EUROGRAPHICS 2017 / L. Barthe and B. Benes
(Guest Editors)

Volume 36 (2017), Number 2

Intrinsic filtering for 2D manifold surface

D. W. Fellner[†1,2] and S. Behnke[2]

[1]TU Darmstadt & Fraunhofer IGD, Germany
[2]Institut für ComputerGraphik & Wissensvisualisierung, TU Graz, Austria

Abstract

The bilateral filter is a classical filtering operator with the goal of preserving edge structure while smoothing signals. The reason it works is mainly on the spatial and range distances are enough close, but it ignores the local relations between signals. In this paper, we introduce a intrinsic filter method for 2D manifold surfaces. This approach builds the connections between desired signal and its neighbors.Therefore, it has a outstanding power on filtering neighbor signal while preserving the important feature. A novel filtering model based on our intrinsic filter is proposed with application in mesh denoising. As the traditional mesh filtering, our framework also has two-stage process: first, we apply geodesic path to build the face normal connections, then filter the face normals basing two different accumulative distance weights. Afterwards, the vertex positions are updated according to the filtered face normals. For accelerating our intrinsic algorithm, we also introduce a simple, fast and effective pattern to compensate the time deficiency in using geodesic path. It performs well on a wide variety of meshes and is competitive with other state-of-the-art methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Due to the popularity of 3D scanners, meshes are becoming more and more accessible. However, the influence of environment and people makes the data contain a lot of high frequency noises. Mesh filtering is a vital preprocessing tool of updating vertex positions in a mesh to achieve perfect goals like denoising, smoothing or enhancement. It typically preserve the obvious geometric structures, while undesirable noise need be discarded. Although a variety of mesh filtering methods achieve satisfactory results [FDCO03, ZFAT11], they have a little imperfect results in dealing with the geometry features such as edges and corners. Because the characteristics of adjacent regions are blended, the output mesh would be imperfect.

The bilateral filter, introduced in [TM98], is a famous edge-preserving image filter, which prefers near values to distant values in both domain and range. Unlike bilateral filter, the guided filters [PSA*04, HST10] set the filtering weights using the intensity difference from another image and has better edge-preserving property. Due to their success in image processing and computational photography and considering normals as a surface signal defined over the original mesh, many attempts have been made to adapt them to geometry processing such as mesh denoising and smoothing [JDD03, ZFAT11, SCBW14]. These filter are able to alleviate the smooth problem, but they still have disadvantages in the process of preserving image/mesh context. For example, choosing a large neighborhood will result in blend of cross-region, while selecting a small one would limit the filtering ability.In our intrinsic filtering framework, we build the connections between filtering signal and its neighbors to effectively solve this problem.

The geodesic filter [GS09], unlike the bilateral filter directly uses the signal differences, considers the accumulated difference between signals on a geodesic path. However, it only uses the adjacent accumulated differences. Our intrinsic filter simultaneously apply another integral differences between signals. Recently, propagated image filter [CW15] was addressed and display its powerful filtering performance. It calculates the accumulated difference not only between the values of adjacent pixels, but also start-end pixels(??) in choosing shortest path. Then it dynamically determines their filtering weight by these two accumulated difference. As the accumulative difference has the ability to estimate the image context, the propagated filter has a more superior edge-preserving property. From our paper, we find that propagated filter is only a special case of our method.

In this paper, inspired by bilateral filter, then based on the theory of signal filtering, we propose a novel filtering method called intrinsic filter for 2D manifold surface. Similar to the previous method, we consider some geometry feature (normals, Gaussian curvature, position and so on) as a surface signal defined over a 2D manifold

† Chairman Eurographics Publications Board

surface. We calculates the filtering weights through using curve integral along the geodesic path. Then the weights are applied to traditional mesh denoising framework and achieve wonderful results. In addition, we introduce a particular pattern for solving the time defeciency in applying geodesic algorithm and obtain even better experimental results. The effectiveness of our approach is illustrated by extensive experimental results in mesh denoising.

2. Relate work

The rise of 3D scanning devices makes captured meshes become more and more easier. However, as the influence of light or devices, meshes which are taken often contain high-frequency noises. Thus mesh denoising, as an important tool of geometry processing, becomes a popular study point [WLDB08]. Due to the large amount of research work in this field, we introduce some methods that have similarities with ours.

The purpose of mesh denoising is removing the noise without damaging the true structure. Because the edge-preserving property of bilateral filter [TM98], it is widely used in image processing [OCDD01, DD02, BC04, WYL*14] and geometry processing such as mesh denoising [FDCO03, JDD03, ZFAT11, SCBW14] and mesh feature enhancement [Wan06].

The papers [FDCO03] and [JDD03] apply bilateral filter to the mesh vertex positions as it is used in image denoising [DD02]. [ZFAT11] employ the bilateral filter to the mesh face normals, then according the filtered normals the process of vertex updated is implemented. [SCBW14] also perform mesh denoising by filtering the face normals, using a generalized cross-bilateral filter. The effectiveness of bilateral filter relies on the range function which reflects the local information of signal, and then applies the averaging weights as the filtered output. However, the difference between the input signal values J_p and J_q can not provides reliable prediction for that between the desired signal values $\bar{J}_p$ and $\bar{J}_q$. Our denoising algorithm also applies the similar approach which iterates face normal filtering and vertex updates. However, our method differs from [ZFAT11] and [SCBW14] which both only consider the spatial and range distances that may smooth the weak edges, also is different from [ZDZ*15] which uses a joint bilateral filter on the normals that may break the sharp corners. As we build the connections between face normals through the geodesic path, obtaining the more accurate filtering weight for denoised algorithm.

The two-stage process including filtering face normals and updating vertex positions has also been adapted by many famous work [YOB02, CC05, SRML08, WFL*15]. The mainly differences between these methods is in their normal filtering strategies. Mean and median filtering and rolling guidance filter [ZSXJ14] are applied in [YOB02] and [WFL*15] respectively. [CC05] automatically select filters according the mesh local sharpness. In [SRML08], face normal filtering is performed by weighted averaging of normals based on the concept of random walks.

Another type of denoising methods employ different strategies to filter the type of vertices which belonging to corner, edge or flat areas on a mesh. The paper [BT11] classify the vertices based on the volume integral invariant while [FYP10] uses a local quadric model to fit the vertices and curvature tensor for obtaining the types of vertices. These two papers [WZY12] and [WYP*15] apply different normal estimation strategies to remove noise.

Other researchers [HS13] apply sparsity optimization to mesh denoising and get a success because the sharp structures are usually sparse on the mesh. They apply L_0 minimization to an edge-based Laplacian operator and effectively preserve the sharp structure of mesh. However, the optimization algorithm prefer flat shapes and may not be suitable for complex meshes. The paper [WYL*14] uses L_1 optimization to recover sharp structures from noisy meshes and also has above-mentioned problems.

3. Intrinsic filtering for 2D manifold surface

In this section, we introduce our intrinsic filtering model for 2D manifold surfaces in detail. Furthermore, we explain that almost all classical filtering methods can be viewed as a simplification of ours.

3.1. Filtering for 2D manifold surface

Suppose we are interested in signals that are defined on 2D manifold surface Ω with values in domain Γ, the filtered output signal $\bar{J}_p$ at point p can be produced by this general form

$$\bar{J}_p = \frac{1}{W_p}\int_{\mathcal{N}(p)} \omega_{p,q} J_q dq\,, \tag{1}$$

where J_q denotes the value of the signal at point q; $\mathcal{N}(p)$ is the neighborhood of the point p; $\omega_{p,q}$ indicates the weight for each neighboring signal, and $W_p = \int_{\mathcal{N}(p)} \omega_{p,q} dq$ is the normalization factor for ensuring the sum of all $\omega_{p,q}$ equal to 1. The construction of $\omega_{p,q}$ is very important, it directly relates to the performance of edge-preserving.

Here, we give the following weight which reflects the cumulative difference of input signal:

$$\omega_{p,q} = g(d^s(\varphi_p,\varphi_q);\sigma_s)g(d^r(\psi_p,\psi_q);\sigma_r)\,, \tag{2}$$

where we choose Gaussian function as kernel function $g(\cdot;\cdot)$, σ_s, σ_r are variance parameters and the function d^s and d^r are defined as:

$$\begin{cases} d^s(\varphi_p,\varphi_q) = (\int_L ||\varphi_{q_{i+1}} - \varphi_{q_i}||^n ds\,)^{1/n} \\ d^r(\psi_p,\psi_q) = (\int_L ||\psi_s - \psi_p||^m ds\,)^{1/m} \end{cases}. \tag{3}$$

Note that L is the geodesic path connecting points p and q. φ and ψ can represent different signal values, for example position, Gaussian curvature and normal on a 2D manifold surface. q_{i+1} and q_i are adjacent points in path L(???).

The cumulative difference of input signal can reflect the local structure of desired signal. The main reason is that we consider the difference not only between adjacent signals but also start-end signals. According to different problems, we can choose suitable parameters φ, ψ, n and m.

Equation 3 which reflects the similar differences between signals can be approximated as:

$$\begin{cases} d^s(\varphi_p,\varphi_q) = (\sum_{q_{i+1},q_i\in L} \|\varphi_{q_{i+1}} - \varphi_{q_i}\|^n \|q_{i+1} - q_i\|\,)^{1/n} \\ d^r(\psi_p,\psi_q) = (\sum_{q_{i+1},q_i\in L} \|\psi_{q_i} - \psi_p\|^m \|q_{i+1} - q_i\|\,)^{1/m} \end{cases}. \tag{4}$$

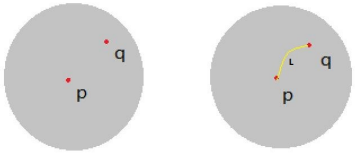

Figure 1: *The relation between bilateral filtering and ours(change).*

Figure 2: *Explain the shortest path.*

We define $\|q_{i+1} - q_i\|$ as a step length in the above formula. As image is a special 2D manifold, some classical filtering methods [TM98, GS09, CW15] are special cases of our method. We think about the 4-neighbor connecting in image, so the step length is $||q_{i+1} - q_i|| = 1$ in image filter. Next, we explain why our method are generalized.

Case 1. Bilateral filter [TM98], as a classical nonlinear filter, uses two Gaussian functions as the spatial and range weights respectively. In our generalized model 4, we use two points p and q to replace the geodesic path L. At the same time, signal functions φ and ψ are defined as $\varphi_x = x$ and $\psi_x = I_x$, where x is the pixel position and I_x respectives the pixel intensity. When $m = 2$ and $n = 2$, our model is simplified as:

$$\begin{cases} d^s(p,q) = ||p-q|| \\ d^r(I_p, I_q) = ||I_p - I_q|| \end{cases}. \tag{5}$$

We can see that the above formulation gets the distance differences, which are used to calculate the weights in bilateral filtering.

Case 2. Geodesic filtering [GS09] only uses accumulative difference in adjacent pixel signals during filtering process. This way gives a high response in image edges, so has a better edge-preserving power than bilateral filtering. Namely, when $n = 2$, the image intensity value I as a signal value φ and $d^r(\psi_p, \psi_q) = 1$, our intrinsic filtering is simplified, gets the distance difference between intensities of geodesic image filtering:

$$\begin{cases} d^s(I_p, I_q) = (\sum_{q_{i+1}, q_i \in L} ||I_{q_{i+1}} - I_{q_i}||^2)^{1/2} \\ d^r(\psi_p, \psi_q) = 1 \end{cases}. \tag{6}$$

Case 3. In a similar way, our intrinsic filtering is simplified as a propagation image filtering [CW15] by $n = 2$, $m = 2$, φ and ψ both represent the pixel intensity. Therefore,

$$\begin{cases} d^s(I_p, I_q) = (\sum_{q_{i+1}, q_i \in L} ||I_{q_{i+1}} - I_{q_i}||^2)^{1/2} \\ d^r(I_p, I_q) = (\sum_{q_i \in L}) ||I_{q_i} - I_p||^2)^{1/2} \end{cases}. \tag{7}$$

It is more comprehensive in considering the difference between signals. It has an outstanding power in preserving the image edge structures.

Our intrinsic filtering builds connections between p and q, which providing more reliable information about the structure of the desired output. This relation can be depicted in Figure 1. Our method inherits the advantage of these filtering algorithms [TM98, GS09, CW15] while compensating their deficiencies, and becomes more adaptive to the context of input signals.

3.2. Filtering mesh geometry

Given a noisy triangle mesh, our goal is to filter the noise while keeping the mesh structure. In order to achieve this purpose, we adapt a two-stage process which be widely used in mesh filtering. Firstly, noisy face normals are filtered iteratively by our intrinsic filtering model. Secondly, according to the filtered face normals, vertex positions are updated iteratively though gradient descent.

Filtering face normals. When we consider normals as a surface signal defined over the original triangle mesh, it is easy to put the intrinsic filtering algorithm to mesh denoising. For a triangle face f_i, its outward normal $\mathbf{n_i}$ can be calculated by outer product easily. Then we consider $\mathbf{n_i}$ as a signal associated with the face centroid c_i. In order to filter the face normals $\mathbf{n_i}$, we need find a path L to connect $\mathbf{n_i}$ and its neighbor $\mathbf{n_j}$. Finally, a filtered face normal $\bar{\mathbf{n}}_\mathbf{i}$ is computed through our intrinsic mesh filtering model:

$$\bar{\mathbf{n}}_\mathbf{i} = \frac{1}{W_i} \sum_{f_j \in \mathcal{N}_i} A_j g(d^s(\mathbf{n_i}, \mathbf{n_j}); \sigma_s) g(d^r(\mathbf{n_i}, \mathbf{n_j}); \sigma_r) \mathbf{n_j}. \tag{8}$$

here,

$$\begin{cases} d^s(\mathbf{n_i}, \mathbf{n_j}) = (\sum_{x, x+1 \in L} ||\mathbf{n_{x+1}} - \mathbf{n_x}||^n)^{1/n} \\ d^r(\mathbf{n_i}, \mathbf{n_j}) = (\sum_{x \in L} ||\mathbf{n_x} - \mathbf{n_i}||^m)^{1/m} \end{cases}, \tag{9}$$

where W_i is the normalization factor, calculated by $||\sum_{f_j \in \mathcal{N}_i} A_j g(d^a(\mathbf{n_i}, \mathbf{n_j}); \sigma_s) g(d^r(\mathbf{n_i}, \mathbf{n_j}); \sigma_r)||$ which ensures that $\bar{\mathbf{n}}_\mathbf{i}$ is a unit normal; A_j is the area of face f_j; $\mathcal{N}_i$ is the neighborhood of face f_i. In our paper, we use the geometrical neighborhood, defined at the paper [ZDZ*15].

Note that equation 8 can be derived from equation 1. A_j is the accumulation of infinitesimal area because the domain of integration is triangle face. Furthermore, we give up the step length which measures the distance between adjacent centroid of triangle faces. We simply make this term be one for facilitating the calculation.

Although the geodesic algorithm gives the shortest distance d^s and d^r, it is computationally expensive. Considering the shortcomings, we introduce a simple, fast and effective pattern for choosing filtering paths which is shown in next section. We find that this operation is about an order of magnitude faster than applying shortest path in obtaining similar results. Furthermore, the filter results that applying the particular paths often are more powerful on dealing with details, like edges and corners. These results are shown in the figure 2.

For triangle mesh, we conduct the experiment in the presence of m and n taking different values. We mainly implement three group experiments. Figure 3 show the denoised results of different conditions. We find that $m = 1$ and $n = 1$ can obtain the best results because of the sparseness of mesh feature. From the figure 3, the L_1 norm is more sensitive to details, so has a better expression.

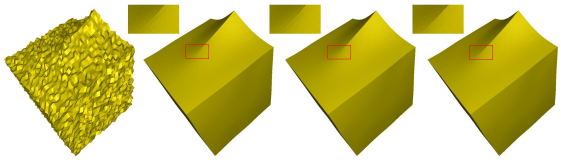

Figure 3: *From the left to right: Noised mesh (Gaussian noise with $0.3\sigma_E$), the results of $m = n = 2$ with step length, $m = n = 2$ without step length, $m = n = 1$ without step length. Because of the sparseness of mesh feature, L_1 norm receives the best results in preserving the mesh structure.*

Updating vertices. After all face normals is filtered, the vertex positions need to be updated to cater to these new normals. We adopt the iterative scheme in the paper [SRML07] to update the vertex positions. Namely, for a face f_i, we calculate its updated vertex positions via the following iteration form

$$\bar{v_i}^{(t+1)} = \bar{v_i}^{(t)} + \frac{1}{|\mathcal{F}_i|} \sum_{j \in \mathcal{F}_i} \bar{\mathbf{n}}_\mathbf{j} [\bar{\mathbf{n}}_\mathbf{j} \cdot (\bar{c_j}^{(t)} - \bar{v_j}^{(t)})], \qquad (10)$$

where $\bar{v_j}^{(t)}$ is the value of $\bar{v_i}$ in the $t-th$ iteration, $\mathcal{F}_i$ is the index set of the incident faces for $\bar{v_i}$, $|\cdot|$ denotes the cardinality of a set, and $\bar{c_j}^{(t)} = \sum_{i=1}^{3} \bar{v_{j_i}}^{(t)}/3$ is the centroid of the triangle f_j. This scheme is actually based on the orthogonality between the normal and the three edges of each face on the mesh. Then adopts a gradient descend process for solving that orthogonality equation in the conditions of L_2 error.

In our experiments, 10 to 20 iterations are sufficient for achieving satisfactory results and approximately up to above conditions. Our filtered process is summarized in Algorithm 1. In the next section, we introduce a simple but effective method for selecting a particular pattern for our propagated mesh filtering.

Algorithm 1 Propagation mesh filtering framework

Input: Initial mesh M_{in}, number of iterations k_{iter}.
Output: Filtered mesh M_{out}.
1: $M^{(0)} = M_{in}$
2: **for** $s = 1$ to k_{iter} **do**
3: Compute face normals n_i of mesh $M^{(s-1)}$;
4: Get paths from $\mathcal{N}_i$ to n_i according to section 4;
5: Compute filtered normals $\bar{\mathbf{n}}_\mathbf{i}$ according to 8;
6: Compute updated mesh $M^{(s)}$ according to $\bar{\mathbf{n}}_\mathbf{i}$;
7: **end for**
8: $M_{out} = M^{k_{iter}}$.

4. Path chosen for mesh filtering

In image, many particular patterns can be chosen because of the regular coordination system. These patterns are simple and easy to be thought about. But in triangle mesh, it is very difficult to obtain a good pattern which can preserve the mesh feature. The easiest way to be thought of is applying the shortest path algorithm on the triangle mesh. However, figure 2 shows this method is time consuming. We solve this problem in this paper within the context of mesh denoising.

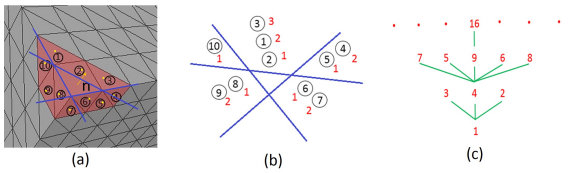

Figure 4: *The process of generation path. (a) the 10 neighbors of current face with its normal* $\mathbf{n}$*. (b) The schematic diagram after projection, dividing and sorting. (c) the pattern. The numbers with circle are the neighbor of current triangle face, the yellow points are the projection points of neighbor face centroid, the blue lines are used to decide the face label, the red numbers are their serial number after sorting, the green lines are connected path.*

Our method is based on the following theory: the areal coordinates of triangle segments the plane into seven regions. Areal coordinates are extremely useful in engineering applications involving triangular subdomains. These make analytic integrals often easier to evaluate, and Gaussian quadrature tables are often presented in terms of area coordinates. In the context of a triangle, the areal coordinates of a point P are the ratios of the areas of PBC, PCA and PAB to the area of the reference triangle ABC. Because the ratios are a plus or a minus, it divide the plane where the triangle is to seven parts. That gives us an enlightenment to obtain a particular pattern for solving the problem of paths.

Path generation. For each triangle f_i, we project its neighborhood $\mathcal{N}_i$ onto a tangent plane basing its normal $\bar{\mathbf{n}}_\mathbf{i}$. In detail, we only project their centroid onto that plane. Then basing the three points of f_i, we use areal coordinations to divide these centroid to seven regions. Note that, if the projection point falls into the face f_i, we throw it away from $\mathcal{N}_i$. In this way, we give each neighbor a label that belong to a specified region. Afterwards, we sort these neighbors belong to the same region according to their face centroid distance to c_i in original noisy mesh. Finally, we provide a particular pattern to assign the path for matching the ordered neighbors. This pattern is defined square-pattern, which is used for generating paths. The entire process is depicted in the figure 4.

Advantages. The above process has the power to protect local mesh feature like edge and corner. To a certain extend, the six regions surrounding f_i depict the local structure of a mesh. Projecting along the normal of f_i makes the faces having similar normal flock together. Furthermore, the division further restricts the normal difference according to areal coordinates. The combination of these two aspects insures that one of six regions has the similarity to f_i. The part including faces ①, ② and ③ has the similar normal to f_i in the figure 4(b). These normals play a important role in applying our intrinsic filtering algorithm.

Figure 4(c) shows the square-pattern for generating path. We use $n^2 (n = 1, 2, ...)$ as connection points, the sequence numbers between $(m-1)^2$ and m^2 including m^2 directly connect the number $(m-1)^2$. In this way, paths are generated. For example, the path between number 8 and f_i is $8-4-1-f_i$, number 16 and f_i is $16-9-4-1-f_i$ and so on.

In the next section, we will introduce the experimental results basing our method.

Figure 5: *Denoising results using different values of σ_r with other parameters fixed.*

5. Results

5.1. Parameters

Our method has four parameters: the number of normal filtering iterations k_{iter}, the number of iterations v_{iter} for vertex update, choosing a radius parameter r determined the geometrical neighborhood, and the standard variance parameters σ_s and σ_r, for the two difference between normals, respectively. In our experiments, we find that $k_{iter} \leq 80$ and $v_{iter} \leq 5$ are enough for achieving satisfactory results. For all results in this paper, we set the same value to σ_s and σ_r for employing the filtering unless stated otherwise. And when their values are between 0.2 and 0.8, a good experimental result can be achieved. Later, we will introduce the function of the standard variance parameters. In addition, we choose $r \in [2d, 6d]$ where d is the average distance between neighboring face centroids across the whole mesh. For CAD model, a bigger r will make a better results. The reason is that some one of six regions provides more reliable normal estimation.

5.2. Results and comparisons

In the following, we compare our denoising results with other methods on synthetic noisy meshes and real meshes.

We generate the synthetic meshes though adding noise to the vertices of a ground truth mesh along the vertex normals. The intensity of the noise is defined by a relative standard deviation $\sigma_E = \sigma / E_{mean}$, where σ is the standard deviation of the Gaussian function and E_{mean} is the average edge length of the ground truth mesh. For a fair comparison, for each method we fine tune the parameters to produce the best results in their parameter space. And furthermore, for synthetic meshes, we also evaluate two quantitative metrics about vertex-based and normal-based mesh-to-mesh error metric separately introduced in papers [BO03] and [NH00] to analyze the difference between our results and others. These two metrics measure mean deviations about vertex and face between denoised mesh and true mesh.

The vertex-based error metric is

$$E_v = \sqrt{\frac{1}{3A(M')} \sum_{P' \in M'} A(P') dist(P', M)^2}, \quad (11)$$

where M and $M^{'}$ reference original noiseless mesh and denoised mesh respectively, $P^{'}$ is a vertex of $M^{'}$ and $dist(P^{'}, M)$ defines the distance between $P^{'}$ and a triangle of M which is closest to $P^{'}$.

The normal-based error metric is used to reveal the average angle offset which is defined as follows

$$E_n = \sum_{f' \in M', f \in M} angle(\mathbf{n}^{'}, \mathbf{n}) / N, \quad (12)$$

Table 1: *Time consumption for different results.*

Model	Vertics	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}
Fandisk (Fig 6)	Vertices	Faces	Time(s)	k_{iter}

where $f^{'}$ and f are triangle faces of $M^{'}$ and M respectively, $\mathbf{n}^{'}$ and $\mathbf{n}$ reference the normal of $f^{'}$ and f, $angle(\mathbf{n}^{'}, \mathbf{n})$ defines the angle between $\mathbf{n}^{'}$ and $\mathbf{n}$ and the last N is the number of triangle faces of a mesh.

Comparison with other methods on the synthetic meshes. Figure **??** shows the effectiveness of our method on preserving weak edges and sharp corners. From the results of first line on a noisy octahedron, the approaches of Fleishman et al [FDCO03] and Jones et al [JDD03] can not well preserve the strong edges, because the spatial weight weaken the strength of filtering. While the rest of methods have the ability to maintain the mesh edge structure, they can not well deal with the sharp corner. The main reason is that the methods of Sun et al [SRML07] and Zheng et al [ZFAT11](l) easily smooth the sharp corner, The approaches of He et al [HS13] and Zhang et al [ZDZ*15] are prone to ambiguity. Because our method divide the neighbor sets, it can well preserve the sharp corner and edge. The second line is the denoised results on a prism with 18 edges with the Gaussian noise intensity $\sigma_E = 0.1$. Other methods can not well restore these weak edges and easily smooth them. We do well in the two models not only because our denoised method but also the function of six regions which give a better estimation in the structure of local mesh.

The models in figures 6 and 7 all selected from [ZDZ*15]. For most of models, our method achieves better results than others according to the table 2. The detail of parameters can be found in the supplementary material.

In the figure 7, we show our method also can deal with different kinds of noise. From the vision, our method almost obtains the similar results comparing with Zhang et al [ZDZ*15]. However, according to the table 2, our method works well.

Comparison with other methods on the real noisy meshes. Figure 8 witnesses the denoised effectiveness on real world 3D objects.

Table 1 provides the timing of our approach for the shown examples, on a PC with an Intel Core $i7-4790K$.

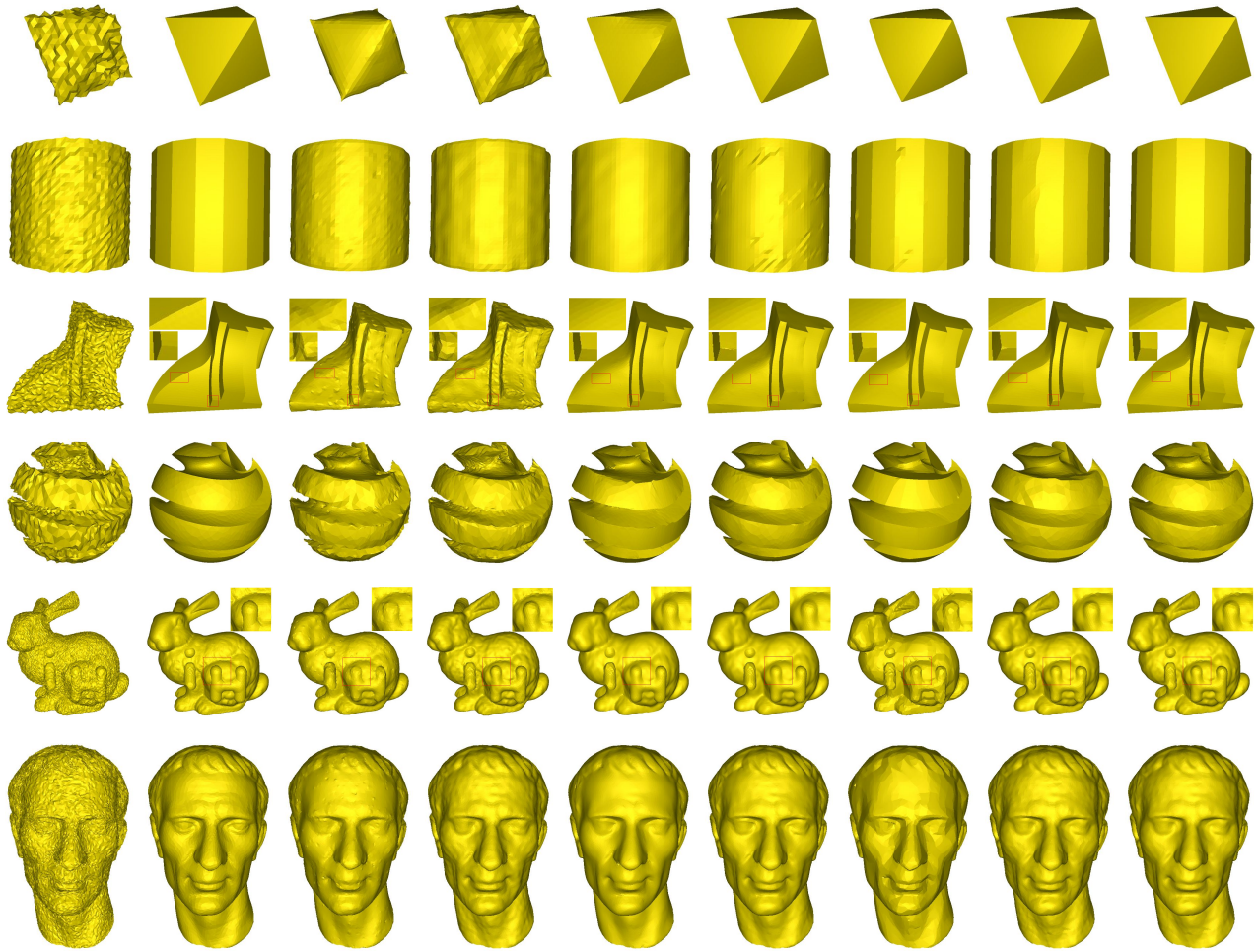

Figure 6: *Comparisons with other methods on synthetic meshes with additive Gaussian noise. The results are from left to right noisy mesh, original mesh , Fleishman et al [FDCO03], Jones et al [JDD03], Sun et al [SRML07], Zheng et al [ZFAT11](l), He et al [HS13], Zhang et al [ZDZ*15] and our result separately. The intensity σ_E of the noise is from the top to bottom 0.3, 0.1, 0.3, 0.3, 0.2 and 0.2.*

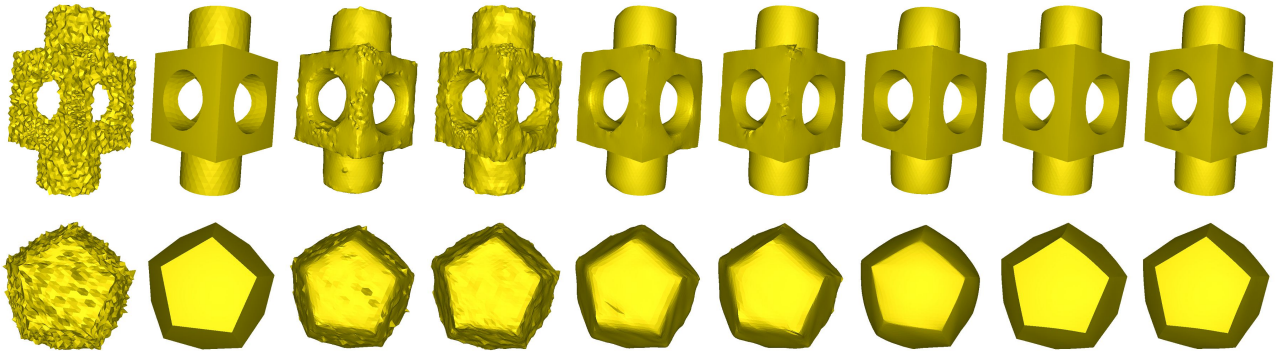

Figure 7: *Comparisons with other methods on synthetic meshes. One mesh with non-uniform sampling and the Gaussian noise intensity is $\sigma_E = 0.4$, the other mesh with impulsive noise and the intensity is $\sigma_E = 0.5$. The results are from left to right noisy mesh, original mesh , Fleishman et al [FDCO03], Jones et al [JDD03], Sun et al [SRML07], Zheng et al [ZFAT11](l), He et al [HS13], Zhang et al [ZDZ*15] and our result separately.*

Table 2: *Quantitative comparisons using two error metrics.*

Model	Error	[FDCO03]	[JDD03]	[SRML07]	[ZFAT11](l)	[HS13]	[ZDZ*15]	ours
Prism (Fig 6)	E_n	4.98	4.48	3.29	3.87	0.71	0.87	**0.46**
	$E_v(\times 10^{-2})$	11.89	10.46	7.45	7.9	3.42	4.09	**2.77**
octahedron (Fig 6)	E_n	12.81	10.65	6.85	5.27	3.33	3.36	**1.00**
	$E_v(\times 10^{-3})$	28.02	14.70	10.71	6.28	10.81	4.90	**3.02**
Fandisk (Fig 6)	E_n	9.73	9.82	4.04	3.35	5.53	2.62	**2.27**
	$E_v(\times 10^{-3})$	18.14	15.10	11.02	8.72	18.79	**6.29**	6.35
sphere (Fig 6)	E_n	12.58	17.36	11.89	**6.70**	12.96	10.17	7.39
	$E_v(\times 10^{-2})$	15.51	8.46	8.88	4.48	12.41	5.65	**4.37**
bunny (Fig 6)	E_n	6.93	5.81	5.89	5.68	7.21	5.35	**5.11**
	$E_v(\times 10^{-4})$	18.36	9.24	9.98	9.35	10.65	7.61	**7.26**
Julius (Fig 6)	E_n	7.70	7.63	7.01	6.21	7.98	6.38	**6.11**
	$E_v(\times 10^{-4})$	10.58	7.71	6.49	**5.53**	8.60	6.02	5.78
block (Fig 7)	E_n	12.72	13.85	5.80	5.31	4.97	3.57	**3.02**
	$E_v(\times 10^{-2})$	17.98	13.35	8.21	6.80	11.60	5.41	**4.99**
twelve (Fig 7)	E_n	11.72	11.09	7.45	7.37	8.46	2.75	**1.78**
	$E_v(\times 10^{-3})$	20.85	17.28	12.06	12.54	20.13	6.16	**5.23**

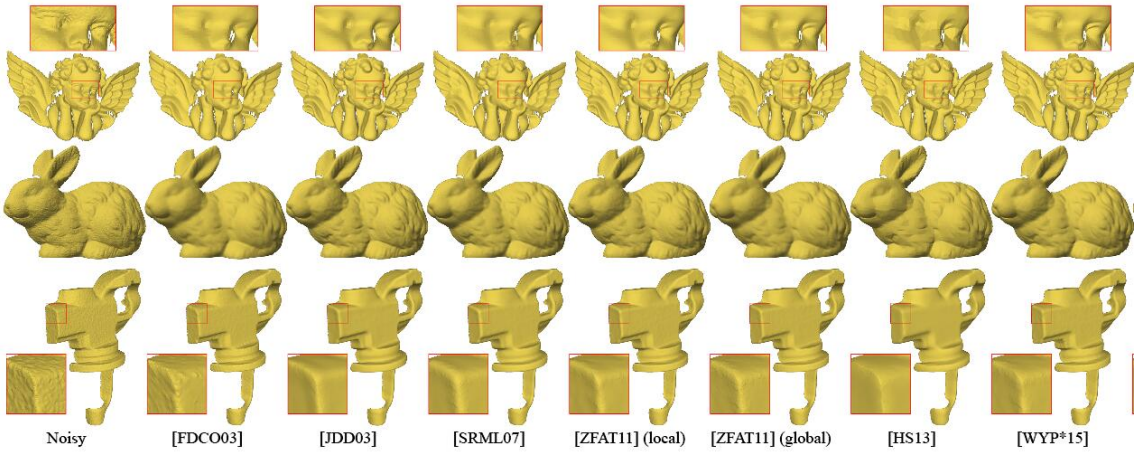

Figure 8: *true mesh.*

5.3. Limitation and discussion

Figure 9 shows the convergence of our method.

6. Conclusion

In this paper, we introduce a intrinsic signal filtering framework for denoising 2D manifold surface. Intrinsically, our method builds the relation between desired filtering signals and its neighbors, making calculate weights more reasonably. Other famous filtering algorithm, such as bilateral, geodesic and propagation filters, can be simplified by our algorithm. Furthermore, we apply our filtering framework to triangular meshes and obtain state-of-the-art performance. We also propose a simple, fast and effective method for choosing path in the process of intrinsic filtering algorithm instead of geodesic path. And, the approach for dividing region protects the local mesh structure in a certain extent, further increasing the effectiveness of mesh filter. Finally, a large number of experiments prove the effectiveness of our method.

Acknowledgements

References

[BC04] BARASH D., COMANICIU D.: A common framework for nonlinear diffusion, adaptive smoothing, bilateral filtering and mean shift. *Image and Vision Computing 22*, 1 (2004), 73–81. 2

[BO03] BELYAEV A., OHTAKE Y.: A comparison of mesh smoothing

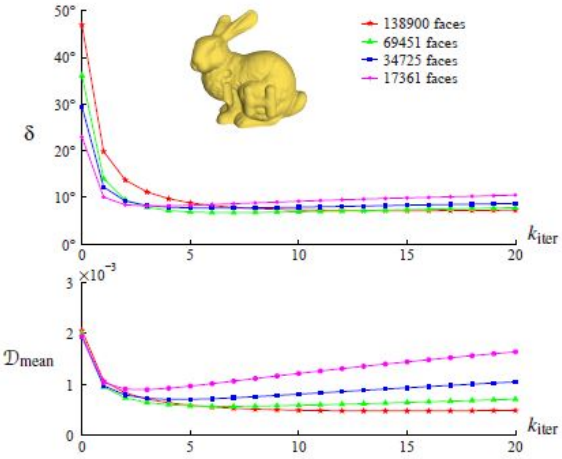

Figure 14: *Convergence plots of error metrics δ and $\mathcal{D}_{mean}$ on bunny models with different resolutions. The values of $\mathcal{D}_{mean}$ are normalized by the bounding box diagonal length of the ground truth model. Our method achieves desired results on lower resolution models faster than higher resolution ones.*

Figure 9: *the convergence.*

methods. In *Israel-Korea Bi-national conference on geometric modeling and computer graphics* (2003), vol. 2, Citeseer. 5

[BT11] BIAN Z., TONG R.: Feature-preserving mesh denoising based on vertices classification. *Computer Aided Geometric Design 28*, 1 (2011), 50–64. 2

[CC05] CHEN C.-Y., CHENG K.-Y.: A sharpness dependent filter for mesh smoothing. *Computer Aided Geometric Design 22*, 5 (2005), 376–391. 2

[CW15] CHANG J.-H. R., WANG Y.-C. F.: Propagated image filtering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), IEEE, pp. 10–18. 1, 3

[DD02] DURAND F., DORSEY J.: Fast bilateral filtering for the display of high-dynamic-range images. In *ACM transactions on graphics (TOG)* (2002), vol. 21, ACM, pp. 257–266. 2

[FDCO03] FLEISHMAN S., DRORI I., COHEN-OR D.: Bilateral mesh denoising. In *ACM transactions on graphics (TOG)* (2003), vol. 22, ACM, pp. 950–953. 1, 2, 5, 6, 7

[FYP10] FAN H., YU Y., PENG Q.: Robust feature-preserving mesh denoising based on consistent subneighborhoods. *IEEE Transactions on Visualization and Computer Graphics 16*, 2 (2010), 312–324. 2

[GS09] GRAZZINI J., SOILLE P.: Edge-preserving smoothing using a similarity measure in adaptive geodesic neighbourhoods. *Pattern Recognition 42*, 10 (2009), 2306–2316. 1, 3

[HS13] HE L., SCHAEFER S.: Mesh denoising via l_0 minimization. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 64. 2, 5, 6, 7

[HST10] HE K., SUN J., TANG X.: Guided image filtering. In *European conference on computer vision* (2010), Springer, pp. 1–14. 1

[JDD03] JONES T. R., DURAND F., DESBRUN M.: Non-iterative, feature-preserving mesh smoothing. In *ACM Transactions on Graphics (TOG)* (2003), vol. 22, ACM, pp. 943–949. 1, 2, 5, 6, 7

[NH00] NEHORAI A., HAWKES M.: Performance bounds for estimating vector systems. *IEEE Transactions on Signal Processing 48*, 6 (2000), 1737–1749. 5

[OCDD01] OH B. M., CHEN M., DORSEY J., DURAND F.: Image-based modeling and photo editing. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), ACM, pp. 433–442. 2

[PSA*04] PETSCHNIGG G., SZELISKI R., AGRAWALA M., COHEN M., HOPPE H., TOYAMA K.: Digital photography with flash and no-flash image pairs. *ACM transactions on graphics (TOG) 23*, 3 (2004), 664–672. 1

[SCBW14] SOLOMON J., CRANE K., BUTSCHER A., WOJTAN C.: A general framework for bilateral and mean shift filtering. *CoRR abs/1405.4734 8*, 9 (2014). 1, 2

[SRML07] SUN X., ROSIN P., MARTIN R., LANGBEIN F.: Fast and effective feature-preserving mesh denoising. *IEEE transactions on visualization and computer graphics 13*, 5 (2007), 925–938. 4, 5, 6, 7

[SRML08] SUN X., ROSIN P. L., MARTIN R. R., LANGBEIN F. C.: Random walks for feature-preserving mesh denoising. *Computer Aided Geometric Design 25*, 7 (2008), 437–456. 2

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on* (1998), IEEE, pp. 839–846. 1, 2, 3

[Wan06] WANG C. C.: Bilateral recovering of sharp edges on feature-insensitive sampled meshes. *IEEE Transactions on Visualization and Computer Graphics 12*, 4 (2006), 629–639. 2

[WFL*15] WANG P.-S., FU X.-M., LIU Y., TONG X., LIU S.-L., GUO B.: Rolling guidance normal filter for geometric processing. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 173. 2

[WLDB08] WANG K., LAVOUÉ G., DENIS F., BASKURT A.: A comprehensive survey on three-dimensional mesh watermarking. *IEEE Transactions on Multimedia 10*, 8 (2008), 1513–1527. 2

[WYL*14] WANG R., YANG Z., LIU L., DENG J., CHEN F.: Decoupling noise and features via weighted l_1-analysis compressed sensing. *ACM Transactions on Graphics (TOG) 33*, 2 (2014), 18. 2

[WYP*15] WEI M., YU J., PANG W.-M., WANG J., QIN J., LIU L., HENG P.-A.: Bi-normal filtering for mesh denoising. *IEEE transactions on visualization and computer graphics 21*, 1 (2015), 43–55. 2

[WZY12] WANG J., ZHANG X., YU Z.: A cascaded approach for feature-preserving surface mesh denoising. *Computer-Aided Design 44*, 7 (2012), 597–610. 2

[YOB02] YAGOU H., OHTAKE Y., BELYAEV A.: Mesh smoothing via mean and median filtering applied to face normals. In *Geometric Modeling and Processing, 2002. Proceedings* (2002), IEEE, pp. 124–131. 2

[ZDZ*15] ZHANG W., DENG B., ZHANG J., BOUAZIZ S., LIU L.: Guided mesh normal filtering. *Computer Graphics Forum (Special Issue of Pacific Graphics 2015) 34* (2015), 23–34. 2, 3, 5, 6, 7

[ZFAT11] ZHENG Y., FU H., AU O. K.-C., TAI C.-L.: Bilateral normal filtering for mesh denoising. *IEEE Transactions on Visualization and Computer Graphics 17*, 10 (2011), 1521–1530. 1, 2, 5, 6, 7

[ZSXJ14] ZHANG Q., SHEN X., XU L., JIA J.: Rolling guidance filter. In *European Conference on Computer Vision* (2014), Springer, pp. 815–830. 2