Bi-Normal Mesh Smoothing Based on Vertex Feature

WULI WANG (Member, IEEE), AND LIMING DUAN (Member, IEEE)

1College of Oceanography and Space Informatics, China University of Petroleum, Qingdao 266580, China
2Engineering Research Center of Industrial Computed Tomography Nondestructive Testing, Ministry of Education, Chongqing University, Chongqing 400044, China

Corresponding authors: Liming Duan (duanliming163@163.com) and Wuli Wang (upcwwl@163.com)

ABSTRACT Aiming at the issue that mesh smoothing is hard to balance in terms of noise removing and feature preserving, in this article, we combine the facet normal representing global geometric features of the mesh with the vertex normal characterizing local details of the mesh and propose a bi-normal mesh smoothing method based on vertex feature selection (BNBVF). Firstly, the guided filtering algorithm is extended to calculate accurately the facet normal in the field of geometric processing. The key of this portion is computing the guided normal of facet by using a geometric neighboring patch with the most consistent normal. Then, an adaptive tensor voting method is employed to divide the vertices of the mesh into feature vertices and non-feature vertices. Thirdly, a method of the neighborhood facets clustering combining with the plane fitting is proposed to calculate the normal of feature vertex, and the weighting average of first-order neighborhood facets of the vertex is applied to compute the normal of non-feature vertex. Finally, the vertices of the mesh are updated iteratively by combining the geometric information of the facet normal and vertex normal to achieve mesh smoothing. Experimental results demonstrate that the superior performance of our proposed algorithm to state-of-the-art approaches in feature preserving and error reducing.

INDEX TERMS Mesh smoothing, feature-preserving, bi-normal, guided filtering, tensor voting.

I. INTRODUCTION

Industrial computed tomography (CT) can clearly, accurately and intuitively display the internal structure, composition, material and defect condition of the detected object in the form of two-dimensional sectional image or three-dimensional image without damage to the detected object. The basic principle is that x-ray or γ-ray with certain energy and intensity is used to irradiate the detected object, and then the attenuation law and distribution of energy in the detected object are calculated. Finally, Industrial CT data of the detected object is obtained by using computer information processing and image reconstruction technology. However, it is affected by some factors such as instrumentation, ray source noise, reconstruction algorithm and human factor during data acquisition, storage and transmission, which makes the noise information inevitably existing in the triangular mesh reconstructed by industrial CT data [1], [2]. The presence of these noises not only reduces the visualization of the mesh, but also brings about some troubles for subsequent processing. Although the existing mesh smoothing methods have achieved satisfactory results, there remains still a challenging issue that non-uniform sampling and multi-scale triangular mesh reconstructed by industrial CT data can effectively preserve geometric detail of the mesh while smoothing.

The current mesh smoothing methods include mainly isotropic smoothing and anisotropic smoothing. The Laplacian method [3] has become a representative of isotropic methods and always favored by researchers because it is fast, easy to implement and widely used. Taubin et al. [4] proposed a two-step Laplacian operator to extend Laplacian method for mesh smoothing. Desbrun et al. [5] extended two-step Laplacian method by geometric flow simulation and applied it to smooth the non-uniform mesh. Meyer et al. [6] improved the geometric flow simulation further and applied it to preserve anisotropic feature. Kim and Rossignac [7]
designed a low-pass/high-pass filter frame with enhancement and attenuation options for smoothing the mesh. Indeed, these isotropic smoothing methods can achieve stable results, however some local details of the mesh are inevitably filtered out due to the inherent characteristics of isotropic filtering. Compared with the isotropic mesh smoothing method, the anisotropic smoothing method can better preserve the geometry and detail features of the mesh while smoothing the mesh. Fleishman et al. [8] extended firstly the bilateral filtering algorithm that is well filter effect in the field of image processing, and applied it to smooth the mesh by means of a fast one-step iterative scheme. However, this method cannot always accurately preserve the details of the mesh. Jones et al. [9] employed a two-step non-iterative scheme to consider normal smoothing and vertex updating as a global optimization problem, which enhances the retention of features to a certain extent. However, the algorithm is slow. Zheng et al. [10] used bilateral filtering to process the facet normal of the mesh, and then updated the vertex position according to the facet normal. Solomon et al. [11] modified bilateral filtering to deal with the facet normal, and combined with the mean shift method to implement a framework for mesh smoothing, which improves the speed and smoothing effect of the bilateral filtering algorithm to some extent. Nevertheless, the framework is not effective to all meshes.

In recent years, a two-stage method (or its variants) [12]–[17] that the facet normal is firstly calculated and then the vertices are iteratively updated according to the facet normal is proposed to achieve mesh smoothing, and it can better preserve the geometric and detailed features of the mesh with certain noise. However, the method cannot effectively eliminate the mutual influence of different geometric structures, resulting in the fuzzy features. The basis of this method is that the facet normal is better than the vertex normal in describing the global geometric variation of the mesh, but the vertex normal can better describe the local detail features of the mesh. Therefore, the non-uniform and multi-scale mesh smoothing only based on geometric information of the facet normal can result in the blurring of the detail features, shape distortion or vertex offset of the mesh [18]–[20]. Moreover, Wei et al. utilize surface normal filtering to avoid the loss of geometric details in bas-relief modeling [21], [22].

In addition, sparse optimization and surface reconstruction for mesh de-noising have increasingly attracted more attention of researchers in recent years. Several novel $l_0$ and $l_1$ sparse optimization method [17], [17], [23]–[25] have been proposed to eliminate robustly and reliably noises while preserving features. However, it is crucial to estimate properly the differential geometric properties of mesh in these methods. Some 3D-surface reconstruction methods [26]–[28] have been presented to remove noisy while recovering feature. Whereas, the effective structure detection method for estimating properly surface normal is essential due to most meshes usually contain complicate structure.

Though the above-mentioned methods can remove the noisy from mesh while preserving features to some extent, there is still great space to improve the effect of mesh smoothing. In this article, the guided filtering is extended to process the facet normal of the mesh, combining the vertex normal calculated by the vertex feature, a novel bi-normal method for mesh smoothing is proposed. The scheme of the presented method is listed in FIGURE 1.

The work foci of this article is three-fold:

1. A novel mesh smoothing method that considers both facet normal and vertex normal is developed.
2. The authors extend the guided filtering to calculate accurately the facet normal according to the patch whose consistency with the facet is best.
3. A method of the neighborhood facets clustering combining with the plane fitting is proposed to calculate precisely the normal of feature vertex.

II. COMPUTING THE FACET NORMAL FIELD BASED ON GUIDED FILTERING

In consideration of the field of image processing, the guided filter is better than the bilateral filtering in feature retaining and arithmetic speed. Inspired by the good performance of the bilateral filtering in the field of mesh smoothing, this article introduce the guided filtering to the field of geometry processing for mesh smoothing.

A. THE GUIDED FILTERING

Guided filtering is a new image filtering method, and its output is a local linear transformation of the guided image. The guided filtering has good edge retention performance similar to bilateral filtering and does not cause gradient inversion. Moreover, if the guidance image is appropriate, the output of the guided filter has more structural information than the input image. Because the guided filtering is simple, effective and the features are kept well, it has been widely used in image smoothing/enhancement, flash/non-flash image, image mapping/gradation processing, de-blurring, up-sampling, optical flow estimation, image segmentation and illumination and rendering [29].

The key to the guided filtering is that can be regarded as a local linear model between the guidance image and the output image [29], [30]. If the output of the guided filter is $q$ and the guidance image is $G$, the linear relationship between the
output and the guided image can be defined as follows:

\[ q_i = a_k G_i + b_k, \quad \forall i \in \omega_k \]  

(1)

where, \( \omega_k \) is a rectangular window whose central pixel is \( k \), and its radius is \( r \). \( a_k \) and \( b_k \) are linear coefficient in window \( \omega_k \), and respectively defined as:

\[ a_k = \frac{1}{|\omega|} \sum_{i \in \omega_k} G_ip_i - \mu_k \overline{p_k} \]  

(2)

\[ b_k = \overline{p_k} - a_k \mu_k \]  

(3)

where, \( \overline{p_k} \) is the mean of input image \( p \) in window \( \omega_k \). According to (1), we can know that the output image must have edge information as long as the guidance image \( G \) has edge information, because \( \nabla q \) is equal to \( a \nabla G \).

**B. FACET NORMAL FILTERING**

Compared with vertex filtering, normal filtering can describe the geometric features of the mesh more conveniently and accurately. For example, the sharp edge can be described by the larger error of normal between two incident faces. The outward unit normal of facet \( f_i \) in the triangular mesh can be defined as:

\[ n_i = \frac{(v_{i2} - v_{i1}) \times (v_{i3} - v_{i1})}{\|v_{i2} - v_{i1}\| \times \|v_{i3} - v_{i1}\|} \]  

(4)

where, \( v_{i1}, v_{i2}, v_{i3} \) are respectively the coordinates of each vertex of facet \( f_i \).

According to (1), the guided filtering output of the unit normal of facet \( f_i \) can be expressed as:

\[ n^*_i = \frac{a_k g_i + b_k}{\|a_k g_i + b_k\|} \]  

(5)

where,

\[ a_k = \frac{1}{|\omega|} \sum_{j \in N_i} g_j n_j - \overline{g} \overline{n}_i \]  

(6)

\[ b_k = \overline{n}_i - a_k \overline{g}_i \]  

(7)

where, \( g_i \) is a guided normal of facet \( f_i \), \( \overline{g}_i \) denotes the mean of the guided normal that come from neighborhood facets of \( f_i \), \( \overline{n}_i \) indicates the mean of the normal that come from neighborhood facets of \( f_i \), \( \omega \) is the number of facets in the neighborhood patch, \( N_i \) represents the first-order geometric neighborhood of facet \( f_i \). The geometric neighborhood consists of the largest set of faces containing the facet \( f_i \) and satisfies the following two conditions. (1) The normal angle between each facet belonging to \( N_i \) and facet \( f_i \) is less than or equal to \( \beta \) which is specified by user, that is, \( \forall f_k \in N_i, \angle(n_k, n_i) \leq \beta \). (2) A single connection is formed by sharing vertices between adjacent facets of \( N_i \). For a non-uniform mesh, the geometric neighborhood includes all neighborhood facets with similar normal, which can achieve better filtering output.

**C. CONSTRUCTING THE GUIDED NORMAL OF FACET**

In the guided filtering, the closer the input is similar to the original signal, the better the filtering output. Likewise, the closer the guiding normal to the facet normal is, the better the filtering effect of the mesh is. Therefore, the key to the quality of the filtered output is the construction of the guided normal.

If there is no noise in the mesh, the normal \( \overline{g}_i \) of the facet itself is a good guided normal. If the noise in the mesh is large, the normal of the facet itself cannot correctly describe the curve surface of the mesh, and using it as the guided normal may bring about erroneous results. Reference [14] proposed that a triangular mesh could be divided into a number of small patch composed of a plurality of facets with similar normal. For each facet \( f_i \), we can find a patch \( P \) that possesses the best consistency among all patches containing the facet \( f_i \). Therefore, we employ the normal of facets in patch \( P \) to calculate the guided normal of facet \( f_i \). All patches that contain facet \( f_i \) can be represented by a patch candidate set \( P_{set}(f_i) = \{ P_k | f_i \in P_k \} \), where \( P_k \) is a union patch composed of all facets that share a vertex with any facet \( f_i \) of the mesh.

The normal consistency \( H(P) \) of each candidate patch \( P \in P_{set}(f_i) \) is calculated as follows:

\[ H(P) = D(P) \cdot S(P) \]  

(8)

where, \( D(P) \) denotes the maximum distance between the normal of two facets in patch \( P \), and \( S(P) \) represents the relative saliency of the edges in the patch.

\[ D(P) = \max_{f_i, f_j \in P} \| n_i - n_j \| \]  

(9)

\[ S(P) = \max_{e_j \in E_P} \frac{\varphi(e_j)}{\varepsilon + \sum_{e_j \in E_P} \varphi(e_j)} \]  

(10)

where, \( E_P \) stands for the set of all inner edges in patch \( P \), \( \varphi(e_j) \) is the saliency of the inner edge and it can be calculated by the normal of two adjacent facets \( f_{i1} \) and \( f_{i2} \) (\( e_j \) represents the common edge of facet \( f_{i1} \) and facet \( f_{i2} \)).

\[ \varphi(e_j) = \|n_{i1} - n_{i2}\| \]  

(11)

It can be seen by above (9) and (10) that the smaller \( D(P) \), the more similar the normal of all facets in the patch are, the smaller \( S(P) \), the more similar the saliency of all the inner edges in the patch. Therefore, the smaller the normal change of each facet in patch \( P \), the smaller the consistency function \( H(P) \).

The guided normal of facet \( f_i \) can be calculated by the weighting average of all facets of patch \( P^* \). Where \( P^* \in C(f_i) \) and \( H(P^*) = \min(H(P), P \in P_{set}(f_i)) \).

\[ g_i = \frac{\sum_{f_j \in P^*} \omega_{f_j} g_{f_j} n_{f_j}}{\sum_{f_j \in P^*} \omega_{f_j} |g_{f_j} n_{f_j}|} \]  

(12)

where, \( \omega_{f_j} = \frac{S_j}{\sum_{f_{j'} \in P^*} \omega_{f_{j'}}} \) denotes the area weighting factor of facet \( f_j \) of patch \( P^* \), \( \omega_{f_j} = \exp(- \| c_i - c_j \|) \) is monotonically
decreasing with respect to the distance between \( c_i, c_j, c_i, c_j \) present the center of facet \( f_i, f_j \) respectively.

III. CONSTRUCTING THE VERTEX NORMAL FIELD OF THE MESH

A. THE VERTEX CLASSIFICATION BASED ON TENSOR VOTING

1) THE TENSOR VOTING

The voting tensor of vertex \( v_i \) of facet \( f_j \) in the triangular mesh can be regarded as the weight sum of the neighborhood triangle normal covariance matrix \( \mathbf{V}_f \) [13], [31], [32]. If the unit normal of facet \( f_j \) is \( \mathbf{n}_{f_j} = (a, b, c) \), then \( \mathbf{V}_f \) can be expressed as:

\[
\mathbf{V}_f = \mathbf{n}_{f_j} \cdot \mathbf{n}_{f_j}^T = \begin{pmatrix} a & b & c \end{pmatrix} \cdot \begin{pmatrix} a & b & c \\ b & b^2 & bc \\ c & bc & c^2 \end{pmatrix}
\]

Tensor voting \( \mathbf{T}_{v_i} \) of vertex \( v_i \) can be defined as:

\[
\mathbf{T}_{v_i} = \sum_{f_j \in \mathcal{N}_f(v_i)} \omega_{f_j} \mathbf{n}_{f_j} \cdot \mathbf{n}_{f_j}^T
\]

where, \( \mathcal{N}_f(v_i) \) represents the first-order neighborhood triangle of vertex \( v_i \), \( \omega_{f_j} \) denotes the weight that is calculated by the area ratio of the neighborhood triangle and the distance between vertex \( v_i \) and each triangle of \( \mathcal{N}_f(v_i) \).

\[
\omega_{f_j} = \frac{S_{f_j}}{\max(S_{\mathcal{N}_f(v_i)})} \exp\left(-\frac{\|\mathbf{c}_{f_j} - \mathbf{v}_i\|}{\max(\|\mathbf{c}_{f_j} - \mathbf{v}_i\|)}\right)
\]

where, \( S_{f_j} \) stands for the area of facet \( f_j \), \( \max(S_{\mathcal{N}_f(v_i)}) \) denotes the max triangle area of \( \mathcal{N}_f(v_i) \), \( \mathbf{c}_{f_j} \) is the center of triangle \( f_j \), and \( v_i \) is the coordinate of vertex \( v_i \).

2) THE VERTEX CLASSIFICATION

Reference [32] proposed a vertex classification method based on the eigenvalues of normal tensor of the vertex, which can divide the vertices of the mesh into plane vertices (non-feature vertices), sharp edge vertices and corner vertices (feature vertices). However, the method can only detect the feature vertices of the high-quality mesh because that employs only the normal information of the vertex first-order neighborhood facets to calculate the vertex normal tensor, no considering the scale information. There may be exist false detection and missed detection of feature vertices when the method is employed to deal with a noisy and non-uniform sampling mesh, such as the mesh reconstructed by industrial CT. Therefore, this article adopt the adaptive tensor voting method proposed in Reference [13] to classify the vertices. This adaptive tensor filtering can not only filter the noise, but also preserve the detailed features of the mesh and achieve a good balance between feature vertices and non-feature vertices. The threshold \( \Omega_i \) for distinguishing vertex types can be defined as follows:

\[
\Omega_i = 0.475 - \sqrt{\frac{(\lambda_1 - \bar{\lambda}) + (\lambda_2 - \bar{\lambda}) + (\lambda_3 - \bar{\lambda})}{3}}
\]

where, \( \lambda_1, \lambda_2, \lambda_3 \) denote the eigenvalues of the tensor voting respectively, and satisfy \( \bar{\lambda} = (\lambda_1 + \lambda_2 + \lambda_3)/3 \). If the vertex is a feature vertex, then \( \Omega_i = 0 \). The pipeline of the vertex classification is as follows.

Step1: \( r_m(m = 1) \) that is the first-order triangle neighborhood of vertex \( v_i \) set to initial scale.

Step2: Calculate the value of \( \Omega_i^{m-1} \) and compare it with \( \Omega^1, \Omega^2 \). If \( \Omega_i^{m-1} < \Omega^1 \), then vertex \( v_i \) is a feature vertex and the neighborhood expansion is stop; if vertex \( v_i \) is a non-feature vertex and the neighborhood expansion is stop too; otherwise, step3 is executed.

Step3: If \( \Omega_i^m > \tau \Omega_i^{m-1} \) is satisfied, neighborhood \( r_m(m \geq 2) \) of the vertex \( v_i \) is expanded to next neighborhood \( r_{m+1} \), go to step 2 to recalculate \( \Omega_i^m \) which compares with \( \Omega^1, \Omega^2 \). Neighborhood \( r_m \) of vertex \( v_i \) stop expanding until \( \Omega_i^m \) can determine whether vertex \( v_i \) is a feature vertex or a non-feature vertex at current scale.

where, \( \Omega^1, \Omega^2 (\Omega^1 < \Omega^2) \) are two predefined reference values for assisting judgment. Extensive experiments show that the results are satisfied in the condition of \( \Omega^1 = 0.03, \Omega^2 = 0.13, \tau = 1.4 \).

The normal tensor voting at scale \( r_m \) can be defined as follows:

\[
\mathbf{T}_{r_m(v)} = \mathbf{T}_{r_{m-1}(v)} + \sum_{f_j \in \mathcal{N}_{r_m}(v)} \omega_{f_j} \mathbf{n}_{f_j} \cdot \mathbf{n}_{f_j}^T
\]

where, \( \mathcal{N}_{r_m}(v) \) represents new added facets at scale \( r_m \), and weight coefficient \( \omega_{f_j} \) can be defined as:

\[
\omega_{f_j} = \frac{S_{f_j}}{\max(S_{\mathcal{N}_{r_m}(v)})} \exp\left(-\frac{m \cdot \|\mathbf{c}_{f_j} - \mathbf{v}_i\|}{\max(m(\|\mathbf{c}_{f_j} - \mathbf{v}_i\|))}\right)
\]

where, \( m \) indicates the order of the neighborhood of vertex \( v_i \), \( \max(S_{\mathcal{N}_{r_m}(v)}) \) denotes the max area of the facet which comes from \( m \)-order neighborhood of vertex \( v_i \), \( S_{f_j} \) represents the area of each facet in \( m \)-order neighborhood of vertex \( v_i \), \( \max(m(\|\mathbf{c}_{f_j} - \mathbf{v}_i\|)) \) stands for the max distance between vertex \( v_i \) and the center of the facet in \( m \)-order neighborhood. In addition, the purpose of adding \( m \) to the numerator of the exponent \( \exp(-\cdot) \) is to ensure that the inner ring neighborhood of vertex \( v_i \) is more influential than the outer ring neighborhood in calculating tensor voting.

B. CALCULATING THE NORMAL FIELD OF THE VERTICES

1) CALCULATING THE NORMAL OF THE NON-FEATURE VERTEX

The non-feature vertex whose change of geometric feature around it is smaller, and the normal of vertex is roughly consistent with the normal of the facet in neighborhood. Therefore, the normal of non-feature vertex can be calculated by the weighting average of the first-order neighborhood
\[ \mathbf{n}_{v_i} = \sum_{f_j \in N_{v_i}} \omega_{f_j} \mathbf{n}_{f_j} \]

where, \( \omega_{f_j} = \frac{S_{f_j}}{\sum_{f_j \in N_{v_i}} S_{f_j}} \) is the weighting factor of the first-order neighborhood facet, \( S_{f_j} \) indicates the area of facet \( f_j \), and \( \mathbf{n}_{f_j} \) represents the normal of facet \( f_j \).

2) **CALCULATING THE NORMAL OF THE FEATURE VERTEX**

Because the feature vertex normal of the mesh has a large difference from the normal of its neighboring facets, there is a big error if the weighting average of the neighborhood facet normal is still employed to calculate the feature vertex normal. In this article, the method of the neighborhood facets clustering combining with the plane fitting [6], [7], [11], [12] is utilized to calculate the normal of feature vertex accurately.

### a: NEIGHBORHOOD FACETS CLUSTERING

The neighborhood facets clustering is the clustering of neighborhood facets with uniform geometric features, such as, normal similarity or smooth variation, around the feature vertex. References [12], [13] employed an energy function to describe clustering as an optimization problem.

\[
E = \sum_{i=1}^{k} \int_{M_i} \rho(x) \mathbf{n} \cdot (\mathbf{n} - \mathbf{n}_i) \, dx
\]

where, \( k \) indicates the sorts of the neighborhood facets, \( M_i \) denotes a certain neighborhood, \( \rho(x) \) is a density function that is a constant, \( \mathbf{n}_i \) represents the normalize weighting average of the normal of all facets in \( M_i \).

According to formula (20), the energy function \( E \) regards the total offset of all facets in the class as the normal representative of the class. If the normal of all vertices in the facet are the same and \( s_j = \int_{f_j} \rho(x) \, dx \) is the area function of facet \( f_j \), then the discrete form of \( E \) can be expressed as:

\[
E = \sum_{i=1}^{k} \left( \sum_{f_j \in M_i} s_j \| \mathbf{n}_j \|^2 - \sum_{f_j \in M_i} s_j \mathbf{n}_j \right) \]

FIGURE 3. Comparisons of the resulted holder mesh with noise in different smoothing methods.
where, \( n_j \) indicates the unit normal of facet \( f_j \). If \( \rho(x) = 1 \), then \( s_j \) is the area of facet \( f_j \).

Neighborhood facets clustering of the feature vertex has been realized after formula (21) is optimized. For detailed optimization solution process, please refer to reference [12].

\( b: \text{CALCULATING THE NORMAL OF THE FEATURE VERTICES} \)

Step1: The least squares method [33] is employed to fit a cubic surface for the vertices of each clustered patch, and the fitting formula is as follows:

\[
f(x, y) = Ax^3 + By^3 + Cx^2y + Dxy^2 + Ex^2 + Fy^2 + Gxy + Hx + ly + J
\]  

(22)

Step2: The facet with the best fitting is searched in each type fitted cubic surface.

Step3: The normal of the feature vertex is calculated by the normal of facet with the best fitting in each type fitted cubic surface.

\[
n_v = \sum_{j=1}^{k} \omega_j n_j
\]

(23)

where, \( \omega_j \) represents the weighting factor of facet \( f_j \) with the best fitting. \( n_j \) indicates the normal of facet \( f_j \) with the best fitting.
**Step4:** Traverse each feature vertex of the mesh and repeat step1∼step3 to generate the normal field of the feature vertices.

The normal of the facet that is the most representative of geometric feature information of the vertex is employed to calculate the normal of the feature vertex, which is more accurate than the weighting average of the normal of the first-order neighborhood facets.

### IV. MESH SMOOTHING BASED BI-NORMAL FILTERING

The facet normal and vertex normal of the mesh describe global geometric features and local details of the mesh respectively. The geometric feature of these two normal is particularly repetitive sometimes, and even almost same in the flat region of the mesh. However, the mesh is mostly segmented smooth and the geometric features of the facet normal and vertex normal are no longer similar in the reverse engineering field based industrial CT. In this case, only the facet normal or the vertex normal is employed to smooth the mesh, which results in blurred details and shape distortion of the mesh. Therefore, in this article, a novel mesh smoothing method based on bi-normal filtering is proposed by

![Image of Figure 6](image.png)

**FIGURE 6.** Performance comparisons of different smoothing methods.

| TABLE 1. Original data of the meshes. |
|--------------------------------------|
| model     | size /mm×mm×mm | area/mm² | volume/mm³ |
| carburetor | 173.02×175.41×167.32 | 298076.17 | 840405.69 |
| cylinder head | 141.16×198.15×108.52 | 235250.42 | 1255598.13 |
TABLE 2. Performance comparisons of different mesh smoothing methods.

| model   | method | $S_{lm}$/mm×mm×mm | $E_{ar}$/mm$^3$ | $E_{vd}$/mm$^3$ | $D_{max}$/mm | $D_{sar}$/mm |
|---------|--------|-------------------|----------------|----------------|--------------|--------------|
| LAP     |        | 171.87×173.88×165.96 | 1846.36 | 2356.82 | 0.75 | 0.086 |
| BF      |        | 172.68×174.79×166.78 | 1278.19 | 1864.08 | 0.42 | 0.065 |
| carburetor | BNF   | 172.07×174.39×165.45 | 939.38 | 830.46 | 0.36 | 0.044 |
| BNF     |        | 172.28×175.39×166.89 | 685.24 | 590.61 | 0.26 | 0.039 |
| JBF     |        | 171.95×174.15×165.32 | 1446.09 | 2066.37 | 0.67 | 0.074 |
| PcFilter |        | 172.79×175.07×166.96 | 432.67 | 268.90 | 0.28 | 0.036 |
| BNBVF   |        | 172.85×175.15×167.61 | 458.60 | 285.43 | 0.26 | 0.032 |
| LAP     |        | 138.35×196.59×107.02 | 2519.56 | 2674.07 | 0.68 | 0.093 |
| BF      |        | 139.79×197.23×107.87 | 1637.32 | 1535.81 | 0.47 | 0.076 |
| cylinder | BNF    | 140.14×197.88×106.96 | 1149.76 | 980.26 | 0.39 | 0.068 |
| BNF     |        | 140.60×198.07×108.87 | 858.09 | 675.54 | 0.31 | 0.041 |
| BNF     |        | 138.62×196.94×107.35 | 2190.85 | 1870.61 | 0.60 | 0.052 |
| PcFilter |        | 140.06×197.89×107.98 | 570.62 | 289.07 | 0.29 | 0.033 |
| BNBVF   |        | 140.65×198.63×107.49 | 580.86 | 332.65 | 0.27 | 0.031 |

combining the vertex normal with the facet normal. The presented method can better preserve the geometric detailed features of the mesh, and the specific steps are as follows.

**Step1:** The guided filtering is extended to calculate accurately the facet normal field of the mesh.

**Step2:** The tensor voting method is used to classify the vertices of the mesh into non-feature vertices and feature vertices.

**Step3:** The vertex normal field of the mesh is respectively calculated according to the vertex type.

**Step4:** Update the vertex iteratively.

1. If the vertex is a non-feature vertex (i.e. plane vertex), the vertex is updated via the facet normal $n^*_i$ of the mesh.

$$v_i^{(t+1)} = v_i^{(t)} + \frac{1}{|N_i|} \sum_{j \in N_i} n^*_j \left[ n^*_j \cdot (c_j^{(t)} - v_i^{(t)}) \right]$$  \hspace{0.5cm} (24)

where, $v_i^{(t)}$ is the value of vertex $v_i$ at the $t$-th iteration, $N_i$ indicates the first-order neighborhood facet set of the vertex $v_i$, and $c_j^{(t)} = (v_j^{(t)} + v_l^{(t)} + v_k^{(t)}) / 3$.

2. If the vertex is a feature vertex (i.e. sharp edge vertex or corner vertex), the vertex is updated by combining the facet normal $n^*_i$ with the vertex normal $n_{v_i}$.

$$v_i^{(t+1)} = v_i^{(t)} + \frac{1-\lambda}{|N_i|} \sum_{j \in N_i} n^*_j \left[ n^*_j \cdot (c_j^{(t)} - v_i^{(t)}) \right] + \lambda n_{v_i}$$  \hspace{0.5cm} (25)

where, $n^*_j$ denotes the facet normal of the first-order neighborhood facet processed by the guided filtering, $n_{v_i}$ indicates the normal of the vertex, $\lambda$ stands for the ratio factor and the range of value is $0 \leq \lambda \leq 1$.

**Step5:** Traverse all vertices of the mesh and repeat step1~step4 to update each vertex.

**Step6:** If the error of between smoothed mesh and the initial mesh or the error between the current smoothed mesh and the former smoothed mesh is less than the setting value, the algorithm is stopped. Otherwise, the algorithm flow return to step1 to continue processing.

V. EXPERIMENTAL RESULTS AND ANALYSIS

To validate the effectiveness in both noise eliminating and feature retaining of the proposed method, the method is compared against the classical and state-of-the-art methods on some meshes, such as, Laplacian (LAP) [3], bilateral filtering (BF) [8], bilateral normal filtering (BNF) [10], joint bilateral filtering (JBF) [14], bi-normal filtering based on optimization (BNFBO) [12], and PcFilter [34]. Since each smoothing method involves many parameters and some of the parameters are changed, the parameters are set according to the recommendations of the original literature, and the experiment is repeated several times to select the best result for comparison.

In order to demonstrate the versatility of the proposed method, the big Gaussian noise is added artificially to the Fandisk mesh and the holder mesh. FIGURE 2 and FIGURE 3 show the comparative results of the noisy
Fandisk and the noisy holder in different smoothing methods. BF is not effective in dealing with such mesh with large-scale noise, and can only remove part of the high-frequency noise. BNF and BNFBO can better eliminate noise, but the edges and detailed features of the mesh need to be improved. The smoothing effects of PcFilter and BNBVF method both are good, and BNBVF method is better in terms of detailed feature-preserving and has the best visualization effect.

FIGURE 4 shows the results of the cylinder head mesh reconstructed by industrial CT image in different smoothing methods. According to Figure 4, the visualization effects of the cylinder head mesh in different smoothing methods can be observed clearly. The smoothing result of LAP is too smooth. The smoothing results of BF and JBF contain still some noises, however, the features can better be preserved. BNFBO, PcFilter, and BNBVF not only can remove the noises of the mesh, but also can better preserve the detailed features of the mesh. In addition, the smoothing results of these three methods are the best.

FIGURE 5 show that the visualization effect of the carburetor mesh smoothed by various methods. The detailed features of the mesh smoothed by LAP are lost, and the surface of the mesh is too smooth. The method of JBF improves the ability of feature preserving to some extent, however, the surface of the mesh is still too smooth. Part of the noises of the mesh is not removed in the methods of BF and BNF, and the feature preserving of the mesh in BF is better. Both BNBVF, BNFBO and PcFilter can better retain the detailed features of the mesh, and the smoothing results are perfect too.

To objectively compare the proposed and existing smoothing methods, the error information of the mesh smoothed by different methods is counted, the specific information is as follow: minimum bounding box size $S_{\text{box}}$, surface area error $E_{\text{surf}}$, volume error $E_{\text{vol}}$, maximum distance of the vertex $D_{\text{max}}$ and average distance of the vertex $D_{\text{ave}}$.

Table 1 shows the related information of the original cylinder head mesh and carburetor mesh. The error information of the meshes smoothed by different methods are shown in Table 2 and Figure 6. According to the data in Table 2, the meshes smoothed by different methods including BNBVF, BNFBO and PcFilter, can better approximate the original mesh, as well as the volume error and surface area error are smaller. Whether the carburetor mesh or the cylinder head mesh, the errors of PcFilter are the smallest, and BNBVF’s is similar. However, the maximum distance and average distance of the vertex of BNBVF are the smallest. A more intuitive performance comparison of different smoothed mesh is demonstrated in FIGURE 6. In addition, the energy function is employed to express the vertex update under PcFilter, which brings about the slow operation speed because of the complicated optimization process.

VI. CONCLUSION

In allusion to the defect that the mesh is hard to balance in terms of mesh smoothing and feature preserving, this article put forward a bi-normal mesh smoothing method by combining the facet normal representing global geometric features of mesh with the vertex normal characterizing local details of mesh. Compared with the existing mesh smoothing methods, the proposed method can better preserve the geometric detail features of the mesh while removing the noising, and the error between the smoothed mesh and the original mesh is smaller. Therefore, the mesh smoothed by the presented method can more accurately approximate the original mesh. The shortcoming is that there are many parameters involved in the method, and these parameters need to be re-adjusted to obtain better smoothing results for different meshes. In addition, the scale factor for evaluating the contribution of the facet normal and the vertex normal has a greater impact on the results. Therefore, for different meshes, how to adjust automatically these parameters to obtain better smoothing results will be the focus of subsequent work to study.

ACKNOWLEDGMENT

The authors thank the AE and anonymous reviewers for their valuable comments, and thank Prof. Ligang Liu from University of Science and Technology of China for his help in code implementation of guided filtering.

REFERENCES

[1] S. Liu, K.-C. Chan, and C. C. L. Wang, “Iterative consolidation of unorganized point clouds,” IEEE Comput. Graph. Appl., vol. 32, no. 3, pp. 70–83, May 2012.
[2] P.-S. Wang, Y. Liu, and X. Tong, “Mesh denoising via cascaded normal regression,” ACM Trans. Graph., vol. 35, no. 6, pp. 232:1–232:12, Nov. 2016.
[3] J. Vollmer, R. Mencel, and H. Muller, “Improved Laplacian smoothing of noisy surface meshes,” Comput. Graph. Forum, vol. 18, no. 3, pp. 131–138, Sep. 1999.
[4] G. Taubin, “A signal processing approach to fair surface design,” in Proc. 22nd Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH), 1995, pp. 351–358.
[5] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, “Implicit fairing of irregular meshes using diffusion and curvature flow,” in Proc. 26th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH), 1999, pp. 317–324.
[6] M. Meyer, M. Desbrun, and P. Schroeder, “Discrete differential-geometry operators for triangulated 2-manifolds,” Visualizat. Math., vol. 12, no. 7, pp. 35–57, Dec. 2003.
[7] B. Kim and J. Rossignac, “GeoFilter: Geometric selection of mesh filter parameters,” Comput. Graph. Forum, vol. 24, no. 3, pp. 295–302, Sep. 2005.
[8] S. Fleishman, I. Dori, and D. Cohen-Or, “Bilateral mesh denoising,” ACM Trans. Graph., vol. 22, no. 3, pp. 950–953, Mar. 2003.
[9] T. R. Jones, F. Durand, and M. Desbrun, “Non-iterative, feature-preserving mesh smoothing,” ACM Trans. Graph., vol. 22, no. 3, pp. 943–949, Mar. 2003.
[10] Y. Zheng, H. Fu, O. K.-C. Au, and C.-L. Tai, “Bilateral normal filtering for mesh denoising,” IEEE Trans. Vis. Comput. Graph., vol. 17, no. 10, pp. 1521–1530, Oct. 2011.
[11] J. Solomon, K. Crane, A. Butscher, and C. Wojtan, “A general framework for bilateral and mean shift filtering,” 2014, arXiv:1405.4734 [Online]. Available: http://arxiv.org/abs/1405.4734
[12] M. Wei, J. Yu, W.-M. Pang, J. Wang, J. Qin, L. Liu, and P.-A. Heng, “Bi-normal filtering for mesh denoising,” IEEE Trans. Vis. Comput. Graphics, vol. 21, no. 1, pp. 43–55, Jan. 2015.
[13] M. Wei, L. Liang, W.-M. Pang, J. Wang, W. Li, and H. Wu, “Tensor voting guided mesh denoising,” IEEE Trans. Autom. Sci. Eng., vol. 14, no. 2, pp. 931–945, Apr. 2017.
[14] W. Zhang, B. Deng, and J. Zhang, “Guided mesh normal filtering,” Pacific Graph., vol. 34, no. 7, pp. 1–12, Jul. 2015.
L. Zhu, M. Wei, J. Yu, W. Wang, J. Qin, and P.-A. Heng, “Coarse-to-fine normal filtering for feature-preserving mesh denoising based on isotropic subneighborhoods,” *Comput. Graph. Forum*, vol. 32, no. 7, pp. 371–380, Oct. 2013.

Z. Bian and R. Tong, “Feature-preserving mesh denoising based on vertices classification,” *Comput. Aided Geometric Des.*, vol. 28, no. 1, pp. 50–64, Jan. 2011.

R. Wang, Z. Yang, L. Liu, J. Deng, and F. Chen, “Decoupling noise and features via weighted $L_1$-analysis compressed sensing,” *ACM Trans. Graph.*, vol. 33, no. 2, pp. 1–12, Mar. 2014.

L. He and S. Schaefer, “Mesh denoising via $L_0$ minimization,” *ACM Trans. Graph.*, vol. 32, no. 4, pp. 1–8, Apr. 2013.

X. Lu, Z. Deng, and W. Chen, “A robust scheme for feature-preserving mesh denoising,” *IEEE Trans. Vis. Comput. Graphics*, vol. 22, no. 3, pp. 1181–1194, Mar. 2016.

M. Centin and A. Signoroni, “Mesh denoising with (geo) metric fidelity,” *IEEE Trans. Visualizat. Comput. Graph.*, vol. 23, no. 1, pp. 1–17, Jan. 2017.

M. Wei, Y. Tian, W.-M. Pang, C. C. L. Wang, M.-Y. Pang, J. Wang, J. Qin, and P.-A. Heng, “Bas-relief modeling from normal layers,” *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 4, pp. 1651–1665, Apr. 2019.

M. Wei, Z. Song, Y. Nie, J. Wu, Z. Ji, Y. Guo, H. Xie, J. Wang, and F. Wang, “Normal-based bas-relief modelling via near-lighting photometric stereo,” *Comput. Graph. Forum*, vol. 39, no. 1, pp. 204–219, Feb. 2020, doi: 10.1111/cgf.13754.

Y. Zhao, H. Qin, X. Zeng, J. Xu, and J. Dong, “Robust and effective mesh denoising using $L_0$ sparse regularization,” *Comput.-Aided Design*, vol. 101, pp. 82–97, Aug. 2018.

H. Zhang, C. Wu, J. Zhang, and J. Deng, “Variational mesh denoising using total variation and piecewise constant function space,” *IEEE Trans. Vis. Comput. Graphics*, vol. 21, no. 7, pp. 873–886, Jul. 2015.

X. Wu, J. Zheng, Y. Cai, and C.-W. Fu, “Mesh denoising using extended ROF model with $L_1$ fidelity,” *Comput. Graph. Forum*, vol. 34, no. 7, pp. 35–45, Oct. 2015.

H. Sheung and C. C. L. Wang, “Robust mesh reconstruction from unoriented noisy points,” in *Proc. SIAM/ACM Joint Conf. Geometric Phys. Modeling (SPM)*, 2009, pp. 13–24.

H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. Zhang, “Edge-aware point set resampling,” *ACM Trans. Graph.*, vol. 32, no. 1, pp. 1–12, Jan. 2013.

O. Remil, Q. Xie, X. Xie, K. Xu, and J. Wang, “Surface reconstruction with data-driven exemplar priors,” *Comput.-Aided Design*, vol. 88, no. 3, pp. 31–41, Jul. 2017.

WULI WANG (Member, IEEE) received the B.S. and M.S. degrees from the China University of Petroleum, in 2001 and 2007, respectively, and the Ph.D. degree from Chongqing University, in 2018. He is currently a Lecturer with the College of Oceanography and Space Informatics, China University of Petroleum. His research interests include image processing, technology and application of industrial CT, mesh generation, and optimization.

LIMING DUAN received the B.S. degrees from the Chengdu University of Science and Technology, in 1985, and the M.S. and Ph.D. degrees from Chongqing University, in 1988 and 1998, respectively. He is currently a Professor with Chongqing University. His research interests include image reconstruction, the technology and application of industrial CT, mesh reconstruction and processing, and reverse design based on industrial CT.

---

[15] L. Zhu, M. Wei, J. Yu, W. Wang, J. Qin, and P.-A. Heng, “Coarse-to-fine normal filtering for feature-preserving mesh denoising based on isotropic subneighborhoods,” *Comput. Graph. Forum*, vol. 32, no. 7, pp. 371–380, Oct. 2013.

[16] L. Zhu, M. Wei, J. Yu, W. Wang, J. Qin, and P.-A. Heng, “Fast guided filter,” 2015, arXiv:1505.00996. [Online]. Available: http://arxiv.org/abs/1505.00996

[17] G. Medioni and C. K. Tang, “Tensor voting: Theory and applications,” in *Proc. RFIA*, 2000, pp. 243–247.

[18] H. S. Kim, H. K. Choi, and K. H. Lee, “Feature detection of triangular meshes based on tensor voting theory,” *Comput.-Aided Design*, vol. 41, no. 1, pp. 47–58, Jan. 2009.

[19] W. Wang, L. Duan, and Y. Bai, “A triangle mesh standardization method based on particle Swarm optimization,” *PLoS ONE*, vol. 11, no. 8, pp. 1–13, Aug. 2016.

[20] M. Wei, J. Huang, X. Xie, L. Liu, J. Wang, and J. Qin, “Mesh denoising guided by patch normal co-filtering via kernel low-rank recovery,” *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 10, pp. 2910–2926, Oct. 2019.

[21] K. He, J. Sun, and X. Tang, “Guided image filtering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[22] K. He and J. Sun, “Fast guided filter,” 2015, arXiv:1505.00996. [Online]. Available: http://arxiv.org/abs/1505.00996

[23] W. Wang, L. Duan, and Y. Bai, “A triangle mesh standardization method based on particle Swarm optimization,” *PLoS ONE*, vol. 11, no. 8, pp. 1–13, Aug. 2016.

[24] M. Wei, J. Huang, X. Xie, L. Liu, J. Wang, and J. Qin, “Mesh denoising guided by patch normal co-filtering via kernel low-rank recovery,” *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 10, pp. 2910–2926, Oct. 2019.

[25] K. He, J. Sun, and X. Tang, “Guided image filtering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[26] K. He and J. Sun, “Fast guided filter,” 2015, arXiv:1505.00996. [Online]. Available: http://arxiv.org/abs/1505.00996

[27] G. Medioni and C. K. Tang, “Tensor voting: Theory and applications,” in *Proc. RFIA*, 2000, pp. 243–247.

[28] H. S. Kim, H. K. Choi, and K. H. Lee, “Feature detection of triangular meshes based on tensor voting theory,” *Comput.-Aided Design*, vol. 41, no. 1, pp. 47–58, Jan. 2009.

[29] W. Wang, L. Duan, and Y. Bai, “A triangle mesh standardization method based on particle Swarm optimization,” *PLoS ONE*, vol. 11, no. 8, pp. 1–13, Aug. 2016.

[30] M. Wei, J. Huang, X. Xie, L. Liu, J. Wang, and J. Qin, “Mesh denoising guided by patch normal co-filtering via kernel low-rank recovery,” *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 10, pp. 2910–2926, Oct. 2019.

[31] M. Wei, J. Huang, X. Xie, L. Liu, J. Wang, and J. Qin, “Mesh denoising guided by patch normal co-filtering via kernel low-rank recovery,” *IEEE Trans. Vis. Comput. Graphics*, vol. 25, no. 10, pp. 2910–2926, Oct. 2019.