Mersin Photogrammetry Journal

https://dergipark.org.tr/en/pub/mephoj

e-ISSN 2687-654X

Analysing detail preserving capabilities of bilateral, laplacian and taubin mesh filtering methods

Erkan Besdok [*1], **Pinar Civicioglu** [2]

[1]*Erciyes University, Department of Geomatics Engineering, Türkiye*
[2]*Erciyes University, Department. of Aircraft Electrics and Electronics, Türkiye*

Keywords
Mesh Filtering
Bilateral Filter
Laplacian Filter
Taubin Filter

Research Article
DOI: 10.53093/mephoj.1349021

Received:24.08.2023
Revised: 09.09.2023
Accepted:25.09.2023
Published:17.10.2023

Abstract

Mesh filtering of surfaces is crucial for noise reduction, feature preservation, and mesh simplification in graphics, visualization, and computer vision. In this paper, the detail preservation capacities of 3 frequently used filters, i.e., Bilateral, Laplacian, and Taubin mesh filters, in mesh filtering have been thoroughly examined by experiments conducted on 4 different test meshes. While the Bilateral filter excels in preserving sharp features due to its integration of geometric proximity with intensity similarity, the Laplacian filter prioritizes smoothness by averaging neighboring vertex positions, and the Taubin filter offers a balanced approach by merging attributes of both Laplacian and high-pass filters. The Bilateral filter's primary strength lies in its ability to maintain sharp features on a mesh, ensuring that intricate details are preserved by considering both the spatial closeness and intensity similarity of vertices. The Laplacian filter, although effective in achieving mesh smoothness, has the propensity to excessively smooth out sharp and defining features, potentially causing a loss of critical details in the processed mesh. The Taubin filter integrates the best of both worlds, ensuring smoothness without excessive mesh shrinkage; however, it might not excel in feature preservation as effectively as the Bilateral filter or smooth as uniformly as the Laplacian filter, making it a middle-ground option for certain applications. The statistical analysis of the experimental results has shown that the Taubin method is statistically a more successful mesh filtering method for the test sets used in this paper.

1. Introduction

Mesh surface filtering [1-7] is a pivotal process in the domain of computer graphics and computational geometry, aimed at enhancing the visual quality and computational efficiency of three-dimensional (3D) mesh models. A 3D mesh is a discrete representation of a surface composed of vertices, edges, and faces. However, due to various factors such as acquisition methods, simplification techniques, or transmission limitations, meshes often contain imperfections, noise, and artifacts that can degrade their appearance and usability. Mesh surface filtering [8-14], therefore, involves the application of algorithms to refine and improve the geometric and topological characteristics of mesh surfaces.

The rationale behind filtering mesh surfaces resides in the pursuit of producing visually pleasing and physically plausible models for applications ranging from video games and virtual reality to medical imaging and architectural design. The overarching goal is to mitigate undesired visual artifacts and irregularities that might stem from processes like 3D scanning, simplification, or transmission over networks. Filtering not only enhances the aesthetics of rendered scenes but also assists in downstream tasks such as mesh compression, collision detection, and finite element analysis.

Advantages of mesh filtering methods are manifold. Firstly, they enable noise reduction [15-21], effectively ameliorating the presence of outliers and spurious geometries that could arise from sensor inaccuracies or data corruption. Secondly, these methods can enhance mesh coherency by addressing issues like cracks and gaps between adjacent faces, thereby facilitating smoother interactions during rendering and simulation. Moreover, filtering contributes to the preservation of salient features while attenuating superfluous details, which proves invaluable in applications where

* **Corresponding Author**

*(ebesdok@erciyes.edu.tr) ORCID ID 0000-0001-9309-375X
(civici@erciyes.edu.tr) ORCID ID 0000-0003-1850-8489

Cite this article

Beşdok, E., & Civicioğlu, P. (2023). Analysing detail preserving capabilities of bilateral, laplacian and taubin mesh filtering methods. Mersin Photogrammetry Journal, 5(2), 67-74

maintaining the fidelity of critical structures is essential. Additionally, by optimizing the mesh representation [8, 22-25], computational overhead is reduced, enabling real-time rendering and interaction even in resource-constrained environments.

However, mesh filtering methods are not devoid of limitations. One notable concern is the potential loss of fine details during the filtering process, as aggressive filtering can inadvertently erase intricate features that are pertinent in certain applications. Furthermore, there is an inherent trade-off between filtering strength and computational cost; complex filtering algorithms may demand significant processing power and memory resources, impeding their applicability on low-end devices. Selecting an appropriate filtering method and parameter configuration can also be nontrivial, necessitating domain expertise and iterative refinement.

In conclusion, mesh surface filtering constitutes a vital facet of modern computer graphics, serving to enhance visual quality, alleviate artifacts, and improve computational efficiency in 3D mesh models. While its advantages encompass noise reduction, coherency enhancement, and feature preservation, caution must be exercised to mitigate potential drawbacks such as detail loss and computational overhead. The ongoing evolution of filtering techniques continues to address these challenges, contributing to the creation of compelling, high-fidelity virtual environments and simulations across diverse domains.

Mesh filtering methods [10, 26-28] have gained considerable attention due to their applications in graphics, computer vision, and geometric modelling. These methods target the removal of noise, preservation of salient features, and simplification of mesh structures. One of the early and widely-adopted techniques is Laplacian smoothing, which averages the positions of neighbouring vertices, but it often over-smooths and may degrade mesh quality. Bilateral mesh filtering [2, 5, 23, 29, 30] emerged as a promising alternative by combining geometric closeness and intensity resemblance, effectively preserving sharp features. However, its computational expense has been a limitation for real-time applications.

Taubin [31] introduced a method that leverages the combination of low-pass and high-pass filters, preventing mesh shrinkage observed in traditional Laplacian approaches. Recently, non-local means and anisotropic diffusion methods have been explored, inspired by their success in image processing. These methods consider wider neighbourhoods or adapt filtering based on local mesh properties. Guided mesh filtering, where the filter operation is guided by another signal, has also shown promising results, especially in texture and feature preservation. Wavelet-based techniques, which decompose the mesh into frequency bands, enable multi-resolution processing and have applications in mesh compression.

Deep learning-based mesh filtering [32, 33], a burgeoning area, employs neural networks to learn optimal filtering parameters from data. While traditional methods rely on hand-crafted heuristics, these learnable filters adapt based on the input, making them versatile. In conclusion, mesh filtering remains an active research domain with methodologies ranging from classical algorithms to modern machine learning approaches, each with its own merits and challenges.

Bilateral mesh filtering maintains sharp features by weighing both geometric proximity and feature similarity, making it particularly suitable for preserving edges but can be computationally intensive. On the other hand, Laplacian filtering smoothens the surface by averaging neighbouring vertices, which can lead to over-smoothing of sharp details if not controlled properly. In contrast, Taubin filtering employs a sequence of low-pass and high-pass filters, ensuring effective smoothing without causing the mesh to shrink, offering a balance between detail preservation and noise reduction.

This paper presents experiments on the use of Bilateral, Laplacian, and Taubin mesh filters in photogrammetry and computer vision. These filters are commonly used in these fields because they are effective at smoothing meshes while preserving sharp features.

The rest of this paper is organized as follows: Section 2 introduces Mesh Filtering Methods. In Section 3, Experiments are presented. In Section 4, Results and Conclusions are given.

2. Mesh Filtering Methods

This section briefly presents the analytical structures, basic features, advantages, and disadvantages of the Bilateral, Laplacian, and Taubin filters used in the Experiments section of this paper.

2.1. Bilateral Mesh Filtering

Bilateral mesh filtering is a method that applies Bilateral filtering principles to 3D mesh data. It aims to smooth the mesh while preserving important features such as edges and corners. The filtering process takes into account both geometric distance and attribute similarity between vertices to determine the filtering weights.

Given a mesh with vertices V and faces F, the filtered position p_i, for vertex p_i can be computed using Equation 1:

$$p_{i'} = \frac{1}{W_i} \sum_{j=1}^{N} w_{ij} \cdot p_j \tag{1}$$

where, N is the number of neighboring vertices of p_i. The p_j represents the neighboring vertex positions. The w_{ij} is the Bilateral weight between vertices p_i and p_j. w_{ij} *is* the normalization term. The w_{ij} is computed as a combination of spatial and range weights by using Equation 2:

$$w_{ij} = w_s(\|p_i - p_j\|) \cdot w_r(p_i, p_j) \tag{2}$$

where, $w_s(\|p_i - p_j\|)$ is the spatial weight based on the geometric distance between vertices p_i and p_j. The $w_r(p_i, p_j)$ is the range weight based on attribute similarity between vertices p_i and p_j. In this filter, $w_s(\|p_i - p_j\|)$ and $w_r(p_i, p_j)$ represent the spatial and range weights, respectively. Bilateral mesh filter iterates

over each vertex in the mesh, computes the weighted sum of neighboring vertex positions, and updates the filtered position accordingly while considering normalization. Pseudo-code of Bilateral mesh filter is given in Figure 1.

Data: Input mesh with vertices V and faces F
Result: Filtered mesh vertices V'
foreach *vertex* $p_i \in V$ **do**
  Initialize filtered position $p'_i = (0, 0, 0)$;
  Initialize normalization term $W_i = 0$;
  foreach *neighboring vertex* p_j *of* p_i **do**
    Calculate spatial weight $w_{\text{spatial}} = w_s(\|p_i - p_j\|)$;
    Calculate range weight $w_{\text{range}} = w_r(p_i, p_j)$;
    Calculate bilateral weight $w_{ij} = w_{\text{spatial}} \cdot w_{\text{range}}$;
    Update filtered position $p'_i = p'_i + w_{ij} \cdot p_j$;
    Update normalization term $W_i = W_i + w_{ij}$;
  Update filtered position $p'_i = \frac{p'_i}{W_i}$;

Figure 1. Pseudo-code of Bilateral mesh filter.

Bilateral mesh filter offers several advantages that make it a valuable technique for enhancing the visual quality and preserving important features of 3D mesh models:

Bilateral filtering is inherently designed to preserve edges and boundaries within the data. This characteristic is crucial for maintaining the sharpness and integrity of important features in the mesh, such as edges, corners, and creases. Unlike traditional smoothing methods that tend to blur edges, bilateral filtering ensures that these features remain well-defined. One of the strengths of bilateral mesh filtering lies in its ability to take into account attributes associated with vertices, such as normal or colours. This enables the filtering process to consider not only geometric proximity but also attribute similarity when computing filtering weights. As a result, attributes are preserved more effectively, contributing to the overall visual fidelity of the mesh. Bilateral filtering effectively reduces noise and small-scale irregularities present in the mesh data. The incorporation of attribute-based filtering helps distinguish between meaningful variations and noise, allowing the method to selectively smooth out noise while retaining genuine geometric and attribute details.

Bilateral filtering is highly adaptable and can be tailored to specific applications and requirements. By adjusting the parameters of the spatial and range weights, users can control the strength of the filtering effect. This adaptability makes bilateral filtering suitable for a wide range of scenarios, from artistic stylization to scientific simulations. Unlike some traditional smoothing methods that may result in blurring and distortion of geometric details, bilateral filtering smooths the mesh while preserving important features. This is particularly advantageous for applications where maintaining the integrity of the mesh's structural characteristics is essential.

Bilateral filtering strikes a balance between noise reduction and feature preservation. It selectively smooths areas that are less critical while leaving important features untouched. This characteristic is particularly valuable for applications where a compromise between overall smoothness and the preservation of key details is required.

The advantages of bilateral mesh filtering extend across various domains, including computer graphics, medical imaging, computer-aided design, and more. It finds applications in rendering, modelling, simulation, and analysis, making it a versatile technique with wide-ranging benefits. While more computationally intensive methods may achieve better results, bilateral filtering strikes a good balance between quality and efficiency. It is often suitable for real-time or interactive applications, offering an effective compromise between filtering strength and computational complexity.

In summary, bilateral mesh filtering is advantageous due to its ability to preserve edges, accommodate attribute-based filtering, reduce noise, offer customizable adjustments, and strike a balance between feature preservation and smoothing. These advantages make it a valuable tool for enhancing the visual quality and fidelity of 3D mesh models across diverse applications.

2.2. Laplacian Mesh Filtering

Laplacian Mesh Filtering is a widely used method for mesh smoothing and denoising. It leverages the Laplacian operator to iteratively update vertex positions based on the local geometric information of neighbouring vertices.

The Laplacian operator quantifies the difference between a vertex and the average of its neighbours, capturing the curvature and shape characteristics of the mesh. Let's denote the Laplacian operator as Δ and the position of a vertex v_i as $\boldsymbol{p}_i = (x_i, y_i, z_i)$. The Laplacian operator applied to the position of a vertex is defined in Equation 3:

$$\Delta \boldsymbol{p}_i = \frac{1}{|\mathcal{N}_i|} \sum_{v_j \in \mathcal{N}_i} (\boldsymbol{p}_j - \boldsymbol{p}_i) \tag{3}$$

where $\mathcal{N}_i$ represents the set of neighboring vertices of v_i.

The basic outline of the Laplacian Mesh Filtering algorithm is given below:

1. Initialize: Given a mesh with vertices $\boldsymbol{p}_i$ and connectivity information.
2. Choose the number of iterations N.
3. For k = 1 to N:
   a. For each vertex v_i:
      i. Compute the Laplacian update: $\Delta \boldsymbol{p}_i = \frac{1}{|\mathcal{N}_i|} \sum_{v_j \in \mathcal{N}_i} (\boldsymbol{p}_j - \boldsymbol{p}_i)$.
      ii. Update the vertex position: $\boldsymbol{p}_i^{(k)} = \boldsymbol{p}_i^{(k-1)} + \lambda \cdot \Delta \boldsymbol{p}_i$,

where λ is a user-defined weight controlling the step size.

The Pseudo-code of Laplacian mesh filtering is given in Figure 2.

In this pseudo-code, $\boldsymbol{p}_i^{(k)}$ represents the position of vertex v_i after k iterations.

Laplacian Mesh Filtering iteratively adjusts vertex positions, redistributing their positions based on the average displacement of neighbouring vertices. This

process tends to smooth out noise and small-scale irregularities in the mesh while preserving overall shape characteristics. The parameter λ controls the extent of the update and should be chosen carefully to achieve the desired smoothing effect without causing over-smoothing or distortion.

Input: Mesh vertices $\{\mathbf{p}_i\}$, Connectivity information $\{\mathcal{N}_i\}$, Number of iterations N, Weight λ
Output: Smoothed mesh vertices
for $k = 1$ *to* N **do**
  for *each vertex* v_i **do**
    Compute Laplacian update: $\Delta\mathbf{p}_i = \frac{1}{|\mathcal{N}_i|}\sum_{v_j \in \mathcal{N}_i}(\mathbf{p}_j - \mathbf{p}_i)$;
    Update vertex position: $\mathbf{p}_i^{(k)} = \mathbf{p}_i^{(k-1)} + \lambda \cdot \Delta\mathbf{p}_i$;

Figure 2. Pseudo-code of Laplacian mesh filtering.

The Laplacian Mesh Filtering method is conceptually straightforward and easy to implement. It involves a simple iterative process that updates vertex positions based on the Laplacian operator, making it accessible to both novice and experienced users.

Laplacian filtering tends to preserve local geometric coherence. It smooths the mesh while maintaining the overall shape characteristics and connectivity of the original mesh. This makes it suitable for applications where preserving essential features is important.

Laplacian filtering is effective in reducing noise and small-scale irregularities in the mesh. By averaging vertex positions with their neighbours, the method can mitigate high-frequency noise that might be present due to data acquisition or other factors. The filtering strength can be controlled using the parameter λ. Users can adjust this parameter to achieve the desired level of smoothing. This level of control is valuable when adapting the filtering to different meshes and requirements.

While Laplacian filtering is efficient at smoothing, it can inadvertently lead to detail loss, particularly in regions with high curvature or intricate features. The iterative nature of the algorithm tends to distribute vertex positions toward an average, potentially diminishing fine details. In some cases, Laplacian filtering can introduce shape distortion, especially when the smoothing process is too aggressive. This might cause unintended changes in the mesh's shape that could impact the overall visual quality or intended characteristics of the model.

Laplacian filtering can lead to uneven smoothing, where some parts of the mesh are smoothed more than others. This is due to the reliance on local neighbourhood information, which might not be uniform across the entire mesh. The performance of Laplacian filtering is highly dependent on the choice of the λ parameter. Selecting an inappropriate value can lead to suboptimal results, such as under-smoothing or over-smoothing. In meshes with irregular connectivity or boundary conditions, Laplacian filtering can sometimes introduce artifacts like shrinkage or expansion of specific regions. This is because the filtering process is sensitive to the local vertex distribution and connectivity.

In summary, Laplacian Mesh Filtering offers a straightforward approach to mesh smoothing with advantages including simplicity, noise reduction, and local coherence. However, it comes with the trade-offs of potential detail loss, shape distortion, and sensitivity to parameter choices. Users should carefully consider these factors and their specific application requirements when choosing Laplacian filtering as a mesh enhancement technique.

2.3. Taubin Mesh Filtering

Taubin Mesh Filtering is a popular method for smoothing and denoising 3D mesh surfaces. It was introduced by Gabriel Taubin in 1995 as an iterative technique that alternates between applying two distinct filters: a Laplacian smoothing filter and a high-pass filter. This approach effectively reduces noise while preserving important features of the mesh.

Given a 3D mesh represented by vertices $(\boldsymbol{v}_i)$ and faces $(\boldsymbol{f}_j)$ with associated normal $(\boldsymbol{n}_i)$, the Laplacian smoothing step can be represented by using Equation 4:

$$\boldsymbol{v}_i^{(k+1)} = \boldsymbol{v}_i^{(k)} + \lambda_1 \cdot \text{Laplacian}(\mathbf{v}_i^{(k)}) \qquad (4)$$

where (λ_1) is a user-defined parameter controlling the amount of smoothing, and $\text{Laplacian}(\mathbf{v}_i^{(k)})$ computes the Laplacian operator on vertex $(\boldsymbol{v}_i)$ at iteration (k).

The Laplacian operator measures the difference between the vertex and the average of its neighbouring vertices, thus smoothing out irregularities. Following the Laplacian smoothing, the high-pass filter step is applied using Equation 5:

$$\boldsymbol{v}_i^{(k+1)} = \boldsymbol{v}_i^{(k+1)} - \lambda_2 \cdot \text{Laplacian}(\mathbf{v}_i^{(k+1)}) \qquad (5)$$

Here, (λ_2) is another user-defined parameter controlling the amount of high-frequency detail preservation. This step effectively compensates for the excessive smoothing introduced by the previous step, enhancing the overall fidelity of the filtered mesh.

The pseudo-code of Taubin Mesh Filtering method is given in Figure 3.

Input : 3D Mesh with vertices $\mathbf{V}$, faces $\mathbf{F}$, normals $\mathbf{N}$, smoothing parameters λ_1 and λ_2, number of iterations K
Output: Filtered mesh vertices $\mathbf{V}$
for $k \leftarrow 1$ **to** K **do**
  for $i \leftarrow 1$ to *num_vertices*($\mathbf{V}$) **do**
    $\mathbf{L}_i \leftarrow \text{ComputeLaplacian}(\mathbf{V}, \mathbf{F}, \mathbf{N}, i)$; // Laplacian operator
    $\mathbf{V}[i] \leftarrow \mathbf{V}[i] + \lambda_1 \cdot \mathbf{L}_i$; // Laplacian smoothing
  for $i \leftarrow 1$ to *num_vertices*($\mathbf{V}$) **do**
    $\mathbf{L}_i \leftarrow \text{ComputeLaplacian}(\mathbf{V}, \mathbf{F}, \mathbf{N}, i)$; // Laplacian operator
    $\mathbf{V}[i] \leftarrow \mathbf{V}[i] - \lambda_2 \cdot \mathbf{L}_i$; // High-pass filtering
return $\mathbf{V}$

Figure 3. Pseudo-code of Taubin mesh filtering.

This pseudo-code outlines the core steps of the Taubin Mesh Filtering method, including Laplacian computation and the iterative application of smoothing and high-pass filtering. The parameters (λ_1), (λ_2), and the number of iterations (K) can be adjusted to achieve the desired level of smoothing and detail preservation for a given mesh.

The Taubin Mesh Filtering method offers several advantages and drawbacks, making it an interesting choice for mesh smoothing and denoising in certain scenarios.

Taubin Mesh Filtering employs an iterative process that alternates between Laplacian smoothing and high-

pass filtering. This approach allows for controlled smoothing while preserving important geometric details. The iterative nature enables users to fine-tune the degree of filtering. The method is effective in reducing noise and artifacts present in mesh data.

The Laplacian smoothing step averages vertex positions, which helps in attenuating high-frequency noise components, resulting in a smoother appearance. The high pass filtering step counteracts excessive smoothing introduced by the Laplacian operation. This ensures that significant geometric features, such as edges and corners, are better preserved compared to methods that solely rely on simple smoothing techniques. The user has control over two crucial parameters: (λ_1) and (λ_2). These parameters influence the amount of smoothing and detail preservation, allowing users to tailor the filtering process to suit the specific characteristics of the input mesh and the desired visual outcome. The method is relatively computationally efficient due to its localized nature. The Laplacian and high pass filtering operations involve neighbouring vertices, making them amenable to parallelization and optimization techniques.

Depending on the chosen parameter values and the number of iterations, aggressive smoothing may result in the loss of fine geometric details. While the high pass filter aims to mitigate this, there can still be instances where essential details are inadvertently smoothed out.

The effectiveness of the Taubin method is closely tied to parameter settings. Selecting appropriate values for (λ_1) and (λ_2) is not always straightforward, and finding the right balance between smoothing and preserving details requires experimentation. In certain cases, excessive filtering iterations can lead to mesh distortion. Particularly, regions with high curvature might exhibit undesirable artifacts due to the iterative nature of the smoothing process.

The Taubin method's effectiveness diminishes in cases where the input mesh has highly irregular or noisy features. For instance, when the noise levels are extremely high or the mesh lacks clear geometric structure, the method might struggle to achieve satisfactory results. Despite its advantages, Taubin Mesh Filtering may necessitate manual intervention to achieve optimal results. Users might need to fine-tune parameters and conduct iterative trials to strike a balance between smoothing and feature preservation.

In conclusion, the Taubin Mesh Filtering method is a versatile approach for smoothing and denoising 3D mesh surfaces. Its iterative nature and parameter control offer flexibility in achieving varying degrees of noise reduction and detail preservation. However, careful consideration of parameter settings and an understanding of its limitations are essential to ensure effective application and avoid unintended consequences such as detail loss or mesh distortion.

3. Experiments

In the experiments, 4 different meshes were used: “David” (Figure 4A), “Roma” (Figure 4B), “Man” (Figure 5A), and “Girl” (Figure 5B).

The “David” test mesh has 72,685 faces, and 36,714 vertices. The “Roma” test mesh consists of 55,847 faces, and 28,254 vertices. The “Man” test mesh contains 30,000 faces, and 15,258 vertices. The “Girl” test mesh has 6,999 faces, and 3,658 vertices. The spatial coordinates of the related mesh’s are given in centimetre. The spatial boundaries for the "Man" are given as: $150 \leq x \leq 200,\ 3.77 \leq y \leq 43.32,\ and\ 2.23 \leq z \leq 67.39$. The spatial boundaries for the "Roma" are given as: $100 \leq x \leq 200,\ 110.31 \leq y \leq 165.95,\ and\ 204.33 \leq z \leq 295.11$. The spatial boundaries for the "David" are given as: $100 \leq x \leq 200,\ 75 \leq y \leq 150,\ 100 \leq z \leq 200$. The spatial boundaries for the "Girl" are given as: $100 \leq x \leq 200,\ 84.14 \leq y \leq 170.63,\ and\ 56.29 \leq z \leq 210.63$.

Corrupted meshes are generated by adding random valued uniform impulsive noise to the vertex positions of the original meshes. The vertex positions of the generated corrupted meshes are repaired by optimizing the internal parameters of the filters used in the experiments.

The optimal values for the related threshold parameters of Bilateral, Laplacian, and Taubin filters have been optimized using the BSA algorithm [34-37]. Taubin filter is applied to related meshes using 5 iterations. BSA is a very powerful, non-recursive, iterative evolutionary search method developed by Çivicioğlu [34]. Evolutionary search algorithms are very popular because they produce useful results in the optimization of non-differentiable, multimodal, and continuous numerical problems. For BSA, the size of the population is set to 20, and the maximum number of iterations is empirically chosen as 100,000. The search space lower and upper bound values have been determined as [low=0; up=1] only for the first iteration. In the following iterations, BSA employed the unbounded search method to obtain the optimum values for the related filters.

The objective function used for BSA is given in Equation 6:

$$\underbrace{argmin}_{parameters\ of\ Filter} \Sigma|CMesh_{Filter} - OMesh| \qquad (6)$$

where $CMesh_{Filter}$, and $OMesh$ denote response of the filter used in the current experiment, and original mesh, respectively.

All the filtering methods used in the Experiments were implemented in MATLAB. The Experiments were conducted by using a computer with Intel(R) Xeon(R) CPU E5-2650 v2 @ 2.60GHz 2.60 GHz (2 CPU), 64GB RAM.

Table 1 shows the ‘Mean Square Error’’ values calculated between the meshes obtained from the experiments and the original mesh. When Table 1 is examined, it is seen that the Taubin method is relatively more successful in filtering out the relevant data.

Table 1. ‘Mean Square Error’ values computed between filtered and original mesh.

Test Set	Noisy	Filtering Methods		
		Bilateral Filer	Laplacian	Taubin
David	0.0837	0.0417	0.0215	0.0108
Roma	0.0954	0.0567	0.0349	0.0181
Man	0.1054	0.0600	0.0381	0.0270
Girl	0.0910	0.0580	0.0276	0.0198

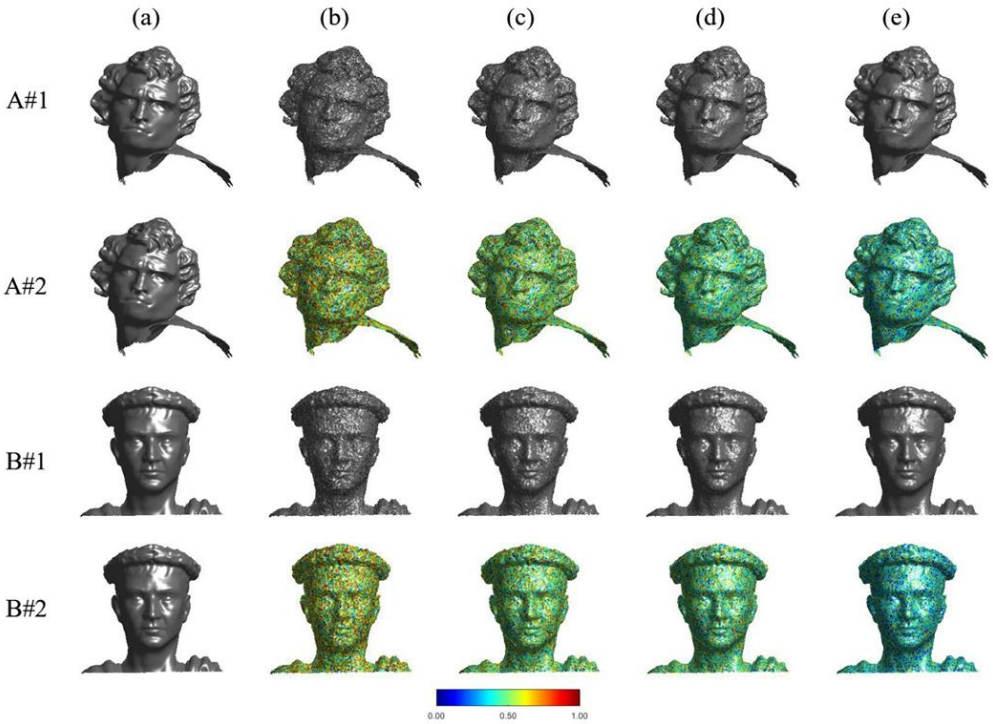

Figure 4. The mesh surface without texture for (a) Original, (b) Corrupted, (c) Bilateral Filtering, (d) Laplacian, and (e) Taubin are illustrated in rows A#1, and B#1. The normalized- displacement values superimposed on the related mesh surfaces as texture for (b) Corrupted, (c) Bilateral Filtering, (d) Laplacian, and (e) Taubin are illustrated in rows A#2, and B#2.

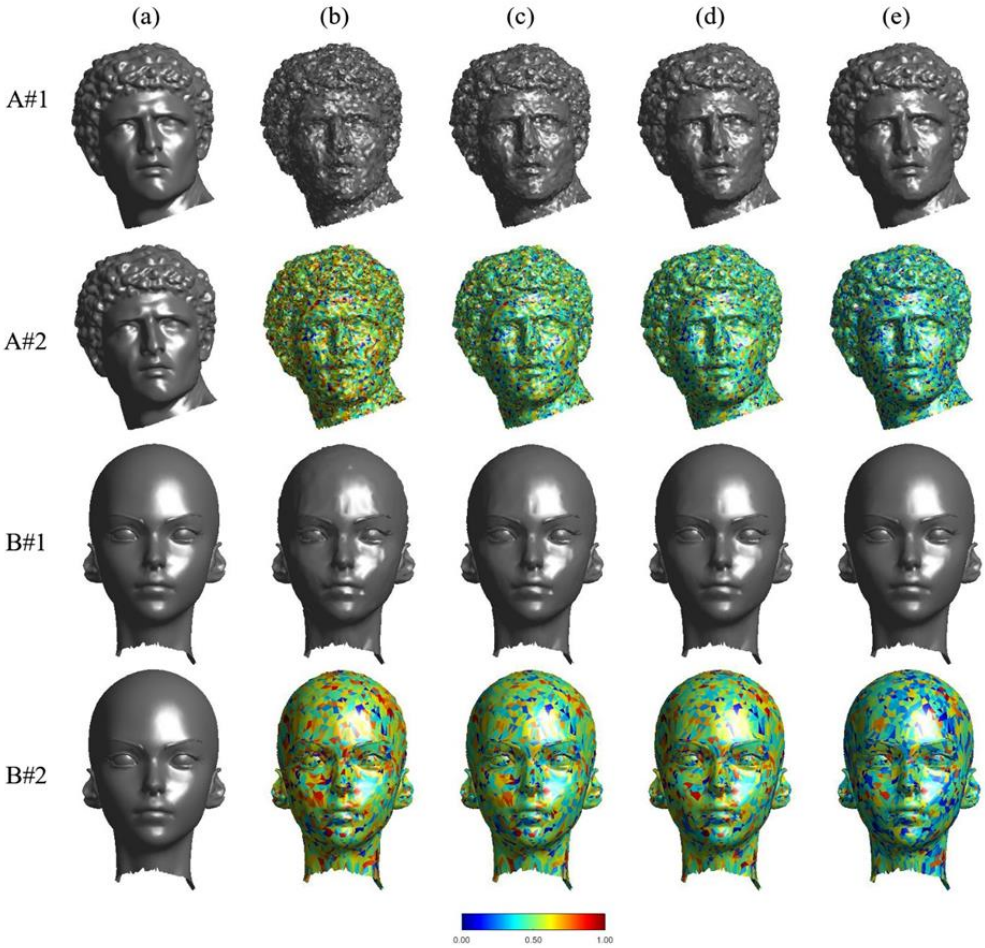

Figure 5. The mesh surface without texture for (a) Original, (b) Corrupted, (c) Bilateral Filtering, (d) Laplacian, and (e) Taubin are illustrated in rows C#1, and D#1. The normalized-displacement values superimposed on the related mesh surfaces as texture for (b) Corrupted, (c) Bilateral Filtering, (d) Laplacian, and (e) Taubin are illustrated in rows C#2, and D#2.

The meshes produced by the relevant filters at the end of the experiments are shown in Figure 4 and Figure 5. Column (a) of Figure 4 and Figure 5 shows the original mesh surface. Column (b) of Figure 4 and Figure 5 shows the corrupted mesh with random valued uniform impulsive noise, where noise ~ **U** [- 0.50; 0.50] as cm., and **U** denotes continuous uniform distribution.

4. Results and Conclusion

In this paper, the detail preservation capabilities of Bilateral, Laplacian, and Taubin mesh filtering methods have been examined in detail using 4 test sets. The Bilateral filter has a relatively more complex analytical structure compared to the Laplacian, and Taubin filters. Taubin filtering is relatively more successful in preventing deformation on the mesh. In concave and convex areas, Laplacian, and Taubin tend to preserve mesh details relatively better. The Bilateral filter causes a partial over-smoothing effect on the edge areas. In contrast, visually, the Bilateral filter tends to produce more continuous surface data. The Laplacian method has caused a partial over-sharp effect on the edge areas. Although the results obtained from the study suggest that the success of the mesh filter is somewhat data-dependent, it has been observed that the Taubin method is more successful in detail preservation compared to other methods used in the experiments.

Author contributions

Erkan Beşdok: Data curation, Software, Validation. **Pınar Çivicioğlu:** Conceptualization, Methodology, Software, Writing, Editing

Conflicts of interest

The authors declare no conflicts of interest.

References

1. Liu, Y., Coombes, M., & Liu, C. (2023). Mesh-based consensus distributed particle filtering for sensor networks. IEEE Transactions on Signal and Information Processing over Networks, 9, 346-356. https://doi.org/10.1109/TSIPN.2023.3278469
2. Liu, B., Li, B., Cao, J., Wang, W., & Liu, X. (2023). Adaptive and propagated mesh filtering. Computer-Aided Design, 154, 103422. https://doi.org/10.1016/j.cad.2022.103422
3. Fábián, G. (2023). Generalized Savitzky–Golay filter for smoothing triangular meshes. Computer Aided Geometric Design, 100, 102167. https://doi.org/10.1016/j.cagd.2022.102167
4. Han, H. D., & Han, J. K. (2022). Modified bilateral filter for feature enhancement in mesh denoising. IEEE Access, 10, 56845-56862. https://doi.org/10.1109/ACCESS.2022.3176961
5. Zhong, S., Song, Z., Liu, Z., Xie, Z., Chen, J., Liu, L., & Chen, R. (2021). Shape-aware mesh normal filtering. Computer-Aided Design, 140, 103088. https://doi.org/10.1016/j.cad.2021.103088
6. Zhao, W., Liu, X., Wang, S., Fan, X., & Zhao, D. (2019). Graph-based feature-preserving mesh normal filtering. IEEE Transactions on Visualization and Computer Graphics, 27(3), 1937-1952. https://do.iorg/10.1109/TVCG.2019.2944357
7. Zhang, J., Deng, B., Hong, Y., Peng, Y., Qin, W., & Liu, L. (2018). Static/dynamic filtering for mesh geometry. IEEE transactions on visualization and computer graphics, 25(4), 1774-1787. https://do.org/10.1109/TVCG.2018.2816926
8. Noel, G., Djouani, K., Van Wyk, B., & Hamam, Y. (2012). Bilateral mesh filtering. Pattern Recognition Letters, 33(9), 1101-1107. https://doi.org/10.1016/j.patrec.2012.02.008
9. Loménie, N., & Stamon, G. (2008). Morphological mesh filtering and α-objects. Pattern Recognition Letters, 29(10), 1571-1579. https://doi.org/10.1016/j.patrec.2008.03.019
10. Kim, B., & Rossignac, J. (2005). Geofilter: Geometric selection of mesh filter parameters. In Computer Graphics Forum, 24(3), 295-302.
11. Leipoldt, K. J., Happich, T., Kreysa, E., & Gemünd, H. P. (1991). Scattering matrix methods for far-infrared metal mesh filters. International Journal of Infrared and Millimeter Waves, 12, 263-274. https://doi.org/10.1007/BF01010300
12. Chen, P. A. (1987). The performance of dielectric coated mesh filter. International Journal of Infrared and Millimeter Waves, 8, 29-33. https://doi.org/10.1007/BF01010643
13. Byrne, D. M., Brouns, A. J., Case, F. C., Tiberio, R. C., Whitehead, B. L., & Wolf, E. D. (1985). Infrared mesh filters fabricated by electron-beam lithography. Journal of Vacuum Science & Technology B: Microelectronics Processing and Phenomena, 3(1), 268-271. https://doi.org/10.1116/1.583243
14. Byrne, D. M., Brouns, A. J., & Case, F. C. (1984). Infrared mesh filters (A). Journal of the Optical Society of America A, 1, 1330.
15. Civicioglu, P. (2009). Removal of random-valued impulsive noise from corrupted images. IEEE Transactions on Consumer Electronics, 55(4), 2097-2104. https://do.org/10.1109/TCE.2009.5373774
16. Civicioglu, P. (2007). Using uncorrupted neighborhoods of the pixels for impulsive noise suppression with ANFIS. IEEE Transactions on Image Processing, 16(3), 759-773. https://doi.org/10.1109/TIP.2007.891067
17. Çivicioğlu, P. (2005). Using LM artificial neural networks and η-closest-pixels for impulsive noise suppression from highly corrupted images. In International Symposium on Neural Networks (pp. 679-684). https://doi.org/10.1007/11427445_110
18. Beşdok, E., Çivicioğlu, P., & Alçı, M. (2005). Using Anfis with circular polygons for impulsive noise suppression from highly distorted images. AEU-International Journal of Electronics and Communications, 59(4), 213-221. https://doi.org/10.1016/j.aeue.2004.11.041

19. Çivicioğlu, P., Alçı, M., & Beşdok, E. (2004). Using an exact radial basis function artificial neural network for impulsive noise suppression from highly distorted image databases. In International Conference on Advances in Information Systems, 383-391. https://doi.org/10.1007/978-3-540-30198-1_39
20. Çivicioğlu, P., Alçı, M., & Beşdok, E. (2004). Impulsive noise suppression from images with the noise exclusive filter. EURASIP Journal on Advances in Signal Processing, 16, 2434–2440. https://doi.org/10.1155/S1110865704403151
21. Çivicioğlu, P., & Alçı, M. (2004). Edge detection of highly distorted images suffering from impulsive noise. AEU-International Journal of Electronics and Communications, 58(6), 413-419. https://doi.org/10.1078/1434-8411-54100262
22. Liu, B., Cao, J., Wang, W., Ma, N., Li, B., Liu, L., & Liu, X. (2018). Propagated mesh normal filtering. Computers & Graphics, 74, 119-125. https://doi.org/10.1016/j.cag.2018.05.003
23. Zhang, W., Deng, B., Zhang, J., Bouaziz, S., & Liu, L. (2015). Guided mesh normal filtering. In Computer Graphics Forum, 34(7), 23-34. https://doi.org/10.1111/cgf.12742
24. Wei, M., Yu, J., Pang, W. M., Wang, J., Qin, J., Liu, L., & Heng, P. A. (2014). Bi-normal filtering for mesh denoising. IEEE Transactions on Visualization and Computer Graphics, 21(1), 43-55. https://doi.org/10.1109/TVCG.2014.2326872
25. Shen, J. G., Zhang, S. Y., Chen, Z. Y., Zhang, Y., & Ye, X. Z. (2009). Mesh sharpening via normal filtering. Journal of Zhejiang University-Science A, 10(4), 546-553. https://doi.org/10.1631/jzus.A0820505
26. Mao, Z., Ma, L., Zhao, M., & Xiao, X. (2006). SUSAN structure preserving filtering for mesh denoising. The Visual Computer, 22, 276-284. https://doi.org/10.1007/s00371-006-0005-7
27. Hou, Q., Bai, L., & Wang, Y. (2005). Mesh smoothing via adaptive bilateral filtering. In International Conference on Computational Science, 273-280. https://doi.org/10.1007/11428848_34
28. Balan, R., & Taubin, G. (2000). 3d mesh geometry filtering algorithms for progressive transmission schemes. Computer-aided design, 32(13), 825-846. https://doi.org/10.1016/S0010-4485(00)00069-5
29. Liu, S., Rho, S., Wang, R., & Jiang, F. (2018). Feature-preserving mesh denoising based on guided normal filtering. Multimedia Tools and Applications, 77, 23009-23021. https://doi.org/10.1007/s11042-018-5735-9
30. Zheng, Y., Fu, H., Au, O. K. C., & Tai, C. L. (2010). Bilateral normal filtering for mesh denoising. IEEE Transactions on Visualization and Computer Graphics, 17(10), 1521-1530. https://doi.org/10.1109/TVCG.2010.264
31. Agathos, A., Azariadis, P., & Kyratzi, S. (2022). Elliptic Gabriel Taubin smoothing of point clouds. Computers & Graphics, 106, 20-32. https://doi.org/10.1016/j.cag.2022.05.009
32. Nousias, S., Arvanitis, G., Lalos, A. S., & Moustakas, K. (2020). Fast mesh denoising with data driven normal filtering using deep variational autoencoders. IEEE Transactions on Industrial Informatics, 17(2), 980-990. https://doi.org/10.1109/TII.2020.3000491
33. Li, X., Li, R., Zhu, L., Fu, C. W., & Heng, P. A. (2020). DNF-Net: A deep normal filtering network for mesh denoising. IEEE Transactions on Visualization and Computer Graphics, 27(10), 4060-4072. https://doi.org/10.1109/TVCG.2020.3001681
34. Civicioglu, P. (2013). Backtracking search optimization algorithm for numerical optimization problems. Applied Mathematics and computation, 219(15), 8121-8144. https://doi.org/10.1016/j.amc.2013.02.017
35. Civicioglu, P., & Besdok, E. (2019). Bernstain-search differential evolution algorithm for numerical function optimization. Expert Systems with Applications, 138, 112831. https://doi.org/10.1016/j.eswa.2019.112831
36. Civicioglu, P., & Besdok, E. (2023). Bernstein-Levy differential evolution algorithm for numerical function optimization. Neural Computing and Applications, 35(9), 6603-6621. https://doi.org/10.1007/s00521-022-08013-7
37. Civicioglu, P., Besdok, E., Gunen, M. A., & Atasever, U. H. (2020). Weighted differential evolution algorithm for numerical function optimization: a comparative study with cuckoo search, artificial bee colony, adaptive differential evolution, and backtracking search optimization algorithms. Neural Computing and Applications, 32, 3923-3937. https://doi.org/10.1007/s00521-018-3822-5

© Author(s) 2023. This work is distributed under https://creativecommons.org/licenses/by-sa/4.0/