The Visual Computer
https://doi.org/10.1007/s00371-023-03161-w

ORIGINAL ARTICLE

Segmentation-driven feature-preserving mesh denoising

Weijia Wang[1] · Wei Pan[2] · Chaofan Dai[3] · Richard Dazeley[1] · Lei Wei[1] · Bernard Rolfe[1] · Xuequan Lu[4]

Accepted: 29 October 2023
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

Abstract
Feature-preserving mesh denoising has received noticeable attention in visual media, with the aim of recovering high-fidelity, clean mesh shapes from the ones that are contaminated by noise. Existing denoising methods often design smaller weights for anisotropic surfaces and larger weights for isotropic surfaces in order to preserve sharp features, such as edges or corners, on the mesh shapes. However, they often disregard the fact that such small weights on anisotropic surfaces still pose negative impacts on the denoising outcomes and detail preservation results on the shapes. In this paper, we propose a novel segmentation-driven mesh denoising method which performs region-wise denoising, and thus avoids the disturbance of anisotropic neighbour faces for better feature preservation results. Also, our backbone can be easily embedded into commonly used mesh denoising frameworks. Extensive experiments have demonstrated that our method can enhance the denoising results on a wide range of synthetic and real mesh models, both quantitatively and visually.

Keywords Mesh denoising · 3D Vision · 3D Processing

1 Introduction

Mesh denoising is a fundamental research problem in geometry processing. The denoised mesh models can be applied to other computer vision tasks such as 3D modelling, computer animation and industrial design [1–3]. The main challenges lie in the removal of noise while preserving the features (e.g. edges, boundaries and corners) on mesh shapes.

Most existing mesh denoising methods focus on analysing local geometries (e.g. a triangular face and its surrounding neighbours) on triangle mesh surfaces. However, it still remains a challenge in recovering sharp features from noisy mesh models. For example, some methods based on face normals [4–7] often consider designing small weights for anisotropic surfaces (i.e. where the orientations of neighbouring faces are significantly different) and large weights for isotropic surfaces (i.e. where neighbouring faces have similar normal orientations), in order to preserve sharp features on mesh shapes. Nonetheless, such small weights on anisotropic surfaces still affect the denoising outcomes. There are also anisotropic mesh denoising methods (e.g. [8]) attempting to preserve sharp edges and corners on mesh shapes during denoising, but they suffer from recovering features that are contaminated by severe noise. A few other methods utilise additional information (e.g. building low-rank matrices based on mesh geometries [9, 10]) for mesh denoising. However, such methods are usually very slow due to high computational complexity.

To address the above issues, we introduce a novel segmentation-driven mesh denoising approach to greatly facilitate feature-preserving mesh filtering. Our key idea

Weijia Wang and Wei Pan are co-first authors.

✉ Weijia Wang
wangweijia@deakin.edu.au

✉ Wei Pan
vpan@foxmail.com

✉ Xuequan Lu
b.lu@latrobe.edu.au

Chaofan Dai
daichaofan@optmv.com

Richard Dazeley
richard.dazeley@deakin.edu.au

Lei Wei
lei.wei@deakin.edu.au

Bernard Rolfe
bernard.rolfe@deakin.edu.au

1 Deakin University, Geelong, VIC, Australia

2 OPT Machine Vision Tech Co., Ltd. (Japan), Tokyo, Japan

3 OPT Machine Vision Tech Co., Ltd., Dongguan, China

4 La Trobe University, Bundoora, VIC, Australia

Published online: 29 November 2023

is to partition the mesh surface into segments using an edge-based operator and utilise the segments to guide denoising. Specifically, we cluster triangular faces with similar geometric information (e.g. face normals) into regions and perform region-wise denoising, which assists with eliminating disturbance from anisotropic neighbours and preserving features in the denoised shapes. Furthermore, our segmentation backbone is flexible and can be easily embedded into commonly used mesh denoising methods. Extensive experimental results have proven that our segmentation-driven approach can achieve desirable mesh denoising outcomes in terms of feature preservation, both quantitatively and visually.

Our main contributions are summarised as follows:

- We propose a novel segmentation-driven mesh denoising approach, where the incorporation of the segmentation step into mesh denoising significantly improves the results in regard to feature preservation.
- We design an edge-based segmentation approach to partition the mesh surface into regions and use them to guide region-wise denoising, which eliminates disturbances from anisotropic neighbouring faces.
- Our pipeline's denoising performance is robust on mesh models contaminated by severe noise.
- Our method is compatible with various denoising methods, which can significantly enhance their performance.

2 Related work

Both mesh segmentation and mesh denoising have been extensively studied over the decades. In this section, we aim to provide an overview of both areas, summarising the employed methodologies.

2.1 Mesh segmentation

Mesh segmentation is partitioning the surface of a mesh into meaningful subsets. The methods are usually divided into two broad categories: (1) *semantic* segmentation, which segments the mesh surface into meaningful clusters based on semantic information (e.g. body parts of a human or areas of a city landscape [11–13]); and (2) *geometric* segmentation, which clusters triangular mesh facets based on geometric criteria such as curvatures and normals.

In this paper, we focus on reviewing *geometric* segmentation methods. They can be divided into two classes: *region-based* and *boundary-based* methods. Methods of the former class usually gather regions with similar geometric information (such as curvature or planarity) together, where the most representative methods are K-means and its variants [14]. Inspired by the iterative fitting scheme in K-means, Cohen-Steiner et al. [15] proposed variational shape approximation (VSA), an iterative scheme to reduce distortion error in order to find the best-fitting regions to cluster mesh facets. In [16, 17], simple linear iterative clustering (SLIC) technique is adopted to efficiently compute super facets with the K-means approach. Similar to the superpixel concept (i.e. perceptual grouping of pixels) in image processing, triangle faces with similar geometric metrics are grouped into super facets in such works. In addition, there are also other region-based clustering methods such as mean-shift [18], medoidshift [19], quick shift [20], hierarchical decomposition [21], primitive fitting [22] and random walks [23].

Segmentation methods of the latter class detect geometric feature boundaries in input meshes, such that each shape can be divided into different regions based on the boundaries. Relevant methods include randomised cuts [24], fuzzy clustering and cuts (FCC) [21], shape diameter function (SDF) [25] and 3D mesh scissoring [26, 27]. These methods heavily rely on the local geometric information of the input mesh, and may easily fail on complicated or extremely noisy meshes.

In addition to the methods above, recent advances in deep learning lead to data-driven methods for mesh segmentation [28, 29]. The majority of them deal with the semantic segmentation problem; see [30] for a comprehensive review.

2.2 Mesh denoising

The Laplacian smoothing methods [31, 32] are early research works in the field of mesh denoising, which perform isotropic smoothing and are thus fast and efficient. Later, a series of enhanced isotropic methods utilising different techniques such as differential properties, volume preservation and pass frequency-controlling were introduced [33–37] to improve mesh denoising outcomes. Nevertheless, while such methods are effective for noise removal, their isotropic nature may blur or shrink sharp features on the mesh shapes.

In order to alleviate such feature-blurring problems, anisotropic methods started to emerge. An early anisotropic denoising work by Hildebrandt and Polthier [8] utilises mean curvature flow to denoise mesh shapes. To better preserve features, two-step methods such as bilateral filtering techniques [5, 38] and others [4, 6, 7, 9, 39] were proposed in the following years. Such methods involve two steps: normal smoothing and vertex updating, which have demonstrated promising outcomes for robust, feature-preserving mesh denoising. In recent years, some researchers also attempt to classify vertices and faces in order to distinguish features during mesh denoising [40–46]. Nonetheless, such classification strategies mainly focus on local neighbourhoods and are usually sensitive to noise. To mitigate the issue, Lu et al. [47, 48] proposed a pre-filtering technique before denoising as a remedy solution, such that the impact brought by excessive noise is significantly reduced.

Another stream of anisotropic mesh denoising method focuses on the sparse perspective, as feature vertices can be computed by solving linear sparse systems. For example, He and Schaefer [49] proposed an L_0-minimisation framework, which utilises a discrete differential operator to preserve mesh features during denoising. While this method is straightforward, the minimisation process (as solving a sparse system) is non-convex and slow. To improve this, Zhao et al. [50] introduced an alternating optimisation strategy to perform L_0-minimisation, which consists of 2 steps (i.e. updating vertex positions and face normals). With similar inspiration and backbones, Lu et al. [48] introduced an L_1-minimisation method to preserve mesh features during denoising. Recently, by constructing half window of the local neighbourhood for each vertex, Pan et al. [51] proposed a half-kernel Laplacian operator to reduce the damages on features while removing noise. However, while this method is fast and effective, it has limited capability for sharp edge preservation on CAD-like models (i.e. models that have sharp edges separating smooth regions, such as cubes and octahedrons).

Over the years, there are works attempting to utilise mesh segmentation results to guide mesh denoising. For instance, Vieira and Shimanda [52] proposed a region-growing algorithm to segment noisy mesh surfaces and then smooth them out. While this is capable on scanned meshes, it can hardly preserve sharp features among the regions. Later, Huang and Ascher [53] proposed a vertex classification technique to segment meshes and guide denoising. However, it requires manually tuning the number of clusters based on the input shapes and does not achieve good feature-preserving results on scanned models. Lagrand et al. [54] proposed a variant of quick shift to automatically optimise the number and distribution of mesh segments, but the clustering result is obtained based on filtered meshes. Recently, Wang et al. [55] utilised Mumford-Shah function to obtain outlines for each noisy mesh and perform feature-preserving denoising. Nevertheless, the segmentation and denoising outcomes may still be unsatisfactory at high noise levels. With the benefits of mesh segmentation in mind, a method that can utilise the advantages of segmentation and guide feature-preserving denoising is thus greatly desired.

3 Method

3.1 Method overview

Figure 1 provides an overview of our segmentation-driven mesh denoising method. The key idea is segmenting the triangular faces on a noisy mesh into regions based on the mesh's features (e.g. edges and corners). Each region is then denoised separately without being affected by neighbouring facets in other regions. In this way, the features can be more effectively preserved in the denoised mesh. Note that for meshes corrupted with relatively higher levels of noise, we first employ an additional pre-filtering step to remove excessive noise. We then perform region segmentation on the pre-filtered mesh so that more accurate segmentation results can be obtained. Finally, we map the segmented regions back to the original unprocessed noisy mesh and utilise the regions to guide mesh filtering. We elaborate the pre-filtering step in Sect. 3.2, demonstrate our segmentation approach in Sect. 3.3, and introduce the denoising techniques in Sect. 3.4.

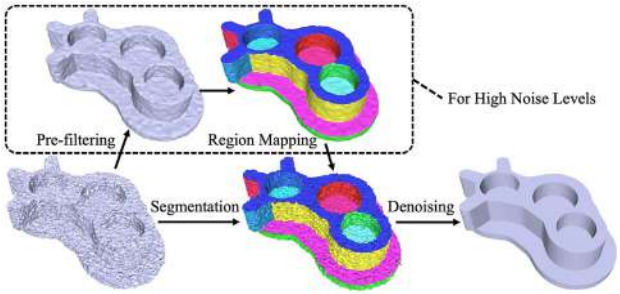

Fig. 1 Our mesh denoising pipeline

3.2 Pre-filtering

We aim to obtain high-quality segmentation results to guide our subsequent denoising operations. For meshes with low noise, we directly perform segmentation on them as their triangular faces are not severely distorted. Nevertheless, the segmentation task becomes extremely difficult on meshes corrupted with high noise levels (i.e. $\geq$ 30% of the mesh's average edge length as per our experimental results), where the face orientations are severely distorted or even flipped. It is extremely difficult to directly perform area segmentation on such mesh shapes, as the severely distorted triangular faces do not provide indicative information for our segmentation process. Figure 2 demonstrates an example of region segmentation at low and high noise levels, where the segmentation result is significantly affected in the latter case. Thus, we introduce this *pre-filtering* step to remove excessive noise to obtain better segmentation results. Inspired by Lu et al. [47], we formulate this pre-filtering step as a convex optimisation problem, where we estimate new vertex positions for the mesh. The objective function is formulated as

$$\min \sum_i \|\tilde{p}_i - p_i\|_2^2 + \alpha \sum_e w(e)\|D(e)\|_2^2 + \beta \sum_e w(e)\|R(e)\|_2^2, \tag{1}$$

where p_i is the original i-th vertex of the input mesh and $\tilde{p}_i$ is its unknown denoised position. $D(e)$ and $R(e)$ are the

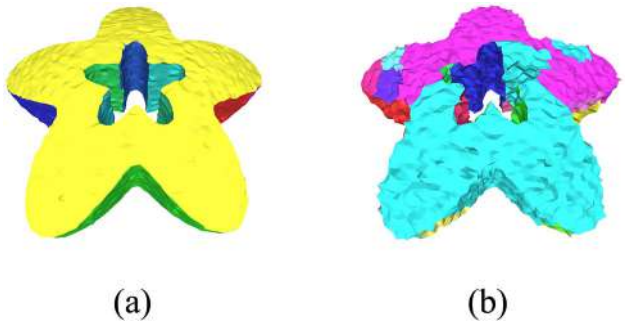

Fig. 2 Demonstration of directly performing segmentation on a mesh shape corrupted with **a** low noise ($0.15l_e$), and **b** high noise ($0.3l_e$)

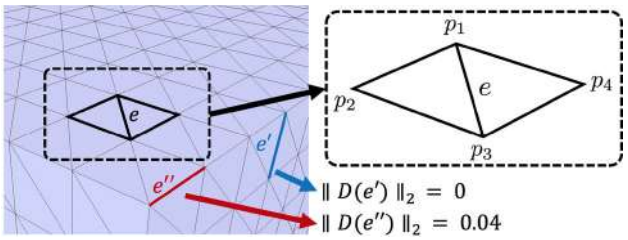

Fig. 3 Demonstration of the edges and vertices. For any edge e, we denote the order of the 4 adjacent vertices as per [49]. Also, for edge e' on a flat surface and edge e'' on a sharp feature, the L_2-norms of their edge operators are 0 and 0.04, respectively, in this example

area-based edge operator and the regulariser that were originally defined in [49]. Both $D(e)$ and $R(e)$ are weighted by a Gaussian function $w(e) = e^{-\left(\frac{\theta}{\sigma_\theta}\right)^2}$; the function itself is designed to preserve features during the pre-filtering step, where θ represents the angle formed by each two adjacent faces' normals, and σ_θ is a scaling threshold for normal similarity. In addition, $D(e)$ and $R(e)$ are, respectively, multiplied by two global weighting parameters, α and β.

We solve Eq. (1) as a sparse linear system. In specific, we firstly conduct 1 unweighted iteration (i.e. without the term $w(e)$), followed by 2 weighted iterations (i.e. with $w(e)$) as per [47]. Also, based on our experimental results, we set α, β and σ_θ from $w(e)$ in Eq. (1) to 0.2, 0.1 and 30 degrees, respectively, as such values lead to satisfactory segmentation results on our test shapes.

3.3 Edge-based region growing segmentation

Segmentation We propose to use edge metrics to perform mesh segmentation. An intuitive way is to utilise the face normal information on the mesh surface to distinguish features. To do so, we set a global threshold N_{thr} which is defined as

$$N_{thr} = \frac{\sum_{e \in E} \cos(n_i, n_j)}{|E|}, \tag{2}$$

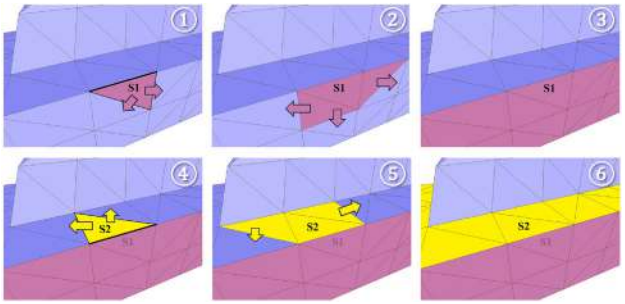

Fig. 4 Demonstration of our edge-based region growing segmentation method, where the sequence number is marked at the top right corner in each step. The seeding face $S1$ will be expanded based on edge-connected normal angles and the edge operators. When the expansion of $S1$ stops, we randomly select an unprocessed face $S2$ and repeat the expansion process, until all faces on the mesh are clustered

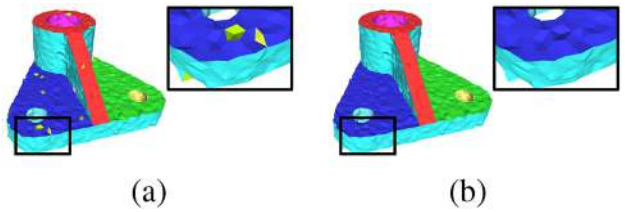

Fig. 5 Segmentation results **a** before and **b** after region refinement

where $|E|$ is the total number of edges in the input mesh, n_i and n_j are the normals of the two faces sharing edge e, and $\cos(n_i, n_j)$ is the cosine value of the angle formed by n_i and n_j.

Nevertheless, since noise distorts the faces' orientations, this global threshold may detect false features in some cases. To alleviate this issue, we introduce $D(e)$ from Eq. (1) as an edge operator to detect edge features in a local manner. As per the definition in [49], for any two adjacent triangles sharing an edge e, we assume the surrounding four vertices are ordered as per Fig. 3. Based on this order, $D(e)$ is defined as

$$D(e) = \begin{bmatrix} \frac{\triangle_{123}(p_4-p_3)\cdot(p_3-p_1)+\triangle_{134}(p_1-p_3)\cdot(p_3-p_2)}{(|p_3-p_1|)^2(\triangle_{123}+\triangle_{134})} \\ \frac{\triangle_{134}}{\triangle_{123}+\triangle_{134}} \\ \frac{\triangle_{123}(p_3-p_1)\cdot(p_1-p_4)+\triangle_{134}(p_2-p_1)\cdot(p_1-p_3)}{(|p_3-p_1|)^2(\triangle_{123}+\triangle_{134})} \\ \frac{\triangle_{123}}{\triangle_{123}+\triangle_{134}} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}, \tag{3}$$

where $\triangle_{123}$ denotes the area of the triangle formed by p_1, p_2 and p_3, and $\triangle_{134}$ is the area of the triangle formed by p_1, p_3 and p_4.

$D(e)$ itself is a 1×3 vector, which essentially describes the feature or non-feature property of a specific edge e. We calculate its L_2-norm to determine whether or not e is a feature edge: the value should be 0 when the adjacent triangle

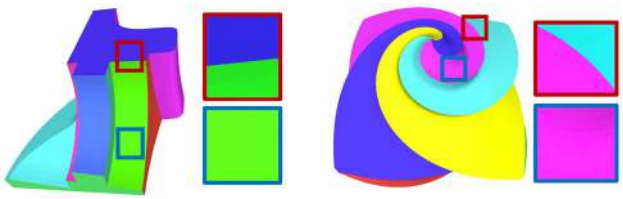

Fig. 6 Segmentation on shapes with mixed types of surfaces. Sharp edges (in red frames) are distinguished by our algorithm using two colours, while smooth regions (in blue frames) are marked in the same colour

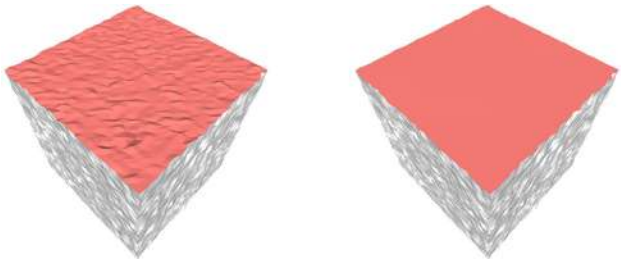

Fig. 7 Demonstration of region-based denoising, where faces in each region (marked in red) in the noisy shape (left) are denoised (right) without considering any neighbours outside the region

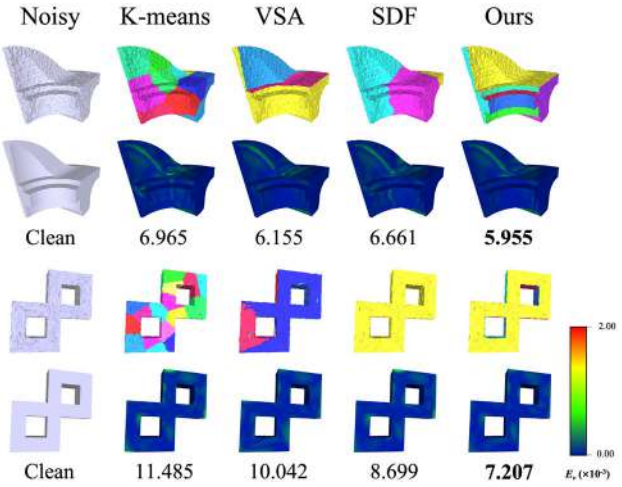

Fig. 8 Different segmentation techniques and their corresponding denoised results. Our segmentation approach leads to the minimum overall $E_v(\times 10^{-3})$ value

pair forms a flat dihedral angle, and increases if the dihedral angle becomes smaller (as shown in Fig. 3). Based on the properties of $D(e)$, we introduce a user-defined threshold D_{thr} to enable the algorithm to detect the feature edges. An edge is defined as a feature edge if the L_2-norm of its $D(e)$ is greater than D_{thr} (i.e. $\|D(e)\|_2 > D_{thr}$). However, merely relying on the term $D(e)$ is also very fragile for detecting feature edges, as shown in Fig. 23. Thus, for our segmentation technique, we use N_{thr} in Eq. (2) as a global prior term and use $D(e)$ as a complementary term. These two terms together form our edge detection metric for our region-growing segmentation algorithm. In Fig. 24, we show the segmentation results about tuning the user-defined threshold D_{thr}.

Figure 4 shows our region segmentation process. We first set a seeding triangle face in the input mesh and assign a region label to it. For each edge of the seeding face, we compute the cosine value of the normal angle formed with the corresponding edge-connected face (i.e. $\cos(n_i, n_j)$), as well as the edge's $D(e)$ value. We then determine if that edge-connected face belongs to the same cluster as the seeding face, by comparing $\cos(n_i, n_j)$ with N_{thr} and $\|D(e)\|_2$ with D_{thr}. If $\cos(n_i, n_j)$ is greater than N_{thr} or $\|D(e)\|_2$ is less than D_{thr}, we assign the cluster label of the seeding face to that connected triangle face. The newly clustered faces with the same label are all considered as seeding faces. We further calculate $\cos(n_i, n_j)$ and $\|D(e)\|_2$ for the remaining edges of the newly clustered faces and repeat this procedure to keep expanding the seeding cluster until no more adjacent triangles that meet the criteria can be found. When this happens, we randomly select a new seeding face from the remaining unclustered faces of the input mesh, and execute the above procedure again. Our segmentation procedure finishes when all faces on the mesh are clustered.

Region refinement Small clusters are sometimes observed after the region-growing segmentation process, as shown in Fig. 5a. Such small areas are leftover triangular faces which were not clustered into their surrounding regions due to noise. We further find that such small clusters usually pose negative impacts on mesh denoising, as discussed later in Sect. 5.4. Thus, we identify such overly small clusters and merge them into nearby ones. In our experiments, we empirically set this threshold to 50 faces, as it can identify the undesired small clusters being left due to noise. Specifically, for each triangle face in the small cluster, we calculate the cosine value of the current face normal and each of its surrounding 2-ring neighbourhood face normal (i.e. their normal similarity), and sum the cosine values within the same cluster. For simplicity, the adjacent cluster label which produces the greatest sum is

ALGORITHM 1: Region Growing Segmentation

Input: Mesh with low noise or pre-processed mesh, and the threshold D_{thr};
Output: Mesh partitioned in segments;
Compute N_{thr};
repeat
  Randomly select an unprocessed face F_i as a seed for a new cluster C;
  Get F_i's edge-connected neighbors $\{F_j\}$ and corresponding edges $\{e_j\}$;
  while $\cos(n_i, n_j) > N_{thr}$ ***or*** $\|D(e_j)\|_2 < D_{thr}$ **do**
    cluster F_j into C;
    mark F_j as a new seed;
  end
until *all faces are clustered*;
Refine the segments;

Table 1 Parameters of the mesh denoising methods

Methods	Number of parameters	Parameters description
L0	3	β_{max}: maximum value of beta
		α_0: initial value for alpha
		λ: weight for the L_0 term in the target function
BNF	3	σ_s: variance parameter for the spatial kernel
		n_{iter}: number of iterations for normal update
		v_{iter}: number of iterations for vertex update
UNF	3	T: threshold for controlling the averaging weights
		n_{iter}: number of iterations for normal update
		v_{iter}: number of iterations for vertex update
GNF	5	r: radius for finding a geometrical neighbourhood
		σ_s:variance parameter for the spatial kernel
		σ_r: variance parameter for the range kernel
		n_{iter}: number of iterations for normal update
		v_{iter}: number of iterations for vertex update
L1	3	σ: variance parameter for the spatial kernel
		n_{iter}: number of iterations for normal update
		v_{iter}: number of iterations for vertex update
HLO	1	$iter$: number of filtering iterations
TGV	3	α_1: first-order parameter
		α_0: second-order parameter
		β: fidelity term
Ours	$1 + X$	D_{thr}: edge-based segmentation threshold
		X: the parameters of other methods

The tuned parameters are shown in the same order in Table 2

assigned to this triangle face. The function is defined as

$$\arg\max_{k} \sum_{j \in S(i)} \cos(n_i, n_j), \tag{4}$$

where k indicates the desired cluster label, n_i is the current face normal, and $j \in S(i)$ represents any face with index j in the 2-ring neighbourhood face set $S(i)$, with its face normal denoted as n_j. The desired label k is chosen from all candidate faces in the face set $S(i)$. The segmentation result after the region refinement process is demonstrated in Fig. 5b, where the small clusters are fused into surrounding regions.

The full pipeline of our edge-based region growing segmentation algorithm is summarised in Algorithm 1. Overall, the term N_{thr} provides a global threshold for guiding feature identification during region segmentation. Based on this global prior term, the use of D_{thr} helps with distinguishing mixed types of surfaces, such as shapes with combinations of sharp edges and soft curves. Figure 6 shows examples of segmentation performance on two mesh shapes, Fandisk and Octaflower, where we show close-ups of sharp regions in red frames and smooth regions in blue frames. As can be seen, smooth surfaces are clustered into the same region while sharp boundaries are distinguished by two colours.

3.4 Mesh denoising

Our segmentation algorithm partitions the triangular faces on the mesh shape into regions, where each region excludes anisotropic neighbouring faces. We then constrain the neighbouring faces and update their normals and vertices within each region. This step excludes negative influence from the neighbouring faces on anisotropic surfaces that may affect the denoising results and prevent feature preservation. The procedure is demonstrated in Fig. 7, where all faces within a specific region (marked in red) are denoised without including any adjacent faces outside the region. In this respect, many local-based mesh denoising techniques can gain benefits from our segmentation step, including [4–6, 48] (which are elaborated in Sect. 4). Experimental results showcase the effectiveness of our segmentation-driven approach in achieving state-of-the-art mesh denoising results, as well as significantly enhancing the performance of the aforementioned local-based denoising methods.

Table 2 Quantitative comparisons in regards to MAE (in degrees) and $E_v(\times 10^{-3})$, where the best results are marked in bold

Models	Methods	MAE	$E_v(\times 10^{-3})$	Parameters
Cube ($\sigma_n = 0.2l_e$) (Fig. 9)	L0	2.692	8.960	(1000, 0.0033, 0.01)
$\|V\|$: 12,288 $\|F\|$: 6,146 $D_{thr} = 0.01$	BNF	2.325	4.758	(0.45, 80, 40)
	UNF	0.460	1.229	(0.35, 20, 50)
	GNF	0.530	1.267	(2, 1, 0.35, 20, 10)
	L1	0.519	1.303	(80, 40, 45)
	HLO	6.459	7.621	(3)
	TGV	0.294	1.163	(0.7, 0.2, 100)
	Ours (BNF)	**0.276**	**0.942**	(0.45, 80, 40)
	Ours (UNF)	0.447	1.197	(0.35, 20, 50)
	Ours (GNF)	0.436	1.207	(2, 1, 0.35, 20, 10)
	Ours (L1)	0.486	1.275	(80, 40, 45)
Casting ($\sigma_n = 0.1l_e$) (Fig. 10)	L0	9.386	2.452	(1000, 0.0051, 0.0001)
$\|V\|$: 5086 $\|F\|$: 10,204 $D_{thr} = 0.001$	BNF	8.039	2.244	(0.35, 20, 10)
	UNF	12.109	4.970	(0.4, 20, 20)
	GNF	10.685	1.682	(2, 1, 0.25, 5, 5)
	L1	6.642	**0.864**	(20, 10, 30)
	HLO	13.213	4.487	(3)
	TGV	**3.514**	0.970	(0.7, 0.05, 100)
	Ours (BNF)	7.731	2.059	(0.35, 20, 10)
	Ours (UNF)	12.486	4.640	(0.4, 20, 20)
	Ours (GNF)	7.189	1.191	(2, 1, 0.25, 5, 5)
	Ours (L1)	6.633	**0.864**	(20, 10, 30)
Double torus ($\sigma_n = 0.2l_e$) (Fig. 11)	L0	**1.109**	7.367	(1000, 0.0024, 0.01)
$\|V\|$: 8702 $\|F\|$: 17,408 $D_{thr} = 0.01$	BNF	8.049	50.788	(0.45, 80, 40)
	UNF	2.869	8.509	(0.35, 20, 50)
	GNF	2.697	10.126	(2, 1, 0.35, 20, 10)
	L1	3.116	9.718	(80, 40, 45)
	HLO	8.781	50.171	(3)
	TGV	1.181	8.132	(0.7,0.2, 100)
	Ours (BNF)	2.520	9.943	(0.45, 80, 40)
	Ours (UNF)	2.832	8.282	(0.35, 20, 50)
	Ours (GNF)	2.403	**7.207**	(2, 1, 0.35, 20, 10)
	Ours (L1)	3.034	9.735	(80, 40, 45)
Dodecahedron ($\sigma_n = 0.4l_e$) (Fig. 12)	L0	12.549	25.771	(1000, 0.0038, 5)
$\|V\|$: 4610 $\|F\|$: 9,216 $D_{thr} = 0.005$	BNF	15.350	13.808	(0.33, 40, 20)
	UNF	14.926	11.117	(0.4, 15, 10)
	GNF	8.833	6.832	(2, 1, 0.2, 75, 20)
	L1	13.846	8.213	(80, 40, 45)
	HLO	9.641	16.432	(5)
	TGV	8.967	8.219	(0.7. 0.2, 100)
	Ours (BNF)	8.976	3.878	(0.33, 40, 20)
	Ours (UNF)	10.366	5.007	(0.4, 15, 10)
	Ours (GNF)	**7.159**	**2.695**	(2, 1, 0.2, 75, 20)
	Ours (L1)	10.449	5.486	(80, 40, 45)

Table 2 continued

Models	Methods	MAE	$E_v(\times 10^{-3})$	Parameters
Icosahedron ($\sigma_n = 0.4l_e$) (Fig. 13)	L0	9.440	5.604	(1000, 0.0022, 1)
$\|V\|$: 10,242 $\|F\|$: 20,480 $D_{thr} = 0.0008$	BNF	9.859	3.380	(0.33, 40, 20)
	UNF	8.361	2.057	(0.4, 15, 10)
	GNF	3.534	1.155	(2, 1, 0.2, 75, 20)
	L1	5.925	1.463	(80, 40, 45)
	HLO	6.472	2.909	(5)
	TGV	3.576	3.391	(0.7, 0.2, 100)
	Ours (BNF)	3.712	0.946	(0.33, 40, 20)
	Ours (UNF)	4.685	1.054	(0.4, 15, 10)
	Ours (GNF)	**3.111**	**0.921**	(2, 1, 0.2, 75, 20)
	Ours (L1)	4.237	1.002	(80, 40, 45)
Cad ($\sigma_n = 0.3l_e$) (Fig. 14)	L0	2.673	202.264	(1000, 0.0013, 1)
$\|V\|$: 19,398 $\|F\|$: 38,792 $D_{thr} = 0.001$	BNF	2.418	207.254	(0.35, 25, 20)
	UNF	2.822	239.814	(0.55, 20, 40)
	GNF	2.730	252.715	(2, 1, 0.25, 25, 20)
	L1	2.807	186.496	(80, 40, 45)
	HLO	11.223	793.123	(5)
	TGV	2.029	377.546	(0.7, 0.2, 100)
	Ours (BNF)	**2.021**	181.763	(0.35, 25, 20)
	Ours (UNF)	2.523	219.908	(0.55, 20, 40)
	Ours (GNF)	2.439	245.837	(2, 1, 0.25, 25, 20)
	Ours (L1)	2.486	**165.722**	(80, 40, 45)

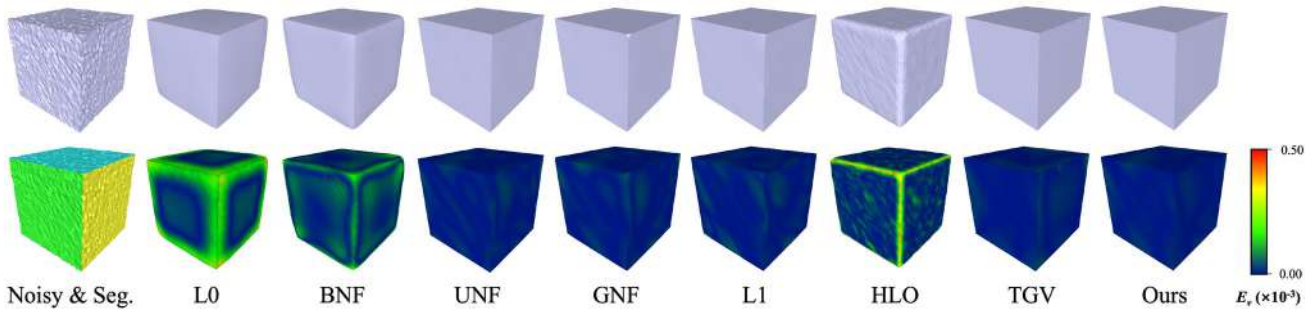

Fig. 9 Coloured E_v for Cube with noise $\sigma_n = 0.2l_e$, with Lee and Wang [38] as our denoise backbone

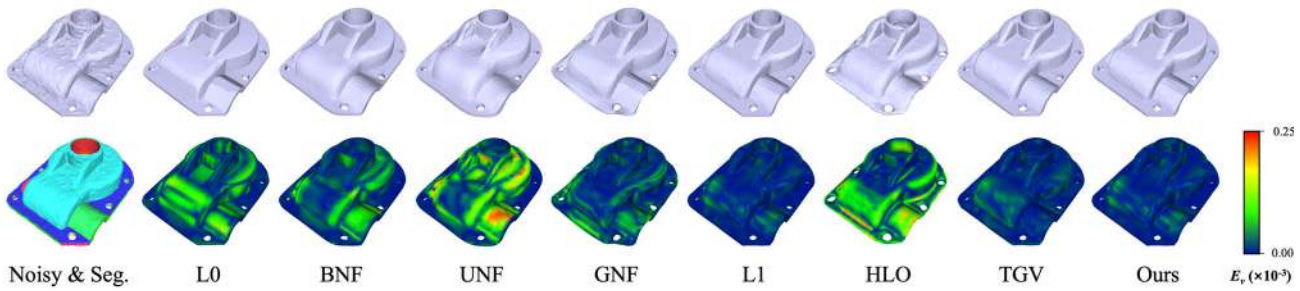

Fig. 10 Coloured E_v for Casting with noise $\sigma_n = 0.1l_e$, with Lu et al. [48] as our denoise backbone

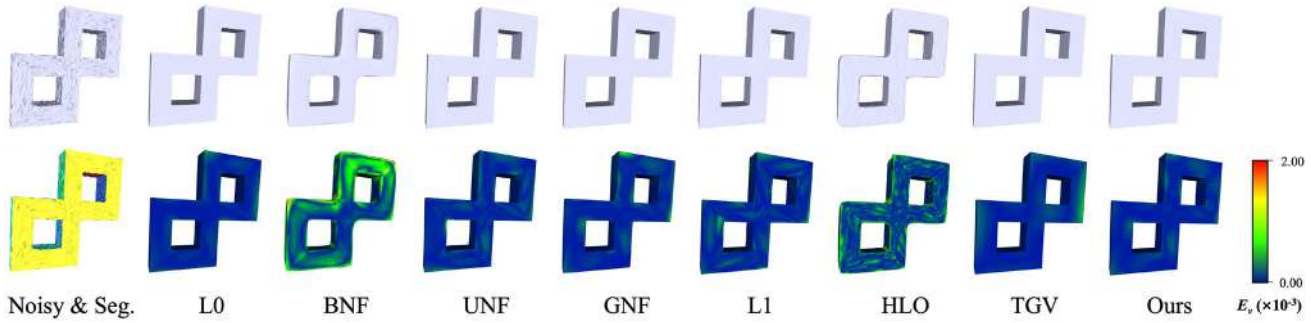

Fig. 11 Coloured E_v for Double Torus with noise $\sigma_n = 0.2l_e$, with Zhang et al. [6] as our denoise backbone

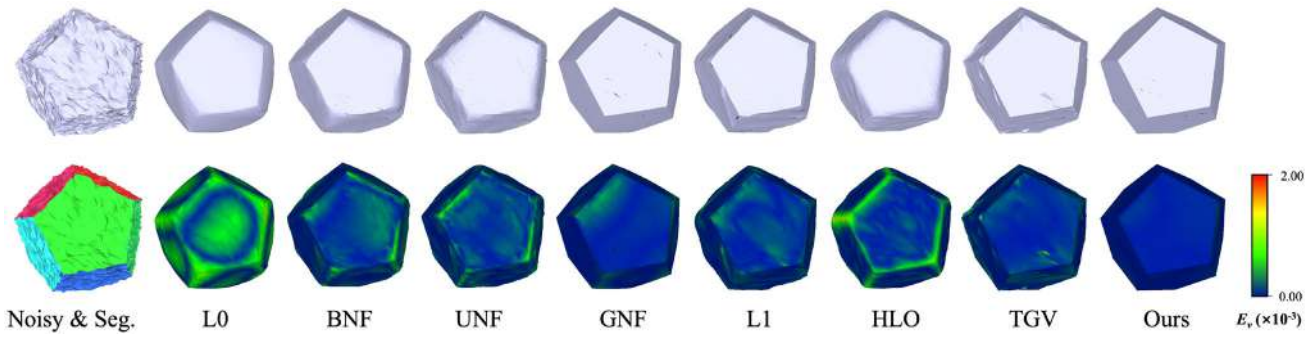

Fig. 12 Coloured E_v for Dodecahedron with noise $\sigma_n = 0.4l_e$, with Zhang et al. [6] as our denoise backbone

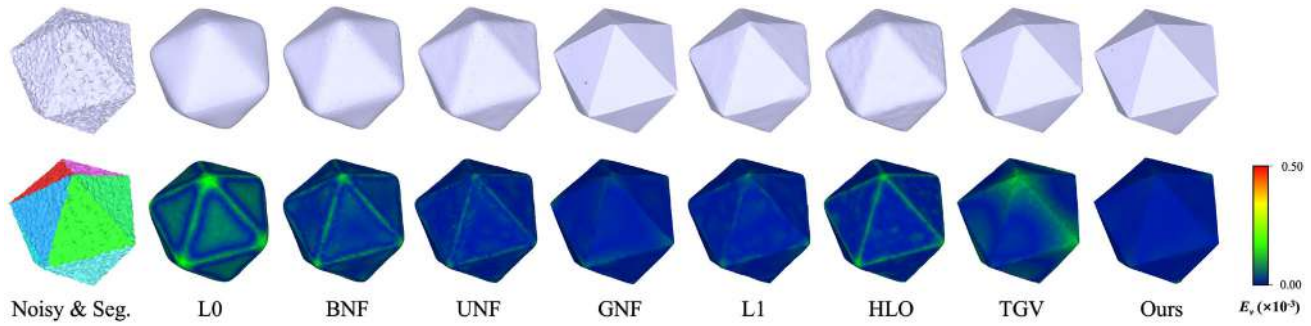

Fig. 13 Coloured E_v for Icosahedron with noise $\sigma_n = 0.4l_e$, with Zhang et al. [6] as our denoise backbone

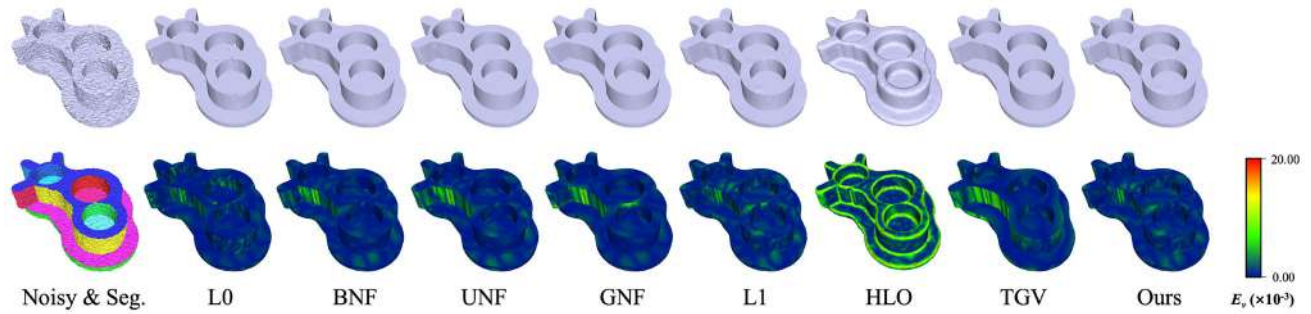

Fig. 14 Coloured E_v for Cad with noise $\sigma_n = 0.3l_e$, with Lu et al. [48] as our denoise backbone

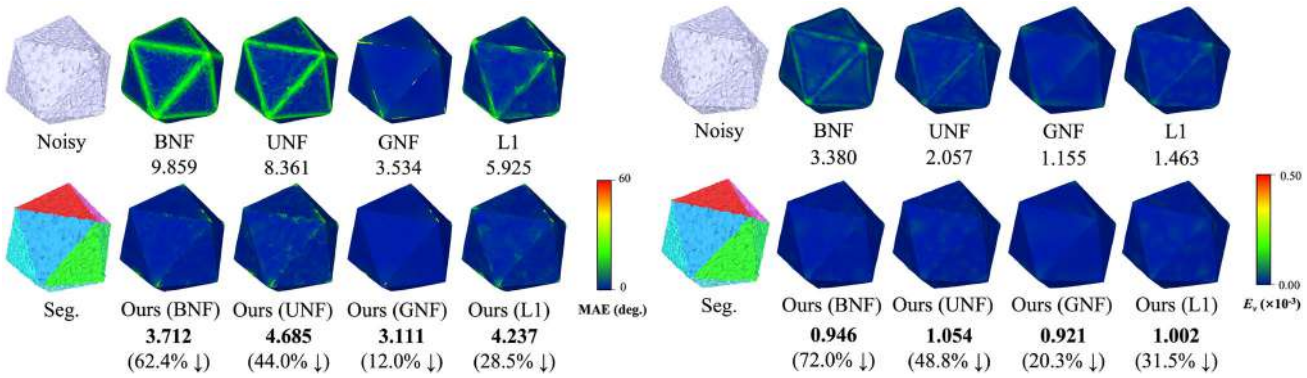

Fig. 15 The MAE values (left, in degrees) and the E_v values (right, $\times 10^{-3}$), along with the changes in percentages after embedding our segmentation backbone

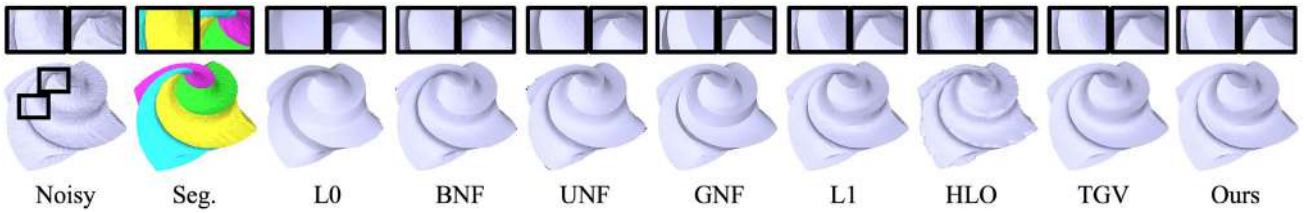

Fig. 16 Visual comparison for Octaflower with noise $\sigma_n = 0.1l_e$, with Lu et al. [48] as our denoise backbone

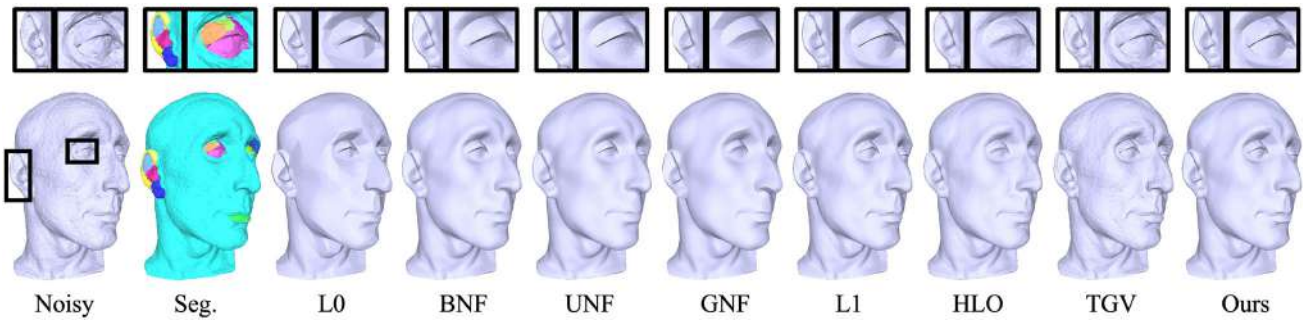

Fig. 17 Visual comparison for Nicolo with noise $\sigma_n = 0.1l_e$, with Lu et al. [48] as our denoise backbone

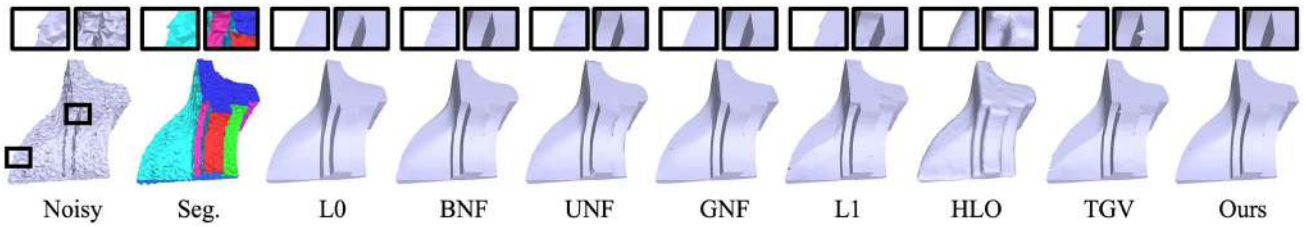

Fig. 18 Visual comparison for Fandisk with noise $\sigma_n = 0.4l_e$, with Zhang et al. [6] as our denoise backbone

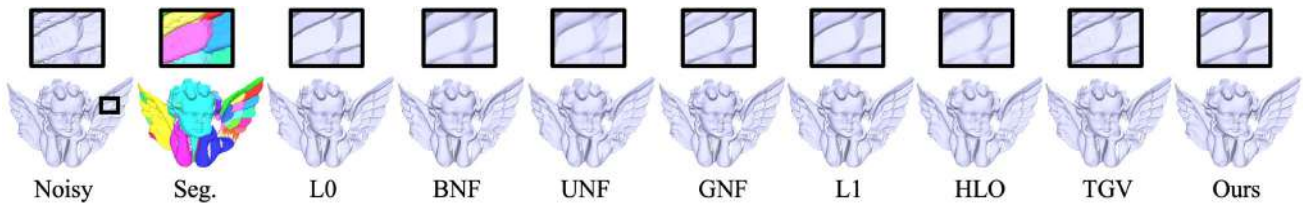

Fig. 19 Visual comparison for a raw scanned Angel mesh, with Lu et al. [48] as our denoise backbone

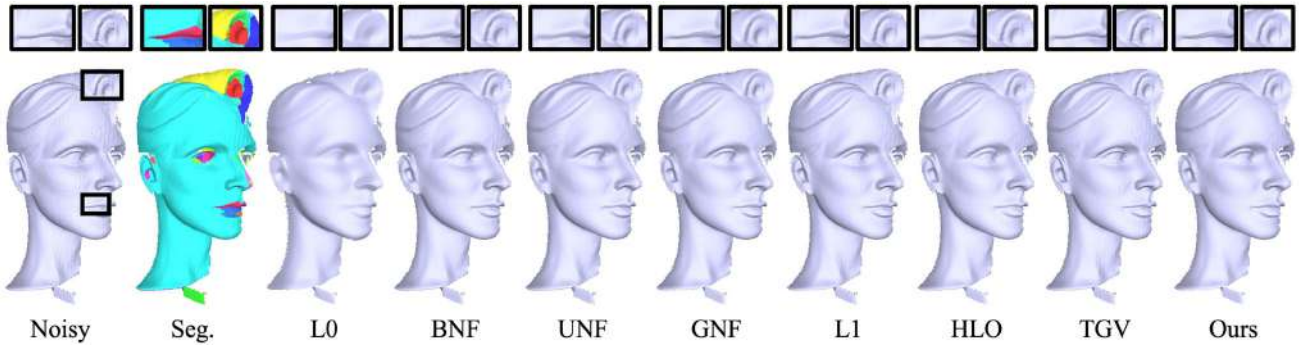

Fig. 20 Visual comparison for a raw scanned Wilhelm mesh, with Zhang et al. [6] as our denoise backbone

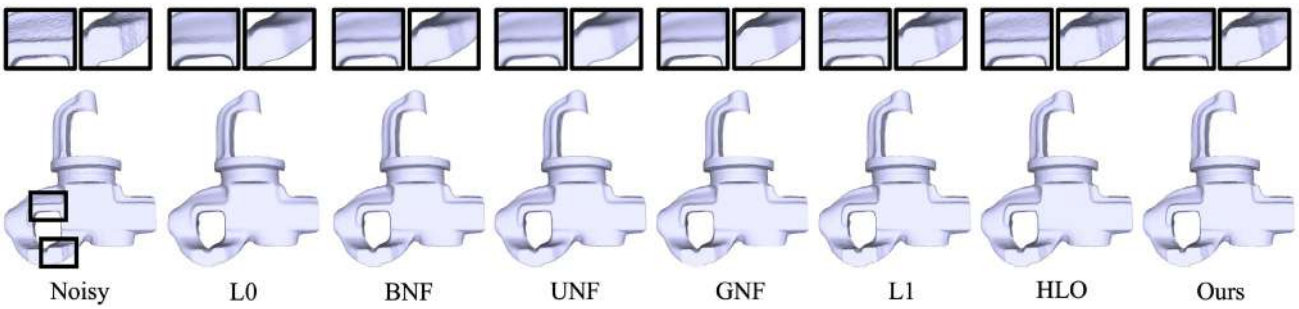

Fig. 21 Visual comparison for a raw scanned Iron mesh, with Lu et al. [48] as our denoise backbone

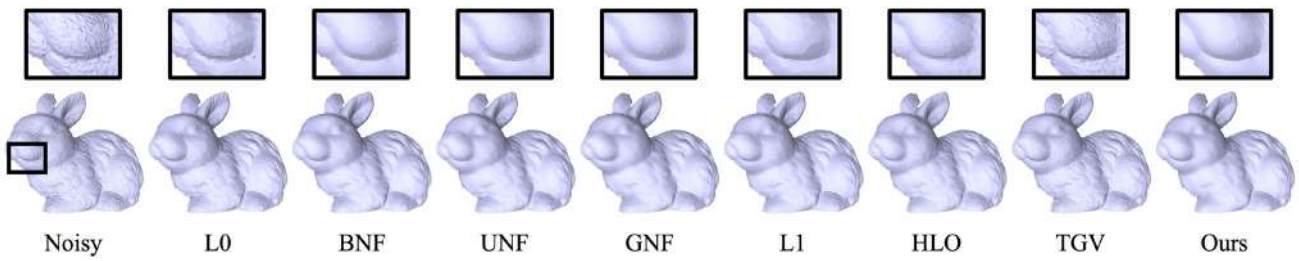

Fig. 22 Visual comparison for a raw scanned Rabbit mesh, with Zhang et al. [6] as our denoise backbone

4 Experimental results

We first compare the performance of different segmentation methods in Sect. 4.1. Then, we select the following methods for denoising comparison: L_0-minimisation (L0) [49], the bilateral normal filter (BNF) [38], the unilateral normal filter (UNF) [4], the guided normal filter (GNF) [6], the L_1-median filter (L1) [48], the half-kernel Laplacian operator (HLO) [51] and total generalised variation (TGV) mesh denoising [56]. We select them since they are the state-of-the-art works over the years, and their source code or executables are available online. For our denoising step, we embed four open-source methods into our segmentation backbone: BNF, UNF, GNF and L1.

Similar to our pipeline, some of the comparison methods also utilise the anisotropic denoising idea. For instance, the bilateral filter in [38] utilises anisotropic diffusion during the denoising task. Similarly, the guidance normal construction process in [6] is inspired by anisotropic denoising (i.e. judging a face patch's normal consistency). Nevertheless, experimental results show that such methods still have limitations on feature preservation and our segmentation-driven backbone can achieve more desirable effects on the denoised meshes.

4.1 Segmentation methods comparison

We compare the following segmentation methods: *K-means segmentation*, a classic method that has been adopted throughout the years [14]; *variational shape approximation (VSA)* [57], which partitions mesh shapes into their most representative parts; the segmentation method based on *shape diameter function (SDF)* [25], which segments mesh shapes based on facet thickness; and finally, our *edge-based seg-*

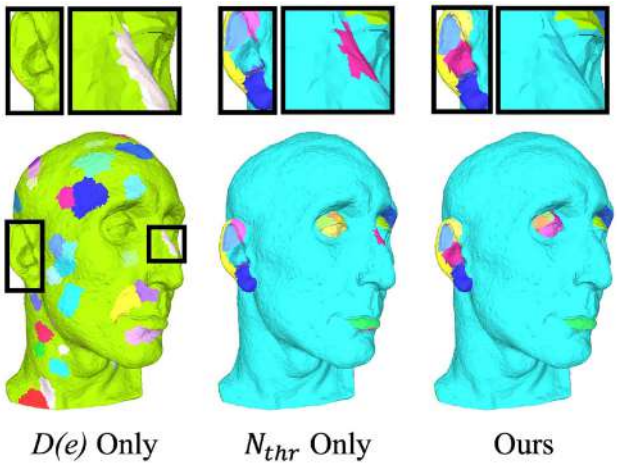

Fig. 23 Different strategies for segmentation on Nicolo with noise $\sigma_n = 0.1l_e$, where our segmentation strategy provides the most informative segmentation result

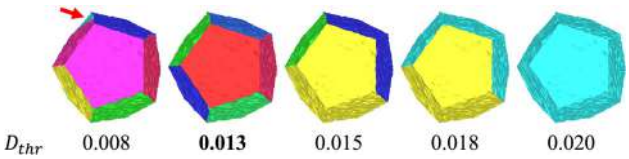

Fig. 24 Effects of tuning D_{thr} for mesh segmentation on a dodecahedron model with noise $\sigma_n = 0.2l_e$

mentation strategy. We exploit these segmentation methods to guide mesh denoising.

Figure 8 shows the segmentation results and the corresponding denoising effects on Fandisk and Double Torus, both corrupted with $\sigma_n = 0.2l_e$ noise. We adopt [6] as the denoising approach and use the vertex-based mesh-to-mesh error metric (E_v) to measure the accuracy of the denoised vertices' positions. We visualise the segmented regions for each shape and list the E_v error for each denoised shape together with its visualisation. As illustrated in the figure, both K-means and VSA cannot correctly capture the edge features on the mesh shapes, and the SDF segmentation method fails to properly partition the mesh shape based on sharp features. As a result, such segmentation methods all lead to sub-optimal denoising results and may result in unwanted folds in the denoised shapes. By contrast, our method accurately clusters all smooth regions and distinguishes the sharp features, leading to the minimum error.

4.2 Parameter setting for denoising

The parameters of the denoising methods for comparison are summarised in Table 1. For all methods, we use the recommended parameters and carefully tune them to obtain desired outputs. Compared with other methods, ours only has one extra parameter (i.e. the segmentation threshold D_{thr}), which is easy to tune. Note that for reasonable and fair comparisons, for each of our segmentation-driven method, we set the parameters the same as the corresponding original method.

4.3 Quantitative denoising evaluations

We firstly show quantitative evaluations on mesh models with synthetic Gaussian noise. Such noisy mesh models are generated by adding zero-mean Gaussian noise with standard deviation σ_n to the corresponding ground truth models. Here, σ_n describes the noise level, which is proportional to the mean edge length l_e of the input mesh, with the parameters shown in Table 2. As the corresponding clean mesh models are known, we employ two common metrics as suggested by previous works [4, 5, 56] to, respectively, evaluate the errors on denoised face normals and vertex positions: the Mean Angular Error (MAE) metric, which measures the average angular error between the face normals on the filtered mesh and the ones of the ground truth mesh; and the aforementioned vertex-based metric E_v, which measures the error of the denoised vertices' positions.

Table 2 lists MAE and E_v over six models for all methods. Our segmentation-driven backbone achieves the minimum E_v metric on all shapes, and achieves competitive results for MAE. To demonstrate our advantages, we visualise the E_v values with colour gradients in Figs. 9, 10, 11, 12, 13 and 14, along with our segmentation results (abbreviated as Seg. in the figures). Our advantages in feature preservation can be especially seen in Figs. 9, 12 and 13: despite L0 and BNF are generally good at dealing with CAD-like shapes, they tend to blur sharp edges on such shapes (which are reflected by the green colour near the edges); TGV generally performs well, but may still not be able to robustly preserve the edges; by contrast, with the assistance from our segmentation backbone, the E_v error on such shapes denoised by our method (especially on the edges) is significantly reduced.

In addition, as demonstrated in Table 2, our method significantly boosts the denoising effects of each original method. Figure 15 demonstrates the improvements on the MAE and E_v metrics, respectively, on the denoised Icosahedron model (the fifth shape in Table 2), showing the metric values and the changes in percentages.

4.4 Visual denoising results

Synthetic models We first compare visual results on various synthetic models corrupted with Gaussian noise. Analogous to previous research [4, 47], we find that the visual comparisons might be inconsistent with MAE and E_v. Thus, we focus on the visual effects in this section.

As shown in Fig. 16, our segmentation-driven approach assists with preserving the sharp tip on the Octaflower mesh while preserving the curve regions. Although the tip is

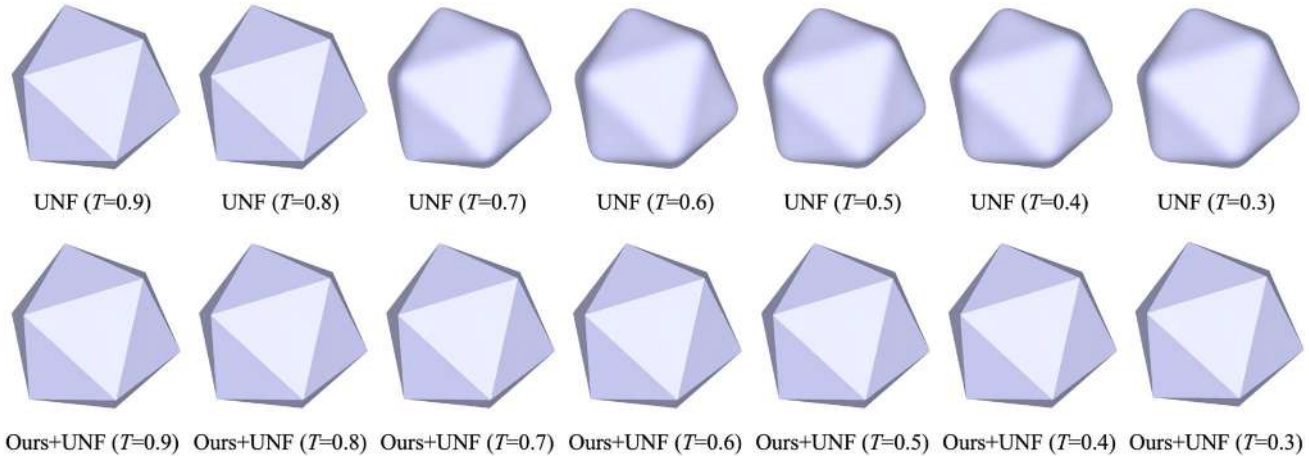

Fig. 25 Stability on denoising Icosahedron ($\sigma_n = 0.1l_e$) with different T using UNF [4]

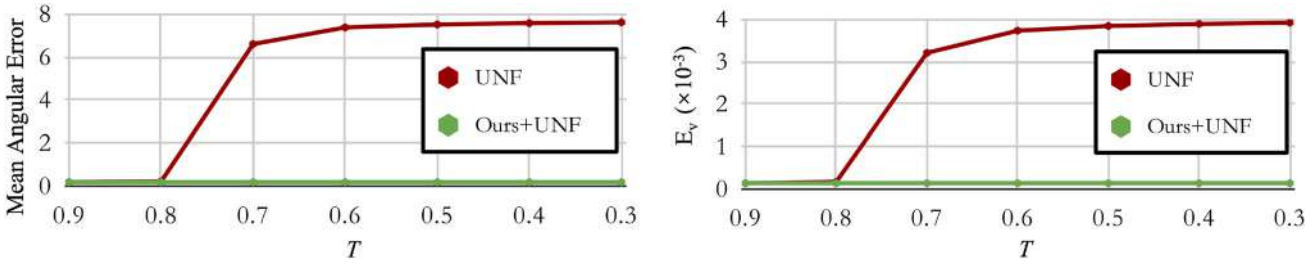

Fig. 26 The changes in MAE (left) and E_v (right) with different T values in Fig. 25

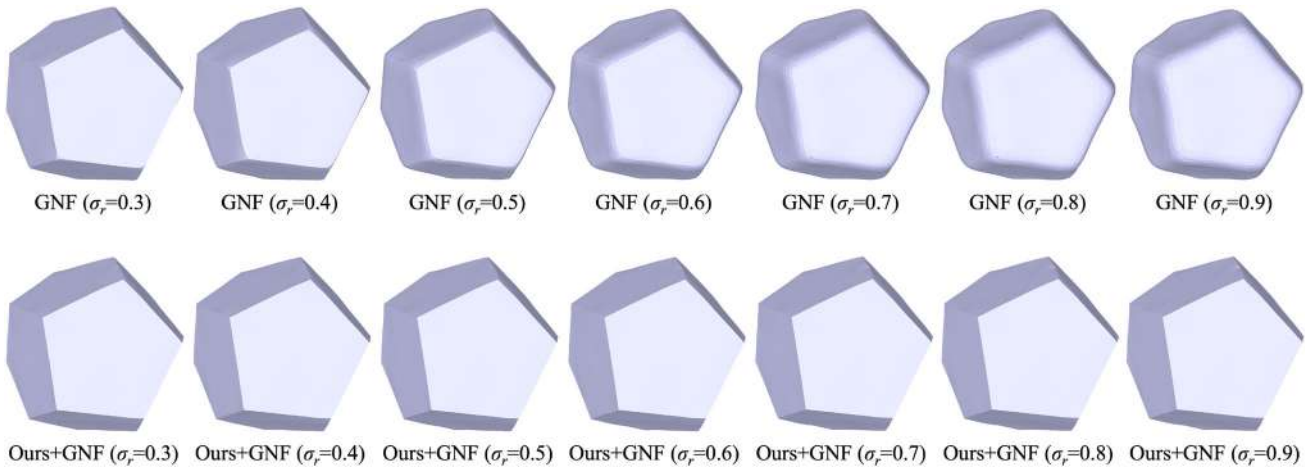

Fig. 27 Stability on denoising Dodecahedron ($\sigma_n = 0.2l_e$) with different σ_r using GNF [6]

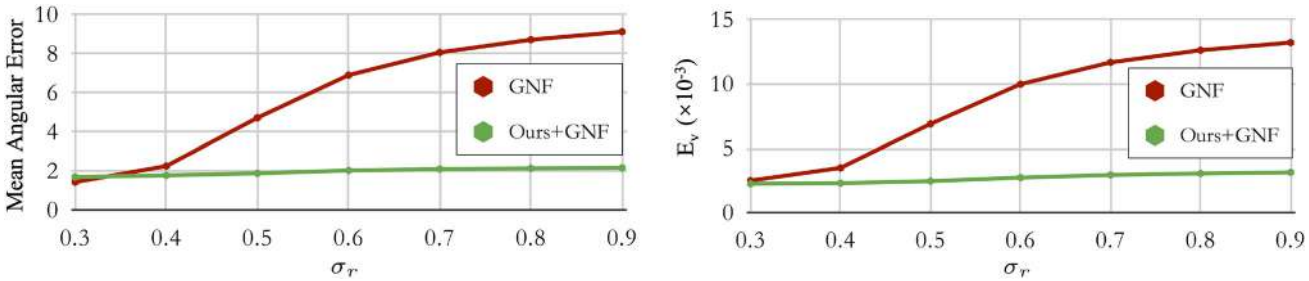

Fig. 28 The changes in MAE (left) and E_v (right) with different σ_r values in Fig. 27

blurred in the noisy shape, our segmentation method robustly recognises it and helps the denoising algorithm to reveal it. Also, unlike Zhang et al. [6] which oversharpens the curved regions, our method keeps the triangular facets on each curve surface within the same region, which assists in maintaining the curvatures of such areas. Similarly, as displayed in Fig. 17, our segmentation approach helps with restoring the details of the ear and the eye on the Nicolo model. Furthermore, most anisotropic methods (such as [8]) often struggle with preserving features at higher noise levels (i.e. $\sigma_n \geq 0.3l_e$) [48]. This can also be seen from Fig. 18, where BNF [38] cannot fully preserve features of the Fandisk model that is corrupted with severe noise. In contrast, our segmentation-driven method can produce smooth surfaces and preserve the sharp features on this model.

Raw scanned models In addition to the synthetic shapes, we compare denoising performance on scanned models corrupted with raw noise. Such shapes come from scanners and their noise is different from Gaussian noise. Despite the fact, our segmentation-driven pipeline can still preserve original features and details of the input meshes. For example, in Fig. 19, the details of the feathers on the Angel model are preserved. Similarly, on the Wilhelm model in Fig. 20, the details on the hair and the lips are maintained. Besides, our segmentation approach can also help with restoring details that are corrupted with the scanner's noise, such as the Iron model in Fig. 21 and the Rabbit model in Fig. 22.

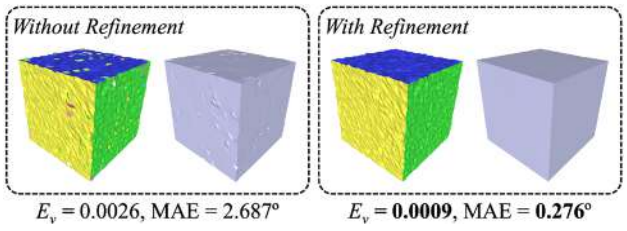

Fig. 29 Denoising results without and with the cluster refinement step, where the MAE and E_v values are smaller with the refinement step

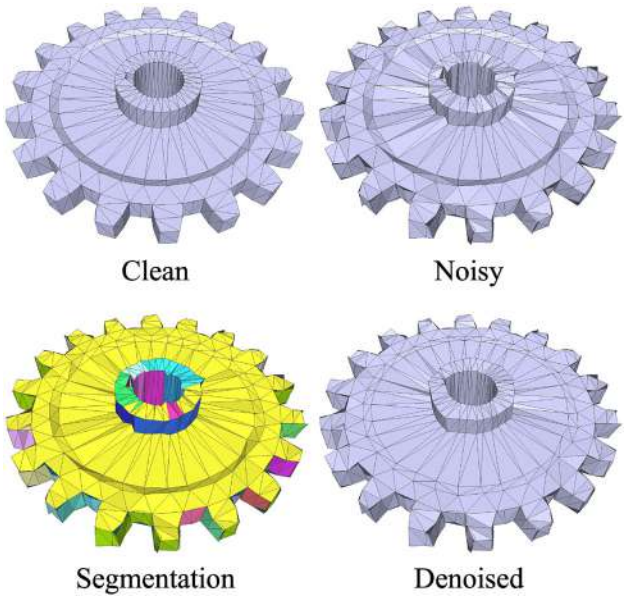

Fig. 30 A failure case of denoising a gear model with extremely sparse triangulation, with [48] as our denoising backbone

5 Discussion

5.1 Edge-based segmentation strategy

For our edge-based region growing segmentation algorithm, we compare against different strategies. The segmentation results in Fig. 23 demonstrate that either only utilising N_{thr} for region growth, or performing segmentation merely using $D(e)$, is not enough. As demonstrated, merely relying on $D(e)$ does not give informative regions, resulting in a large region with small patches. Only using N_{thr} for region growing may result in unwanted clusters, such as the extra magenta area on the nose, and some detailed regions (e.g. earhole) are not well-segmented. By contrast, our combined approach leverages both criteria and leads to a much more reasonable segmentation result that facilitates denoising.

5.2 Segmentation parameter

We demonstrate the effect of tuning the parameter D_{thr} and the corresponding segmentation results in Fig. 24. Setting D_{thr} to 0.013 (as marked in bold) for the dodecahedron shape produces the optimal segmentation result, where all edges are correctly identified. Setting the D_{thr} too low may output false features, while setting it too high may make the algorithm overlook some of the feature edges. In both cases, the segmentation results are sub-optimal.

5.3 Algorithms stability

Our segmentation result also increases the stability of the original algorithms, making them more robust against parameter variations. For example, Fig. 25 shows the denoising results of Icosahedron using the original version of UNF and the version with our segmentation backbone, tested with different values of threshold T. For the original UNF method, the visual results are quite sensitive to different T values. After embedding our backbone, the sharp edges are generally well preserved, even with different T. This can also be seen from the line charts of the computed MAE and E_v values in Fig. 26.

Similarly, Fig. 27 shows the denoising results on Dodecahedron using GNF itself and its embedded version with different σ_r values. The line charts of the obtained MAE and E_v values are shown in Fig. 28. These results demonstrate

that our approach can significantly boost the stability of the existing mesh denoising methods.

5.4 Refinement step

As previously mentioned, we apply a region refinement step after the initial segmentation step. In our experiments, we find that refining based on face numbers works well, but users might also perform region refinement based on other criteria such as face areas. The refinement step indeed assists in improving the quality of the denoised mesh. As shown in Fig. 29, small clusters are considered as anisotropic surfaces without the refinement step, even if they originally belong to the same region as their surrounding faces. As a consequence, it leaves pits and rough edges on the denoised mesh. In contrast, after applying the refinement step, the denoised shape has cleaner surfaces and tidier edges, achieving lower MAE and E_v values.

5.5 Limitations

Our method has a few limitations. For example (and similar to other methods such as [6]), for any model with extremely sparse and irregular triangulation, our segmentation backbone can hardly guide the denoising process. Figure 30 demonstrates segmentation and denoising results on a gear model with extremely sparse and irregular triangulation. Although our edge-based segmentation result can reasonably partition the mesh's geometric regions, the denoised gear model does not fully reveal its original shape. This can be seen at the edges, where the sparse triangles are folded.

6 Conclusion

We present a novel segmentation-driven mesh denoising framework, which facilitates the benefits brought by geometric mesh segmentation and helps with achieving feature-preserving mesh denoising. Moreover, it can be easily integrated with existing mesh denoising methods with demonstrated robustness. Both visual and quantitative results confirm that our method enables better feature-preserving mesh denoising outcomes, especially for models corrupted with severe noise.

Availability of data and materials The test mesh data set being presented in this paper is available on reasonable request.

Declarations

Conflict of interest The authors declare that there are no conflicts of interest.

References

1. Ströter, D., Mueller-Roemer, J.S., Weber, D., Fellner, D.W.: Fast harmonic tetrahedral mesh optimization. Vis. Comput. **38**(9), 3419–3433 (2022). https://doi.org/10.1007/s00371-022-02547-6
2. Jia, S., Zhang, W., Wang, G., Pan, Z., Yu, X.: A real-time deformable cutting method using two levels of linked voxels for improved decoupling between collision and rendering. Vis. Comput. **39**(2), 765–783 (2022). https://doi.org/10.1007/s00371-021-02373-2
3. Prashant, G.: A survey of modeling, rendering and animation of clouds in computer graphics. Vis. Comput. **37**(7), 1931–1948 (2020). https://doi.org/10.1007/s00371-020-01953-y
4. Sun, X., Rosin, P.L., Martin, R., Langbein, F.: Fast and effective feature-preserving mesh denoising. IEEE Trans. Vis. Comput. Graph. **13**(5), 925–938 (2007)
5. Zheng, Y., Fu, H., Au, O.K.C., Tai, C.L.: Bilateral normal filtering for mesh denoising. IEEE Trans. Vis. Comput. Graph. **17**(10), 1521–1530 (2011)
6. Zhang, W., Deng, B., Zhang, J., Bouaziz, S., Liu, L.: Guided mesh normal filtering. Comput. Graph. Forum **34**(7), 23–34 (2015)
7. Lu, X., Liu, X., Deng, Z., Chen, W.: An efficient approach for feature-preserving mesh denoising. Opt. Lasers Eng. **90**, 186–195 (2017)
8. Hildebrandt, K., Polthier, K.: Anisotropic filtering of non-linear surface features. Comput. Graph. Forum **23**(3), 391–400 (2004)
9. Lu, X., Schaefer, S., Luo, J., Ma, L., He, Y.: Low rank matrix approximation for 3d geometry filtering. IEEE Trans. Vis. Comput. Graph. **28**(4), 1835–1847 (2022)
10. Li, X., Zhu, L., Fu, C.W., Heng, P.A.: Non-local low-rank normal filtering for mesh denoising. Comput. Graph. Forum **37**(7), 155–166 (2018)
11. Yaz, I.O., Loriot, S.: Triangulated surface mesh segmentation. In: CGAL User and Reference Manual (2022). https://doc.cgal.org/5.4.1/Manual/packages.html#PkgSurfaceMeshSegmentation
12. Grzeczkowicz, G., Vallet, B.: Semantic segmentation of urban textured meshes through point sampling. ISPRS Ann. Photogram., Remote Sens. Spat. Inf. Sci. **2**, 177–184 (2022)
13. Hu, Z., Bai, X., Shang, J., Zhang, R., Dong, J., Wang, X., Sun, G., Fu, H., Tai, C.L.: Vmnet: voxel-mesh network for geodesic-aware 3d semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 15488–15498 (2021)
14. Lian, J., Li, H., Li, N., Cai, Q.: An adaptive mesh segmentation via iterative K-means clustering. In: Proceedings of 2021 Chinese Intelligent Systems Conference, pp. 193–201. Springer Singapore, Singapore (2022)
15. Cohen-Steiner, D., Alliez, P., Desbrun, M.: Variational shape approximation. In: ACM SIGGRAPH 2004 Papers, pp. 905–914 (2004)
16. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
17. Simari, P., Picciau, G., De Floriani, L.: Fast and scalable mesh superfacets. Comput. Graph. Forum **33**(7), 181–190 (2014)
18. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. **24**(5), 603–619 (2002)
19. Sheikh, Y.A., Khan, E.A., Kanade, T.: Mode-seeking by medoidshifts. In: 2007 11th IEEE International Conference on Computer Vision, pp. 1–8 (2007). https://doi.org/10.1109/ICCV.2007.4408978
20. Vedaldi, A., Soatto, S.: Quick shift and kernel methods for mode seeking. In: Computer Vision – ECCV 2008, pp. 705–718. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)

21. Katz, S., Tal, A.: Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Trans. Graph. (TOG) **22**(3), 954–961 (2003)
22. Attene, M., Falcidieno, B., Spagnuolo, M.: Hierarchical mesh segmentation based on fitting primitives. Vis. Comput. **22**(3), 181–193 (2006)
23. Lai, Y.K., Hu, S.M., Martin, R.R., Rosin, P.L.: Fast mesh segmentation using random walks. In: Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling, pp. 183–191 (2008)
24. Golovinskiy, A., Funkhouser, T.: Randomized cuts for 3d mesh analysis. In: ACM SIGGRAPH Asia 2008 papers, pp. 1–12 (2008)
25. Shapira, L., Shamir, A., Cohen-Or, D.: Consistent mesh partitioning and skeletonisation using the shape diameter function. Vis. Comput. **24**(4), 249 (2008)
26. Zheng, Y., Tai, C.L., Au, O.K.C.: Dot scissor: a single-click interface for mesh segmentation. IEEE Trans. Vis. Comput. Graph. **18**(8), 1304–1312 (2011)
27. Lee, Y., Lee, S., Shamir, A., Cohen-Or, D., Seidel, H.P.: Mesh scissoring with minima rule and part salience. Comput. Aided Geometr. Des. **22**(5), 444–465 (2005)
28. Kalogerakis, E., Hertzmann, A., Singh, K.: Learning 3d mesh segmentation and labeling. In: ACM SIGGRAPH 2010 papers, pp. 1–12 (2010)
29. Liu, C.M., Luan, W.N., Fu, R.H., Pang, H.B., Li, Y.H.: Attention-embedding mesh saliency. Vis. Comput. **39**(5), 1783–1795 (2022)
30. Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., Garcia-Rodriguez, J.: A survey on deep learning techniques for image and video semantic segmentation. Appl. Soft Comput. **70**, 41–65 (2018)
31. Vollmer, J., Mencl, R., Müller, H.: Improved Laplacian smoothing of noisy surface meshes. Comput. Graph. Forum **18**(3), 131–138 (1999)
32. Field, D.A.: Laplacian smoothing and Delaunay triangulations. Commun. Appl. Numer. Methods **4**(6), 709–712 (1988)
33. Liu, X., Bao, H., Shum, H.Y., Peng, Q.: A novel volume constrained smoothing method for meshes. Graph. Models **64**(3–4), 169–182 (2002)
34. Kim, B., Rossignac, J.: Geofilter: Geometric selection of mesh filter parameters. Comput. Graph. Forum **24**(3), 295–302 (2005)
35. Nehab, D., Rusinkiewicz, S., Davis, J., Ramamoorthi, R.: Efficiently combining positions and normals for precise 3d geometry. ACM Trans. Graph. **24**(3), 536–543 (2005)
36. Nealen, A., Igarashi, T., Sorkine, O., Alexa, M.: Laplacian mesh optimization. In: Proceedings of GRAPHITE'06, pp. 381–389 (2006)
37. Su, Z.X., Wang, H., Cao, J.J.: Mesh denoising based on differential coordinates. In: Proc. of IEEE Int'l Conf. on Shape Modeling and Applications 2009, pp. 1–6 (2009)
38. Lee, K.W., Wang, W.P.: Feature-preserving mesh denoising via bilateral normal filtering. In: Proc. of Int'l Conf. on Computer Aided Design and Computer Graphics (2005)
39. Chen, S., Wang, J., Pan, W., Gao, S., Wang, M., Lu, X.: Towards uniform point distribution in feature-preserving point cloud filtering. arXiv:2201.01503 (2022)
40. Wei, M., Yu, J., Pang, W., Wang, J., Qin, J., Liu, L., Heng, P.: Bi-normal filtering for mesh denoising. Vis. Comput. Graph., IEEE Trans. **21**(1), 43–55 (2015)
41. Fan, H., Yu, Y., Peng, Q.: Robust feature-preserving mesh denoising based on consistent subneighborhoods. IEEE Trans. Vis. Comput. Graph. **16**(2), 312–324 (2010)
42. Bian, Z., Tong, R.: Feature-preserving mesh denoising based on vertices classification. Comput. Aided Geometr. Des. **28**(1), 50–64 (2011)
43. Wang, J., Zhang, X., Yu, Z.: A cascaded approach for feature-preserving surface mesh denoising. Comput.-Aided Des. **44**(7), 597–610 (2012)
44. Zhu, L., Wei, M., Yu, J., Wang, W., Qin, J., Heng, P.A.: Coarse-to-fine normal filtering for feature-preserving mesh denoising based on isotropic subneighborhoods. Comput. Graph. Forum **32**(7), 371–380 (2013)
45. Wang, J., Yu, Z.: A novel method for surface mesh smoothing: applications in biomedical modeling. In: Proceedings of the 18th International Meshing Roundtable, IMR 2009, October 25–28, 2009, Salt Lake City, UT, USA, pp. 195–210 (2009)
46. Wei, M., Liang, L., Pang, W.M., Wang, J., Li, W., Wu, H.: Tensor voting guided mesh denoising. IEEE Trans. Autom. Sci. Eng. **14**(2), 931–945 (2017)
47. Lu, X., Deng, Z., Chen, W.: A robust scheme for feature-preserving mesh denoising. IEEE Trans. Vis. Comput. Graph. **22**(3), 1181–1194 (2016)
48. Lu, X., Chen, W., Schaefer, S.: Robust mesh denoising via vertex pre-filtering and 1-median normal filtering. Comput. Aided Geometr. Des. **54**, 49–60 (2017)
49. He, L., Schaefer, S.: Mesh denoising via l0 minimization. ACM Trans. Graph **32**(4), 64:1-64:8 (2013)
50. Zhao, Y., Qin, H., Zeng, X., Xu, J., Dong, J.: Robust and effective mesh denoising using L0 sparse regularization. Comput.-Aided Des. **101**, 82–97 (2018)
51. Pan, W., Lu, X., Gong, Y., Tang, W., Liu, J., He, Y., Qiu, G.: Hlo: half-kernel Laplacian operator for surface smoothing. Comput.-Aided Des. **121**, 102807 (2020)
52. Vieira, M., Shimada, K.: Surface mesh segmentation and smooth surface extraction through region growing. Comput. Aided Geometr. Des. **22**(8), 771–792 (2005)
53. Huang, H., Ascher, U.: Surface mesh smoothing, regularization, and feature detection. SIAM J. Sci. Comput. **31**(1), 74–93 (2008). https://doi.org/10.1137/060676684
54. Legrand, H., Thiery, J.M., Boubekeur, T.: Filtered quadrics for high-speed geometry smoothing and clustering. Comput. Graph. Forum **38**(1), 663–677 (2019)
55. Wang, C., Liu, Z., Liu, L.: Feature-preserving Mumford-Shah mesh processing via nonsmooth nonconvex regularization. Comput. Graph. **106**, 222–236 (2022). https://doi.org/10.1016/j.cag.2022.06.006
56. Liu, Z., Li, Y., Wang, W., Liu, L., Chen, R.: Mesh total generalized variation for denoising. IEEE Trans. Vis. Comput. Graph. **28**(12), 4418–4433 (2021). https://doi.org/10.1109/TVCG.2021.3088118
57. Cohen-Steiner, D., Alliez, P., Desbrun, M.: Variational shape approximation. ACM Trans. Graph. **23**(3), 905–914 (2004)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Weijia Wang is currently pursuing his Ph.D. at Deakin University, Australia. Prior to that, he was awarded Master of Information Technology (with Distinction) and Bachelor of Science at the University of Melbourne, Australia. His research interests include 3D geometry processing, deep learning and multimedia.

Wei Pan is currently working in OPT Machine Vision Corp. as a senior Research and Development Engineer. Prior to that, he worked as a research fellow at VISTA lab, Shenzhen University after he got his Ph.D. degree from Singapore University of Technology and Design. His research interests include computational geometry, 3D algorithms, machine learning and computer-aided design.

Chaofan Dai has earned a master's degree from Wuhan University of Science and Technology. His academic focus spanned specialized research methodologies involving 2D data processing, high-performance acceleration, and 3D data processing. During his tenure as an intern at OPT Machine Vision Group, he adeptly applied these methodologies to push forward the boundaries of research within these domains.

Richard Dazeley is the Deputy Head of School for the School of Information Technology (SIT) at Deakin University. His research interests include reinforcement learning and the human-alignment problem—focusing on multi-objective, explainable, safe, ethical, interactive and multi-agent reinforcement learning methods. His research interests also include knowledge representation, acquisition and prudence analysis and their incorporation into machine learning methods.

Lei Wei received his Ph.D. degree from School of Computer Engineering, Nanyang Technological University (NTU). His research interests include haptic rendering, interaction and collaboration, collision detection, human-computer interaction, function-based and physics-based modelling, as well as web visualisation.

Bernard Rolfe is the Associate Dean, Research for the Faculty of Science, Engineering and Built Environment (SEBE) at Deakin University. His research interests include Additive Manufacturing, Sheet Metal Forming, People Centred Manufacturing, Material Modelling, Lightweight Design, and Multi-material optimisation.

Xuequan Lu is a Senior Lecturer at La Trobe University in Australia and is currently an IEEE Senior Member. He worked as a research fellow in Singapore for two years. Prior to that, he received his Ph.D. in the College of Computer Science and Technology at Zhejiang University. His research interests mainly fall into the category of visual data computing, including geometry modelling, processing and analysis, animation/simulation, 2D data processing and analysis.