang@acae.cuhk.edu.hk; Fax: (852)2603 6002; Tel: (852)2609 8052

Manuscript received (insert date of submission if desired). Please note that all acknowledgments should be placed at the end of the paper, before the bibliography.

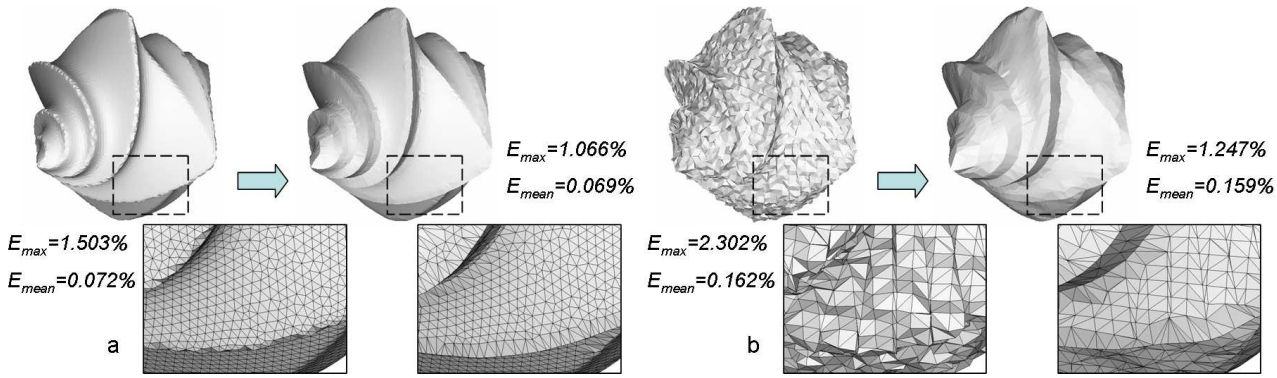

Fig. 1. Bilateral recovering of sharp edges: (a) our approach can successfully reconstruct the sharp edges that are rounded on the given mesh generated by the ShrinkWrap algorithm [36]; (b) we can successfully recover the sharp edges even if Gaussian noise ($\sigma = 1/5$ of the mean edge length) is corrupted, where the noisy-free mesh is generated by a Marching-Cubes algorithm [26]. The models are generated from the distance-field of the mesh model available in [48].

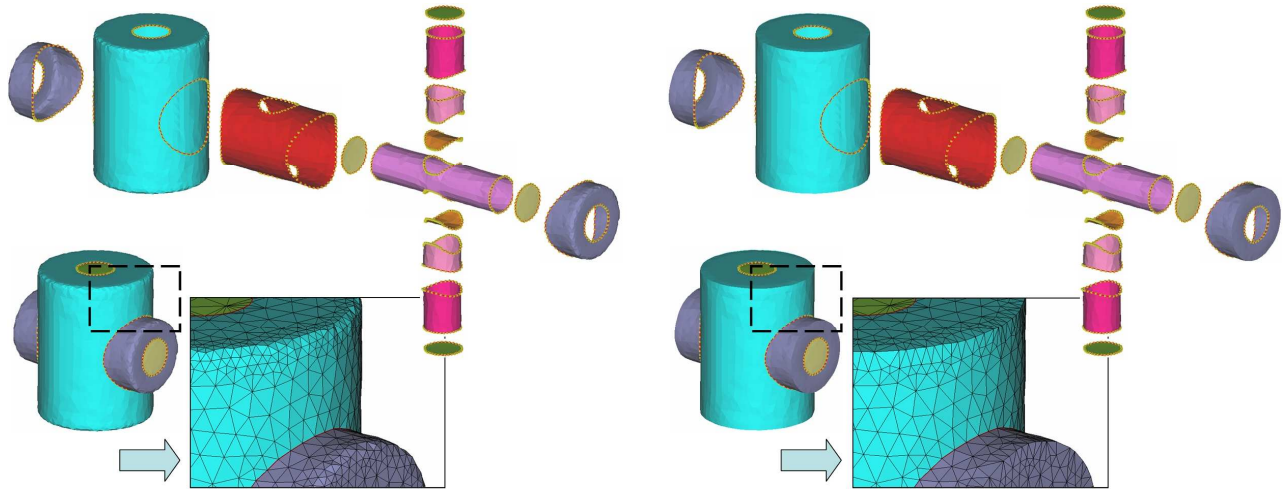

Fig. 2. Recovering sharp edges on a non-manifold model which is assembled from several mesh patches (left), so that on the resultant mesh (right) the sharpness can be easily detected by every edge's dihedral angle.

[22] and [23] have been presented in literature. Both algorithms adopt Hermite data as input. They work well when each cell contains no more than one sharp feature or complex edges (i.e. edges with more than one intersection with a surface). The approach in [43] solved the problem with more than one sharp feature by integrating the adaptive grid generation method and the improved isosurface extraction algorithm. The algorithms presented in [33] and [35] improved the output of MC algorithms based on optimization techniques and smoothing operations. Their approaches are based on mesh evolution towards a given implicit surface with simultaneous control of the mesh vertex position and mesh normals. The above approaches all rely on either the underlying mathematical surface representation (so that the sampling frequency could be infinite) or the Hermite data equipped with surface normals. However as discussed at the beginning of this paper, if the volume or implicit data was not sampled from an existed surface representation, neither Hermite datasets nor ADFs could be given. This leads to the motivation of our research – to reconstruct sharp edges on feature-insensitive sampled triangular meshes without Hermite data.

Anisotropic Mesh Smoothing

In recent years, many mesh smoothing algorithms have been proposed. Some of which are isotropic (see [10], [24], and [40]), which therefore indiscriminately smooth noise and small features (i.e., the sharp edges on a noisy model will become extremely rounded before the model becoming smooth). Thus, techniques for anisotropic mesh smoothing have been developed to solve the problem of feature preservation during fairing (ref. [3], [8], [11], [16], and [29]). The idea behind these approaches is to modify the diffusion equation to make it non-linear or so call anisotropic through curvature tensors. However, by our experimental implementations, it is not easy for the anisotropic diffusion approaches to form singlewide sharp edges as what our approach recovers – especially when noises are corrupted so that curvature tensors are noisy. A crease-enhancing diffusion was introduced in [34] to sharpen creases while maintaining the mesh connectivity. This approach fails on noisy datasets too. All above approaches employed local connectivity to compute geometry properties, which greatly depends on a regular mesh representation. To overcome this limitation, the bilateral filter was recently used to denoise a given surface (ref. [12] and [20]). Also, the authors in [30] conducted the bilateral filter in the reconstruction of surfaces from scattered data. Our sharp edge recovering

method borrows the idea from [20] – using bilateral filters to process geometry on the given mesh.

Another crease-sharpening approach proposed in [39] separated the diffusion in two phases: the normal vectors are first processed to form sharpness, and the positions of vertices on a given mesh are then moved to satisfying the optimized normal vectors. Nevertheless, their processing is conducted on an implicit representation; when converting the surface into an explicit representation, we are still facing the problem of sharp edge reconstruction.

Sharp Feature Detection and Recovering

As our purpose is to recover sharp edges while maintaining the shape of smooth regions on the given mesh, we need to distinguish the sharp and non-sharp regions. Edges are the most important element on a polygonal mesh to represent sharp features. Sharp edges are usually recognized and enhanced from its neighboring connectivity graph. In [19], a method was presented for extracting sharp edges on a multi-resolution organization. Their method is based on the measurement of dihedral angles. When the sharp features are rounded by small radius, a lot of "sharp" edges are detected – so that a thinning process is applied to form patches of the surface. Watanabe and Belyaev in [46] used the identification of perceptually salient curvature extremes to detect curvature features. However, the sharpness geometry is not really reconstructed on edges in above approaches. In contrast, the proposed algorithm here modifies the geometry and connectivity around sharp edges so that they can be easily recognized by their dihedral angle.

Curvature tensors have also been employed in [19] and [29] to detect and recover sharp features. As mentioned above, the curvature tensor based methods have two problems: 1) the support size greatly depends on the local connectivity – so that regular meshes are needed; 2) they are very sensitive to noises. Considering the methods in [19] and [29], if noises are introduced on the sharp edges, it is very difficult to reproduce singlewide sharp edges robustly. This is because that the diffusion process does not make the noisy surface converge to original sharp edges. Benefited from the robustness of bilateral filtering, our recognition and reconstruction algorithms for sharp-features work well on noisy datasets.

Recently, in [2], the authors developed two filters: *Edge-Sharpener* and *Bender* to enhance sharp features at chamfered edges. Unfortunately, small sharp features sometimes may be rounded but not chamfered in the surface reconstruction from a feature-insensitive sampled dataset, which makes their algorithm fail. In this paper, a robust mesh-sharpening algorithm is developed to enhance sharp features on a given model no matter whether they are chamfered or rounded. Also, the non-manifold models are addressed in the same approach. Our proposed approach works well on noisy datasets but [2] works only on the data with chamferred edges from MC algorithms..

The work of [14] is also worthy to mention, where the feature curves and corners are extracted and recovered from point clouds. However, although their method is developed for point clouds, points are assumed to be in the distribution of two-manifold surface like when recovering sharp edges and corners. By this, the methods developed in [14] cannot be directly applied to the non-manifold structures with several patches joining at one edge or corner point. Of course, if we separate a non-manifold object into several two-manifold patches, we can make [14] work on individual patch. Whereas, the collaboration of the recovering procedures taken on several patches is by no means an easy job. The method proposed in this paper provides a general approach, which can easily recover sharp edges and corners on either manifold or non-manifold objects under the same framework. There is no such work found in current literature.

Remeshing

Our algorithm employs remeshing techniques to incrementally remove degenerated triangles formed through bilateral filtering. Similar to the methods in [6], [17], [25], [38], [44], and [47], two remeshing operators: *edge collapse* and *mesh slicing*, are conducted, but with some modification to preserve sharp features and prevent flips of triangles. One may ask why not directly apply a remeshing approach on the result of the bilateral filtering. This is because that we only want to remesh the region degenerated from sharp edges while retaining the rest part so that the volume shrinkage is prevented. Therefore, we need to develop a detector to distinguish the sharp and non-sharp regions, and then recover sharp edges only on the sharp regions. Directly applying remeshing to the bilateral filtering results will have no way to prevent the shrinkage led by the bilateral filtering.

1.2 Contribution

In this paper, the bilateral filtering technique is employed together with the dynamical remeshing operators to recover sharp edges on a given triangular mesh surface. Compared with other existed approaches for similar purposes, our method shows the following advantages:

- A robust general approach is developed for recovering sharp edges on either manifold or non-manifold models from feature insensitive sampling, while previously approaches all focus on two-manifold surfaces or points in two-manifold like distribution. In our approach, sharp edges are reconstructed only from the given mesh surface – no additional normal is requested. The connectivity of a given mesh around the sharp features is optimized in order to let a sharp edge be easily detected by its dihedral angle after the processing.
- With the help of the operators in bilateral filtering manner, a robust sharp feature detector is introduced in this paper, which will be effected by neither the irregular connectivity nor the noisy geometry.
- A novel two-phase recovering algorithm is developed to separate the geometry recovering and the mesh recovering so that the result shows neither flipped nor overlapped triangles.
- Besides, only the mesh around sharp features is reproduced in our approach – the recovering in this means prevents the shrinkage effect by keeping the non-sharp region static.

2 Bilateral Filtering

The bilateral filtering technology will be reviewed in this section. The bilateral filter, which is originally conducted in image processing [37] (see also [42]), is a nonlinear filter derived from Gaussian blur, with a feature preservation term that decreases the weights of pixels as a function of intensity differences. In [37], the bilateral filtering for image $I(p)$ at the pixel p is formulated as

$$E(p) = \sum_{q\in N(p)} W_c(\|p-q\|)W_s(|I(p)-I(q)|)I(q) \Big/ \sum_{q\in N(p)} W_c(\|p-q\|)W_s(|I(p)-I(q)|) \quad (1)$$

where $N(p)$ is the neighborhood of p,

$$W_c(t) = e^{-t^2/2\sigma_c^2} \quad (2)$$

is the standard Gaussian filter with parameter σ_c, and

$$W_s(t) = e^{-t^2/2\sigma_s^2} \quad (3)$$

is a similarity weight function for feature-preserving with parameter σ_s that penalizes large variation in intensity.

The authors in [12] and [20] recently applied the Bilateral filter to smooth and denoise 3D meshes while preserving sharp features. Different methods were conducted in [12] and [20] to capture surface smoothness, Jones et al. [20] used the normal on polygons while Fleishman et al. [12] adopted vertex normals. For the neighborhood of a vertex v, [20] considered all triangles with the distance from v to their center less than a threshold, while only the 1-ring adjacent vertices were utilized in [12]. Our proposed approach also conduct the bilateral filter to drift vertices, with which the sharpness could be elegantly recovered. The predictor in our approach follows the manner of [20] – adopting the normals on triangles and the centers of triangles. Similar to [20], the bilateral filter is weighted by the area of triangles to account for variations in the sampling rate of the surface. Using $N(v)$ to denote the set of triangles contributing to the position of a vertex v, and $\hat{n}(f)$ to represent the unit normal of f, the formula of a bilateral filter for v is

$$v' = \frac{1}{k(v)} \sum_{f\in N(v)} p_f(v) a_f W_c(\|c(f)-v\|)W_s(\|p_f(v)-v\|) \quad (4)$$

where a_f and $c(f)$ are the area and center of f, $p_f(v)$ is the projection point of v on the plane of f served as a predictor in the bilateral filter

$$p_f(v) = v + < c(f)-v, \hat{n}(f) > \hat{n}(f), \quad (5)$$

$k(v)$ is a normalization factor

$$k(v) = \sum_{f\in N(v)} a_f W_c(\|c(f)-v\|)W_s(\|p_f(v)-v\|), \quad (6)$$

and $W_c(t)$ and $W_s(t)$ are as given in Eq. (2) and (3).

Simply applying the bilateral filter to the vertices on a given model in an iterative way may break the regularity of its mesh. Thus, Jones et al. in [20] only moved vertices once plus a previous mollification. However, through our tests, we find that only one run cannot produce sharpness. Repeated filtering is needed. In our approach, the bilateral filter is integrated with remeshing operators to iteratively recover sharp edges.

2.1 Bilateral Filtering on Non-manifold Models

In this section, we address some modifications on the bilateral filter to let it work for non-manifold models. A non-manifold object $\mathbf{M}$ is usually stored as a collection of two-manifold mesh patches

$$\mathbf{M} = \bigcup_{i=0}^{n} \mathbf{M}_i, \quad (7)$$

where each mesh surface patch $\mathbf{M}_i$ is defined as a pair (K,V), with K as a simplicial complex specifying the connectivity of vertices, edges, and faces (i.e., the topological graph of $\mathbf{M}_i$), and $V = \{v_1, \cdots, v_m\}$ as the set of vertices defining the shape of a polyhedral patch in $\Re^3$. The continuities between patches are preserved by storing linkers on boundary vertices. The vertices belonging to different patches sharing the same position are called *common vertices*, whose positions should be kept consistent when processing meshes (see Fig. 2 – the small yellow cubes show common vertices). Manifold object could be represented in this framework either, but with only one patch.

For a vertex $v \in \mathbf{M}_i \subset \mathbf{M}$ on a given model $\mathbf{M}$, if $v \in \mathbf{M}_i$ is an inner vertex, we do not want its position to be effected by the triangles $f \notin \mathbf{M}_i$, so its contributing face set $N(v)$ includes only triangles $f \in \mathbf{M}_i$ with the distance from f to v less than $2\sigma_c$, which is also called the support size of a bilateral filter. For a boundary vertex v_b, as its position is jointly determined by several patches around it, triangles on all these patches are included in $N(v_b)$.

3 Bilateral Sharp Feature Recognition

The region degenerated from sharp features during an insensitive sampling is determined through identifying vertices with great surface normal variation. With the help of the robust bilateral filter, our algorithm can successfully recognize sharp feature regions on noisy datasets in four steps:

- Bilateral mollification;
- Sharp-vertices detection;
- Isolating sharp-vertices through detecting sharp-region-edges and sharp-region-faces;
- Restoring each vertex to its original position before bilateral mollification.

For a vertex $v \in \mathbf{M}_i \subset \mathbf{M}$, let $F(v)$ contain all triangles neighboring to v and $\hat{n}(f)$ give the unit normal of f, we can conduct the following formula to compute the smoothness at v:

$$\tau(v) = \inf\{< \hat{n}(f_i), \hat{n}(f_j) >\} \quad \forall f_i, f_j \in F(v). \quad (8)$$

Note that, to ensure the stability of evaluating $\hat{n}(f)$ in $\tau(v)$, those degenerated triangles (i.e., those with extremely small or large angles) are excluded from $F(v)$. When $\tau(v) < \cos\theta$ (θ is a user-defined angle for sharpness criterion), we then treat v as a candidate vertex on sharp features – called *sharp-vertex*. Our sharp-vertex detection method borrows some idea from what Smith and Brady used to detect sharp edges on an image in [37]. Since the bilateral filter lately applied to sharpen a mesh also depends on the normals of triangles around sharp-vertices, the detection and recovering methods here are consistent.

In order to make our detector robust to the irregularity on given meshes, we give up adopting local connectivity but use the Euclidean distance to identify contributed sur-

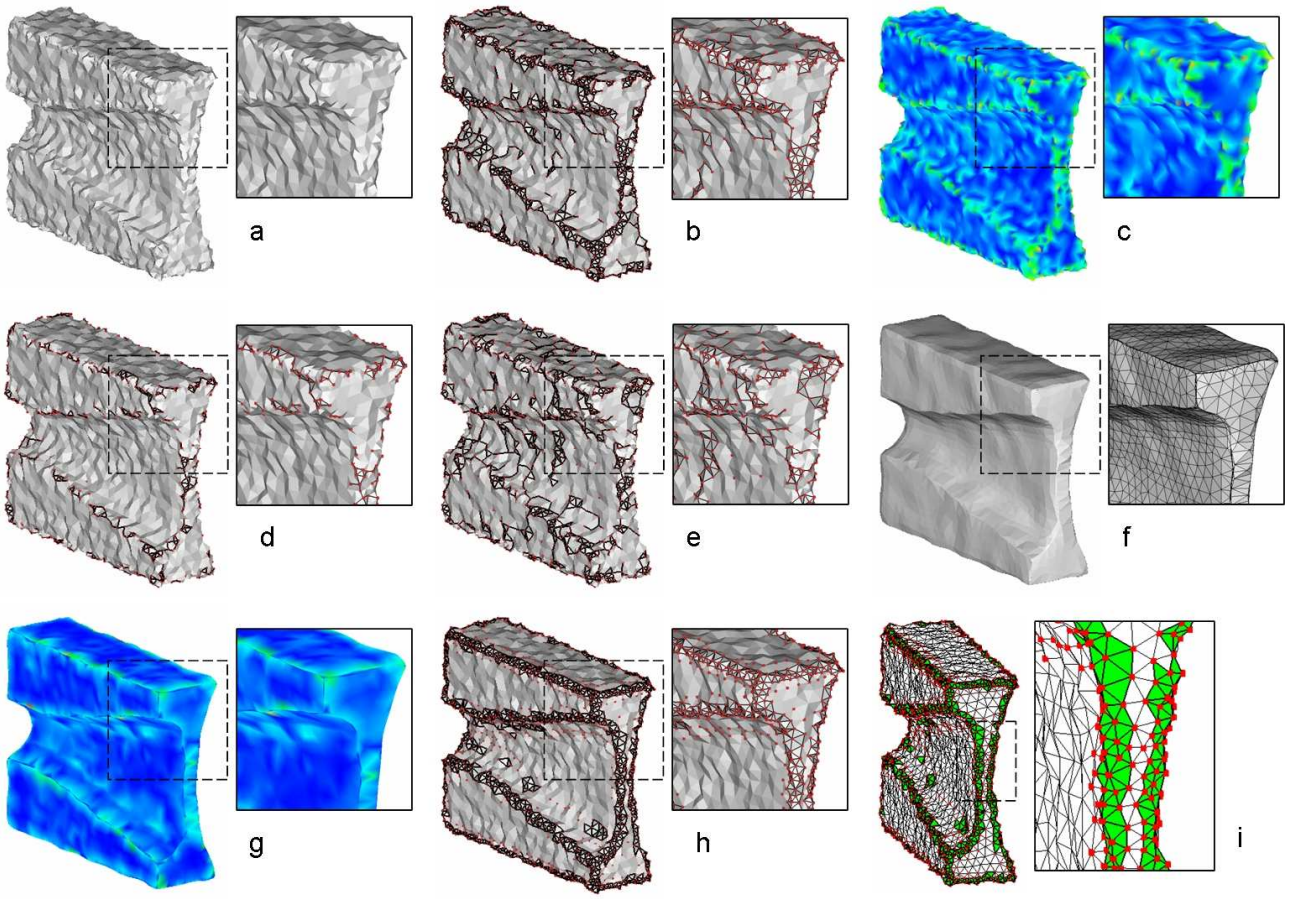

Fig. 3. Comparison of different sharp-feature detection methods on a model corrupted with Gaussian noisy (a) (with $\sigma = 0.2\|e\|$, where $\|e\|$ is the mean edge length). (b) gives the sharp feature detection result from the dihedral angle based method [2] (with the dihedral angle threshold $\theta = \cos^{-1} 0.75$). (c) shows the color map of its mean curvature. (d, e) show the curvature tensors based detection results from [29] (with the curvature threshold $1/\|e\|$ and $0.5/\|e\|$). After 30 passes of the bilateral filtering, the noisy model becomes (f), whose colorful mean curvature map is as (g). The result of our bilateral sharp region recognition is illustrated in (h) (with $\sigma_c = 1.5\|e\|$ and $\cos\theta = 0.75$). Red cubes denote the *sharp-vertices*, while the *sharp-region-edges* are represented with black line segments. (i) *sharp-region-faces* are colored in green on the result of our bilateral sharp feature recognition.

face normals at v. The definition of $F(v)$ is similar to $N(v)$ in Eq.(4) but with the support size reduced by half – i.e., σ_c instead of $2\sigma_c$. To efficiently detect the triangles with their distance to a vertex less than σ_c, we conduct the space subdivision technique to reduce the searching range. The space bounding M is divided into $L \times M \times N$ voxels with the width σ_c. A triangle face f is considered as contributing to a voxel if its bounding box has some overlap with the voxel. By this space subdivision, for the vertex falling in a voxel, its neighboring triangles are only searched among the triangles contributing to this voxel and its 26 neighbors.

Considering about noisy datasets, a mollification process is introduced to solve the normal variation problem. We perform several passes of bilateral filtering (Eq. (4-6)) on a noisy model M before using Eq. (8) to measure the smoothness at each vertex on M. This is called bilateral mollification.

Only detecting sharp-feature vertices is insufficient. Some sharp-vertices belonging to different sharp features may be directly linked by edges if the features are very close to each other. For example, the sharp-vertices shown in the zoom-view of Fig. 3i should be classified into two groups. Here, we conduct the following filtering steps to detect *sharp-region-edges* and *sharp-region-faces*, by which the sharp-vertices on different features are isolated. The filtering steps are:

- Set each triangle with *three* sharp-vertices as a *pseudo-sharp-triangle* (i.e., not *true*), while every edge of a *pseudo-sharp-triangle* is considered as a *pseudo-sharp-edge*;
- For each *pseudo-sharp-triangle*, if its area is increased after the bilateral mollification, its three edges will be marked as *non-sharp*;
- Each *pseudo-sharp-edge* not being marked as *non-sharp* is classified as a *sharp-region-edge*;
- Each triangle with three *sharp-region-edges* is assigned as a *sharp-region-triangle*.

The second step comes from the observation that the triangles with its vertices belong to different sharp features are usually enlarged after the bilateral mollification. Even if the given model is noisy-free, it is also important to apply the bilateral mollification step to isolate sharp-vertices belonging to different features.

Fig. 3 shows a comparison of the sharp feature recognition results from the dihedral angle based method [2], the curvature tensor based method [29], and our approach. All are on a model corrupted with Gaussian noisy ($\sigma = 1/5$ of the mean edge length). Fig. 3c gives the color map of mean

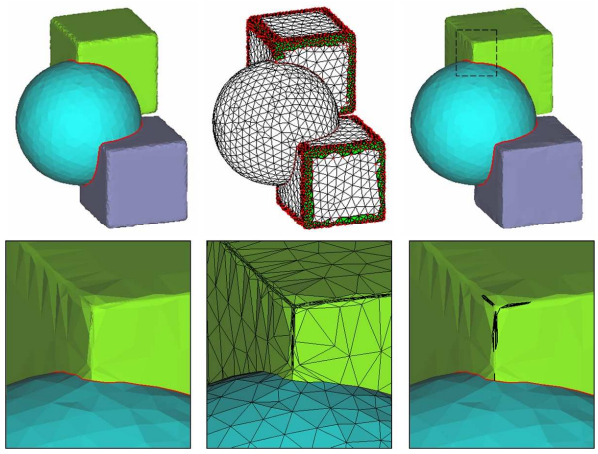

Fig. 4. Result of geometry recovering. For the given non-manifold model (top-left), after the sharp regions have been recognized (top-middle), the geometry of sharpness can be reconstructed by several passes of bilateral filtering (top-right). However, the regularity of given mesh is broken (bottom-left and bottom-middle) and a sharp edge is difficult to be detected by its dihedral angle (see bottom-right – very few edge's dihedral angle is greater than the threshold, the ones greater are specified by black lines).

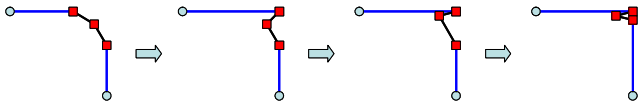

Fig. 5. In some extreme case, the normal of a triangle may be flipped in a bilateral filtering – the figures give the cross-sections of a model under bilateral sharpening where the red nodes are pseudo-sharp vertices.

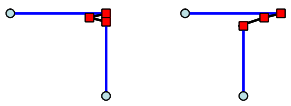

Fig. 6. It is difficult to distinguish the flip singularity (left – the cross-section illustration of overlapped triangles) from an extreme sharp feature (right – the cross-section presentation of an extreme sharp feature).

curvatures on the surface, where red represents the maximal value and blue denotes the minimum. It is not difficult to find that both the dihedral angle based method and the curvature tensor base scheme fail on the noisy model (see Fig. 3b, 3d, and 3e). After thirty passes of bilateral filtering, the noisy model becomes as shown in Fig. 3f. We then measure the sharp feature vertices by Eq. (8) with threshold $\sigma_c = 1.5\|e\|$ and $\cos\theta = 0.75$. Finally, the position of each vertex on the model is restored (see Fig.3h and Fig. 3i). Our result is much better than the results from other two methods, and adjacent sharp features are successfully isolated (i.e., neither sharp-region-edges nor sharp-region-faces link them – see Fig. 3i).

One may ask why not apply the other two methods on the model after bilateral mollification to detect sharp features. This is because that, as shown in the mesh representation on Fig. 3f, the bilateral filtering result usually drifts a lot of vertices and edges around the "sharp" region. Caused by the degenerate triangles, the curvature tensors and the dihedral angles cannot be stably evaluated. For instance, Fig. 3g shows the color map of mean curvatures on the model of Fig. 3f, where extreme curvatures do not always occur on the "sharp" regions. For our method in Eq.(8), the stability is guaranteed since those skinny triangles have been excluded from the detector.

4 Bilateral Sharp Edge Recovering

The algorithm for recovering sharp edges is addressed in this section. First of all, the positions of all *sharp-vertices* are optimized by several passes of bilateral filtering – this is called geometry recovering; then, the sharpened geometry shape is encoded into Hermite data units to govern the shape and connectivity modification on the given mesh – this is named as mesh recovering.

4.1 Geometry Recovering

The bilateral filter (ref. Eq. (4-6)) is applied on all *sharp-vertices* to form the geometry of sharpness. As mentioned in [12], directly applying the bilateral filer for image processing to a 3D surface has three problems: 1) irregularity; 2) shrinkage; 3) drifting. For the irregularity, we follow the manner of Jones et al. [20] to overcome this issue through weighting the contributed triangles by their areas. As the bilateral filter is not volume preserved, shrinkage will occur on the given mesh. To avoid that, only the sharp-vertices are filtered in our algorithm, and the other static non-sharp-vertices will take the role as a keel to prevent shrinking. For the problem of vertex drifting, Fleishman et al. in [12] avoided this problem by moving vertices only along their normals. However, this will also prevent the generation of sharp features. Jones et al. in [20] only moved vertices once plus a previous mollification – but this will not produce sharpness too. As repeated filtering is needed, we solve the vertex drifting problem by separating the geometry and the mesh recovering of sharp edges.

One bottleneck of applying the bilateral filter to sharpen a three-dimensional surface is how to efficiently determine $N(v)$ on every sharp-vertex. We also conduct the space partition method presented in above section to reduce the computational cost. This time, the voxel size is chosen as $2\sigma_c$.

The result after directly applying the bilateral filter to sharp-vertices does not really produce sharp edges. The filter only changes the position of vertices but do not modify the connectivity of meshes. As shown in Fig. 4, a lot of edges are drifted and crowded at the region where expected to have singlewide sharp edges. Because of the vertex drifting, the mesh becomes irregular around sharpened region, so many degraded triangles appears. In some extreme cases, the normals of some triangles may be flipped as illustrated in Fig. 5. Not only irregularity but also the sharpness detection problem occurs – since many thin and small triangles crowd together, the dihedral angles indicating sharpness cannot be stably evaluated. See the bottom right figure in Fig. 4, only a small number of edges can pass the sharpness detection of dihedral angles. The only way to solve this problem is to optimize the connectivity of meshes on the given model during the sharp edge recovering, which leads to the so called mesh recovering algorithm pre-

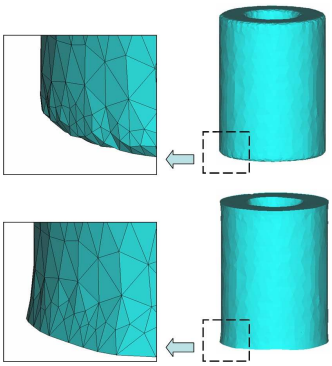

Fig. 7. The given model (top) may be over sharpened (bottom) if we optimize its connectivity during bilateral filtering.

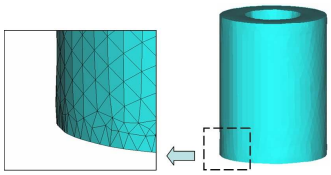

Fig. 8. The over-sharpening problem is solved after separating the geometry recovering and the mesh recovering.

sented in the section below.

4.2 Mesh Recovering

Although directly applying bilateral filters on the sharp-vertices does not give us an expected mesh surface, it at least provides a good profile of the final surface with sharp features. What currently needed is to modify connectivity so to improve the regularity of meshes and the sharpness at pseudo-sharp edges while still maintaining the shape generated by the iterations of bilateral filtering.

If we directly optimize the connectivity on the meshes after geometry recovering (e.g., adopting the method in [6]), we will face several difficulties: Firstly, the edges representing sharp features need to be detected and preserved – but as mentioned above, they are hardly identified among the edges crowded at the sharp regions. Secondly, it is a tough job to eliminate flipped triangles since this kind of modification involves local geometry changes, which are difficult to preserve the original shape produced by bilateral filtering. Also, it is not easy to identify whether a triangle is flipped; in other words, we cannot distinguish the flipped triangles from the triangles forming extreme sharp features (e.g., as illustrated in Fig. 6). Lastly, even if we conduct the method presented in section 3 to detect sharp-region-edges and preserve them during mesh optimization, we can hardly retain sharp corners since they cannot be recognized.

What happens if the mesh connectivity is optimized during the procedure of geometry recovering? From our tests, we find that this may generate over-sharpened results, i.e., the sharpness is over-enhanced. This is because that after the shape and normal of a triangle is modified by a remeshing operator, we lost its original position and orientation to serve as a predictor in later bilateral filtering steps. Especially when some triangles are removed, the accuracy of predictors is decreased since the predictors of a bilateral filter is based on a statistical error norm (ref. [20]). An example result of over-sharpening on a cylinder is shown in Fig. 7.

Our method to overcome all above difficulties is by encoding the result of geometry recovering into a Hermite data collection. Then, this Hermite data collection is employed to govern the mesh recovering algorithm which moves the sharp-vertices while optimizes the connectivity on sharp-region-triangles. With the help of this separation of geometry recovering and mesh recovering, the over-sharpening problem is solved (e.g., the result in Fig. 8).

The position and orientation of a triangle f after geometry recovering is encoded as a vector consists of the center position $c(f)$, the area a_f, the unit normal $\hat{n}(f)$ of f, which is called a Hermite data unit of f. After encoding Hermite data unit of every triangle, the positions of sharp-vertices on the model M are restored to its original location. Then, three operators are iteratively applied to modify both the geometry and the connectivity in the region to be sharpened, which include one geometric operator – *VertexSharpener* and two topological operators – *MeshSlicing* and *EdgeCollapse*.

VertexSharpener – the purpose of this geometric operator is to move every sharp-vertex on $\mathbf{M}$ to a new position predicted by the background Hermite data units. For a sharp-vertex $v \in \mathrm{M}_i \subset \mathrm{M}$, we use Eq. (4) – the bilateral filter to determine its new position v'. However, the triangles in Eq. (4) are replaced by the previously stored Hermite data units. All Hermite data units with their distance to v less than $2\sigma_c$ are considered. Since the area of each triangle is record and contributes to v', the skinny triangles have less influence on the result.

The *VertexSharpener* updates the positions of sharp-vertices for several runs. In each pass, whether the new position v' will flip the normals of triangles around v is detected. If normal flip occurs, we adopt the binary search method to determine a closest position v^* to v' on vv' without normal flip, and move the vertex to v^*. For a very noisy given model, several passes of Laplacian smoothing [40] are applied on the sharp-vertices after encoding the Hermite data units – so that the regularity of meshes are enhanced before applying the *VertexSharpener* operator. Note that to avoid the vibration of face normals caused by the dynamic remeshing, the normal vector of each triangle is recorded and will not be updated during the whole mesh recovering process.

MeshSlicing – this operator together with the following *EdgeCollapse* operator are conducted to remove extreme thin triangles generated by *VertexSharpener*. As mentioned in [41] and [6], extreme thin triangle can be classified as

- Needles: triangles, whose longest edge is much longer than the shortest one;
- Caps: triangles with an angle close to π.

The *MeshSlicing* is from the work of [6] to convert caps into needles, but the slicing here is only applied to the sharp-triangles. Note that the recorded normal of each sharp-triangle must be copied to the newly created triangles to make the flip prevention work.

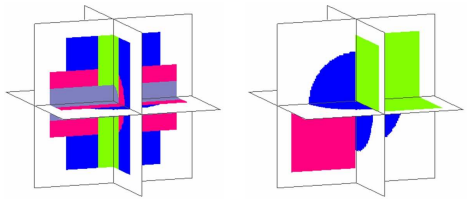

Fig. 9. Model II (left) and III (right) are constructed from implicit models, where different colors represent that different material stuffs are filled – so heterogeneous models are given.

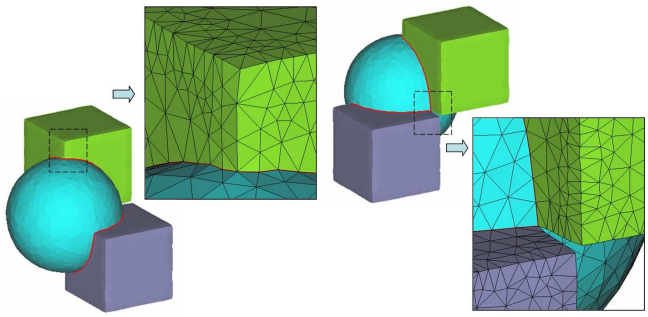

Fig. 10. The bilateral recovering result of sharp edges on Model III.

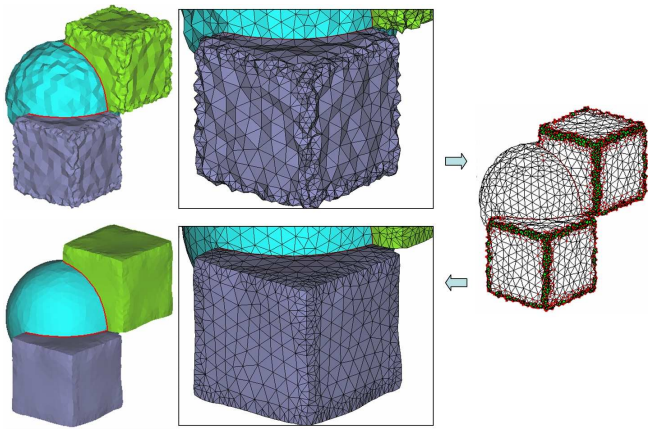

Fig. 11. After corrupting Gaussian noisy ($\sigma = 1/5$ of the mean edge length) on Model III, the sharp region and edges can still be successfully recognized and reconstructed.

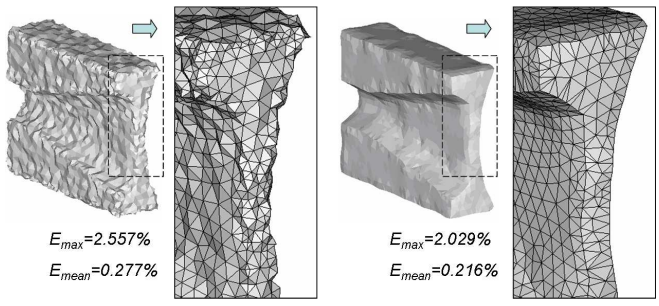

Fig. 12. The recovering result of Model IV – a forming plate corrupted with Gaussian noises.

EdgeCollapse – this operator is employed to remove extreme thin triangles. The *EdgeCollapse* actually removes needles among all sharp-region triangles. Considering about an edge $e \in \mathrm{M}_i \subset \mathrm{M}$, if it satisfies all the following conditions, the edge will be collapsed:

- e is a sharp-region-edge;
- the length of e not greater than $\alpha\sigma_c$;
- if the positions of its two endpoints are moved to their middle point, no normal of their adjacent triangles will be flipped;
- the collapse will not lead to topology degradation (ref. [18]).

When applying the *EdgeCollapse* operator to a mesh surface, we insert all qualified candidate edges into a minimal heap which sorts edge lengths. Then, the edges are iteratively removed from top of the heap and collapsed until the length of top edge in the heap is greater than $\alpha\sigma_c$. In all our experimental tests, we choose $\alpha = 0.4$.

4.3 Algorithm Overview

The algorithm for mesh recovering applies above operators iteratively on the given model so that sharp edges are reconstructed successfully. The overall algorithm is summarized as below.

- Apply the bilateral filter to sharp-vertices to construct the geometry sharpness and encode the deformed triangles in a Hermite data collection;
- Restore the position of each sharp-vertex;
- Relax the connectivity on sharp-vertices – by the Laplacian operator;
- Repeat that:
  - Apply *VertexSharpener* on M for ten runs;
  - Apply *MeshSlicing* on M to convert caps into needles;
  - Apply *EdgeCollapse* on M to eliminate needles;
- Using *VertexSharpener* to finalize the sharpened geometry on M for two runs.

The vertex relaxation at the beginning of recovering algorithm plus later flip-prevented *VertexSharpener* operator can avoid the normal flips (e.g., the case shown in Fig. 5). Even if the normal of a triangle has been flipped in the Hermite data units, it does not matter. As long as the Hermite data units give correct shape after sharpening, the orientation of a triangle on the final sharpened mesh depends only on its normal after vertex relaxation, so the normal flip is absolutely prevented. For the loop in the mesh recovering algorithm, usually only three to five runs give excellent sharpening results.

Note that if a noisy dataset is given, one pass of bilateral filtering is finally applied to all the non-sharp-vertices (not including the sharp-vertices) on the resultant model to further smooth its geometry, which serves as a *post-filtering*. Since the operations on sharp and non-sharp regions are separately conducted, the non-sharp region will serve as support to prevent the volume shrinkage when processing the sharp region; vice versa.

5 Result and Discussion

Our bilateral recovering algorithm has been tested on a range of noisy and noisy-free models. The examples are constructed either directly from a volume model or through a surface-volume-remeshing conversation routine. In all the examples tested, when the original shape was sampled

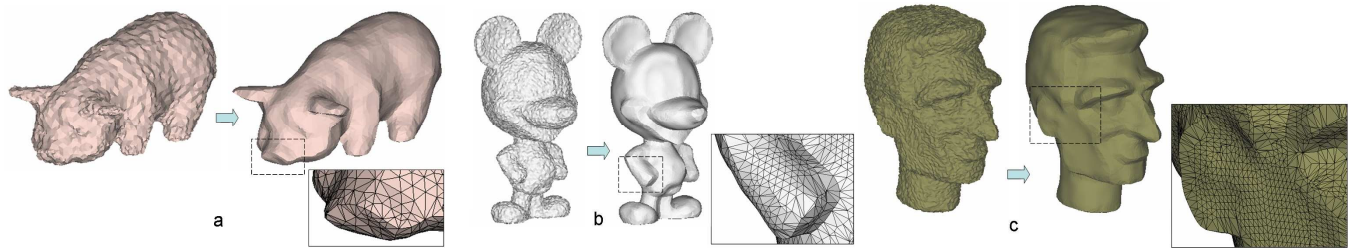

Fig. 13. The recovering results of complex noisy models.

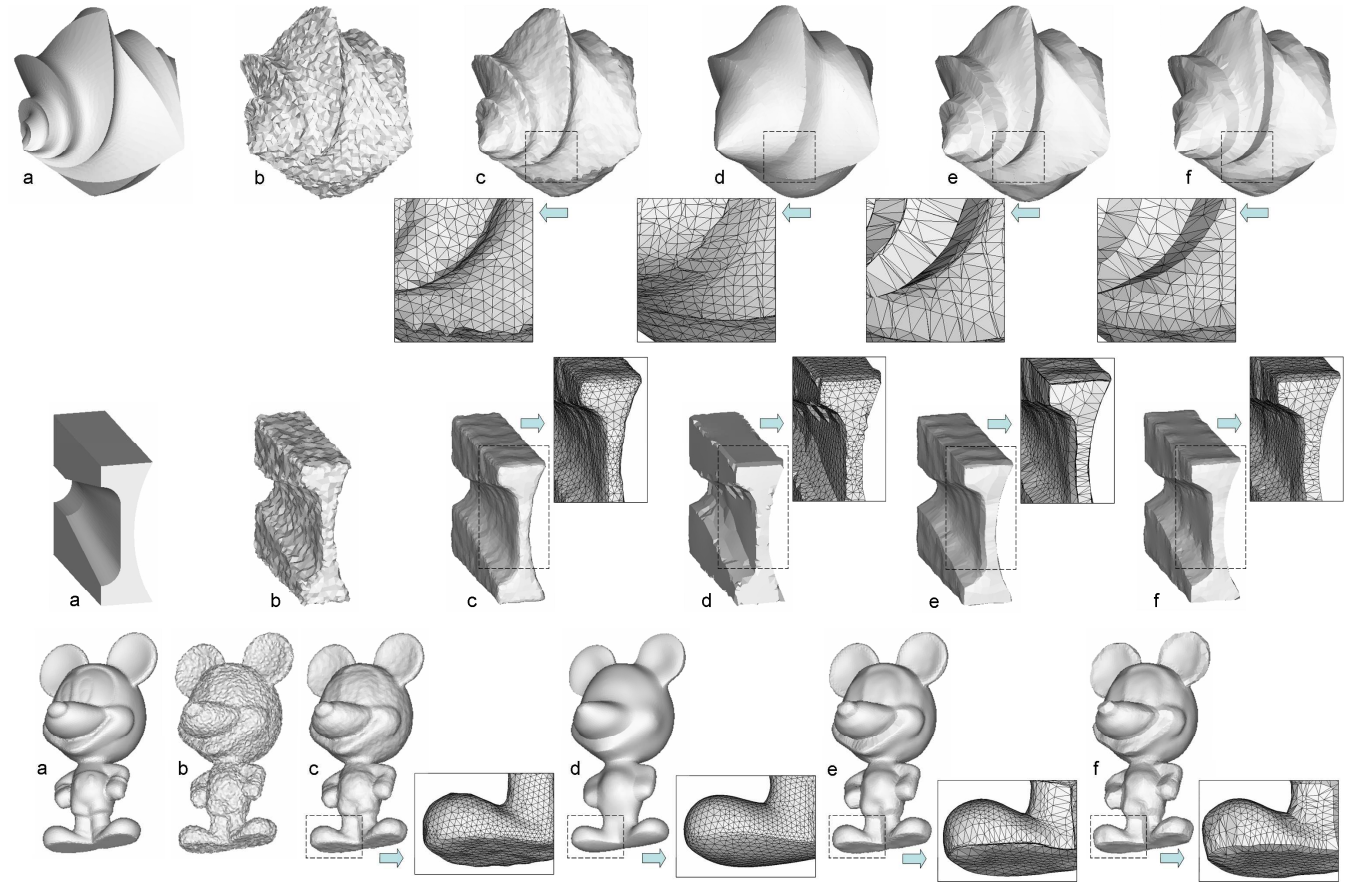

Fig. 14. For given models in (a), after Gaussian noises are corrupted (see (b)), the sharp edge recovering results by the anisotropic diffusion [29] (c), the crease-enhancing diffusion [34] (d), 30 passes of bilateral filtering [20] (e), and ours (f) are compared.

with a sufficiently high rate, most of the sharp features can be successfully recovered. Also, we have observed that our algorithm significantly reduces the shape errors (both the Hausdorff distance E_{max} and the mean distortion E_{mean}) by recovering sharp edges – the errors are measured through the publicly available Metro tool [7].

Our first example is a freeform flower model that has been shown in Fig. 1. Both noisy-free and noisy datasets have been tested. The second example is a non-manifold model, which has been previously shown in Fig.2. The model is directly constructed from an implicit representation (see the left of Fig. 9). The third testing example is also non-manifold (see the right of Fig. 9 and Fig. 10); while comparing to the result after geometry recovering in Fig. 4, the sharp edges are easily detected and preserved by dihedral angles on the resultant model in Fig. 10. Fig. 11 demonstrates the functionality of our approach on the third model corrupted with Gaussian noises ($\sigma = 1/5$ of the mean edge length). The fourth model is a mechanical model – forming plate, which has previously shown in Fig. 3 also with Gaussian noises. The recovering result is given in Fig. 12. The method proposed in this paper also works well on complex models (e.g., the models in Fig.13).

In Fig. 14, the processing results from the anisotropic diffusion [29], the crease-enhancing diffusion [34], the bilateral filtering (with 30 passes), and our approach are compared on two noisy models ($\sigma = 1/5$ of the mean edge length). By the anisotropic diffusion, when a noisy model is given, it is difficult to correct the mean curvatures to let the diffusion process converge to sharp edges. The same problem happens on the crease-enhancing diffusion – effected by noises, some sharp features are incorrectly chamfered or blurred. Iteratively applying bilateral filters will make many edges crowded at the sharp region. The results from our method show the best performance in recovering sharp edges while preserving the detail geometry. In term of computing time,

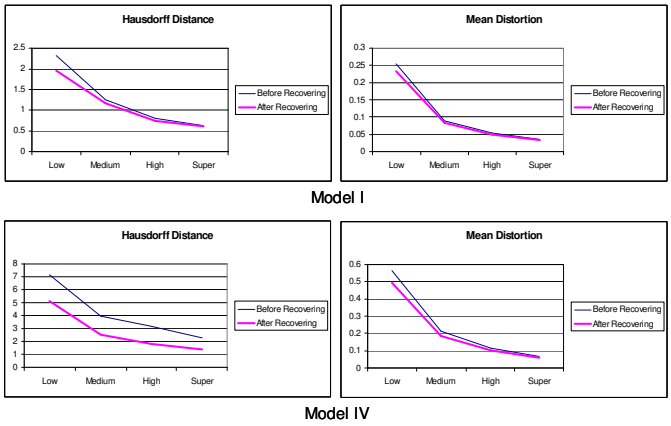

Fig. 15. With the increases of sampling rate, the model processed by our approach gives less error; even if a very dense sampling rate is applied to remesh a model, our algorithm can still reduce some aliasing error. In this test, four different sampling densities are conducted to convert the original model into a volume representation, and then is reconstructed into a mesh model: low – sampled at 64x64x64; medium – the model is sampled by a 128x128x128 grid; high – 192x192x192; and in super high resolution – 256x256x256 samples are conducted.

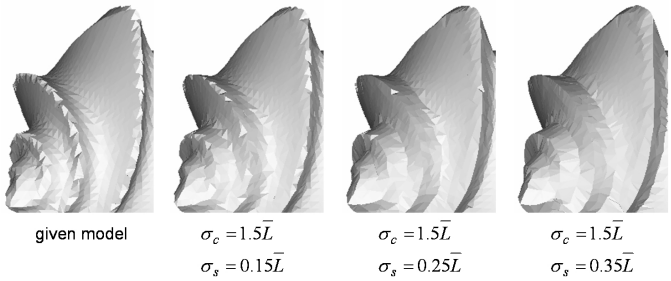

Fig. 16. The effects of increasing σ_s with a fixed σ_c.

our approach approximately takes the same time as about 500 passes of anisotropic diffusion or 700 iterations of crease-enhancing diffusion. The detail computing time on example models is listed in Table 1.

Another interesting investigation concerns the evaluation of the error reduction on a given model sampled with different densities. Theoretically, our sharp feature recovering algorithm should somewhat increase the accuracy of an feature-insensitive sampled model no matter in which rate it is sampled; also, with the increase of sampling rate, the model produced by our approach is expected to give less error. Our tests prove these two opinions. Fig. 15 shows our results on model I and IV – both are tested in four different sampling densities.

5.1 Parameters

When using our approach to reconstruct sharp features on a given model, two parameters need to be chosen: σ_c and σ_s. The first parameter σ_c actually determines the support size of our filter and the sharp feature detector. It could be defined in an interactive manner: let the user select a point of mesh where the surface is expected to be sharp, and then the support size is iteratively increased until the normal variation of triangles falling in the support greater than the threshold (we usually choose 0.75). With a fixed σ_c, the value of parameter σ_s indicates the ability to

TABLE 1
COMPUTATIONAL STATISTICS

Model	Figure	Triangles	$\sigma_s / \|e\|$	$\sigma_c / \|e\|$	Time (s)
I – noisy free	1a	25.1k	1.5	0.5	99.2
I – noisy	1b	12.8k	1.5	0.5	79.3
II	2	8.3k	1.1	0.25	21.9
III – noisy free	4 & 10	7.2k	1.1	0.5	25.7
III – noisy	11		2.0	0.5	55.5
IV – noisy	3 & 12	6.5k	1.5	0.5	30.4
V – pig	13a	9.8k	1.5	0.25	46.8
VI – Mickey	13b	30.3k	2.0	0.5	189.7
VII – head	13c	34.7k	2.0	0.5	238.5

*All computations are performed on a standard PC with PIV 2.6GHz CPU + 512MB RAM.

form sharp features, which is determined in a trial-and-error manner. We usually start from $\sigma_s = 0.1\sigma_c$; if some sharp edges are missed on the result, we then increase σ_s until sharp edges and corners are well recovered. The optimal parameters used in our examples are listed in Table 1 – all are measured relative to the mean edge length. Fig. 16 shows an example effect of increasing σ_s with a fixed σ_c.

The feature missing problem reported in the approach of [2] could be well solved since we can adjust the coefficient σ_c to increase the support size of bilateral predictor – by which the missed features can be recovered.

5.2 Limitations

Similar to the sharpening algorithm presented in [2], the feature that blends smoothly into a flat area may be miss-sharpened if the sampling rate is not able to generate enough great normal variations around aliasing regions. Also, some unwanted sharpening will be given on small radius blends. These two problems are both effected by an insufficient sampling rate – so that they can in some measure be solved if a denser sampling is provided. For the problem of unwanted sharpening, it can also be overcome by interactively introducing the intension of users about where to sharpen and where not.

In this paper, the nearness is determined through Euclidean distances. As mention in [12], Geodesic distance is more appropriate to determine the nearness of a bilateral filter – however, the computing of Geodesic distance is much more expensive. We need to seek a balance between the quality and the efficiency. In our current implementation, we adopt Euclidean distance based detector and predictor, which give good results at most examples.

6 CONCLUSION

We have presented a robust general approach with the assistant of bilateral filters to recover sharp edges on feature-insensitive sampled triangular meshes, which interpolate or approximate the shape sampled on regular grids. Our approach has two phases: bilateral sharp feature recognition and bilateral sharp edge recovering. The experimental re-

sults prove that our method works robust on various models. Sharp edges are constructed only from given mesh surface, i.e., no additional normal is requested. The shape quality of elements around the sharp edges is controlled while preserving the sharpness. Sharp edges on a non-manifold structure can also be well recovered. After applying our recovering approach on insensitive sampled meshes, the sharp edges are reconstructed and can be easily detected through dihedral angles.

REFERENCES

[1] V. Adzhiev, E. Kartasheva, T. Kunii, A. Pasko, and B. Schmitt, "Cellular-functional modeling of heterogeneous objects," *Proceedings of the seventh ACM symposium on Solid modeling and applications*, pp.192-203, Saarbrücken, Germany; 2002.

[2] M. Attene, B. Falcidino, M. Spagnuolo, and J. Rossignac , "Sharpen&Bend: Recovering curved edges in triangle meshes produced by feature-insensitive sampling," *IEEE Trans. on Visualization and Computer Graphics*, vol.11, no.2, pp.181-192, 2005.

[3] C. L. Bajaj and G. Xu, "Anisotropic diffusion of surfaces and functions on surfaces," *ACM Trans. on Graphics*, vol.22, no.1, pp.4-32, 2003.

[4] A. Biswas, V. Shapiro, and I. Tsukanov, "Heterogeneous material modelin g with distance fields," *Computer-Aided Geometric Design*, vol.21, pp. 215-24 2, 2004.

[5] J. Bloomenthal, "An implicit surface polygonizer," *Graphics Gems IV*, P. S. Heckbert, eds., Published: AP Professional (Academic Press), Boston, pp.324-349, 1994.

[6] M. Botsch and L. P. Kobbelt, "A robust procedure to eliminate degenerate faces from triangle meshes," *Proceedings of the Vision Modeling and Visualization Conference 2001*, pp.283-290, 2001.

[7] P. Cignoni, C. Rocchini, R. Scopigno, "Metro: measuring error on simplified surfaces," *Computer Graphics Forum*, vol.17, no.2, June 1998, pp.167-74.

[8] A. U. Clarenz, U. Diewald, and M. Rumpf, "Anisotropic geometric diffusion in surface processing," *Proceedings of IEEE Visualization 2000*, pp.397-405.

[9] D. Cohen-Or, D. Levin, and A. Solomovici, "Three-dimensional distance field metamorphosis," *ACM Trans. on Graphics*, vol.17, no.2, April 1998, pp.116-141.

[10] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," *Proceedings of SIGGRAPH 99*, pp.317-324, 1999.

[11] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Anisotropic feature-preserving denoising of height fields and bivariate data," Graphics Interface, pp.145-152, 2000.

[12] S. Fleishman, I. Drori, D. Cohen-Or, "Bilateral mesh denoising," *ACM Trans. on Graphics*, vol.22, no.3, pp. 950-953, 2003.

[13] S. Frisken, R. Perry, A. Rockwood, and R. Jones, "Adaptively sampled distance field: a general representation of shapes for computer graphics," *Proceedings of ACM SIGGRAPH 2000*, 2000, pp.249-254.

[14] S. Gumhold, X. Wang, and R. McLeod, "Feature extraction from point clouds," *Proceedings of 10th International Meshing Roundtable*, pp.293-305, 2001.

[15] T. He, L. Hong, A. Kaufman, A. Varshney, and S. Wang, "Voxel based object simplification," *IEEE Visualization'95 Proc.*, pp.296-303.

[16] K. Hildebrandt and K. Polthier, "Anisotropic filtering of non-linear surface features", *Computer Graphics Forum*, vol.23, no.3, 2004.

[17] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," *Proceedings of SIGGRAPH 93*, pp.19-26.

[18] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," *Extended_TR_UW_CSE_1993-01-01*, http://www.research.microsoft.com/~hhoppe.

[19] A. Hubeli and M. Gross, "Multiresolution feature extraction for unstructured meshes," Proceedings of Visualization '01, pp.16-25, 2001.

[20] T. R. Jones, F. Durand, and M. Desbrun, "Non-iterative, feature-preserving mesh smoothing," *ACM Trans. on Graphics*, vol.22, no.3, pp. 943-949, 2003.

[21] T. Ju, "Robust repair of polygonal models," *ACM Trans. on Graphics*, vol.23, no.3, Aug. 2004, pp.888-895.

[22] T. Ju, F. Losasso, S. Schaefer, and J. Warren, "Dual contouring of Hermite data," *ACM Trans. on Graphics*, vol.21, no.3, July 2002, pp.339-46.

[23] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel, "Feature sensitive surface extraction from volume data," *Computer Graphics Proceedings. SIGGRAPH 2001.*, pp.57-66, 2001. New York, NY, USA.

[24] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel, "Interactive multi-resolution modeling on arbitrary meshes," *Proceedings of SIGGRAPH 1998*, pp.105-114, 1998.

[25] J. Lawrence and T. Funkhouser, "A painting interface for interactive surface deformations," *Proceedings of Pacific Graphics 2003*.

[26] T. Lewiner, H. Lopes, A.W. Vieira, and G. Tavares, "Efficient implementation of marching cubes' cases with topological guarantees," *Journal of Graphics Tools*, vol.8, pp.1-15, 2003.

[27] W. Lorensen and H. Cline, "Marching cubes: a high resolution 3D surface construction algorithm," *Computer Graphics*, vol.21, pp.163-169, 1987.

[28] L. Markosian, J. M. Cohen, T. Crulli, and J. Hughes, "Skin: a constructive approach to modeling free-form shapes," *Proceedings of SIGGRAPH 99*, pp.393-400, 1999.

[29] M. Meyer, M. Desbrun, P. Schroder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," Proceeding of Visualization and Mathematics, 2002.

[30] A. Miropolsky and A. Fischer, "Reconstruction with 3D geometry bilateral filtering," *Proceeding of ACM Symposium on Solid Modeling and Applications 2004*, pp.225-229.

[31] G. M. Nielson and B. Hamann, "The asymptotic decider: resolving the ambiguity in marching cubes," *Proc. of IEEE Visualization'91*, pp.83-91, 1991.

[32] F. S. Nooruddin and G. Turk, "Simplification and repair of polygonal models using volumetric techniques," *IEEE Trans. on Visualization and Computer Graphics*, vol.9, no.2, April-June 2003, pp.191-205.

[33] Y. Ohtake and A. Belyaev, "Dual-prime mesh optimization for polygonized implicit surfaces with sharp features," *ACM Solid Modeling Symposium*, K. Lee and N. Patrikalakis (Eds.), 2003, pp. 171-178.

[34] Y. Ohtake, A. Belyaev, and I. Bogaevski, "Mesh regularization and adaptive smoothing," *Computer-Aided Design*, vol.33, pp.789-800, 2001.

[35] Y. Ohtake, A. Belyaev, and A. Pasko, "Dynamic mesh optimization for polygonized implicit surfaces with sharp features," *The Visual Computer*, vol.19, pp.115-126, 2003.

[36] K. van Overveld and B. Wyvill, "Shrinkwrap: An efficient adaptive algorithm for triangulating an iso-surface," *The Visual Computer*, vol.20, no.6, pp.362-379, 2004.

[37] S.M. Smith and J.M. Brady, "SUSAN - a new approach to low level image processing," *International Journal of Computer Vision*, vol.23, no.1, pp.45-78, May 1997.

[38] V. Surazhsky and C. Gotsman, "Explicit surface remeshing," *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry*, pp.20-30, 2003.

[39] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher, "Geometric surface smoothing via anisotropic diffusion of normals," *Proceedings of the conference on Visualization '02*, pp.125-132, 2002.

[40] G. Taubin, "A signal processing approach to fair surface design," *In Proceedings of SIGGRAPH 95*, pp.351-358, 1995.

[41] J. F. Thompson, B. K. Soni, and N. P. Weatherill. *Handbook of Grid Generation*. CRC Press: Florida, 1999.

[42] C. Tomasi and R. Manduchi., "Bilateral filtering for gray and color images," *Proceedings of the Sixth International Conference on Computer Vision*, pp. 839-846, 1998.

[43] G. Varadhan, S. Krishnan, Y. J. Kim, and D. Manocha, "Feature-sensitive subdivision and isosurface reconstruction," *Proceedings of IEEE Visualization 2003*, 2003, pp.99-106.

[44] J. Vorsatz, C. Rössl, L. P. Kobbelt, and H.-P. Seidel, "Feature sensitive remeshing," *Computer Graphics Forum*, vol.20, no.3, pp.393-402, 2001.

[45] M. Y. Wang and X. Wang, "A level-set based variational method for design and optimization of heterogeneous objects," *Computer-Aided Design*, vol.37, pp.321-337, 2004.

[46] K. Watanabe and A. Belyaev, "Detection of salient curvature features on polygonal surfaces," *Computer Graphics Forum*, vol.20, no.3, pp.385-392, 2001.

[47] W. Welch and A. Witkin, "Free-form shape design using triangulated surfaces," *Proceedings of SIGGRAPH 94*, pp.247-256, 1994.

[48] http://www.mpi-sb.mpg.de/~ohtake/software/.

Charlie C. L. Wang is currently an Assistant Professor at the Department of Automation and Computer-Aided Engineering, the Chinese University of Hong Kong. He gained his BEng (1998) in Mechatronics Engineering from the Huazhong University of Science and Technology, MPhil (2000) and PhD (2002) in Mechanical Engineering from the Hong Kong University of Science and Technology. His current research interests include design automation and optimization, geometric modeling, CAD/CAM, reverse engineering, and computer graphics.