Mesh Denoising Using Quadric Error Metric

Antonio W. Vieira*†
* *Departamento de Ciências Exatas*
Universidade Estadual de Montes Claros
Montes Claros, Brazil
Email: awilson@dcc.ufmg.br

Armando A. Neto†, Douglas G. Macharet†, Mario F. M. Campos†
† *Departamento de Ciências da Computação*
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil
Email: {aaneto, doug, mario}@dcc.ufmg.br

(a) (b) (c)

Figure 1. (a) Stanford Bunny model with 35.947 vertices. (b) Corrupted with zero mean Gaussian noise with standard deviation $\sigma = 0.4$ of mean edge length. (c) Denoised with our method. Notice that no oversmoothing occurs along the surface and that, despite huge noise level, sharp details were well recovered in the eyes and also enhanced in the ears.

***Abstract*—In this work we present a new method for mesh denoising that uses an operator based on the Quadric Error Metric. This operator is able to estimate the local shape of the surface for each vertex, despite severe noise condition, distinguishing corners, edges and smooth regions in order to best adjust the vertex geometry to recover piecewise smoothing while preserving sharp features. Our method results in a simple algorithm for mesh denoising that can also be used to enhance sharp features present in the surface corrupted by noise. A frequency response analysis is also presented in order to evaluate the characteristics of this operator in the frequency spectrum of the mesh.**

***Keywords*-Point cloud; 3D Mesh denoise; Quadric error metric.**

I. Introduction

The use of three-dimensional ($3D$) point clouds is increasing in many areas of research and development. $3D$ scanning tools and computer vision techniques provide reconstruction of many objects and structures using range images or unstructured point cloud, like those used in robotics mapping. Surface extraction algorithms, such as $MarchingCubes$, allow the access to large point clouds sampled from implicit set of theoretical or physical data, for example in computed tomography and seismic analysis. Point cloud sets are subject to noise from various sources and its attenuation or removal, while preserving features, is an important preprocessing step for most applications.

While high-frequency filtering algorithms usually attenuate noise by smoothing the surface, a denoising algorithm should implement more specialized operators to identify and preserve sharp features in the model that should not be attenuated by the smoothing process. The area of digital image processing has produced many techniques for dealing with noise in images and most of them can be adapted to filter $3D$ point cloud sets. This is specially the case of point clouds obtained from range images, where a matrix structure is present and allows connectivity information among vertices and a support tangent space is well defined for all points.

When considering general $3D$ point cloud sets, one should take into account that no matrix structure is present, nor a support tangent space can be defined for all points. Hence, in order to adapt image processing algorithms to general $3D$ point clouds, a structure to define connectivity among points and to estimate a support tangent space to each point is usually provided.

Triangle meshes are a widely used data structure to define connectivity for general $3D$ point clouds and can represent surfaces of arbitrary topology, with or without boundary. However, when the source of a point cloud doesn't provide

connectivity, a mesh reconstruction algorithm, as presented in [1], [2], [3], must be used to obtain the connectivity as a triangle mesh. The support tangent plane for each vertex is obtained from its unitary normal vector $\mathbf{n}$, which can be estimated as the average of normals from its incident triangles.

In our mesh based denoising method we consider, as input, a point cloud structured in a triangle mesh. The advantage of a mesh structure is that smoothing and denoising are topology preserving procedures and, once the surface topology and connectivity are defined for the point cloud, just geometric adjustments are necessary. This adjustment is usually calculated for each vertex $\mathbf{v}$ using an operator

$$T(\mathbf{v}) = \mathbf{v} + k\mathbf{n} \ ,$$

where $\mathbf{n}$ is a vector normal at the vertex $\mathbf{v}$ in the mesh and k is estimated by the application of some filter. This operator constrains the space for adjustment of a vertex $\mathbf{v}$ to a one dimensional space defined by its normal vector. Therefore, the neighboring vertices cannot influence the direction for adjustment. In our method, a new approach is used, where, instead of estimating the amount of adjustment for each vertex along its normal, a new position to each vertex is directly computed as an optimization problem based on the Quadric Error Metric (QEM). We define the operator

$$F(\mathbf{v}) = -A^{-1}\mathbf{b} \ ,$$

where the 3×3 matrix A and the 1×3 vector $\mathbf{b}$ are blocks from the 4×4 matrix

$$Q = \begin{pmatrix} A & \mathbf{b} \\ \mathbf{b}^T & c \end{pmatrix}$$

calculated for the vertex $\mathbf{v}$ as the sum of quadrics associated to the tangent plane of its neighboring vertices, as detailed in Section III. Using this operator we have, at first, an unconstrained space for the adjustment of the vertex $\mathbf{v}$ that will be constrained to a single point using information from its neighboring vertices. This strategy was inspired in the mesh simplification process by Garland and Heckbert [4] that first introduced the QEM and its properties to describe local geometry on the mesh.

The main contribution of this work is to present a feature preserving approach, based on the properties of quadrics, to adjust the geometry of a noisy mesh in order to obtain a piecewise smooth mesh, while preserving details of its sharp features. Our method requires a single step and does not use iterative procedures, which is advantageous when compared to other iterative procedures that are executed repeatedly for many iterations depending on the noise level. Finally, our method is able to recover some corrupted details even in severe noise conditions.

II. Related Work

Although the processes of mesh denoising and mesh smoothing are different in terms of purposes, they are closely related and have been studied as common tools for signal filtering similar to those used for image processing. In [5], Taubin generalizes the classical discrete Fourier analysis used for two-dimensional discrete surface of intensity and range images in order to reduce the problem of surface smoothing of arbitrary topology to a low-pass filtering.

Still using the ideas from image processing, a mesh denoising algorithm based on locally adaptive Wiener filtering was proposed in [6] using a multi-resolution approach. It has disadvantage that the original connectivity of the mesh is not preserved in the process. An extension of QEM was used by Page et al. in [7] to develop a denoising technique combined with mesh simplification that, due to local smoothing and other computations, leads to results with corrupted mesh quality.

The connectivity and mesh quality were well preserved when filtering was applied in the normal direction of the vertex, using the local operator $T(\mathbf{v}) = \mathbf{v} + k\mathbf{n}$ to adjust each vertex $\mathbf{v}$, where k is a filtered adjustment value and $\mathbf{n}$, the vertex normal. This approach was used by Fleishman et al. in [8], where the main contribution is an adaptation of the Bilateral Filtering [9] from image denoising to $3D$ meshes, presenting an algorithm that is conceptually simple and easy to implement. An improvement to the Fleishman's algorithm using similar operator was presented by Madrazo et al. in [10] to generalize the method for point cloud denoising without any connectivity information in the input data set. It was inspired in the Moving Least Squares [11] and M-estimators robust statistics theory. Similar approach was proposed by Jones et al. [12].

Much effort has been invested to perform mesh denoising while preserving features. In [13], Fleishman et al. presented a new technique that determines a piecewise smooth surface using Moving Least-squares in order to define the sharp features to be preserved in the surface. Unnikrishnan and Hebert [14] proposed a method for denoising non-manifold point clouds, fitting local degenerate high-order polynomials to the data in order to represent and estimate high-frequency variations in point-sampled surfaces. In [15], Sun et al. used a two-step method based on the idea of random walks for mesh denoising, consisting of face normal filtering followed by vertex position update to integrate the denoised face normals in a least-squares manner. More recently, Fan, Yu and Peng [16] introduced a piecewise surface denoising method that seeks to preserve sharp features and volume, also based on bilateral filtering.

Image filtering based approaches, as Bilateral Filtering, try to locally fit a plane to the surface at each point, followed by the application of the filter to the neighborhood of the point. In such methods, a single plane is fitted to a point and serves as a parametrization, over which the filter is applied. However, as pointed out by Fleichman et al. [8], a point on a sharp edge, that these algorithms aim to preserve, resides on two planes rather than one. A known property of the QEM

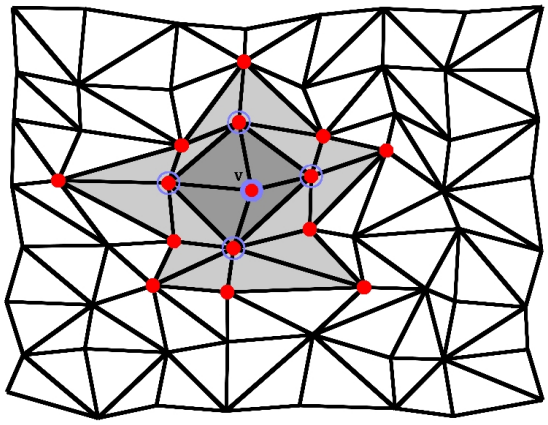


Figure 2. Example of a vertex $\mathbf{v}$ and its $N_2(\mathbf{v})$ neighbors in the mesh.

is that, given a point whose neighborhood is defined by a set of two or more planes, it is able to adjust this point so it best fits to all planes. This property has been used for mesh simplification algorithms [4], [17], [18], [19].

Many works on mesh simplification have also outlined their potential on mesh smoothing and denoising. While studying mesh simplification algorithms, Michael Garland [20] experimented his techniques using QEM to reduce noise in meshes by exploring its smoothing effect.

In this work, we explore the quadrics properties to adjust vertex position without proceeding with any simplification in the mesh in order to remove noise.

III. Methodology

A. Data Structure

We consider a point cloud structured in a triangle mesh as a set $M = \{V, T\}$, where V is a set of vertices and T a set of triangle faces. Although in our current implementation we consider a triangulated surface, the algorithm can be easily extended for any polyhedral surface or raw point cloud as long as a neighboring and vertex normals are provided.

For the purpose of this work, we define a first order neighborhood of a vertex $\mathbf{v}_i \in V$ as a set $N_1(\mathbf{v}_i)$ of vertices that share a common edge with $\mathbf{v}_i$, including itself. A second order neighborhood of $\mathbf{v}_i$ is the set $N_2(\mathbf{v}_i)$ of vertices that share a common edge with any vertex $\mathbf{v}_j \in N_1(\mathbf{v}_i)$. Recursively, a k^{th} order neighborhood of $\mathbf{v}_i$ is the set $N_k(\mathbf{v}_i)$ of vertices that share a common edge with any $\mathbf{v}_j \in N_{k-1}(\mathbf{v}_i)$.

Fig. 2 shows an example of a $N_2(\mathbf{v})$ neighboring set of a vertex $\mathbf{v}$. Notice that, due to the irregular connectivity of the usual mesh, the vertices in the boundary of the neighboring set may have distances to $\mathbf{v}$ that vary significantly. In order to well balance the influence of each $\mathbf{u} \in N_k(\mathbf{v})$, filtering based algorithms define a weight based on $\|\mathbf{u} - \mathbf{v}\|$.

Many data structures can be used to efficiently access the neighborhood of a vertex in a mesh. In this work we use Topological Mesh Operators [21] that is a data structure based on Half-Edge data structure and is also specialized for triangle meshes operations in connectivity and topology.

B. Quadric Error Metric

The QEM was proposed by Garland and Heckbert [4] for mesh simplification based on edge collapsing that preserves the local similarity between the simplified and original surfaces. Originally, the quadrics were defined, for each vertex $\mathbf{v}$, as the sum of quadrics associated to the support planes of the triangles incidents to $\mathbf{v}$.

For the purpose of our denoising algorithm, we will define the quadrics for each vertex $\mathbf{v}$ as the quadric associated to the plane given by the normal vector $\mathbf{n}$ estimated for $\mathbf{v}$.

The plane π defined from the normal vector $\mathbf{n}$ of vertex $\mathbf{v}$ is the set of all vectors $\mathbf{w}$ for which $\mathbf{n}^T\mathbf{w} = d$, where $\mathbf{n} = (n_x, n_y, n_z)^T$ is the unit normal vector to $\mathbf{v}$, $\mathbf{w} = (x, y, z)^T$ is a vector in $\mathbb{R}^3$ and d is a constant related to the perpendicular distance from origin to π. Hence, the signed distance from $\mathbf{w}$ to the plane π is given by

$$D(\mathbf{w}, \pi) = \mathbf{n}^T\mathbf{w} - d,$$

which can be written as

$$D(\mathbf{w}, \pi) = (n_x, n_y, n_z, -d)^T (x, y, z, 1).$$

Writing $\bar{\mathbf{n}} = (n_x, n_y, n_z, -d)^T$ and $\bar{\mathbf{w}} = (x, y, z, 1)$ we have the distance from $\mathbf{w}$ to the plane π in a simple notation

$$D(\mathbf{w}, \pi) = \bar{\mathbf{n}}^T\bar{\mathbf{w}}.$$

The squared distance from $\mathbf{w}$ to the plane π is then calculated using associative properties of matrix multiplication as the bilinear form

$$D(\mathbf{w}, \pi)^2 = \left(\bar{\mathbf{n}}^T\bar{\mathbf{w}}\right)^2 = \left(\bar{\mathbf{w}}^T\bar{\mathbf{n}}\right)\left(\bar{\mathbf{n}}^T\bar{\mathbf{w}}\right) = \bar{\mathbf{w}}^T\left(\bar{\mathbf{n}}\bar{\mathbf{n}}^T\right)\bar{\mathbf{w}}.$$

The 4×4 matrix $\bar{\mathbf{n}}\bar{\mathbf{n}}^T$ is called the quadric $Q_{\mathbf{v}}$ associated to the vertex $\mathbf{v}$. Once $Q_{\mathbf{v}}$ is defined for each $\mathbf{v} \in V$, the squared distance from $\mathbf{w} \in \mathbb{R}^3$ to the plane π tangent in $\mathbf{v}$ is given by the bilinear operation

$$D(\mathbf{w}, \pi)^2 = \bar{\mathbf{w}}^T Q_{\mathbf{v}} \bar{\mathbf{w}},$$

where

$$Q_{\mathbf{v}} = \bar{\mathbf{n}}.\bar{\mathbf{n}}^T = \begin{pmatrix} n_x n_x & n_x n_y & n_x n_z & n_x d \\ n_x n_y & n_y n_y & n_y n_z & n_y d \\ n_x n_z & n_y n_z & n_z n_z & n_z d \\ n_x d & n_y d & n_z d & d^2 \end{pmatrix}.$$

An important property of quadrics is that they can be summed up easily for the entire neighborhood $N_k(\mathbf{v})$ of a vertex $\mathbf{v}$, and the summed displacement $F(\mathbf{w})$ of a point $\mathbf{w}$ relative to all tangent planes in that neighborhood is retrieved in a single bilinear application

$$F(\mathbf{w}) = \bar{\mathbf{w}}^T \left(\sum_{\mathbf{u} \in N_k(\mathbf{v})} Q_{\mathbf{u}} \right) \bar{\mathbf{w}}.$$

C. Algorithm Outline

Given a vertex $\mathbf{v} \in V$, it is expected that its position in the neighborhood $N_k(\mathbf{v})$ has the smallest possible displacement to the shape defined by the tangent spaces in that neighborhood. So, in order to best fit $\mathbf{v}$ in the shape defined by $N_k(\mathbf{v})$, we adjust it to a new position $\mathbf{w} \in \mathbb{R}^3$ that minimizes the summed displacement from $\mathbf{w}$ to all tangent planes in $N_k(\mathbf{v})$. This $\mathbf{w}$ is the one that returns the smallest value from the bilinear form

$$F(\mathbf{w}) = \bar{\mathbf{w}}^T \left(\sum_{\mathbf{u} \in N_k(\mathbf{v})} Q_{\mathbf{u}} \right) \bar{\mathbf{w}}.$$

In order to well model our optimization problem, we write this summed quadrics in blocks

$$\begin{pmatrix} A & \mathbf{b} \\ \mathbf{b}^T & c \end{pmatrix} = \sum_{\mathbf{u} \in N_k(\mathbf{v})} Q_{\mathbf{u}}.$$

Therefore we have a more feasible bilinear form to evaluate,

$$F(\mathbf{w}) = \begin{pmatrix} \mathbf{w}^T & 1 \end{pmatrix} \begin{pmatrix} A & \mathbf{b} \\ \mathbf{b}^T & c \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ 1 \end{pmatrix},$$

or equivalently,

$$F(\mathbf{w}) = \mathbf{w}^T A \mathbf{w} + 2\mathbf{b}^T \mathbf{w} + c.$$

The minimal value for the bilinear form $F(\mathbf{w})$ occurs in $\mathbf{w}$ such that

$$\nabla F = 2A\mathbf{w} + 2\mathbf{b} = 0.$$

Hence, we have to solve only a simple 3×3 linear system to find the optimal position $\mathbf{w}$ for vertex $\mathbf{v}$. If A is not a singular matrix, $\mathbf{w}$ is given by

$$\mathbf{w} = -A^{-1}\mathbf{b}.$$

Fig. 3 illustrates the behavior of our algorithm in a sharp region of the mesh. We give a vertex $\mathbf{v}$ and its second order neighbors $N_2(\mathbf{v}) = \{v, u_1, u_2, u_3, u_4\}$. Note that, the point $\mathbf{w}$ that minimizes summed distance to all tangent spaces in $N_2(\mathbf{v})$ should lie in the region marked with an empty circle. This point adjustment in sharp regions of the mesh is a strong characteristic of our method which allows feature preserving and enhancing, while smoothing noise in planar regions as illustrated in Fig. 4. In our experiments, we present an analysis of this behavior, concerning the frequency response of our methodology.

The aforementioned process leads to our denoising algorithm that is described in the pseudocode presented in Algorithm 1. The inputs are a mesh $M = \{V, T\}$ and a parameter k to the order of neighborhood. The output is a mesh $\bar{M} = \{\bar{V}, T\}$ where only the geometry is changed and connectivity is preserved.

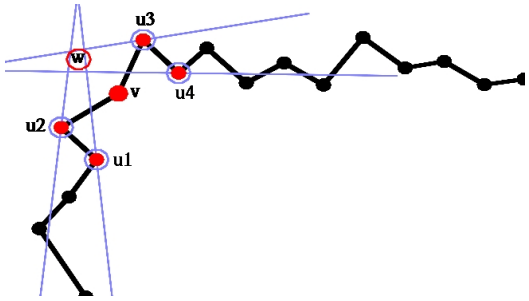


Figure 3. Example of a local sharp feature point adjustment.

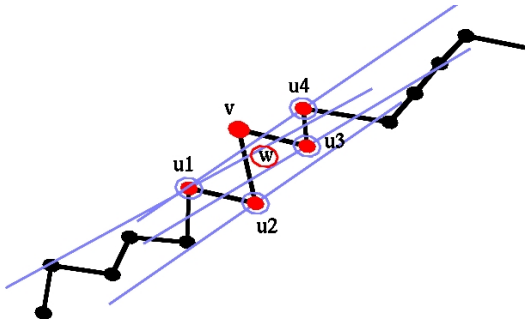


Figure 4. Example of a planar region point adjustment.

IV. Experiments

A. Results

In this section we discuss the experimental results. Firstly, our experiments were performed using a computer with the following configuration:

- *Hardware*: Notebook Dell Inspiron 1525 with an Intel Core 2 Duo CPU running at 2.0 GHz and 2 GB RAM;
- *Software*: Operational system Ubuntu Linux 9.04, compiler gcc and graphic library resources of OpenGL.

In our first and second experiments, we used models with natural acquisition noise in order to explore the potentials of

Algorithm 1 Denoise(M, k)

for each $\mathbf{v} \in V$ **do**
  Set Quadric $Q_{\mathbf{v}}$
end for
for each $\mathbf{v} \in V$ **do**
  $Q = Q_v$
  for each $\mathbf{u} \in N_k(\mathbf{v})$ **do**
    $Q = Q + Q_{\mathbf{u}}$
  end for
  $A = Q(1..3, 1..3)$
  $\mathbf{b} = Q(1..3, 4)$
  $\mathbf{w} = -A^{-1}\mathbf{b}$
  $\mathbf{v} = \mathbf{w}$
end for

our method in real situations. In Fig. 5 (a) and (b) we show a CSG model with sharp features that has been corrupted by the surface extraction algorithm, without any artificial noise, and its denoised version using our algorithm. The same model is shown in Fig. 5 (c) and (d) where the mesh edges are shown as an example of the way the algorithm adjusts vertex positions that tend to concentrate in sharp regions. Notice that the vertices near the sharp feature edges on the model moved along its tangent plane to lie close to the edge, enhancing it. Here we can also see that, although preserving connectivity, the quality of the mesh is also improved in this case considering the triangle aspect ratio that is enhanced in smooth regions of the surface.

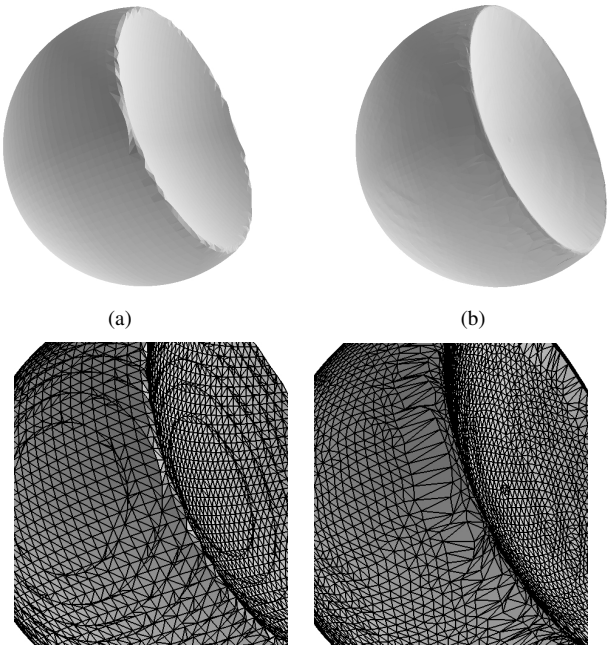


Figure 5. Details on the quality of the mesh. Original flat (a), denoised flat (b) and connectivity details on corrupted sharp region feature (c) recovered by our method (d).

Our method has presented some peculiar behavior compared to traditional filter based methods. While most filter based methods, for mesh denoising, tend to reduce volume due to the shrinkage effect, our method doesn't have this drawback. It is also remarkable that while most methods tend to oversmooth the surface, ours tends to enhance features. In our second experiment, using a scanned face model from INRIA Gamma team [22], this behavior is highlighted as shown in Fig. 6 where a smooth surface have its sharp feature enhanced.

Many algorithms in mesh denoising are based on iterative Bilateral Filtering approach [8], [12], [1], [16]. Because of this aspect, in our third experiment, we have implemented the Bilateral Mesh Filtering algorithm as presented in [8], in order to compare the results in our experiments.

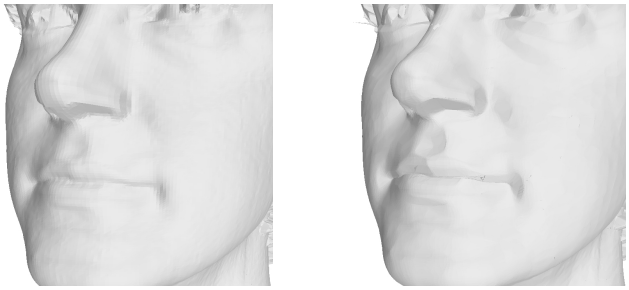


Figure 6. Example of a feature enhancing effect of our method. (a) original and (b) processed mesh.

For this comparison, we used the Stanford dragon model [23] with 437,645 vertices (Fig. 7). This model was corrupted with three different levels of zero mean gaussian noise with standard deviation σ proportional to the mean edge length in the mesh. The difference from the denoised vertices position and their original position were calculated and the standard deviation of residual noise is presented in Table I for each noise level.

Fig. 8 shows the results for $\sigma = 0.2$ mean edge, Fig. 9 shows results for $\sigma = 0.4$ mean edge and Fig. 10 shows results for $\sigma = 0.8$ mean edge. The neighborhood size in all experiments were set to $k = 2$. While our results were obtained in a single iteration, Bilateral Filtering results are presented after three iterations.

We have also compared the execution time for this experiments in the computer described above. For the noisy model on Fig. 8, the average time for performing our algorithm to denoise the model was 13 seconds, while Bilateral Filtering took average time of 4 seconds to perform each iteration. The results from our method are presented after a single iteration, while the results from Bilateral Filtering are presented after 3 successive iterations on the same model. Hence, the final execution time for the Bilateral Mesh Filtering was of the same order as our method. We remark that sequential application of Bilateral Filtering to the models is able to recover high level of smoothness, but with high damage to sharp features, as occurs with some image filters.

Another result of our experiments is presented in Fig. 1, where the Stanford Bunny model [23], with 35,947 vertices, is corrupted with zero mean Gaussian noise with standard deviation $\sigma = 0.4$ of mean edge length and denoised with our method. It is possible to notice that no oversmoothing

Table I
RESIDUAL NOISE COMPARISON.

Added noise (% mean edge)	**Residual noise (% mean edge)**	
	Bilateral Filtering	***Our method***
20	4.21	6.57
40	15.17	11.18
80	33.35	18.82

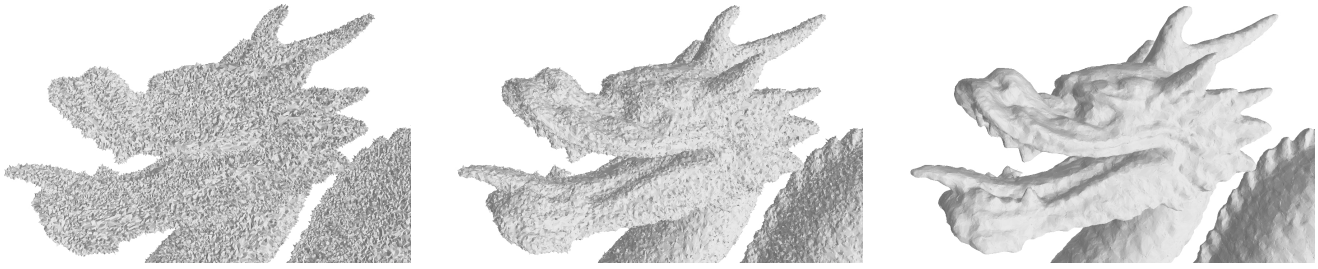
Figure 7. Original dragon model with 437,645 vertices and 871,414 triangle faces.

(a) Noise of 20% mean edge (b) Bilateral Filtering (c) Our Method

Figure 8. Visual comparison of Bilateral Filtering and our method in a mesh corrupted with zero mean gaussian noise, with $\sigma = 0.2$ mean edge. Our result is too close to that from Bilateral Filtering, but it's possible to notice that details were enhanced in our method.

(a) Noise of 40% mean edge (b) Bilateral Filtering (c) Our Method

Figure 9. Visual comparisons of Bilateral Filtering and our method in a mesh corrupted with zero mean gaussian noise, with $\sigma = 0.4$ mean edge. While the noise level increases, the advantage of our method is evident since most features are well recovered.



(a) Noise of 80% mean edge (b) Bilateral Filtering (c) Our Method

Figure 10. Visual comparison of Bilateral Filtering and our method in a mesh corrupted with zero mean gaussian noise, with $\sigma = 0.8$ mean edge. In this severe noise level condition, our method still recovers most features and our results are significantly better than Bilateral Filtering results.

occurs along the surface and that, despite large noise levels, sharp details were well recovered in the area of the eyes and also enhanced in the ears.

B. Frequency Response Analysis

This section presents an analysis concerning the frequency response of our methodology. The analysis shows a brief idea about what happens with the frequency spectrum of a mesh corrupted with gaussian noise and denoised with our method.

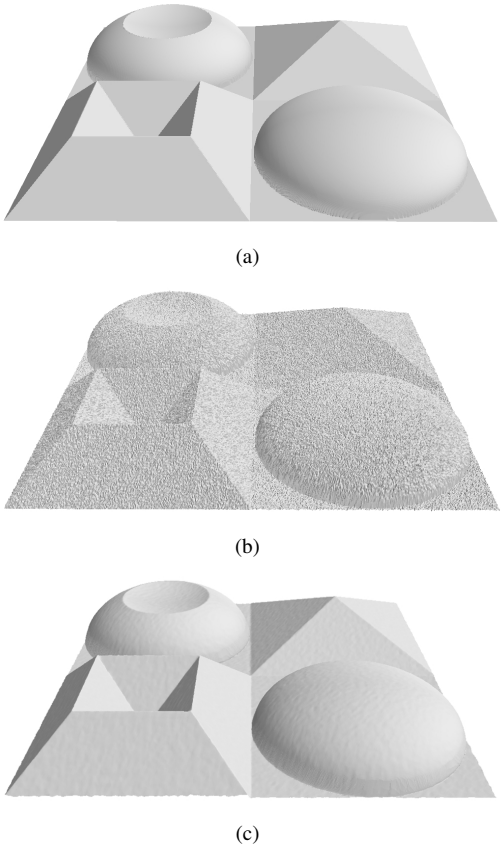
In order to apply the standard FFT (Fast Fourier Transform) for the frequency analysis in meshes, we need a mesh with single support plane for the signal and with homogeneous matricial distribution of vertices, which is not the general case. However, as presented in [24], the frequency spectrum for general meshes can be calculated over patches of the mesh that shares a single support plane using a parametrization over a regular matricial grid to recover an approximation of the original patch with the necessary homogeneous distribution of vertices.

The mesh used in this experiment, shown in Fig. 11(a), is a single patch with planar support for the signal and with non-homogeneous distribution of vertices. For the frequency analysis purposes, we used a parametrization over a matricial grid as proposed in [24].

Gaussian noise were added along the normal on each vertex of the model. The noise was set with zero mean and standard deviation of $\sigma = 40\%$ mean edge length.

Initially, we calculate $\mathcal{F}_1$ as the FFT of the mesh corrupted by noise 11(b). After that, we process the mesh using the proposed algorithm and then calculate $\mathcal{F}_2$ as the FFT of the resulting filtered mesh 11(c). Taking the difference between the absolute values of $\mathcal{F}_1$ and $\mathcal{F}_2$, we can see an approximation of the frequency response of the proposed operator. This response is presented in Fig. 12. A smooth filter was applied to this signal for visualization purposes.

It is possible to see that the operator has band-pass filtering features, what was expected according to the results. This promotes the noise suppression of high-frequencies. It can also be observed a low gain value around the zero frequency (mean value), differently of what occurs in smoothing methods presented in the literature. Other characteristic of our method are the gains in the middle frequencies, that may explain the enhancement of the features in the mesh. This characteristic certainly contributes to the effects seen in Fig. 6(b).



(a)

(b)

(c)

Figure 11. Frequency response analysis: (a) Original mesh, (b) corrupted by gaussian noise and (c) denoised by our method.

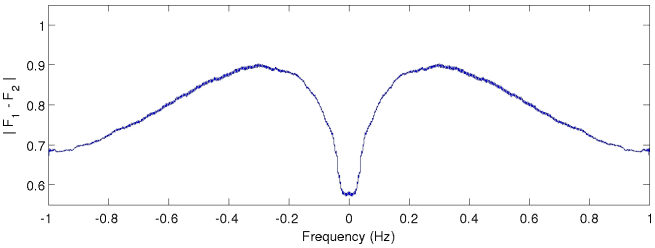


Figure 12. Frequency response generated by the proposed operator.

V. Conclusions and Future Work

We presented a new approach for mesh denoising that performs an adjustment of the geometry of a noisy mesh in order to obtain a piecewise smooth mesh, recovering some details even for highly noisy meshes.

In our experiments, in low noise condition, the proposed methodology presented similar qualitative results while compared with Bilateral Mesh Filtering. For meshes with high noise levels, however, our method recovered more details than those from Bilateral Mesh Filtering.

The time performance of the present algorithm, in a single iteration, is worse than the time performance on a single iteration of the Bilateral Mesh Filtering. However, since Bilateral Mesh Filtering is an iterative process and generally performed more than once, the final time of our method is the same order of Bilateral Mesh Filtering.

Experiments on mesh denoising algorithms usually simulates zero mean gaussian noise with standard deviation from $\sigma = 15\%$ up to 20% of mean edge length. We have tested our method in huge noise condition, $\sigma = 20\%$ up to 80% mean edge, and it was able to recover some corrupted details

even in this condition.

According to the frequency analysis, we observe that there are gain values on the middle frequencies. This means that, if this operator is applied successively, it can reinforce these features.

Bilateral Filtering has been the basis for many denoising algorithms in literature, and we expect that adapting our method to those algorithms should lead to better results. We intend to adapt the quadrics sum to a weighted sum using the results from the frequency response analysis in order to achieve a better balance in smoothing and feature enhancing.

REFERENCES

[1] E. Medeiros, L. Velho, and H. Lopes, "A topological framework for advancing front triangulations," in *Proceedings of the XVI Brazilian Symposium on Computer Graphics and Image Processing*. IEEE Press, 2003, pp. 45–51.

[2] A. Sharf, T. Lewiner, G. Shklarski, S. Toledo, and D. Cohen-Or, "Interactive topology-aware surface reconstruction," in *Siggraph 2007 (ACM Transaction on Graphics)*, vol. 26, no. 3. San Diego: ACM, august 2007, pp. 43.1–43.9.

[3] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–7, 2009.

[4] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1997, pp. 209–216.

[5] G. Taubin, "A signal processing approach to fair surface design," in *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1995, pp. 351–358.

[6] J. Peng, V. Strela, and D. Zorin, "A Simple Algorithm for Surface Denoising," in *Proceedings of the conference on Visualization*. Washington, DC, USA: IEEE Computer Society, 2001, pp. 107–112.

[7] D. L. Page, Y. Sun, A. F. Koschan, J. K. Paik, and M. A. Abidi, "Simultaneous mesh simplification and noise smoothing of range images," in *Proceedings of IEEE International Conference on Image Processing*, 2002.

[8] S. Fleishman, I. Drori, and D. Cohen-Or, "Bilateral mesh denoising," *ACM Transactions on Graphics (TOG)*, vol. 22, no. 3, pp. 950–953, 2003.

[9] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 1998, p. 839.

[10] B. M. Madrazo, L. H. de Figueiredo, and L. Velho, *Geometric Desig and Computing - Seattle 2003*. SIAM, 2004, ch. Point Cloud Denoising.

[11] D. Levin, "Mesh-independent surface interpolation," *Geometric Modeling for Scientific Visualization.*, pp. 37–49, 2003.

[12] T. R. Jones, F. Durand, and M. Desbrun, "Non-iterative, feature-preserving mesh smoothing," in *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*. New York, NY, USA: ACM, 2003, pp. 943–949.

[13] S. Fleishman, D. Cohen-Or, and C. T. Silva, "Robust moving least-squares fitting with sharp features," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 544–552, 2005.

[14] R. Unnikrishnan and M. Hebert, "Denoising Manifold and Non-Manifold Point Clouds," in *18th British Machine Vision Conference (BMVC)*, September 2007.

[15] X. Sun, P. L. Rosin, R. R. Martin, and F. C. Langbein, "Random walks for feature-preserving mesh denoising," *Computer Aided Geometric Design*, vol. 25, no. 7, pp. 437–456, 2008.

[16] H. Fan, Y. Yu, and Q. Peng, "Robust feature-preserving mesh denoising based on consistent subneighborhoods," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 312–324, 2010.

[17] L. Velho, "Mesh simplification using four-face clusters," in *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*. Washington, DC, USA: IEEE Computer Society, 2001, p. 200.

[18] Y. Wu, Y. He, and H. Cai, "Qem-based mesh simplification with global geometry features preserved," in *GRAPHITE '04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*. New York, NY, USA: ACM, 2004, pp. 50–57.

[19] A. W. Vieira, T. Lewiner, L. Velho, H. Lopes, and G. Tavares, "Stellar mesh simplification using probabilistic optimization," *Computer Graphics Forum*, vol. 23, no. 4, pp. 825–838, 2004.

[20] M. Garland, "Quadric-Based Polygonal Surface Simplification," Ph.D. dissertation, School of Computer Science - Carnegie Mellon University, 5000 Forbes Avenue - Pittsburgh, PA 15213-3891, may 1999. [Online]. Available: http://mgarland.org/

[21] T. Lewiner, H. Lopes, E. Medeiros, G. Tavares, and L. Velho, "Topological mesh operators," *Computer Aided Geometric Design*, vol. 27, no. 1, pp. 1–22, 2010.

[22] (2010, may) INRIA Gamma team. Web. [Online]. Available: http://www-roc.inria.fr/gamma

[23] (2010, may) The Stanford 3D Scanning Repository. Web. [Online]. Available: http://graphics.stanford.edu/data/3Dscanrep/

[24] M. Pauly and M. Gross, "Spectral processing of point-sampled geometry," in *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 2001, pp. 379–386.