

# Constraint-based Point Set Denoising using Normal Voting Tensor and Restricted Quadratic Error Metrics

Sunil Kumar Yadav, Ulrich Reitebuch, Martin Skrodzki,  
Eric Skrodzki, and Konrad Polthier  
Freie Universität Berlin, Arnimallee 6, 14195 Berlin, Germany

August 7, 2018

## Abstract

In many applications, point set surfaces are acquired by 3D scanners. During this acquisition process, noise and outliers are inevitable. For a high fidelity surface reconstruction from a noisy point set, a feature preserving point set denoising operation has to be performed to remove noise and outliers from the input point set. To suppress these undesired components while preserving features, we introduce an anisotropic point set denoising algorithm in the normal voting tensor framework. The proposed method consists of three different stages that are iteratively applied to the input: in the first stage, noisy vertex normals, are initially computed using principal component analysis, are processed using a vertex-based normal voting tensor and binary eigenvalues optimization. In the second stage, feature points are categorized into corners, edges, and surface patches using a weighted covariance matrix, which is computed based on the processed vertex normals. In the last stage, vertex positions are updated according to the processed vertex normals using restricted quadratic error metrics. For the vertex updates, we add different constraints to the quadratic error metric based on feature (edges and corners) and non-feature (planar) vertices. Finally, we show our method to be robust and comparable to state-of-the-art methods in several experiments.

Keywords: Point Set Denoising, Normal Voting Tensor, Binary Eigenvalues Optimization, Quadratic Error Metric

## 1 Introduction

Point sets arise naturally in almost all kinds of three-dimensional acquisition processes, like 3D scanning. As early as 1985, they have been recognized as fundamental shape representations in computer graphics, [Levoy and Whitted(1985)]. Thus, they have manifold applications e.g. in face recognition ([Boehnen and Flynn(2005)]), traffic accident

1

2

3

4

6

1 analysis ([Buck et al.(2007)Buck, Naether, Braun, Bolliger, Friederich, Jackowski et al.]),  
2 or archeology ([Levoy et al.(2000)Levoy, Pulli, Curless, Rusinkiewicz, Koller, Pereira et al.]).

3 However, during the acquisition process, due to mechanical limitations and  
4 surrounding conditions, noise and outliers are inevitably added to the point set.  
5 These artifacts have to be removed in a post-processing step to obtain a cleaned  
6 point set, which can be used in further steps like surface reconstruction, com-  
7 puter aided design (CAD), or 3D printing. There exists a variety of denoising  
8 methods focused on removing outliers and noise from the input point set to  
9 create a high fidelity output. These methods do not only aim at removing the  
10 undesired components, but also try to preserve sharp features of the geometry.  
11 High frequency components like corners or edges should be preserved and not  
12 be smoothed out. This is a challenging task as both features and noise are high  
13 frequency components and thus ambiguous in their nature.

14 Most state-of-the-art denoising methods are designed to work on triangle  
15 meshes. Compared to this setup, working on point sets and preserving sharp  
16 features is more difficult as explicit connectivity information is not present.  
17 Also, we assume the input to be given without any normals. However, as point  
18 sets take up less storage space and as the surface reconstruction is easier on a  
19 noise-free point set, we aim for an intrinsic smoothing method to work directly  
20 on the noisy point set input.

21 Our method is focused on the preservation of sharp features while removing  
22 noise and outliers from an input point set. The proposed algorithm follows an it-  
23 erative three-step point set denoising scheme. (1) Noisy vertex normals process-  
24 ing using a vertex-based *normal voting tensor* (NVT) and *binary eigenvalues opti-  
25 mization* (BEO) similar to [Yadav et al.(2017)Yadav, Reitebuch and Polthier].  
26 (2) Feature points detection using an anisotropic covariance matrix. For the  
27 update of vertex positions, we use (3) a variation of the quadratic error norm  
28 adjusted to different kinds of feature points. Steps (1) to (3) are iteratively  
29 applied until a satisfactory output has been generated.

## 30 1.1 Related Work

### 31 1.1.1 Point-based Methods

32 In general, point sets appear as natural output of 3D scan devices. The increase  
33 in computational costs while processing polygonal meshes with growing size is  
34 partly responsible that points got recognized as primitives for surface represen-  
35 tation, cf. [Amenta and Kil(2004)]. One major drawback in this approach is  
36 the absence of connectivity information, which sets the task to declare surface  
37 normals. Here, [Amenta and Kil(2004)] especially proposes a definition utilizing  
38 surfels, which are points equipped with normals. Usually point clouds do not  
39 carry normals, so we have to rely on methods which determine these robustly and  
40 with high quality. The authors Mitra and Nguyen [Mitra and Nguyen(2003)]  
41 suggest a calculation of point set normals and an analysis under consideration  
42 of density, neighborhood sizes, and the presence of noise.

43 We are interested in point set denoising coupled with feature preservation.

There are several works approaching these two properties directly. A first one was published by Fleishman et al. [Fleishman et al.(2003)Fleishman, Drori and Cohen-Or] – serving as a representative despite the fact that it deals with meshes instead of point clouds. As it does not use the mesh information, so it can be transferred to the point set setting. They use a bilateral filtering of points in normal direction in local neighborhoods. Another one is the anisotropic smoothing of point sets ([Lange and Polthier(2005)]), where the authors use an anisotropic geometric curvature flow. Besides the high dependency on suitable neighborhoods, which the authors cannot compute directly, the proposed algorithm does not detect features explicitly, but incorporates feature detection into an anisotropic Laplacian. The more recent work [Sun et al.(2015)Sun, Schaefer and Wang] is based on the idea of sparsity methods and includes  $L_0$  minimization. Originating from image denoising, they set up an energy consisting of the 3D signal to be optimized coupled with an  $L_0$  optimization applied to a differential operator on the signal.

Processing of normals, point positions, and an edge-aware upsampling offers the opportunity for an iterative application. In this setting, we are going to compare our algorithm with that of [E. and A.(2016)], called “moving robust principal component analysis” (MRPCA). The idea is – like the previous – based on sparsity methods, which takes sparsity-algorithms and adapt them to geometry processing problems. They perceive the point cloud as a collection of overlapping two-dimensional subspaces and do not rely – in contrast to other procedures – on oriented normals as input. The method is robust against outliers and capable of denoising the point cloud while handling sharp features.

Recently, Zheng et al. ([Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao]) proposed an extension of edge-aware image processing and mesh denoising to point clouds. In their four-staged approach, feature candidates are detected, employing a feature structure by the  $l_1$ -medial skeleton, calculating and equipping these with multiple normals, and selecting guiding normals by using kNN patches with its normals being most consistent. In this terms, the algorithm is even capable of high intensive noise while preserving important geometric features.

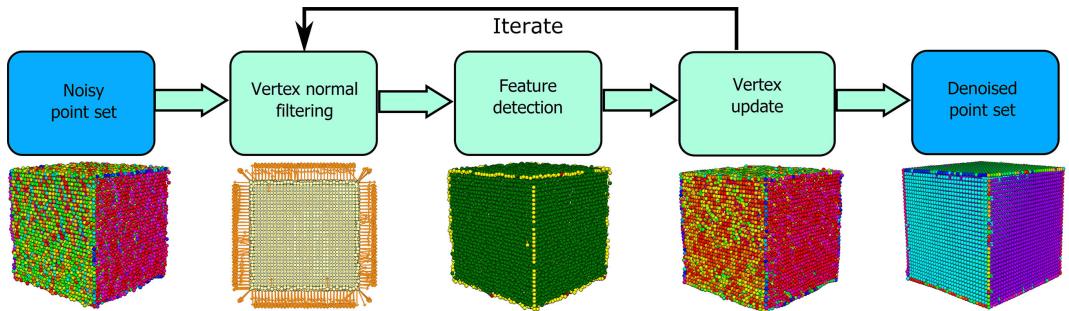


Figure 1: The pipeline of the proposed algorithm. Our method consists of three different stages, which are iterated until a desired output is produced.

**1    1.1.2   Surface Reconstruction with Feature Preservation**

**2** One of the processes most affected by noise and outliers in a point set is  
**3** that of surface reconstruction. A thorough introduction is given in the survey  
**4** [Berger et al.(2017)Berger, Tagliasacchi, Seversky, Alliez, Guennebaud, Levine et al.].  
**5** All following techniques aim at preserving features while simultaneously per-  
**6** form denoising in the surface reconstruction process. In the context of local  
**7** smoothness priors, the moving least squares (MLS) approach has a major im-  
**8** pact. Developed in large parts by Levin [Levin(2003)], MLS underwent a lot of  
**9** modifications. Guennebaud et al. [Guennebaud and Gross(2007)] modified the  
**10** MLS idea by replacing the concept of finding well-defined tangent planes by fit-  
**11** ting spheres as higher order approximations to the surface. This change makes  
**12** the method more robust – especially in sparsely sampled regions, where a well  
**13** defined tangent plane might not exist. Their method is denoted as “algebraic  
**14** point set surfaces” (APSS) and will serve as comparison to our algorithm. The  
**15** method of Öztireli et al. [Öztireli et al.(2009)Öztireli, Guennebaud and Gross]  
**16** aims at overcoming the sensitivity of MLS to outliers and the effect of smoothing  
**17** out small or sharp features. They combine MLS with local kernel regression to  
**18** create a new implicit description of the surface, making it robust to noise, out-  
**19** liers, and even sparse sampling. Their method of “robust implicit moving least  
**20** squares” (RIMLS) will be the third algorithm we compare to. More recently,  
**21** Chen et al. [Chen et al.(2013)Chen, Guennebaud, Barla and Granier] set their  
**22** focus on a new MLS formalism using higher-order approximations – like APSS  
**23** – incorporating discrete non-oriented gradient fields, yielding a continuous im-  
**24** plicit representation.

**25** Turning to hierarchical partitioning, Ohtake et al. [Ohtake et al.(2003)Ohtake, Belyaev, Alexa, Turk and Se-  
**26** propose “multi-level partitioning of unity implicits” (MPU). Their technique  
**27** consists of an octree-based top-down structure, where points in a cell and nearby  
**28** are approximated by either a bivariate quadratic polynomial or an algebraic  
**29** trivariate quadric. An adjustment parameter for the level of smoothness guar-  
**30**antees the handling of noise with respect to an error residual tolerance.

**31** Considering piecewise smooth priors and partition based methods, Fleish-  
**32** man et al. [Fleishman et al.(2005)Fleishman, Cohen-Or and Silva] concentrate  
**33** with their robust moving least squares (RMLS) on the handling of sharp fea-  
**34** tures. They use the robust statistics tool of forward-search paradigm to choose  
**35** small sets of points excluding outliers, continuing through the cloud, and eval-  
**36** uating observations monitored by statistical estimates. Wang et al. [Wang et al.(2013)Wang, Yu, Zhu and Cao]  
**37** robustly compute a feature preserving normal field by mean-shift clustering and  
**38** a least median of squares (LMS) regression scheme, providing local partitions,  
**39** to which edge-preserving smoothing is applied by fitting multiple quadrics. Due  
**40** to the locality, feature fragmentation at sharp edges may occur.

**41** Taking sparsity and neighboring normals into account, Avron et al. [Avron et al.(2010)Avron, Sharf, Greif a]  
**42** use global  $L_1$  optimization on these normals, observing that differences between  
**43** them should be sparse, yet large values should reflect sharp features. Similar to

the approach in RIMLS, [Huang et al.(2013)Huang, Wu, Gong, Cohen-Or, Ascher and Zhang] suggests the edge-aware resampling (EAR) of the point cloud. This is a feature-sensitive method under the guidance of the locally optimal projection (LOP) ([Lipman et al.(2007)Lipman, Cohen-Or, Levin and Tal-Ezer]) in a two-staged approach, starting their robust smoothing and resampling process in regions with similar normal distribution, while approaching the edges in terms of both smoothing and resampling in a second step.

## 1.2 Contribution

On a noisy point set, it is a challenging task to decouple noise components and sharp features, which is essential for a noise-free point set reconstruction. As shown in Figure 1, our algorithm consists of three different stages, which are iteratively applied until a satisfactory output has been computed. In the first stage, which is vertex normal filtering, we extend the concept of face normal processing of Yadav et al. [Yadav et al.(2017)Yadav, Reitebuch and Polthier] to the more general setup of vertex normal processing. Although our vertex normal processing is similar to the face normal processing of Yadav et al. [Yadav et al.(2017)Yadav, Reitebuch and Polthier], we define a vertex-based *Normal Voting Tensor* (NVT) based on the variation of vertex normals. In terms of noise sensitivity, vertex normals are more sensitive compared to face normals. Therefore, we modify the weighting scheme in the neighborhood selection to make the algorithm robust against different levels of noise. Noise and sharp features are decoupled using the spectral analysis of the vertex-based NVT and noise components are suppressed using *Binary Eigenvalues Optimization* (BEO). In the second stage, we introduce an anisotropic covariance matrix using the filtered vertex normals to detect feature points (edges and corners) robustly on the noisy input point set. In the last stage, we update the vertex positions based on quadratic error metrics. A corresponding quadratic error metric is used based on different feature points. The proposed vertex update method helps the algorithm to preserve sharp features with minimum shrinkage during the denoising process.

## 2 Method

Let us consider a nonuniform and noisy input point set  $\mathbf{V} = \{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_{n-1}\} \subset \mathbb{R}_{32}^3$  sampling a surface with  $n \in \mathbb{N}$  denoting the number of vertices. We assume these data points to be acquired e.g. by a 3D laser scanner and not to be equipped with vertex normals. Thus, a first normal field on the vertices is computed following [Hoppe et al.(1992)Hoppe, DeRose, Duchamp, McDonald and Stuetzle], which results in consistently oriented normals. Despite the fact, that there are more recent works dealing with consistent normal fields on point sets, we decided to use [Hoppe et al.(1992)Hoppe, DeRose, Duchamp, McDonald and Stuetzle], as the implementation is simple and it works well with all the models we used for our experiments. This is mostly due to the fact, that we process

1 and smooth the normals further, so a consistent initial normal field is suffi-  
 2 cient for our purposes. We denote the normal at vertex  $\mathbf{v}_i$  by  $\mathbf{n}_i$  and the  
 3 normal field by  $\mathbf{N} = \{\mathbf{n}_0, \mathbf{n}_1, \dots, \mathbf{n}_{n-1}\} \subset \mathbb{R}^3$ . For a given vertex  $\mathbf{v}_i$ , we de-  
 4 note by  $\Omega_i := \{j \in \mathbb{N} \mid \mathbf{v}_j \in \mathbf{V} \cap B_r(\mathbf{v}_i), i \neq j, j < n\}$ , a set of indices of the  
 5 geometric neighborhood of vertex  $\mathbf{v}_i$ , i.e. indices of all points from  $\mathbf{V}$  that  
 6 have distance less or equal  $r$  to  $\mathbf{v}_i$ , where  $r$  is a global parameter. We fa-  
 7 vor a geometric neighborhood over a combinatorial  $k$ -nearest neighborhood as  
 8 [Yadav et al.(2017)Yadav, Reitebuch and Polthier] found it to be more robust.  
 9 In the following, we will denote by  $\tilde{\cdot}$  those elements updated in one iteration,  
 10 which will then serve as input to the next one.

## 11 2.1 Vertex Normal Filtering

12 This is the first of three iteratively applied steps of the proposed method. Here,  
 13 noisy vertex normals are filtered and denoised using a vertex-based *Normal*  
 14 *Voting Tensor* (NVT) and *Binary Eigenvalues Optimization* (BEO) similar to  
 15 [Yadav et al.(2017)Yadav, Reitebuch and Polthier]. We describe both in the fol-  
 16 lowing.

### 17 2.1.1 Vertex-based Normal Voting Tensor (NVT)

18 Covariance matrices compute the variance of an entity in a well defined domain.  
 19 For example, consider a vertex  $\mathbf{v}_i$  and its nearest neighbors  $\mathbf{v}_j$ ,  $j \in \Omega_i$ . The  
 20 covariance matrix on the edges  $\mathbf{v}_j - \mathbf{v}_i$  of the nearest neighbor graph computes  
 21 the variance of the vertex  $\mathbf{v}_i$  in  $\mathbb{R}^3$ . Similarly, the covariance matrix of vertex  
 22 normals  $\mathbf{n}_j$ ,  $j \in \Omega_i$  in a well defined neighborhood computes the anisotropic  
 23 nature of a point set in that region. To analyze the anisotropic nature of a  
 24 shape – in this case a point set surface – we define an object related to the  
 25 covariance matrix, namely the vertex-based NVT, which is computed using the  
 26 neighboring vertex normals:

$$T_i = \frac{1}{\sum_{j \in \Omega_i} w_{ij}} \sum_{j \in \Omega_i} w_{ij} \mathbf{n}_j \otimes \mathbf{n}_j, \quad (1)$$

27 where  $\Omega_i$  is the geometric neighborhood centered at  $\mathbf{v}_i$  and the symbol  $\otimes$  rep-  
 28 presents the outer product  $\mathbf{n}_j \mathbf{n}_j^T$ . The weights  $w_{ij}$  are computed based on the  
 29 similarity between the neighbor normals. To define  $w_{ij}$ , we follow the local  
 30 binary neighborhood concept, where the central vertex normal  $\mathbf{n}_i$  is compared  
 31 to neighborhood vertex normals  $\mathbf{n}_j$  and assigns a binary value  $w_{ij} \in \{0, 1\}$   
 32 according to the normal difference. Formally, we define the weight term  $w_{ij}$  as

$$w_{ij} = \begin{cases} 1 & \text{if } \angle(\mathbf{n}_i, \mathbf{n}_j) \leq \rho \\ 0 & \text{if } \angle(\mathbf{n}_i, \mathbf{n}_j) > \rho, \end{cases} \quad (2)$$

33 where  $\rho \in \mathbb{R}_{>0}$  is given by the user and denotes a local binary neighbor-  
 34 hood threshold, which is used to select vertices  $\mathbf{v}_j$ ,  $j \in \Omega_i$  with similar nor-  
 35 mals to  $\mathbf{n}_i$ . The weight term  $w_{ij}$  is not the exact weight function used in

[Yadav et al.(2017)Yadav, Reitebuch and Polthier] because vertex normals are more sensitive to noise than face normals, for example at sharp features. Therefore, a harder cut-off is necessary to maintain geometrical features while smoothing the point set. Figure 2 shows a comparison between the proposed weighting scheme (2), the bilateral weighting from [Yadav et al.(2018)Yadav, Reitebuch and Polthier], and the weighting function used in Yadav et al. [Yadav et al.(2017)Yadav, Reitebuch and Polthier]. As it can be seen, the harder cut-off function is more effective in terms of feature preservation than Yadav et al. [Yadav et al.(2017)Yadav, Reitebuch and Polthier] or [Yadav et al.(2018)Yadav, Reitebuch and Polthier].

The term  $\mathbf{T}_i$  is a tensor. By construction, it is symmetric and positive semidefinite and can be represented in terms of its spectral components:

$$\mathbf{T}_i = \sum_{\ell=1}^3 \lambda_{i,\ell} \mathbf{x}_{i,\ell} \otimes \mathbf{x}_{i,\ell}, \quad (3)$$

where  $\lambda_{i,\ell}$  and  $\mathbf{x}_{i,\ell}$  are the corresponding eigenvalues and eigenvectors. Let us consider the eigenvalues to be sorted in decreasing order  $\{\lambda_{i,1} \geq \lambda_{i,2} \geq \lambda_{i,3} \geq 0\}$ . Thus, we can rewrite  $\mathbf{T}_i$  as:

$$\begin{aligned} \mathbf{T}_i = & (\lambda_{i,1} - \lambda_{i,2}) \mathbf{x}_{i,1} \otimes \mathbf{x}_{i,1} + (\lambda_{i,2} - \lambda_{i,3}) (\mathbf{x}_{i,1} \otimes \mathbf{x}_{i,1} + \mathbf{x}_{i,2} \otimes \mathbf{x}_{i,2}) \\ & + \lambda_{i,3} (\mathbf{x}_{i,1} \otimes \mathbf{x}_{i,1} + \mathbf{x}_{i,2} \otimes \mathbf{x}_{i,2} + \mathbf{x}_{i,3} \otimes \mathbf{x}_{i,3}). \end{aligned} \quad (4)$$

Here, the first term of the right hand side is known as the stick tensor and has only one dominant eigenvalue in the normal direction. The second term is spanned by the two dominant eigenvectors, such that the normal direction is defined in the direction of the least dominant eigenvector. This term is known as the plate tensor. The third term is spanned by all eigenvectors and does not have a well defined normal direction, cf. [Medioni(2000)]. From the above description, it is clear that the vertex-based NVT captures the anisotropic nature of a point set and feature points can be easily detected using the eigenvalues of  $\mathbf{T}_i$ . That is, if there is only one dominant eigenvalue then it is a planar point, if two eigenvalues are dominant then it is an edge, and if all eigenvalues are equally dominant then it is a corner.

### 2.1.2 Binary Eigenvalues Optimization (BEO)

In our method, the vertex-based NVT is applied as a denoising operator on a noisy point set. Furthermore, as we have discussed in the last section, the vertex-based NVT is capable of detecting features on point sets as shown in Figure 1 (third column). However, on a noisy point set, the behavior of the spectral components of the vertex-based NVT will change.

Let us assume that a point set is corrupted by random noise with standard deviation  $\sigma_n$ . Due to the presence of noise, the eigenvalues of the vertex-based NVT will change. For example, on a planar area, one eigenvalue will remain dominant, but the other two eigenvalues will be non-zero and proportional to  $\sigma$ . Similarly, on an edge of the sampled geometry, the least dominant eigenvalue

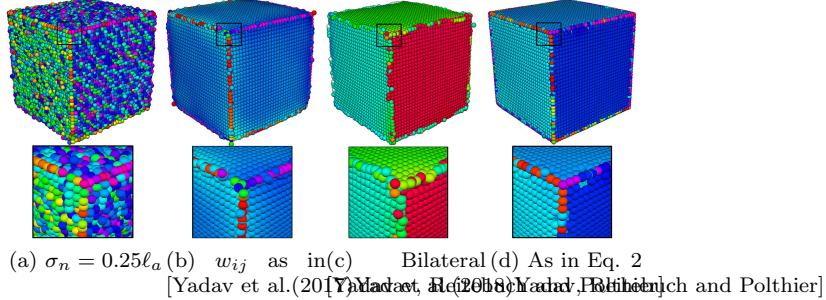


Figure 2: A comparison between different weighting functions for neighborhood selection. (a) The cube model is corrupted with a Gaussian noise ( $\sigma_n = 0.25\ell_a$ ) in random directions, where  $\ell_a$  is the average distance between vertices of the point set. (b) The output obtained by using the weighting function mentioned in Yadav et al. [Yadav et al.(2017)Yadav, Reitebuch and Polthier]. (c) The output obtained by using the bilateral weighting function (smooth functions) mentioned in Yadav et al. [Yadav et al.(2018)Yadav, Reitebuch and Polthier]. (d) The output obtained by using the proposed weighting function of Equation (2). Vertices of the cube are colored based on the variation of vertex normals. We computed the scalar product between the normal of a random vertex and rest of the vertex normals to show the level of denoising and feature preservation.

1 will be proportional to the applied noise, i.e.  $\lambda_{i,3} \propto \sigma$ . On a corner of the  
2 sampled geometry we expect  $\lambda_{i,1}, \lambda_{i,2}, \lambda_{i,3} \gg \sigma$ . To remove these noise effects  
3 from the vertex-based NVT, the eigenvalues of  $\mathbf{T}_i$  should be modified. That is,  
4 on a planar area and on an edge the least dominant eigenvalues should be zero  
5 and at a corner all eigenvalues should be equally dominant. In order to achieve  
6 this, we turn to binary optimization.

7 The concept of binary eigenvalues optimization is applied to the eigenvalues  
8 of  $\mathbf{T}_i$ , where each eigenvalue will be assigned a binary value  $\tilde{\lambda}_{i,\ell} \in \{0, 1\}$  to re-  
9 move noise components effectively. Similar to [Yadav et al.(2017)Yadav, Reitebuch and Polthier],  
10 a threshold value  $\tau \in \mathbb{R}_{>0}$  is used for the BEO. The term  $\tau$  is a global param-  
11 eter given by the user and should be chosen according to the noise intensity,  
12 i.e.  $\tau \propto \sigma$ , and to be smaller than the dominant eigenvalue(s). We will de-  
13 note by  $\tilde{\lambda}_{i,\ell}$  the modified eigenvalues of the vertex-based NVT after BEO. The  
14 modification is based on feature classification:

- At corners of the sampled geometry, when considering the point set, the smallest eigenvalue should still be bigger than the threshold value, i.e.  $\lambda_{i,3} \geq \tau$ . Hence, we set:

$$\tilde{\lambda}_{i,\ell} = 1, \quad \ell \in \{1, 2, 3\} \quad \text{if } \lambda_{i,3} \geq \tau.$$

- At edges of the sampled geometry, in the noisy point set, the least dom-  
inant eigenvalue should be smaller than the threshold value, i.e.  $\lambda_{i,3} < \tau$

and  $\lambda_{i,2} \geq \tau$ . Hence, we set:

$$\tilde{\lambda}_{i,1} = \tilde{\lambda}_{i,2} = 1, \quad \tilde{\lambda}_{i,3} = 0 \quad \text{if} \quad \lambda_{i,2} \geq \tau, \quad \lambda_{i,3} < \tau.$$

- In the last case, we check for planar areas of the geometry. Having  $\lambda_{i,2} < \tau$  and  $\lambda_{i,3} < \tau$  shows that the only dominant eigenvalue is  $\lambda_{i,1}$ . Hence, we set:

$$\tilde{\lambda}_{i,1} = 1, \quad \tilde{\lambda}_{i,2} = \tilde{\lambda}_{i,3} = 0 \quad \text{if} \quad \lambda_{i,1} \geq \tau, \quad \lambda_{i,3}, \lambda_{i,2} < \tau.$$

The BEO procedure presented here will remove the noise components from the eigenvalues of the vertex-based NVT.

### 2.1.3 Vertex Normal Denoising

To remove noise components from the vertex normals, we project the noisy vertex normals towards smooth normals by multiplication of the vertex-based NVT to the corresponding vertex normal. By the preceding BEO, this multiplication procedure will suppress noise in weak eigendirections and will strengthen vertex normals in strong eigendirections.

Before multiplication, we have to recompute the modified vertex-based NVT by using the same eigenvectors with the eigenvalues optimized in the BEO:

$$\tilde{\mathbf{T}}_i = \sum_{\ell=1}^3 \tilde{\lambda}_{i,\ell} \mathbf{x}_{i,\ell} \otimes \mathbf{x}_{i,\ell}. \quad (5)$$

To remove noise, we multiply the corresponding vertex normal with the modified tensor  $\tilde{\mathbf{T}}_i$ . The multiplication will lead to noise removal while retaining sharp features:

$$\tilde{\mathbf{n}}_i = d\mathbf{n}_i + \tilde{\mathbf{T}}_i \mathbf{n}_i = d\mathbf{n}_i + \sum_{\ell=1}^3 \tilde{\lambda}_{i,\ell} \langle \mathbf{x}_{i,\ell}, \mathbf{n}_i \rangle \mathbf{x}_{i,\ell}, \quad (6)$$

where  $d \in \mathbb{R}_{>0}$  denotes a damping factor to control the denoising speed of the vertex normals. We use  $d = 3$  for all experiments. Finally, the updated normal  $\tilde{\mathbf{n}}_i$  is normalized.

## 2.2 Feature Detection

This is the second of three iteratively applied steps of the proposed method. Here, we classify the point set into three different categories: corners, edges, and planar points. This will be done using the spectral analysis of a weighted covariance matrix. The idea to identify points and their features is substantiated in the follow-up treatment of point position updates in the upcoming third subsection where we use the notion of a quadratic error metric applied differently to the occurring feature-assigned points. The weighted covariance matrix is defined using filtered vertex normals, which makes the proposed algorithm more

<sup>1</sup> robust against feature points misclassification, which can lead to feature blurring  
<sup>2</sup> artifacts.

<sup>3</sup> To detect feature points on a point set with filtered vertex normals, we  
<sup>4</sup> consider the weighted covariance matrix:

$$\mathbf{C}_i = \frac{1}{\sum_{j \in \Omega_i} \tilde{w}_{ij}} \sum_{j \in \Omega_i} \tilde{w}_{ij} (\mathbf{v}_j - \bar{\mathbf{v}}) \otimes (\mathbf{v}_j - \bar{\mathbf{v}}), \quad (7)$$

<sup>5</sup> where the weights  $\tilde{w}_{ij}$  are similar to Equation (2), but are now utilizing the  
<sup>6</sup> filtered vertex normals of Section 2.1. Formally, the terms  $\bar{\mathbf{v}}$  and  $\tilde{w}_{ij}$  are defined  
<sup>7</sup> as:

$$\bar{\mathbf{v}} = \frac{1}{\sum_{j \in \Omega_i} \tilde{w}_{ij}} \sum_{j \in \Omega_i} \tilde{w}_{ij} \mathbf{v}_j, \quad \tilde{w}_{ij} = \begin{cases} 1 & \text{if } \angle(\tilde{\mathbf{n}}_i, \tilde{\mathbf{n}}_j) \leq \rho \\ 0 & \text{if } \angle(\tilde{\mathbf{n}}_i, \tilde{\mathbf{n}}_j) > \rho \end{cases}. \quad (8)$$

<sup>8</sup> Similar to the vertex-based NVT,  $\mathbf{C}_i$  is also a symmetric and positive semidef-  
<sup>9</sup>inite matrix and can be represented in terms of its spectral components:

$$\mathbf{C}_i = \sum_{\ell=1}^3 \mu_{i,\ell} \mathbf{y}_{i,\ell} \otimes \mathbf{y}_{i,\ell}, \quad (9)$$

<sup>10</sup> where  $\mu_\ell$  and  $\mathbf{y}_\ell$  are the corresponding eigenvalues and eigenvectors. Let us con-  
<sup>11</sup>sider the eigenvalues to be sorted in decreasing order  $\{\mu_{i,1} \geq \mu_{i,2} \geq \mu_{i,3} \geq 0\}$ .  
<sup>12</sup> In the proposed method, they are used to classify the points as follows, utilizing  
<sup>13</sup>the same threshold parameter  $\tau$  as in Section 2.1.2:

- On a planar area, there will be two dominant eigenvalues and their cor-  
 responding eigenvectors should be spanning the tangent plane. The least  
 dominant eigenvalue will be smaller than the feature threshold  $\tau$ . Therefore,  
 we classify planar points as

$$\mathbf{V}_f = \{\mathbf{v}_i \in \mathbf{V} \mid \mu_{i,1}, \mu_{i,2} \geq \tau, \mu_{i,3} < \tau\}.$$

- On an edge, there will be one dominant eigenvalue and the corresponding  
 eigenvector aligns with the edge direction. Therefore, we classify edge  
 points as

$$\mathbf{V}_e = \{\mathbf{v}_i \in \mathbf{V} \mid \mu_{i,1} \geq \tau, \mu_{i,2}, \mu_{i,3} < \tau\}.$$

- Finally, on a corner, either all eigenvalues are dominant or none of them  
 is significant. Therefore, corner points are set to

$$\mathbf{V}_c = \{\mathbf{v}_i \in \mathbf{V} \mid (\mu_{i,1}, \mu_{i,2}, \mu_{i,3} \geq \tau) \vee (\mu_{i,1}, \mu_{i,2}, \mu_{i,3} < \tau)\}.$$

<sup>14</sup> Points at a corner, an edge, and planar points are represented in the following  
<sup>15</sup> by  $\mathbf{V}_c = \{\mathbf{v}^c\}$ ,  $\mathbf{V}_e = \{\mathbf{v}^e\}$ , and  $\mathbf{V}_f = \{\mathbf{v}^f\}$  respectively, such that we obtain the  
<sup>16</sup> following disjoint union  $\mathbf{V} = \mathbf{V}_c \sqcup \mathbf{V}_e \sqcup \mathbf{V}_f$ .

## 2.3 Constraint-based Vertex Position Update

In this final of three iteratively applied steps of the proposed method, we update the vertex positions. This is done utilizing distance-based constraints, where the resulting updated point set remains within a prescribed distance to the input noisy point set. To compute the optimal position of a vertex w.r.t. the smoothed vertex normal, restricted quadratic error metrics are used in this algorithm, inspired by the work of [Garland and Heckbert(1997)]. The restriction to the quadratic error metric is introduced based on the different feature points and the vertex position is updated utilizing distance-based constraints.

We allow the user to provide a parameter  $\varepsilon \in \mathbb{R}_{>0}$  bounding the maximum deviation  $d_i$  between an initial noisy point and its corresponding iteratively updated point  $\tilde{\mathbf{v}}_i$ .

### 2.3.1 Vertex Update at Corners

Let us consider a point  $\mathbf{v}_i^c \in \mathbf{V}_c$ , labeled as corner point in Section 2.2. We will find its updated position  $\tilde{\mathbf{v}}_i^c$  by minimizing the following energy function:

$$\min_{\tilde{\mathbf{v}}_i^c} \sum_{j \in \Omega_i} \|\tilde{\mathbf{n}}_j \cdot (\tilde{\mathbf{v}}_i^c - \mathbf{v}_j)\|^2. \quad (10)$$

Each of the neighboring vertices  $\mathbf{v}_j$  is equipped with a corresponding filtered normal direction  $\tilde{\mathbf{n}}_j$ . Thus, we can associate a plane based at each neighboring vertex given by the normal direction. In an ideal case, these planes would all meet in a point – the exact position of the vertex  $\mathbf{v}_i$ . But as noise is present and the planes will in general not intersect in a point, we define the error of the vertex  $\tilde{\mathbf{v}}_i^c$  as the sum of squared distances to these planes.

Note that we do not weight the neighbors in Equation (10), but take all  $j \in \Omega_i$  into account equally. That is because we rely on the highly unstable intersection of multiple planes in  $\mathbb{R}^3$ . The more we take into account, the more likely it is for them to even out noise effects and give a faithful reconstruction of the corner.

Minimizing Equation (10) boils down to solving a linear system and the new position can be computed directly, which will be given by the following equation:

$$\mathbf{t}_i^c = \left( \sum_{j \in \Omega_i} \tilde{\mathbf{n}}_j \otimes \tilde{\mathbf{n}}_j \right)^{-1} \sum_{j \in \Omega_i} \left( \tilde{\mathbf{n}}_j \otimes \tilde{\mathbf{n}}_j \mathbf{v}_j \right),$$

where  $\mathbf{t}_i^c$  is a temporary vertex position. Before updating the position of  $\mathbf{v}_i^c$  to  $\tilde{\mathbf{v}}_i^c$ , we compute the deviation  $d_i$  between  $\mathbf{t}_i^c$  and the corresponding original vertex from the noisy point set. If  $d_i$  is within the user-prescribed limit  $\varepsilon$  we move this corner point to  $\mathbf{t}_i^c$ , otherwise we don't move this point:

$$\tilde{\mathbf{v}}_i^c = \begin{cases} \mathbf{t}_i^c & \text{if } d_i \leq \varepsilon \\ \mathbf{v}_i^c & \text{if } d_i > \varepsilon \end{cases}, \quad (11)$$

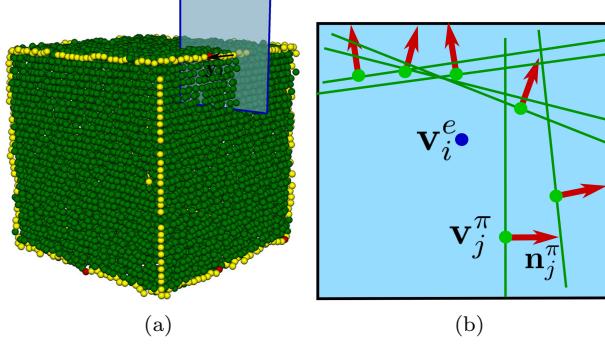


Figure 3: A visual representation of the vertex update scheme at edges. Figure (a) shows the plane  $\mathbf{H}_i$ , which is defined by  $\mathbf{y}_1$  for edge vertex  $\mathbf{v}_i^e$ . Neighbor vertices of  $\mathbf{v}_i^e$  and corresponding vertex normals are projected onto this plane as shown in Figure (b).

where  $\varepsilon$  is the aforementioned user input which limits the deviation between the original noisy and updated corner points. By the above equation, corner points are moved at most  $\varepsilon$  during their position update in the direction of a minimum distance to all neighboring planes spanned by the respective normals.

### 2.3.2 Vertex Update at Edges

Let us consider a point  $\mathbf{v}_i^e \in \mathbf{V}_e$ , labeled as edge point in Section 2.2. Here, the weighted covariance matrix  $\mathbf{C}_i$  has only one dominant eigenvalue  $\mu_{i,1}$  and the corresponding eigenvector  $\mathbf{y}_{i,1}$  aligns with the edge direction. We define a plane

$$\mathbf{H}_i = \{x \in \mathbb{R}^3 \mid \langle \mathbf{y}_{i,1}, x \rangle = \langle \mathbf{y}_{i,1}, \mathbf{v}_i^e \rangle\}.$$

As shown in Figure 3, we project all neighborhood vertices  $\mathbf{v}_j$ ,  $j \in \Omega_i$  and their respective vertex normals  $\tilde{\mathbf{n}}_j$  to  $\mathbf{H}_i$ , denoting the corresponding projections by  $\mathbf{v}_j^\pi$  and  $\tilde{\mathbf{n}}_j^\pi$ :

$$\begin{aligned}\mathbf{v}_j^\pi &= \mathbf{v}_j - \langle (\mathbf{v}_j - \mathbf{v}_i^e), \mathbf{y}_{i,1} \rangle \mathbf{y}_{i,1}, \\ \tilde{\mathbf{n}}_j^\pi &= \tilde{\mathbf{n}}_j - \langle \tilde{\mathbf{n}}_j, \mathbf{y}_{i,1} \rangle \mathbf{y}_{i,1}.\end{aligned}$$

Now, we define a quadratic energy function similar to Equation (10):

$$\min_{\tilde{\mathbf{v}}_i^e} \sum_{j \in \Omega_i} \left( \|\tilde{\mathbf{n}}_j^\pi \cdot (\tilde{\mathbf{v}}_i^e - \mathbf{v}_j^\pi)\|^2 + \frac{1}{|\Omega_i|} \|\mathbf{y}_{i,1} \cdot (\tilde{\mathbf{v}}_i^e - \mathbf{v}_j^\pi)\|^2 \right). \quad (12)$$

The above energy function is defined on the plane  $\mathbf{H}_i$ . In comparison to Equation (10), we include an additional summand. This is necessary, because the matrix of the linear system resulting from Equation (12) without the summand would not be invertible. The reason is, that the least dominant eigenvalue

along the normal being zero would reduce the rank of the corresponding matrix. We choose this additional summand which is directed along the edge to create an orthonormal basis as the plane  $\mathbf{H}_i$  is spanned using the other two eigenvectors of the weighted covariance matrix  $\mathbf{C}_i$ . Including the summand, we can minimize Equation (12) once more by solving a linear system which results in the equation:

$$\mathbf{t}_i^e = \left( \sum_{j \in \Omega_i} \tilde{\mathbf{n}}_j^\pi \otimes \tilde{\mathbf{n}}_j^\pi + \mathbf{y}_{i,1} \otimes \mathbf{y}_{i,1} \right)^{-1} \sum_{j \in \Omega_i} \left( \tilde{\mathbf{n}}_j^\pi \otimes \tilde{\mathbf{n}}_j^\pi \mathbf{v}_j + \mathbf{y}_{i,1} \otimes \mathbf{y}_{i,1} \mathbf{v}_i^e \right),$$

where the summand  $\mathbf{y}_1 \otimes \mathbf{y}_1$  ensures that the matrix is invertible. The term  $\mathbf{t}_i^e$  is a temporary vertex position and we once more compute the deviation between  $\mathbf{t}_i^e$  and the corresponding original vertex from the noisy point set to modify the edge vertex accordingly:

$$\tilde{\mathbf{v}}_i^e = \begin{cases} \mathbf{t}_i^e & \text{if } d_i \leq \varepsilon \\ \mathbf{v}_i^e & \text{if } d_i > \varepsilon \end{cases}. \quad (13)$$

For each edge vertex, the above equation computes the optimal position by minimizing the distance between the lines, which are defined by the projected vertex normals. This operation effectively preserves sharp features along edges and removes noise effectively.



Figure 4: This figure shows the effect of the proposed constraint-based vertex position update scheme: Figure (a) represents the point set reconstructed by using Equation (14) not only for flat regions but also for feature points. Figure (b) represents the result obtained by the proposed scheme, where flat regions follow Equation (14), edges are reconstructed using Equation (13) and corner positions are updated using Equation (11). Note how the corner itself is not recovered in (a), but is recovered in (b).

<sup>1</sup> **2.3.3 Vertex Update on Flat Regions**

<sup>2</sup> Let us consider a point  $\mathbf{v}_i^f \in \mathbf{V}_f$ , labeled as point within a planar area by  
<sup>3</sup> Section 2.2. In this flat region, the matrix  $\mathbf{C}_i$  has two dominant eigenvalues.  
<sup>4</sup> In order to remove noise in these regions, we allow to move the vertex  
<sup>5</sup> position only in direction of the corresponding vertex normal  $\tilde{\mathbf{n}}_i$ . Thereby,  
<sup>6</sup> we follow the approach of [Sun et al.(2007a)Sun, Rosin, Martin and Langbein,  
<sup>7</sup> Zheng et al.(2017b)Zheng, Li, Wu, Liu and Gao]. We use an energy function  
<sup>8</sup> similar to that of Equation (10), but with the restriction to only move in nor-  
<sup>9</sup> mal direction. Similar to edge and corner vertex updates, we first compute the  
<sup>10</sup> deviation  $d_i$  and then update the vertex position according to

$$\tilde{\mathbf{v}}_i^f = \begin{cases} \mathbf{v}_i^f + \frac{\alpha}{\sum_{j \in \Omega_i} W_{ij}} \sum_{j \in \Omega_i} W_{ij} \langle \tilde{\mathbf{n}}_j, \mathbf{v}_j - \mathbf{v}_i^f \rangle \tilde{\mathbf{n}}_i & \text{if } d_i \leq \varepsilon \\ \mathbf{v}_i^f & \text{if } d_i > \varepsilon \end{cases}, \quad (14)$$

<sup>11</sup> where  $\alpha$  is a user-controlled parameter to limit the amount of smoothing on flat  
<sup>12</sup> regions and  $W_{ij}$  is a combination of a similarity and a closeness function:

$$W_{ij} = \exp\left(-\frac{16|\tilde{\mathbf{n}}_i - \tilde{\mathbf{n}}_j|^2}{\delta^2}\right) \cdot \exp\left(-\frac{4|\mathbf{v}_j - \mathbf{v}_i^f|^2}{\delta^2}\right), \quad (15)$$

<sup>13</sup> where  $\delta$  is half the diameter of the point set  $\Omega_i$ .

<sup>14</sup> Even though the update scheme for flat regions as given in Equation (14)  
<sup>15</sup> seems most elaborate, the combination with simpler schemes for both cor-  
<sup>16</sup> ners (11) and edges (13) is more effective in practice, as shown in Figure 4.

<sup>17</sup> **2.4 Method Summary**

<sup>18</sup> In the previous Sections 2.1, 2.2, and 2.3, we have presented the three key steps  
<sup>19</sup> of our smoothing method. By iteratively applying these three steps to a noisy  
<sup>20</sup> point set input, the proposed algorithm produces a noise-free point set with  
<sup>21</sup> proper sharp features.

<sup>22</sup> **3 Experiments, Results and Discussion**

<sup>23</sup> We evaluated the capacity of our algorithm on various kinds of point set models  
<sup>24</sup> corrupted with synthetic noise (Figures 5, 6, 9, 10, 12) and real scanned data  
<sup>25</sup> (Figures 13, 14). We compared our method with five state-of-the-art denoising  
<sup>26</sup> Methods [E. and A.(2016)], [Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao],  
<sup>27</sup> [Guennebaud and Gross(2007)], [Oztireli et al.(2009)Oztireli, Guennebaud and Gross],  
<sup>28</sup> and [Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie]. Methods [Guennebaud and Gross(2007)]  
<sup>29</sup> and [Oztireli et al.(2009)Oztireli, Guennebaud and Gross] are implemented in  
<sup>30</sup> MeshLab. The results of Methods [E. and A.(2016)], [Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao],  
<sup>31</sup> and [Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie] are provided by the authors.

<sup>32</sup>

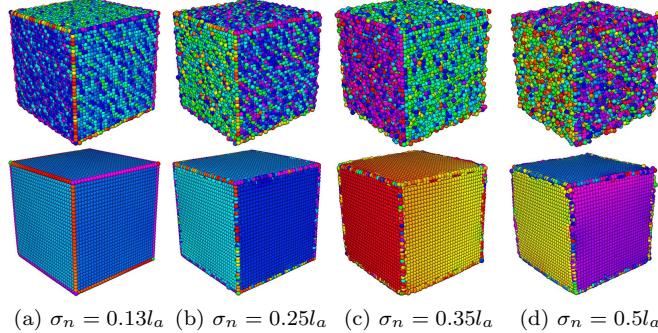


Figure 5: A visual representation of feature preservation analysis. The Cube is corrupted with different levels of noise ( $\sigma_n = 0.13l_a, 0.25l_a, 0.35l_a, 0.5l_a$ ) in random direction. To measure the feature preservation capability of the proposed algorithm, we computed MAD (mean angular deviation) and from Figure (a)-(d) MAD are 2.99, 4.15, 6.4 and 6.48. Vertices are colored based on the variation of vertex normals. We computed the scalar product between the normal of a random vertex and rest of the vertex normals to show the level of denoising.

### 3.1 Parameters Tuning

We introduced several parameters: geometric neighbor radius  $r$ , dihedral angle threshold  $\rho$  (Equation (2)), eigenvalue threshold  $\tau$  (Section 2.1.2), damping factor  $d$  (Equation (6)), distance-based constraint  $\varepsilon$  (Equations (11), (13), (14)), total number of iterations  $p$ , and vertex-diffusion speed  $\alpha$  (Equation (14)). Throughout the whole experimentation, we fixed  $\alpha = 0.1$  and  $d = 3$ . The radius  $r$  of the geometric neighborhood depends on the resolution of the input point set and it is fixed to be twice the average distance  $l_a$  between the vertices of the point set. The average distance between the vertices is computed using 6 nearest neighbors of each vertex. In this paper, we also fix the distance-based constraint  $\varepsilon = 2r$  for the experimentation purpose (except Figure 7). Effectively, there are only 3 parameters  $(\tau, \rho, p)$  to tune the results. In the quantitative comparison, see Table 1, the parameters are mentioned in the following format:  $(\tau, \rho, p)$ . For Methods [E. and A.(2016)], [Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao], and [Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie], we mention “Default” in the parameter column because the corresponding smooth models are provided by their authors. For the method [Guennebaud and Gross(2007)], we used the parameters  $(h, \#\{iterations\}, \alpha)$ , and for [Oztireli et al.(2009)Oztireli, Guennebaud and Gross], we used  $(\sigma_r, \sigma_n)$ , both listed in Table 1.

The eigenvalue threshold  $\tau$  depends on the noise intensity on a point set. The bigger the noise intensity, the larger the value of  $\tau$  should be chosen. We use  $\tau \in \{0.25, \dots, 0.4\}$  for synthetic data and  $\tau \in \{0.05, \dots, 0.1\}$  for real data because in our experiment real data point sets have smaller noise intensity compared to synthetic data point sets. We iterate several times ( $p \in \{20, \dots, 100\}$ ) for best

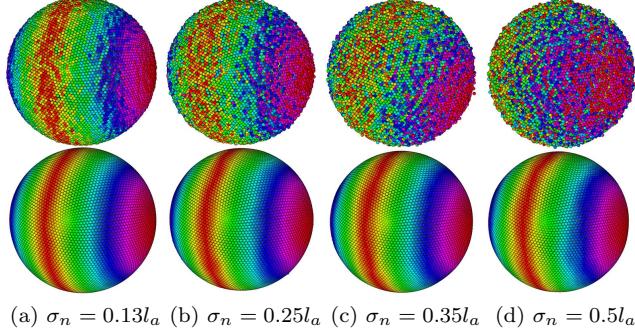


Figure 6: Shrinkage analysis during the denoising process. The Sphere model is corrupted with different levels of noise ( $\sigma_n = 0.13l_a, 0.25l_a, 0.35l_a, 0.5l_a$ ) in random direction. An average of the  $L_2$ -norm of each vertex is computed to show a shrinkage effect in the proposed method. From Figure (a)-(d) the  $L_2$ -norms are 0.984, 0.983, 0.982 and 0.9811. Vertices are colored based on the variation of vertex normals. We computed the scalar product between the normal of a random vertex and rest of the vertex normals to show the level of denoising.

1 results. The term  $\rho \in \{0.8, \dots, 0.95\}$  is the threshold to select the neighbor  
 2 components and it is computed using the scalar product between the neighbor  
 3 vertex normals. On a CAD model, we choose a high threshold value because of  
 4 sharp features and on CAGD models, we choose a small threshold value because  
 5 of smoother features compared to CAD models. The distance-based constraint  
 6  $\varepsilon$  is one of the most important parameter in the proposed algorithm. The effect  
 7 of this parameter is shown in Figure 7. As it can be seen, a small value of  $\varepsilon$   
 8 leads to less shrinkage (small  $E_v$ ) but does not remove all noise components.

### 9    3.2 Quantitative Analysis

10 We performed several experiments regarding the quantitative analysis of the  
 11 proposed algorithm. In general, shrinkage and feature blurring are two main  
 12 challenges during the denoising process. In this section, we show the behavior  
 13 of the proposed algorithm against different levels of noise in terms of shrinkage  
 14 and feature preservation.

15 To quantify shrinkage, we performed the denoising process with a unit Sphere  
 16 and computed the  $L_2$ -norm of each vertex using the following equation:

$$E_v = \frac{1}{n} \sum_{i=0}^{n-1} \|\mathbf{v}_i\|^2. \quad (16)$$

17 For the original Sphere  $E_v = 1.0$  and due to shrinkage effect the value of  $E_v$   
 18 decreases. As shown in Figure 6, the shrinkage effect increases with noise in-  
 19 tensity but at the same time these changes are not significant. The value of  $E_v$

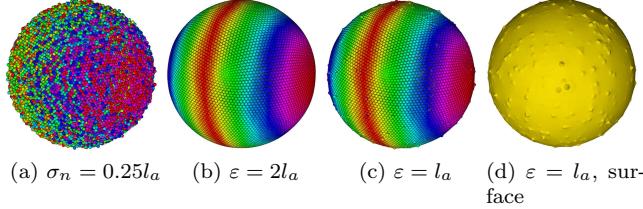


Figure 7: Effect of the distance-based constraint  $\varepsilon$ . Figure (a) shows a sphere, which is corrupted by Gaussian noise in random direction with standard deviation  $\sigma_n = 0.25l_a$  and  $l_a$  is the average distance between vertices of the point set. Figure (b) represents a desirable noise-free output with  $\varepsilon = 2l_a$  and when  $\varepsilon = l_a$ , the proposed method is not able to remove all noise components. The  $L_2$ -norms are 0.984 and 0.981 for Figure (b) and (c) respectively. Vertices are colored based on the variation of vertex normals. We computed the scalar product between the normal of a random vertex and rest of the vertex normals to show the level of denoising.

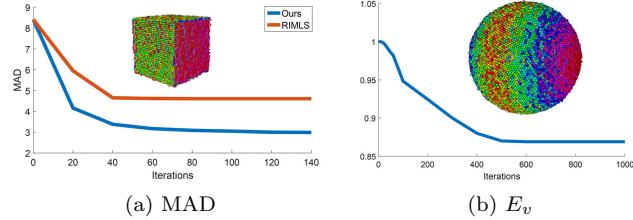


Figure 8: A visual representation of the convergence of the proposed algorithm. Figure (a) shows the orientation error convergence on a Cube model ( $\sigma_n = 0.13l_a$ ) and Figure (b) demonstrates the  $L_2$ -norm variation with iterations on a noisy Sphere ( $\sigma_n = 0.25l_a$ ).

also depends on the distance-based constraint  $\varepsilon$ . As shown in Figure 7, with a bigger value of  $\varepsilon$ , it is possible that  $E_v$  will be bigger but at the same time it gives a smoother result compared to a small value of  $\varepsilon$ .

For further shrinkage analysis, we reconstructed triangulated surfaces using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin] (Figure 9-12). To compute the closeness between the ground truth model and the denoised model, we use an  $L_2$  vertex-based error metric, which is defined as in [Sun et al.(2007b)Sun, Rosin, Martin and Langbein]:

$$D_v = \sqrt{\frac{1}{3 \sum_{k \in F} a_k} \sum_{i \in V} \sum_{j \in F_v(i)} a_j \text{dist}(\tilde{\mathbf{v}}_i, T)^2}$$

where  $F$  and  $V$  are treated as the triangular element set and the set of vertices

1  
2  
3

4

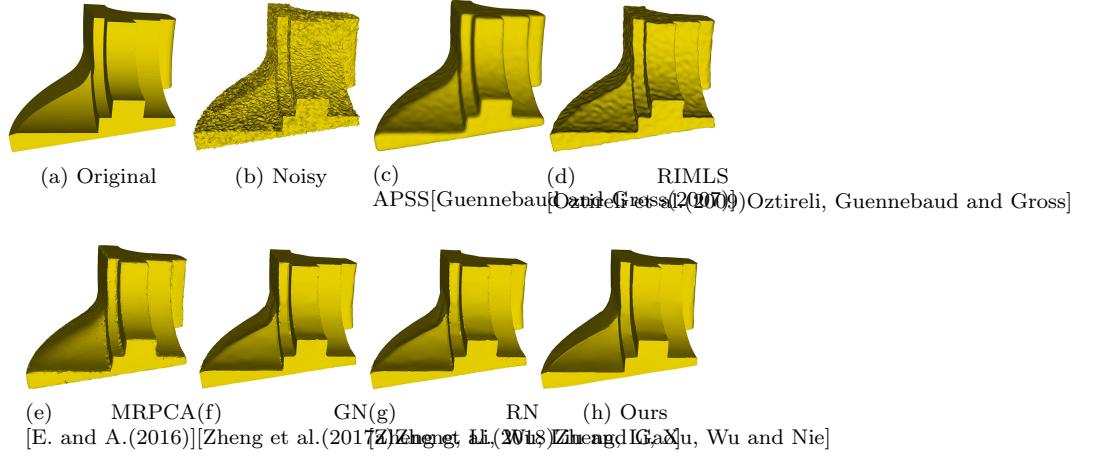


Figure 9: The Fan disk model corrupted by Gaussian noise ( $\sigma = 0.28l_e$ ), where  $l_e$  is the average distance between the vertices of the model. It can be seen that the proposed method is able to preserve sharp features effectively compared to state-of-the-art methods. Surfaces are reconstructed using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin].

1 respectively after the triangulated surface reconstruction. The terms  $a_k$  and  
2  $a_j$  are the corresponding face areas. The distance  $\text{dist}(\tilde{\mathbf{v}}_i, T)$  is the closest  
3  $L_2$ -distance between the newly computed vertex  $\tilde{\mathbf{v}}_i$  and the triangle  $T$  of the  
4 reference model.

5 Table 1: Quantitative Comparison  
6

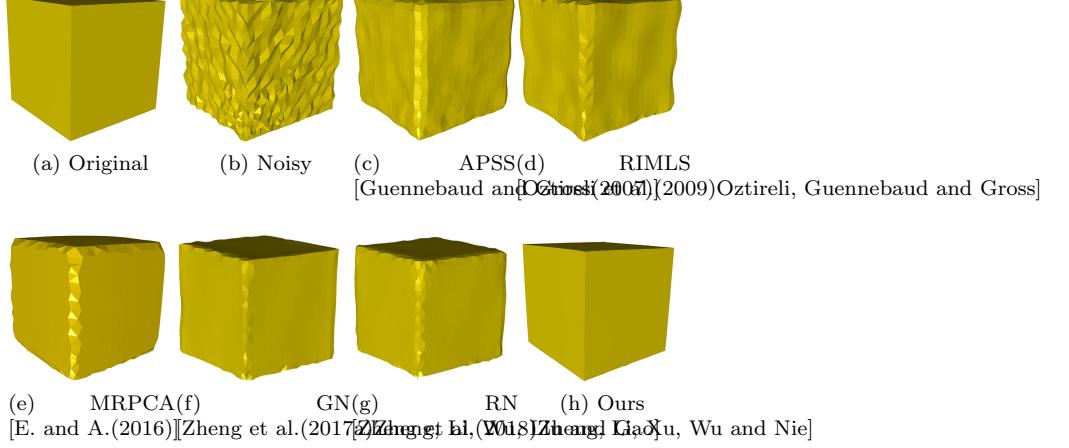


Figure 10: The Cube model with non-uniform distribution of vertices, corrupted by Gaussian noise ( $\sigma_n = 0.3l_e$ ), where  $l_e$  is the average distance between the vertices of the model. It can be seen that the proposed method is able to preserve sharp features effectively compared to state-of-the-art methods and does not create bumpy structures. Surfaces are reconstructed using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin].

Models	Methods	MAD	$D_v \times 10^{-3}$	Parameters
Cube $ V  = 1906$ Figure 10	[Guennebaud and Gross(2007)]	5.56	3.24	(2, 45, 0.5)
	[Oztireli et al.(2009)Oztireli, Guennebaud and Gross]	4.62	5.41	(4, 0.75)
	[E. and A.(2016)]	4.60	3.37	Default
	[Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao]	3.48	7.51	Default
	[Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie]	4.47	6.46	Default
	<b>Ours</b>	<b>2.85</b>	<b>1.65</b>	(0.3, 0.95, 150)
Rocker arm $ V  = 24106$ Figure 11	[Guennebaud and Gross(2007)]	<b>5.13</b>	22.6	(4, 15, 0.5)
	[Oztireli et al.(2009)Oztireli, Guennebaud and Gross]	5.26	<b>21.4</b>	(4, 1)
	[E. and A.(2016)]	6.31	33.0	Default
	[Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao]	8.14	118.7	Default
	[Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie]	6.26	72.12	Default
	<b>Ours</b>	7.56	43.26	(0.25, 0.9, 80)
Fan disk $ V  = 25894$ Figure 9	[Guennebaud and Gross(2007)]	<b>3.72</b>	1.7	(4, 15, 0)
	[Oztireli et al.(2009)Oztireli, Guennebaud and Gross]	6.6	1.81	(4, 0.75)
	[E. and A.(2016)]	13.67	1.56	Default
	[Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao]	4.57	1.81	Default
	[Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie]	4.34	1.4	Default
	<b>Ours</b>	4.4	<b>1.39</b>	(0.3, 0.9, 150)
Octahedron $ V  = 40242$ Figure 12	[Guennebaud and Gross(2007)]	3.35	0.27	(2, 45, 0.5)
	[Oztireli et al.(2009)Oztireli, Guennebaud and Gross]	4.31	0.39	(4, 0.75)
	[E. and A.(2016)]	4.6	0.32	Default
	[Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao]	1.2	0.52	Default
	[Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie]	1.37	0.49	Default
	<b>Ours</b>	<b>1.11</b>	<b>0.19</b>	(0.25, 0.9, 80)

1 To quantify feature preservation, we check the orientation error between the  
2 denoised model and the ground truth. Mean angular deviation (MAD) is defined  
3 to measure the orientation error:

$$MAD = \frac{1}{n} \sum_{i=0}^{n-1} \angle(\bar{\mathbf{n}}_i, \tilde{\mathbf{n}}_i), \quad (17)$$

4 where  $\bar{\mathbf{n}}_i$  and  $\tilde{\mathbf{n}}_i$  are vertex normals of the ground truth model and the denoised  
5 model respectively. Figure 5 shows that MAD is large when noise intensity is  
6 high. So with bigger noise, the orientation error will be bigger compared to  
7 lower noise. As it can be seen from Figure 5, for  $\sigma_n = 0.13l_a$  to  $\sigma_n = 0.35l_a$ , the  
8 output models are noise-free with sharp features. However, with  $\sigma_n = 0.5l_a$ , we  
9 are not able to preserve all sharp features.

10 Table 1 shows the comparison of the proposed method with five state-of-  
11 the-art methods. As it can be seen, for the Cube model, our method not only  
12 reconstructs sharp features (low MAD) but also produces minimum shrinkage  
13 (low  $D_v$ ) compared to the current state-of-the-art methods. For the Rocker arm  
14 model, the proposed method is not as good as APSS [Guennebaud and Gross(2007)]  
15 and RIMLS [Oztireli et al.(2009)Oztireli, Guennebaud and Gross] methods in  
16 terms of MAD and  $D_v$ . However, Figure 11 shows that our method produces  
17 smoother umbilical regions with enhanced sharp features. For the Fan disk  
18 model, the proposed algorithm performs better compared to state-of-the-art  
19 methods in terms of feature preservation. However, it produces more volume  
20 shrinkage compared to APSS [Guennebaud and Gross(2007)]. Similar to the  
21 Cube model, our algorithm outperforms state-of-the-art methods in terms of  
22 feature preservation and volume shrinkage.

23 Figure 8 shows the convergence property of the proposed algorithm. Fig-  
24 ure 8(a) shows the orientation error with a noisy Cube model and it is almost  
25 constant after 100 iterations. As it can be seen from the figure, the proposed  
26 method has better convergence rate compared to RIMLS [Oztireli et al.(2009)Oztireli, Guennebaud and Gross].  
27 Our method has improved convergence rate because of the BEO (binary eigen-  
28 values optimization), which assigns binary values to the eigenvalues of the  
29 vertex-based NVT. By assigning zero to the least dominant eigenvalue, our  
30 algorithm removes noise components faster compared to state-of-the-art meth-  
31 ods. Similarly, Figure 8(b) shows the variation of  $E_v$  w.r.t. iterations. As  
32 the number of iterations increases, the value of  $E_v$  decreases, which leads to  
33 shrinkage effect. In the proposed method, the shrinkage effect is controlled us-  
34 ing the distance-based constraint  $\varepsilon$ . As it can be seen from Figure 8, the value  
35 of  $E_v$  is almost constant after 400 iterations because the value of  $\varepsilon$  is set to  
36 approximately twice of the point set resolution.

### 37 3.3 Visual Comparison with State-of-the-art Methods

38 For visual comparison, we reconstruct triangulated meshes using the “ball pivot-  
39 ing” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin]  
40 after the point set denoising. Fan disk (Figure 9) and Cube (Figure 10) mod-

els have non-uniform vertices corrupted with Gaussian noise in random directions. Figure 10 shows that the proposed method produces a noise-free model with sharp features without creating any false and bumpy features like APSS [Guennebaud and Gross(2007)] and RIMLS [Oztireli et al.(2009)Oztireli, Guennebaud and Gross] methods. MRPCA [E. and A.(2016)], GN (guided normals) [Zheng et al.(2017a)Zheng, Li, Wu, Liu and Gao], and RN (rolling normals) [Zheng et al.(2018)Zheng, Li, Xu, Wu and Nie] remove noise effectively from flat regions but edges and corners are not reconstructed properly. Similarly, Figure 9 shows that our method reconstructs not only sharp features, but also shallow features (around flat regions). APSS and RIMLS preserve different levels of features also but do not remove noise components effectively. MRPCA removes noise effectively but does not reconstruct shallow features. Visually, GN and RN produce an output which is quite similar to the proposed algorithm. However, our method produces better quantitative measures compared to GN and RN. The Rocker arm model (Figure 11) has a considerably non-uniform mesh and our method better enhances sharp features (around cylindrical regions) and removes noise more effectively compared to state-of-the-art methods. In terms of quantitative analysis, APSS (lowest MAD) and RIMLS (lowest  $D_v$ ) are better compared to the proposed method. Figure 12 shows the capability of the proposed method to produce better corners and edges compared to MRPCA, APSS, and RIMLS and the results obtained by GN and RN methods look quite similar to the proposed method.

For real data, Figure 13 shows that our method removes noise effectively while retaining features on the surface compared to [Guennebaud and Gross(2007)] and [Oztireli et al.(2009)Oztireli, Guennebaud and Gross]. RN smooths out fine levels of features. GN and the proposed method produce quite similar results. Similarly, Figure 14 represents the applicability of the proposed method in medical data analysis. As it can be seen from the figure, our method removes noise effectively from the spherical regions and retains sharp features in the cylindrical regions. Guennebaud et al. [Guennebaud and Gross(2007)] and Öztireli et al. [Oztireli et al.(2009)Oztireli, Guennebaud and Gross] are not able to remove the noise components properly. MRPCA, GN, and RN remove noise components effectively. However, these methods blur the sharp features in the cylindrical region.

Figure 15 shows the robustness of the proposed method against irregular sampling of data points. The Gargoyle model is scanned by a laser scanner and has a highly irregular sampling. As it can be seen from the figure, our method produces a noise-free point set of the Gargoyle model without blurring different levels of features.

## 4 Conclusion

In this paper, we presented a simple and effective tensor multiplication algorithm for feature-preserving point set denoising. The proposed method is basically an extension of the ENVT-based mesh denoising [Yadav et al.(2017)Yadav, Reitebuch and Polthier]. Similar to the concept of the ENVT, in the proposed algorithm, we used vertex-

1 based NVT and the spectral analysis of this tensor leads to decoupling of fea-  
2 tures from noise. Noise components are removed by the multiplication of the  
3 vertex-based NVT to the corresponding vertex normal. The concept of binary  
4 eigenvalues optimization not only enhances sharp features but also improves the  
5 convergence rate of the method. Local binary neighborhood selection helps to  
6 select similar vertices in the neighborhood to compute the vertex-based NVT  
7 to avoid feature blurring during the denoising process. After the vertex normal  
8 filtering, we classify feature points into edges, corners and flat regions using  
9 an anisotropic covariance matrix. For the vertex update stage, we introduced  
10 restricted least square error metrics, which are different for different kinds of  
11 features. The vertex position reconstruction using restricted quadratic error  
12 metrics helps the algorithm to recreate the sharp edges and corners. The ex-  
13 perimental results show the effectiveness of the proposed algorithm.

14 Our method is capable of handling noise, but yields erroneous results under  
15 high noise intensities. This is based on the fact, that noise has a great impact on  
16 the normal estimation and the NVT construction, and we use them throughout  
17 the whole process. Another issue arises when the input point set is highly  
18 irregular. As we have shown in Figure 15, our method is robust up to a moderate  
19 level of irregularity but it is possible that with extreme irregular sampling, the  
20 output may not be satisfactory.

21 During the denoising process, we tuned the different parameters manually to  
22 get the desired results. we need to find an optimal combination of the parameters  
23 automatically. A direction we are going to investigate in the future.

## 24 Acknowledgments

25 This research was supported by the DFG Collaborative Research Center TRR  
26 109, “Discretization in Geometry and Dynamics”.

## References

- [Levoy and Whitted(1985)] Levoy, M, Whitted, T. The use of points as a display primitive. University of North Carolina, Department of Computer Science; 1985.
- [Boehnen and Flynn(2005)] Boehnen, C, Flynn, P. Accuracy of 3d scanning technologies in a face scanning scenario. In: IEEE Fifth International Conference on 3D Digital Imaging and Modeling. 2005, p. 310–317.
- [Buck et al.(2007)] Buck, Naether, Braun, Bolliger, Friederich, Jackowski et al.]  
Buck, U, Naether, S, Braun, M, Bolliger, S, Friederich, H, Jackowski, C, et al. Application of 3d documentation and geometric reconstruction methods in traffic accident analysis with high resolution surface scanning, radiological msct/mri scanning and real data based animation. *Forensic science international* 2007;170(1):20–28.
- [Levoy et al.(2000)] Levoy, Pulli, Curless, Rusinkiewicz, Koller, Pereira et al.]  
Levoy, M, Pulli, K, Curless, B, Rusinkiewicz, S, Koller, D, Pereira, L, et al. The digital michelangelo project: 3d scanning of large statues. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques. 2000, p. 131–144.
- [Yadav et al.(2017)] Yadav, Reitebuch and Polthier] Yadav, SK, Reitebuch, U, Polthier, K. Mesh denoising based on normal voting tensor and binary optimization. *IEEE Transactions on Visualization and Computer Graphics* 2017;PP(99):1–1. doi:10.1109/TVCG.2017.2740384.
- [Amenta and Kil(2004)] Amenta, N, Kil, YJ. Defining point-set surfaces. ACM Trans Graph 2004;23(3):264–270. URL: <http://doi.acm.org/10.1145/1015706.1015713>. doi:10.1145/1015706.1015713.
- [Mitra and Nguyen(2003)] Mitra, NJ, Nguyen, A. Estimating surface normals in noisy point cloud data. In: Proceedings of the Nineteenth Annual Symposium on Computational Geometry. ACM; 2003, p. 322–328. URL: <http://doi.acm.org/10.1145/777792.777840>. doi:10.1145/777792.777840.
- [Fleishman et al.(2003)] Fleishman, Drori and Cohen-Or] Fleishman, S, Drori, I, Cohen-Or, D. Bilateral mesh denoising. ACM Trans Graph 2003;22(3):950–953. URL: <http://doi.acm.org/10.1145/882262.882368>. doi:10.1145/882262.882368.
- [Lange and Polthier(2005)] Lange, C, Polthier, K. Computer Aided Geometric Design. Computer Aided Design 2005;22:680–692.
- [Sun et al.(2015)] Sun, Schaefer and Wang] Sun, Y, Schaefer, S, Wang, W. Denoising point sets via  $l_0$  minimization. Comput Aided Geom Des 2015;35:2–15. URL: <http://dx.doi.org/10.1016/j.cagd.2015.03.011>. doi:10.1016/j.cagd.2015.03.011.

- 1 [E. and A.(2016)] E., M, A., C. Point cloud denoising via moving  
 2 rpca. Computer Graphics Forum 2016;36(8):123–137. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13068>. doi:10.1111/cgf.13068.
- 5 [Zheng et al.(2017a)] Zheng, Li, Wu, Liu and Gao] Zheng, Y, Li, G, Wu, S,  
 6 Liu, Y, Gao, Y. Guided point cloud denoising via sharp feature skele-  
 7 tons. Vis Comput 2017a;33(6-8):857–867. URL: <https://doi.org/10.1007/s00371-017-1391-8>. doi:10.1007/s00371-017-1391-8.
- 9 [Berger et al.(2017)] Berger, Tagliasacchi, Seversky, Alliez, Guennebaud, Levine et al.]  
 10 Berger, M, Tagliasacchi, A, Seversky, LM, Alliez, P, Guen-  
 11 nebaud, G, Levine, JA, et al. A survey of surface reconstruction  
 12 from point clouds. Comput Graph Forum 2017;36(1):301–329. URL:  
 13 <https://doi.org/10.1111/cgf.12802>. doi:10.1111/cgf.12802.
- 14 [Levin(2003)] Levin, D. Mesh-independent surface interpolation. In: Geometric  
 15 Modeling for Scientific Visualization; vol. 3. 2003, p. 37–49.
- 16 [Guennebaud and Gross(2007)] Guennebaud, G, Gross, M. Algebraic point  
 17 set surfaces. ACM Trans Graph 2007;26(3). URL: <http://doi.acm.org/10.1145/1276377.1276406>. doi:10.1145/1276377.1276406.
- 19 [Oztireli et al.(2009)] Oztireli, Guennebaud and Gross] Oztireli, C, Guen-  
 20 nebaud, G, Gross, M. Feature preserving point set surfaces based on  
 21 non-linear kernel regression. Computer Graphics Forum 2009;28(2):493–  
 22 501.
- 23 [Chen et al.(2013)] Chen, Guennebaud, Barla and Granier] Chen, J, Guen-  
 24 nebaud, G, Barla, P, Granier, X. Non-oriented mls gradient fields.  
 25 Computer Graphics Forum 2013;32(8):98–109. URL: <http://dx.doi.org/10.1111/cgf.12164>. doi:10.1111/cgf.12164.
- 27 [Ohtake et al.(2003)] Ohtake, Belyaev, Alexa, Turk and Seidel] Ohtake, Y,  
 28 Belyaev, A, Alexa, M, Turk, G, Seidel, HP. Multi-level partition of  
 29 unity implicits. ACM Trans Graph 2003;22(3):463–470. URL: <http://doi.acm.org/10.1145/882262.882293>. doi:10.1145/882262.882293.
- 31 [Fleishman et al.(2005)] Fleishman, Cohen-Or and Silva] Fleishman, S, Cohen-  
 32 Or, D, Silva, CT. Robust moving least-squares fitting with sharp features.  
 33 ACM Trans Graph 2005;24(3):544–552. URL: <http://doi.acm.org/10.1145/1073204.1073227>. doi:10.1145/1073204.1073227.
- 35 [Wang et al.(2013)] Wang, Yu, Zhu and Cao] Wang, J, Yu, Z, Zhu, W, Cao,  
 36 J. Feature-preserving surface reconstruction from unoriented, noisy point  
 37 data. Computer Graphics Forum 2013;32(1):164–176. URL: <http://dx.doi.org/10.1111/cgf.12006>. doi:10.1111/cgf.12006.

- [Avron et al.(2010)] Avron, Sharf, Greif and Cohen-Or] Avron, H, Sharf, A,  
 Greif, C, Cohen-Or, D.  $l_1$ -sparse reconstruction of sharp point set surfaces.  
 ACM Trans Graph 2010;29(5):135:1–135:12. URL: <http://doi.acm.org/10.1145/1857907.1857911>. doi:10.1145/1857907.1857911.
- [Huang et al.(2013)] Huang, Wu, Gong, Cohen-Or, Ascher and Zhang] Huang, H, Wu, S, Gong, M, Cohen-Or, D, Ascher, U, Zhang, HR. Edge-aware point set resampling. ACM Trans Graph 2013;32(1):9:1–9:12. URL: <http://doi.acm.org/10.1145/2421636.2421645>. doi:10.1145/2421636.2421645.
- [Lipman et al.(2007)] Lipman, Cohen-Or, Levin and Tal-Ezer] Lipman, Y, Cohen-Or, D, Levin, D, Tal-Ezer, H. Parameterization-free projection for geometry reconstruction. ACM Trans Graph 2007;26(3). URL: <http://doi.acm.org/10.1145/1276377.1276405>. doi:10.1145/1276377.1276405.
- [Hoppe et al.(1992)] Hoppe, DeRose, Duchamp, McDonald and Stuetzle] Hoppe, H, DeRose, T, Duchamp, T, McDonald, J, Stuetzle, W. Surface reconstruction from unorganized points. SIGGRAPH Comput Graph 1992;26(2):71–78. URL: <http://doi.acm.org/10.1145/142920.134011>. doi:10.1145/142920.134011.
- [Yadav et al.(2018)] Yadav, Reitebuch and Polthier] Yadav, SK, Reitebuch, U, Polthier, K. Robust and high fidelity mesh denoising. IEEE Transactions on Visualization and Computer Graphics 2018;:1–1doi:10.1109/TVCG.2018.2828818.
- [Medioni(2000)] Medioni, G. Tensor voting: Theory and applications. 2000.
- [Garland and Heckbert(1997)] Garland, M, Heckbert, PS. Surface simplification using quadric error metrics. In: Proceedings of the 24th annual conference on Computer graphics and interactive techniques. 1997, p. 209–216.
- [Sun et al.(2007a)] Sun, Rosin, Martin and Langbein] Sun, X, Rosin, P, Martin, R, Langbein, F. Fast and effective feature-preserving mesh denoising. IEEE transactions on visualization and computer graphics 2007a;13(5).
- [Zheng et al.(2017b)] Zheng, Li, Wu, Liu and Gao] Zheng, Y, Li, G, Wu, S, Liu, Y, Gao, Y. Guided point cloud denoising via sharp feature skeletons. The Visual Computer 2017b;33(6-8):857–867.
- [Zheng et al.(2018)] Zheng, Li, Xu, Wu and Nie] Zheng, Y, Li, G, Xu, X, Wu, S, Nie, Y. Rolling normal filtering for point clouds. Computer Aided Geometric Design 2018;URL: <http://www.sciencedirect.com/science/article/pii/S0167839618300189>. doi:<https://doi.org/10.1016/j.cagd.2018.03.004>.

- 1 [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin]  
2      Bernardini, F, Mittleman, J, Rushmeier, H, Silva, C, Taubin, G.  
3      The ball-pivoting algorithm for surface reconstruction. IEEE Trans-  
4      actions on Visualization and Computer Graphics 1999;5(4):349–359.  
5      doi:10.1109/2945.817351.
- 6 [Sun et al.(2007b)Sun, Rosin, Martin and Langbein] Sun, X, Rosin, P, Mar-  
7      tin, R, Langbein, F. Fast and effective feature-preserving mesh de-  
8      noising. Visualization and Computer Graphics, IEEE Transactions on  
9      2007b;13(5):925–938.

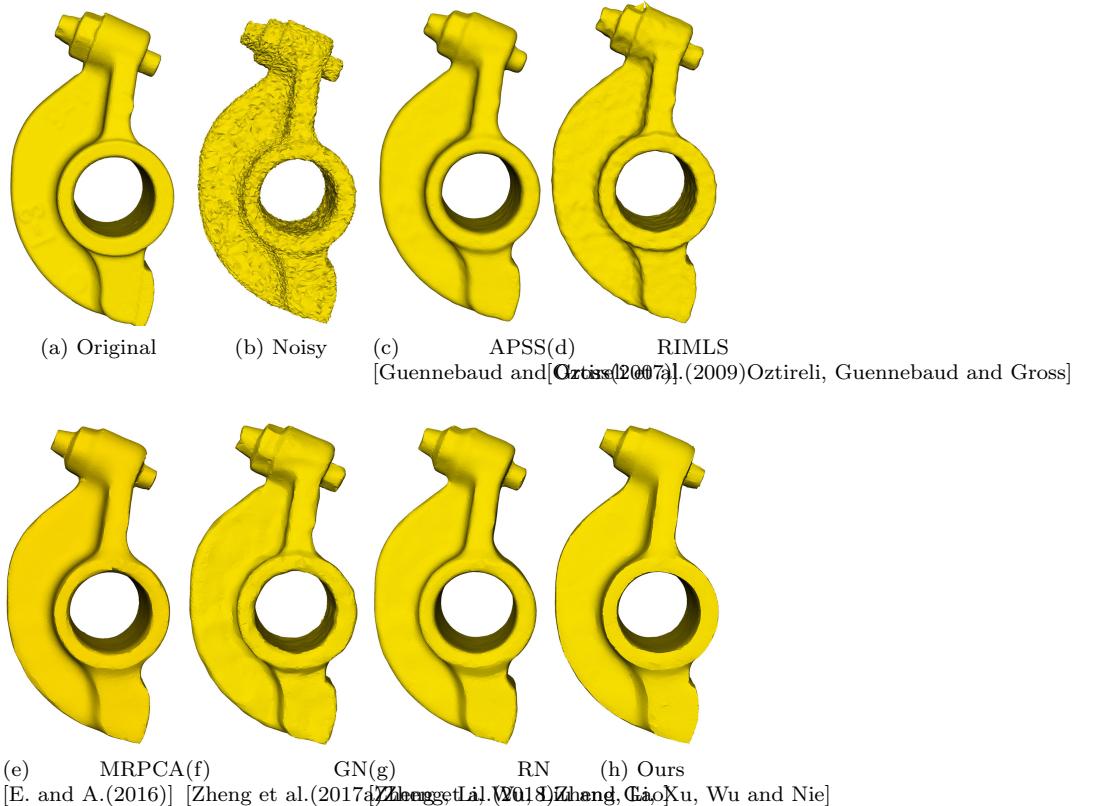


Figure 11: The Rocker arm model corrupted by Gaussian noise ( $(\sigma_n = 0.3l_e)$ ) in normal direction. The results are produced by state-of-the-art methods and our proposed method. The proposed method removes noise effectively and also enhances the sharp features around the cylindrical region. Surfaces are reconstructed using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin].

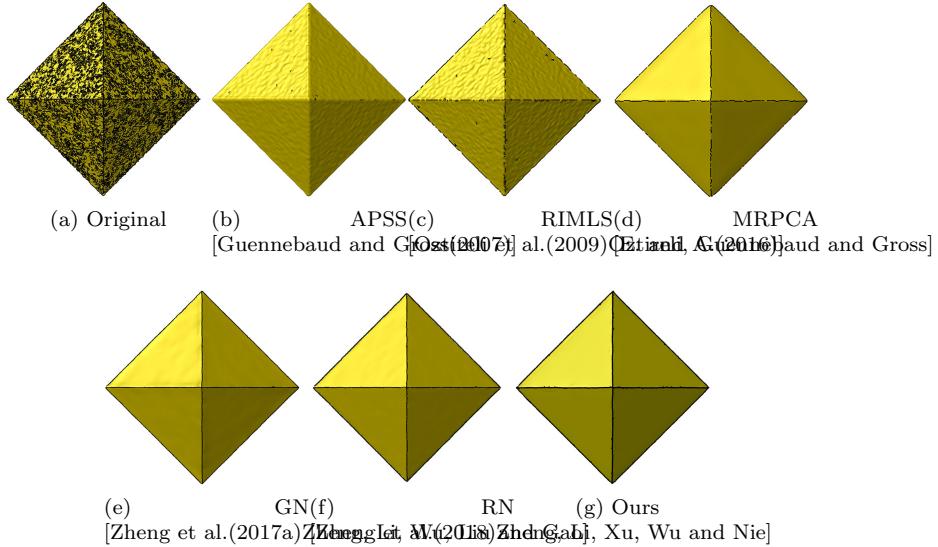


Figure 12: The Octahedron model, which is corrupted by Gaussian noise and results produced by the proposed method and state-of-the-art methods. Surfaces are reconstructed using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin].

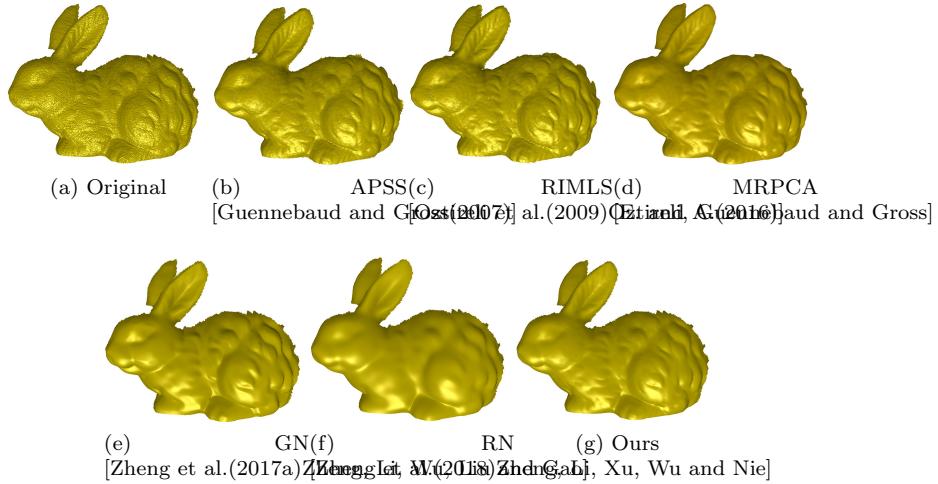


Figure 13: Real data of the Rabbit model, acquired by a 3D scanner and results produced by the proposed method and state-of-the-art methods. Surfaces are reconstructed using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin].

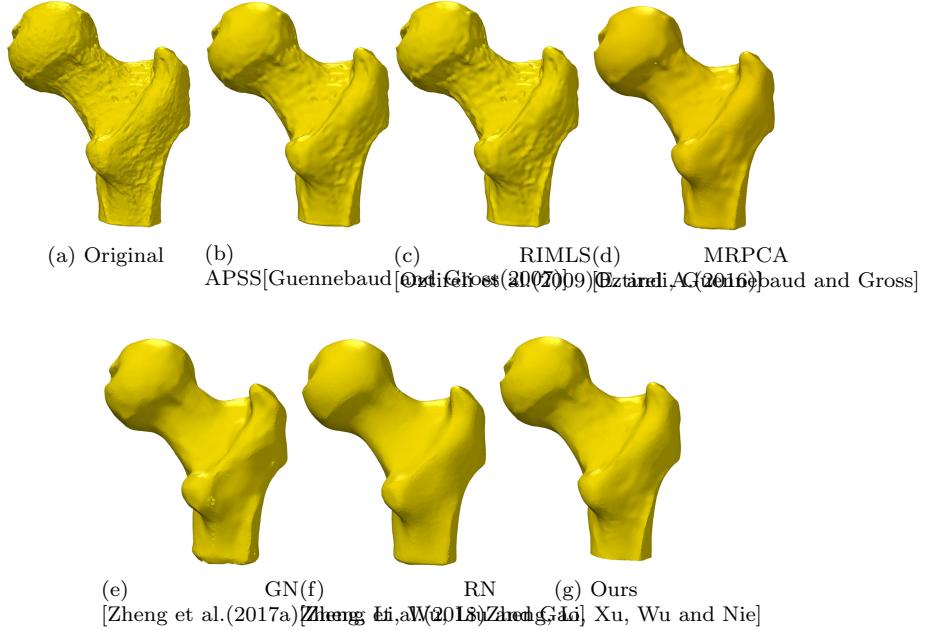


Figure 14: The Ball joint model – which is corrupted by scanner noise – and results produced by the proposed method and state-of-the-art methods. Surfaces are reconstructed using the “ball pivoting” algorithm [Bernardini et al.(1999)Bernardini, Mittleman, Rushmeier, Silva and Taubin].

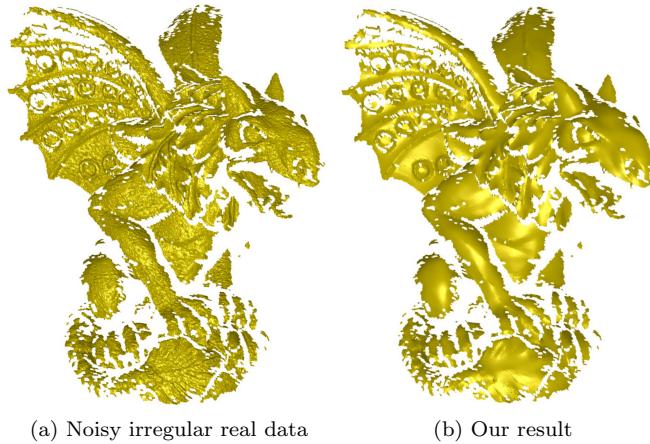


Figure 15: Robustness against irregular data points. Figure (a) shows the noisy irregular data points of the Gargoyle model and Figure (b) shows the result obtained by the proposed method.