

Article

Sparse Regularization-Based Approach for Point Cloud Denoising and Sharp Features Enhancement

Esmeide Leal ¹, German Sanchez-Torres ^{2,*} and John W. Branch ³

¹ Faculty of Engineering, Universidad Autónoma del Caribe, Barranquilla 080001, Colombia; esleal@uac.edu.co

² Faculty of Engineering, Universidad del Magdalena, Santa Marta 470004, Colombia

³ Facultad de Minas, Universidad Nacional de Colombia, Sede Medellín 050041, Colombia; jwbranch@unal.edu.co

* Correspondence: gsanchez@unimagdalena.edu.co

Received: 24 April 2020; Accepted: 20 May 2020; Published: 5 June 2020



Abstract: Denoising the point cloud is fundamental for reconstructing high quality surfaces with details in order to eliminate noise and outliers in the 3D scanning process. The challenges for a denoising algorithm are noise reduction and sharp features preservation. In this paper, we present a new model to reconstruct and smooth point clouds that combine L1-median filtering with sparse L1 regularization for both denoising the normal vectors and updating the position of the points to preserve sharp features in the point cloud. The L1-median filter is robust to outliers and noise compared to the mean. The L1 norm is a way to measure the sparsity of a solution, and applying an L1 optimization to the point cloud can measure the sparsity of sharp features, producing clean point set surfaces with sharp features. We optimize the L1 minimization problem by using the proximal gradient descent algorithm. Experimental results show that our approach is comparable to the state-of-the-art methods, as it filters out 3D models with a high level of noise, but keeps their geometric features.

Keywords: point cloud denoising; 3D surface reconstruction; sparse representation

1. Introduction

With the rapid expansion of 3D scanning devices, the process of capturing and scanning real objects has become a common task in many areas, ranging from medicine and entertainment to industry and 3D printing. Despite significant development in the precision of 3D scanning technology, the raw data produced by the scanning devices inevitably contains noise and outliers caused by the inherent measurement error of 3D devices and the digitalization process. Herein lies the importance of denoising the point cloud in a pre-processing step before proceeding with surface reconstruction or shape analysis. The goal of the denoising algorithms is to suppress noise and outliers while preserving the sharp features such as edges and corners. Unlike denoising methods focused on triangular meshes, methods focused on point clouds do not have connectivity information, introducing an additional challenge. Denoising point clouds with sharp features is a complex problem as the features and noise are high frequency and therefore difficult to distinguish.

Many point set surfaces are piecewise smooth almost everywhere except for a few features such as corners and edges [1,2]. This means that these features are sparse, allowing a sparsity analysis in the point clouds to be conducted to estimate them. We can measure the sparsity of a solution using either the L0 norm or the L1 norm. The L0 norm counts the number of non-zero elements in a vector, directly measuring the sparsity, but it is challenging to optimize due to its non-convexity; furthermore, the L1 norm can approximate the L0 norm. The L1 norm is convex, and under certain conditions,

produces sparse solutions. Some works exploit the sparsity theory on point clouds [1–4], and applying this theory is motivated by the field of sparse signal reconstruction and compressed sensing [5,6]. These works have attempted to overcome the problems related to noisy point clouds; the algorithms proposed in these studies perform well for feature preservation with a certain level of noise. Still, when the noise scale is larger or impulsive noise is present, they usually do not do well. Although the L0 norm produces sparser solutions than the L1 norm, the application of the L0 minimization can over-flatten and over-sharpened effects for small geometric features.

In this paper, we propose a robust method that focuses on removing noise and outliers while preserving the sharp features in a point cloud. Our approach comprises two iterative steps: (1) the normal estimation, finding a regression plane equidistant to all heights in a local neighborhood to then calculate the normal at the plane; and (2) based on the estimation of the normals, the position of the points is updated, using the orthogonal distance of the noisy point to the local regression plane, shifting the point along the normal direction projecting it onto the plane. This two-step procedure is repeated until a minimum error threshold is reached.

Our work is motivated by three observations: (1) L1-median filtering data-fidelity term encourages us to find a local regression plane to approximate the input points while discarding the noise and outliers; (2) points that belong to the same local smooth region will have similar normals, and the differences between them should be sparse, while large differences in values would reveal sharp features; and (3) points in a local neighborhood must comply with a local planarity criterion, except in the sharp features.

The three principal contributions of this paper are as follows: We present a two-step method for both normal estimation and point position update measuring the sparsity of sharp features while discriminating between noise and features. First, an adaptive weighted strategy is used to improve the normal estimation on sharp features. Second, a point cloud denoising method is developed that is sufficiently robust to large scale noise, outliers, and impulsive noise, outperforming some state-of-the-art point clouds denoising methods.

2. Related Work

Point cloud denoising algorithms can be roughly divided into six categories as follows: moving least squares (MLS)-based methods, locally optimal projection (LOP)-based methods, sparsity-based methods, non-local similarity-based methods, graph-based methods, and normal smoothing-based methods. In this section, we are interested in point cloud denoising coupled with features preservation.

2.1. MLS-Based Methods

The MLS [7] methods approximate a noisy input point cloud with a smooth surface by projecting the noisy points onto the MLS surface. Three steps are required to project each point: (1) finding a local reference domain to each point, (2) defining a function above the reference domain by fitting a bivariate polynomial using its neighboring points, and (3) computing the projection by evaluating the polynomial at the origin. MLS methods have some drawbacks, because they are not robust to outliers. The projection procedure can be unstable in high curvature regions and for low sampling rate and can over-smooth the surface. Several variants to this method have been proposed to correct the cited problems and for handling sharp features; e.g., algebraic point set surfaces (APSS) [8], and robust implicit MLS (RIMLS) [9].

2.2. LOP-Based Methods

Without the use of normal information, LOP-based [10] methods generate a set of points called particles using the L1 median and a regularization term. This method projects the points onto an underlying surface while enforcing a uniform distribution of points. A variation to this method is the weighted LOP (WLOP) [11]; this method produces more evenly distributed points on the surface. Edge aware resampling (EAR) [12] improves sharp features by modifying LOP to use normal information as

a weight function. However, LOP-based methods can produce over-smoothing because of the use of local operators.

2.3. Non-Local Similarity-Based Methods

These methods are inspired by the image processing techniques non-local mean (NLM) [13] and the block-matching and 3D filtering (BM3D) [14] algorithms, and they exploit the concepts of self-similarity between small surface patches in the point cloud. The direct application of this concept to the point cloud is not straightforward, because the point cloud structure does not exhibit a regular disposition such as the pixels in an image in their structure. The methods based on MLN or BM3D better preserve structural features under a high level of noise. One of the first works extending the NLM algorithm to operate in point clouds is [15]. This method uses the MLS surface and its polynomial coefficients as neighborhood descriptors to find similar patches or neighborhoods. In [16], the authors generalized the BM3D, searching for similar neighborhoods globally in the point cloud using the iterative closest point (ICP) algorithm. Low-rank matrix representation is used in [4], wherein the authors use dictionary representation from the noisy patches to smooth 3D patches. The drawback of this method is its computational complexity given the global point clouds search.

2.4. Graph-Based Methods

Graph-based methods treat the point cloud as a signal in a graph, and the smooth surface is chosen by using graph filters. In [17], the authors use the graph of the k -nearest neighborhood to represent a point cloud as a signal, and carry the smoothing out via convex optimization. The method in [18] used weighted graph Laplacian over the normals and total variation L1-norm as the regularized term to model two kinds of additive noise. Using a bipartite graph, they establish a linear relationship between the points and normals to proceed with the optimization. In [19], the authors proposed the use of a graph Laplacian regularization (GLR) and low manifold model to find self-similarity between patches to the smooth point cloud, avoiding the direct smoothing of point coordinates or point normals. In [20], the authors computed local tangent planes based on a graph and then reconstructed the point cloud by the weighted average of its projections in the tangent planes.

2.5. Normal Smoothing-Based Methods

Normal smoothing methods are focused on estimating noise-free normals, followed by an updating of the position of the points based on the clean normals. In [21], the authors used a robust version of principal component analysis (PCA) to estimate the normal vector; the authors proposed weight factors that were inversely proportional to the sum of the distance to the mean. The weights defined in this way make the method robust to outliers and noise. They proposed a simple solution to avoid data shrinkage using bootstrap bias correction. The method iteratively smooths the surface and preserves sharp features.

In [22], the authors used the multi-normal guided concept (GN) to correctly estimate the normals in edges and corners, based on the observation that points on the same side of the edges have the same normal orientation. The first step is to detect the edges and then refine them using the L1-medial skeleton of point cloud algorithm [23], followed by an estimation of the multi-normals, and finally, an updating of the position by optimizing a height-based function. In [24], the authors used the same multi-normal concept for denoising called rolling normal filtering (RN). The first step here was the same as for [22], but for the point position update, they introduced an energy term to avoid point cloud deformation close to the edges. The method is robust to the noise and preserves edges and corners. Yadav et al. [25] used Constraint-based normal voting tensor (CVN) analysis and binary optimization to estimate noise-free normal. Next, to update the position of points, the method was used to classify each point on the cloud as edge, corner, or planar point, and based on this classification, they proposed three optimization procedures for each type of point. The method is effective for denoising the point cloud and preserves the sharp features.

2.6. Sparsity-Based Methods

These methods are based on the theory of sparse representation of certain geometric features of the point cloud. The sparsity-based method assumes local planarity for the optimization model. Recently, attention has been devoted to sparsity-based methods in geometry processing [26]. These methods comprise two sparse modeling steps. The first carries out a sparse reconstruction of the normals by solving a global minimization problem with sparse prior regularization. The prior model can be the L0-norm or L1-norm. The local planarity comprises the following assumption: If two points belong to the same smooth region, its normal vectors will be similar; therefore, the gradient should be sparse. Based on the smoothed normals, it updates the point positions following a second sparse global model optimization. Methods in [1,2] follow this strategy. Recently, authors in [3] proposed a method called moving robust principal component analysis (RMPCA), using weighted minimization of the point deviations from a local reference plane to preserve sharp features. However, when the noise level is high, over-smoothing or over-sharpening occurs. Our approximation belongs to sparsity-based methods and is in the same spirit of RMPCA, but the difference lies in that we use sparsity in both data fitting and the prior term. Our method uses the L1 median for the fitting term and the L1 norm for the regularization term. For the proposed method, we apply a local sparse optimization strategy based on proximal gradient.

3. Robust Point Cloud Denoising

As in previous point cloud denoising methods [22,24,25], our method comprises two steps, namely normal denoising and point position update.

Starting from a noisy point cloud P near an unknown surface S , the goal of the proposed algorithm is to find a noise-free point set P' that conserves the features of the original point cloud. We use local neighborhoods to each point $p_i \in P$. The main idea is to define a local reference tangent plane \mathfrak{F}_p to every p_i in the point cloud and determine its normal vector n_i , and then shift the point p_i in the normal direction to a distance $\tau_i \in \mathbb{R}$, obtaining a new position $p'_i = p_i + \tau_i n_i$, $p'_i \in P'$. The new position p'_i being the projection of point p_i onto the tangent plane \mathfrak{F}_p , which is the linear approximation of the surface S at point p_i , as shown in Figure 1.

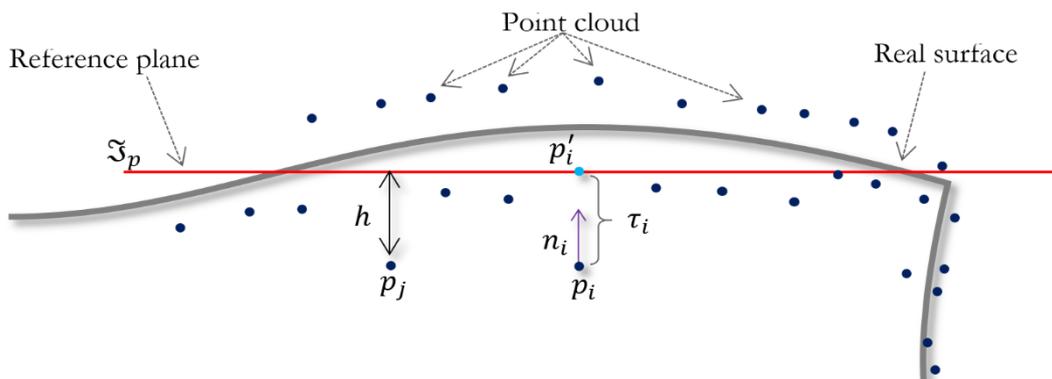


Figure 1. The point p_i is projected onto the reference plane. \mathfrak{F}_p is a linear approximation to the surface.

The normal n_i and the displacement τ_i , are computed iteratively by adjusting the tangent plane \mathfrak{F}_p to the neighborhood $N_g(p_i)$. To estimate the tangent plane \mathfrak{F}_p , we are looking for an equidistance height to all heights of the points $p_j \in N_g(p_i)$ over \mathfrak{F}_p . To estimate the local reference tangent plane \mathfrak{F}_p , we minimize a cost function (Equation (2)) concerning τ and n , subject to the constraint $\|n\| = 1$.

3.1. L1-Median

The L1-median is a robust estimator related to the multivariate median, and is defined to be the point p , which minimizes the sum of Euclidean distances to all points in the data set $\{p_j\}_{j \in J}$.

$$\arg \min_x \sum_{j \in J} \|p_j - p\| \quad (1)$$

The L1-median is insensitive to outliers and noise when compared with the mean [10]. We use the L1-median as a data-fidelity term.

3.2. L1 Sparse Regularization

L1 regularization has been applied for feature selection [27], sparse signal reconstruction [28], signal processing as image decomposition [29], and basis pursuit [30]. Although L0 regularization produces the sparsest solution, under certain constraints, L1 regularization produces a sparse solution [31]. Image processing has successfully applied L1 regularization to preserve fine details and edges through the minimization of the gradient [32]. This is conceptually named total-variation regularization or TV, and is used to measure the sparsity of the gradient. The proposed method uses TV for normal estimation to preserve the sharp features.

3.3. Cost Function

To denoise the noisy point cloud, we integrate L1-median height filter and L1 regularization of gradient or total variation. The normal is obtained, minimizing the following energy functional:

$$\min_{n, \tau} E_f + \lambda E_{reg} \quad (2)$$

where E_f is the fidelity term, E_{reg} is the regularization term, λ is the regularization parameter, and $\|n\| = 1$.

The L1-median height fidelity term E_f is defined as follows:

$$E_f = \sum_{q_j \in N_g(p_i)} \|h_i - \tau_i\| \psi(h_i, \tau_i) \theta(\|p_i - q_j\|) \quad (3)$$

where $h_i = n_i^t(p_i - q_j)$, $\psi(h_i, \tau_i) = e^{-(h_i - \tau_i)^2 / \sigma_h^2}$, and $\theta(\|p_i - q_j\|) = e^{-d^2 / \sigma_d^2}$.

It is used to fit a robust hyperplane in the neighborhood of the sampled point p_i , and to then estimate the normal vector with respect to the hyperplane. We minimize the sum of Euclidean distances of the orthogonal projections (height) of points $q_j \in N_g(p_i)$, with respect to the hyperplane (Equation (3)). τ_i is the height to be found, which minimized the orthogonal projections of each point q_j to the hyperplane.

$$\arg \min_x \sum_{j \in J} \|p_j - x\| \quad (4)$$

The estimation of the hyperplane shows robustness to large deviations of points q_j . The outliers are identified by the L1-median height filter, which penalizes points q_j with high orthogonal projections or heights h_i with respect to the hyperplane.

Consequently, points q_j with considerable heights, h_i , are probably located passing through the sharp features; these points are possible outliers. As such, we propose an adaptive weighting strategy, which adaptively assigns the weight of each point as a function of the height. Thus, the weighting term $\psi(\cdot)$, adaptively encourages the reduction of the influence of points q_j with high h_i values, and σ_h is the height parameter, which controls sensitivity to outliers. Thus, the term $\psi(\cdot)$ only considers points located in the same smooth region to estimate the normal vector.

The L1 norm regularization term E_{reg} is defined as

$$E_{reg} = \sum_{n_j \in N_g(n_i)} w_{i,j} \|n_i - n_j\|_1 \quad (5)$$

where $w_{i,j} = e^{-\left(\frac{1-n_i^t n_j}{1-\cos(\sigma_n)}\right)^2}$, which is introduced as a measure of sparsity to preserve the sharp features and smooth the underlying surface. If a point cloud is piecewise smooth, many of the gradients in the normals field n (consistently oriented) tend to be zero; in contrast, the large values of the gradient only indicate sharp features. This means that normals $n_j \in N_g(n_i)$ in a neighborhood must be similar, where $w_{i,j}$ is the normal weight function, and σ_n is the angle parameter that measures the similarity between normals n_j , which is customarily is set to $\sigma_n = 15^\circ$.

3.4. Model Optimization

We optimize n_i and τ_i subject to $\|n_i\| = 1$, using a minimizing strategy defined as follows:

$$\min_{n_i, \tau} \sum_{q_j \in N_g(p_i)} \|h_i - \tau_i\| \psi(h_i, \tau_i) \theta(\|p_i - q_j\|) + \lambda \sum_{n_j \in N_g(n_i)} w_{i,j} \|n_i - n_j\|_1. \quad (6)$$

This approach finds the optimal values of n_i and τ_i by alternating optimization strategies, a procedure shown in Algorithm 1.

Algorithm 1. Model optimization

```

1 Initialization:  $\tau_0 = 0$ 
2 repeat
  j=0
  repeat
3   fix  $\tau_k$ , solve for  $n_{k+1}$  as minimum of Equation (6).
4   fix  $n_k$ , solve for  $\tau_{k+1}$  as minimum of Equation (6).
    $p_{k+1} = p_k + \tau_k n_k$ 
5   until  $\|p_{k+1} - p_k\|_2^2 < \varepsilon$ 
   edge_points_correction()
14 until  $j > j_{max}$ 

```

We solve the energy minimization problem regarding n having fixed τ . Since the minimization problem is non-differentiable due to the regularization term E_{reg} , we use the proximal gradient descent method [33] as an optimization strategy.

3.4.1. The n_0 Parameter Initialization

First, we estimate the initial normal set to each $p_i \in P$ using only the equation corresponding to the fidelity term E_f , with $\tau = 0$ for the optimization. Similar to [34], we use the constraint $\|n\| = 1$, and compose the Lagrange form of Equation (3) to compute the derivative with respect to n , obtaining

$$L(n, \lambda) = \sum_{q_j \in N_g(p_i)} \|h_i\| \psi(h_i) \theta(\|p_i - q_j\|) + \frac{\lambda}{2} (1 - \|n_i\|^2) \quad (7)$$

$$L_n(n, \lambda) = \sum_{q_j \in N_g(p_i)} \omega_i(p_i - q_j) (p_i - q_j)^t n_i - \lambda n_i = 0 \quad (8)$$

with weight $\omega_i = \psi(h_i) \theta(\|p_i - q_j\|) / \|h_i\|$

We can see that the term $\omega_i(p_i - q_j)(p_i - q_j)^t$ on Equation (8) is a symmetric and definite positive matrix (weighted covariance matrix), and we can rewrite it depending on n as

$$Cm(n)n = \lambda n \tag{9}$$

where $Cm(n) = \sum_{q_j \in N_g(p_i)} \omega(p_i - q_j)(p_i - q_j)^t$

Equation (6) is an eigensystem and can be solved iteratively as follows:

$$Cm(n_k)n_{k+1} = \lambda_{k+1}n_{k+1} \tag{10}$$

where λ_{k+1} is the smallest eigenvalue of $Cm(n_k)$, and n_{k+1} is an orthonormal eigenvector. We start the initialization with $n_0 = 0$, i.e., $Cm(0)n_1 = \lambda_1 n_1$, is the first iteration.

3.4.2. Optimization of n

Keeping τ and $\psi(h_i, \tau_i)$ fixed to solve Equation (6), $\psi(h_i, \tau_i)$ is treated as a constant because it is a practical way to make it computationally tractable. Thus, the fidelity term E_f has gradient ∇E_f as follows:

$$\nabla E_f(n) = \sum_{q_j \in N_g(p_i)} \eta_i (h_i - \tau_i)(p_i - q_j)^t \tag{11}$$

with weight $\eta_i = \psi(h_i, \tau_i)\theta(\|p_i - q_j\|)/\|h_i - \tau_i\|$. η_i is undefined when $h_i = \tau_i$; therefore, when $\|h_i - \tau_i\| < 10^{-3}$, we set $\eta_i = \theta(\|p_i - q_j\|)$.

Setting $d = n_i - n_j$, we define the proximal mapping (or operator), associated with a convex non-differentiable function $h()$, as follows:

$$prox_{h,\gamma}(d) = \arg \min_z (h(z) + \frac{1}{2\gamma}\|z - d\|_2^2) \tag{12}$$

The proximal gradient descent has an iteration form as follows:

$$d^{k+1} = prox_{h,\gamma}(d^k - \gamma \nabla E_f(n^k)) \tag{13}$$

where $\gamma > 0$ is a scalar termed step size, and d^{k+1} is computed iteratively until convergence. The proximal operator corresponding to the L1-norm or regularization term E_{reg} is the following shrinkage or soft thresholding function:

$$prox_{h,\gamma}(d_{ic}) = \begin{cases} d_{ic} - \gamma \lambda w_{ij} & \text{if } d_{ic} > \gamma \lambda w_{ij} \\ 0 & \text{if } |d_{ic}| \leq \gamma \lambda w_{ij} \\ d_{ic} + \gamma \lambda w_{ij} & \text{if } d_{ic} < -\gamma \lambda w_{ij} \end{cases} \tag{14}$$

where d_{ic} is each component of normal vector d_i .

3.4.3. Optimization of τ

With n fixed, we solve Equation (6) for τ , which shows that the fidelity term E_f has gradient ∇E_f .

$$\nabla E_f(\tau) = \sum_{q_j \in N_g(p_i)} \eta_i (h_i - \tau_i) = 0 \tag{15}$$

By solving $\nabla E_f(\tau)$, we obtain an iterative solution, which yields the following local update equation:

$$\tau_i^{k+1} = \frac{\sum_{q_j \in N_g(p_i)} \eta_i h_i}{\sum_{q_j \in N_g(p_i)} \eta_i} \tag{16}$$

where $\eta_i = \frac{\psi(h_i, \tau_i) \theta(\|p_i - q_j\|)}{\|h_i - \tau_i\|}$. The parameters n and τ are iteratively optimized using Equations (13) and (16) until convergence; η_i is undefined when $h_i = \tau_i$. Therefore, when $\|h_i - \tau_i\| < 10^{-3}$, we set $\eta_i = \theta(\|p_i - q_j\|)$.

3.5. Point Position Update and Point Border Correction

In the last stage of the denoising method, we follow the update vertex position with a distance-based constraint proposed by [25], where the resulted point cloud P is bounded within a prescribed distance to the input point cloud.

3.5.1. Point Position Update

The authors in [25] propose a parameter provided by the user $\varepsilon \in R^+$, bounding the maximum deviation d_i between the initial noisy point cloud and its corresponding iteratively denoised version point $p'_i \in P'$. The update position point p'_i for our algorithm is determined as follows:

$$p'_i = \begin{cases} p_i + \tau_i n_i & \text{if } d_i \leq \varepsilon \\ p_i & \text{if } d_i \geq \varepsilon \end{cases} \quad (17)$$

where d_i is computed as the difference between p_i and the corresponding original point in the noisy point cloud. The parameter ε is set to $4h$, i.e., $\varepsilon = 4h$.

To make our algorithm more robust against edge artifacts and blurring, we detect the edge and corner points, and it corrects its position to obtain cleaner and more defined borders.

3.5.2. Edge Points Detection

To detect sharp features in the point cloud, we refer to the method proposed in [22], which uses the normals associated to each point in P and measures the normal variability into the neighborhood; if the variability is lower than a predefined threshold, th is labeled as edge point. The similarity between normal vectors n_i and n_j is defined as follows:

$$w_n(n_i, n_j) = \exp\left(-\frac{\|n_i - n_j\|^2}{2\sigma_n^2}\right) \quad (18)$$

where σ_n is an angle threshold; using Equation (18), we define the normal variation in $N_g(n_i)$ as follows:

$$V_n(i) = \frac{1}{|N_g(n_i)|} \sum_{j \in N_g(n_i)} w_n(n_i, n_j). \quad (19)$$

All points that satisfy $V_n(i) < th$ are labeled as edge points.

3.5.3. Edge Point Correction

After edge points detection and taking advantage of the fact that the estimated normals near the edges and corners belong to surfaces on one side or another of the sharp features, we propose a scheme to correct the position of points that belong to edges or corners, which present a deviation from the corner or borderline. As shown in Figure 2, we find the closest point p_j with normal vector n_j on an opposite surface to the edge point p_i and its normal n_i . Next, we project point p_i onto the plane that contains point p_j and its normal n_j . The new position is computed as

$$d_{proj} = n'_j(p_i - p_j). \quad (20)$$

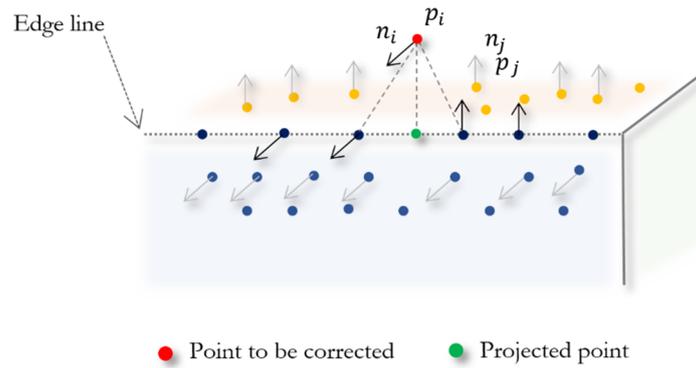


Figure 2. Edge points correction.

We only correct the point positions that meet the following criteria:

$$p_{corr} = \begin{cases} p_i - d_{proj} \cdot n'_j & \text{if } \delta < |d_{proj}| < \rho \cdot h \\ p_i & \text{other case} \end{cases} \quad (21)$$

where h is the average distance between the points of the point cloud, ρ is a fixed value that represents the percentage of the maximum shift of point p_i towards the edge line, and δ is a fixed value close to zero.

4. Experimental Results and Discussion

The proposed method was implemented in MATLAB and run on a laptop with Intel Core i7-2670QM CPU, 2.20 GHz processor, and 8 GB RAM. We tested the method using several point clouds with sharp features and smooth surfaces including irregular sampling. Additionally, synthetic and real scanned noisy point clouds were used to validate our method. The synthetic models were contaminated with Gaussian noise and impulsive noise along the normal directions or random directions. Different levels of Gaussian noise with zero mean and standard deviation σ were applied to the models; the standard deviation was proportional to the average distance between the points of the ground-truth point clouds. The noise of raw scanned data was natural. We compared our method with eight state-of-the-art denoising approaches including two MLS-based methods, APSS [8] and RIMLS [9]; one LOP based method, EAR [12]; one sparsity-based method, MRPCA [3]; one graph-based method, GLR [19]; and three normal smoothing-based methods, CNV [25], RN [24], and GN [22]. Methods APSS and RIMLS were implemented in MeshLab software. The GLR code and EAR software were provided by the authors, as were the results of the MRPCA, CNV, RN, and GN methods.

4.1. Parameter Selection and Tuning

Our method presented seven parameters: the sparsity parameter λ , the height sensitivity σ_h , the distance action range σ_d , the bound displacement ρ , the low bound δ , the radius of neighborhood r , and the total number of iterations k .

The sparsity regularization parameter λ , depends on the desired gradient sparsity level and affects the reconstruction of sharp features and the smoothness of the point cloud. A larger λ yields a smoother result. We observed that $\lambda = 0.2$ worked well for all the testing point sets used in the experiments.

The displacement ρ was fixed throughout all experiments with $\rho = 0.7$. Parameter δ was fixed throughout all experiments with $\delta = 10^{-4}$. Parameter h was the average distance between the points. We computed the value of h , taking the six nearest neighbors to each point. The distance action range σ_d , and the height sensitivity σ_h , are user-defined values given in terms of h . In all the experiments, the radius r of the neighborhood was set to σ_d , i.e., $r = \sigma_d$; a smaller value of σ_d leads to faster computation because the neighborhood $N_g(p_i)$ is small, and large values may cross sharp features and over smooth

the results. Alternatively, $r = \sigma_d$ can be chosen as a function of the local point density. In the results of the experiments, we chose the values of this parameter constant, tuned to achieve visually appealing results. The height sensitivity, σ_h , controls the outliers in the point cloud; small values of σ_h preserve the model features, while large values only preserve the salient features.

The values of σ_h and σ_d depend on the level of noise. The bigger the noise level, the larger the value of these parameters that should be chosen. We used $\sigma_d \in \{1.5h, 2h, 3h, 4h\}$ and $\sigma_h \in \{0.5h, 0.7h, 0.9h\}$ for synthetic data and $\sigma_d \in \{1.5h, 2h\}$ and $\sigma_h \in \{0.1h, 0.2h, 0.3h\}$ for real scanned point clouds. The difference of parameter values between synthetic and real models was because the level of noise in real models was lower than synthetic models. The number of iterations k for the best results was set at $k \in \{10, 16, 20, 50\}$. At last, there were only three parameters for our algorithm to tune the results (σ_h, σ_d, k) .

In our comparison experiments, we used the following parameter set for the eight selected state-of-the-art methods. For the methods [3,12,22,24], we mention Default in the parameter Table 1, because the corresponding smooth models were provided by the authors in [8]; we reported a tuple (scale, # of iterations, α); [9] = (σ_r, σ_n) ; [12] = (Default values); [25] = (τ, ρ, p) , and [19], the parameter settings in their paper. Our method = (σ_h, σ_d, k) .

Table 1. Parameter settings of comparative methods for different models.

Methods	Cube	Fandisk	Rocker Arm	Octahedron
EAR	Default	Default	Default	Default
APSS	(2, 45, 0.5)	(4, 15, 0)	(4, 15, 0.5)	(2, 45, 0.5)
RIMLS	(4, 0.75)	(4, 15, 0)	(4, 1)	(4, 0.75)
MRPCA	Default	Default	Default	Default
GLR	Paper	Paper	Paper	Paper
GN	Default	Default	Default	Default
RN	Default	Default	Default	Default
CNV	(0.3, 0.95, 150)	(0.3, 0.9, 150)	(0.25, 0.9, 80)	(0.25, 0.9, 80)
Ours	(0.98 h, 4 h, 30)	(0.7 h, 3 h, 16)	(0.7 h, 3 h, 14)	(0.7 h, 2 h, 20)

4.2. Quantitative Analysis

We used the following three metrics in our quantitative analysis: feature preservation, accuracy, and signal-to-noise ratio.

4.2.1. Error Metrics

To quantify feature preservation, we measured the orientation error between the smoothed point cloud and the ground truth. Mean angular deviation (MAD) was defined to measure the orientation error as follows:

$$MAD = \frac{1}{n} \sum_{i=0}^{n-1} \angle(\bar{n}_i, \hat{n}_i) \quad (22)$$

where \bar{n}_i and \hat{n}_i are the point normals corresponding to the ground truth and the smoothed point cloud, respectively.

To quantify the accuracy, i.e., the closeness between the ground truth model and the smoothing model, we used the mean-squared-error (MSE) metric, which measures the average of the squared Euclidean distances between the ground truth points and their closest denoising points, and vice versa between the denoised points and their closest ground truth points, and finally, the average between two measures gives the MSE. If the ground truth model and the smoothed model are $P_1 = \{p_i\}_{i=1, n_1}$ and $P_2 = \{q_j\}_{j=1, n_2}$, the point clouds can be of different sizes, i.e., $n_1 \neq n_2$. The MSE is defined as follows:

$$MSE = \frac{1}{2n_1} \sum_{p_i \in P_1} \min_{q_j \in P_2} \|p_i - q_j\|_2^2 + \frac{1}{2n_2} \sum_{q_j \in P_2} \min_{p_i \in P_1} \|q_j - p_i\|_2^2 \quad (23)$$

Finally, the signal-to-noise ratio (SNR) is measured in dB and is defined as follows.

$$SNR = 10 \log \frac{\frac{1}{n_2} \sum_{q_i \in P_2} \|q_i\|_2^2}{MSE} \quad (24)$$

4.2.2. Metric Evaluation

Tables 2 and 3 shows the comparison between our method and competing state-of-the-art methods highlighting cells in gray indicating the best performance. In Table 2 we can also see that the MSE and SNR metrics are the lowest values compared to eight state-of-the-art methods. For the Fandisk model, the proposed method reached the second position in all three metrics. For the Rocker Arm, our method outperformed the state-of-the-art methods in the MAD metric, but with MSE and SNR our method was better than the APSS method. For the Octahedron model, our method outperformed all metrics of the state-of-the-art methods.

Table 2. Error metrics by comparative methods for each one of the 3D objects.

	Methods	EAR	APSS	RIMLS	MRPCA	GLR	GN	RN	CNV	Ours
Cube	MAD	4.3634	5.5588	4.4552	4.4341	6.2429	3.4878	4.4531	2.8668	2.6985
	MSE	0.0183	0.0125	0.0117	0.0273	0.0336	0.0330	0.0352	0.0064	0.0046
	SNR (dB)	39.438	42.613	43.272	35.749	33.936	33.920	33.305	48.394	51.418
Fandisk	MAD	4.4038	5.0465	5.6874	3.7932	7.7937	2.9186	3.1585	3.5273	2.9691
	MSE	0.0073	0.0057	0.0060	0.0067	0.0257	0.0060	0.0038	0.0108	0.0039
	SNR (dB)	45.105	46.168	45.965	45.525	39.653	45.963	48.006	43.410	47.833
Rocker Arm	MAD	5.9647	4.8825	4.9493	6.0163	7.1012	7.7694	5.7894	7.1894	4.2611
	MSE	0.1392	0.0468	0.0717	0.1345	0.2554	0.6141	0.5873	0.1651	0.0665
	SNR (dB)	36.116	40.830	38.988	36.234	33.450	29.717	29.885	35.340	39.300
Octahedron	MAD	1.8779	3.9838	4.8495	4.9541	5.2574	1.3606	1.3776	1.0415	1.0196
	MSE	9.5E4	0.0014	0.0011	0.0014	0.0016	0.0074	0.0074	7.0E4	5.6E4
	SNR (dB)	54.384	51.007	52.846	51.006	50.008	55.731	55.631	57.057	58.931

Table 3. The results of the error metrics of different compared methods for two Block and Trim-star objects varying the noise levels.

	Methods	EAR	APSS	RIMLS	GLR	Ours	
$\sigma = 0.1$ h	Block	MAD	3.8083	4.2386	3.1723	2.9909	2.9232
		MSE	0.0693	0.0641	0.0466	0.0469	0.0477
		SNR(dB)	34.518	34.842	36.238	36.199	36.124
	Trim-star	MAD	5.0802	4.7813	4.1111	7.0203	3.6042
		MSE	0.0634	0.0324	0.0408	0.1105	0.0370
		SNR(dB)	29.572	32.459	31.472	27.120	31.855
$\sigma = 0.2$ h	Block	MAD	6.2682	8.2802	4.1979	4.9876	3.5737
		MSE	0.1339	0.1191	0.0688	0.0911	0.0551
		SNR(dB)	31.676	32.149	34.552	33.310	35.492
	Trim-star	MAD	6.9177	7.0610	5.3676	8.3487	4.8816
		MSE	0.1068	0.0525	0.0573	0.1456	0.0522
		SNR(dB)	27.341	30.353	30.011	25.914	30.363
$\sigma = 0.3$ h	Block	MAD	7.6707	11.629	4.6145	9.2070	4.4352
		MSE	0.1573	0.1640	0.1029	0.2735	0.0784
		SNR(dB)	30.980	30.753	32.816	28.488	33.955
	Trim-star	MAD	8.1821	10.9294	6.7664	10.495	5.8392
		MSE	0.0695	0.0995	0.0822	0.1828	0.0632
		SNR(dB)	29.158	27.576	28.465	24.915	29.608

Noise levels and the comparison method are (a) Cube $\sigma = 0.3$ h method CVN, (b) Fandisk $\sigma = 0.28$ h method RN, (c) Rocker Arm $\sigma = 0.3$ h, and (d) Octahedron $\sigma = 0.3$ h.

Table 3 shows a comparison between four state-of-the-art methods, with three different levels of noise, i.e., $\sigma = 0.1$ h, $\sigma = 0.2$ h, and $\sigma = 0.3$ h. We can see that the MAD grew as the noise increased. Thus, if the noise level is high, the orientation error will be larger compared to the lower noise level.

Table 3 shows that the proposed method achieved the best results for the MAD metric, with all levels of noise, but for the level of noise $\sigma = 0.1$ h, RIMLS achieved the best MSE and SNR. However, for the level of noise $\sigma = 0.2$ h and $\sigma = 0.3$ h, our method outperformed the competing methods. In all the experiments, the proposed method achieved the best results on average in all three metrics and all noise levels. The visual results of the experiment are shown in Figure 3. For each one of the objects, we illustrated the original model, the noisy model, the results of the best method when comparing methods, and the results of our approach.

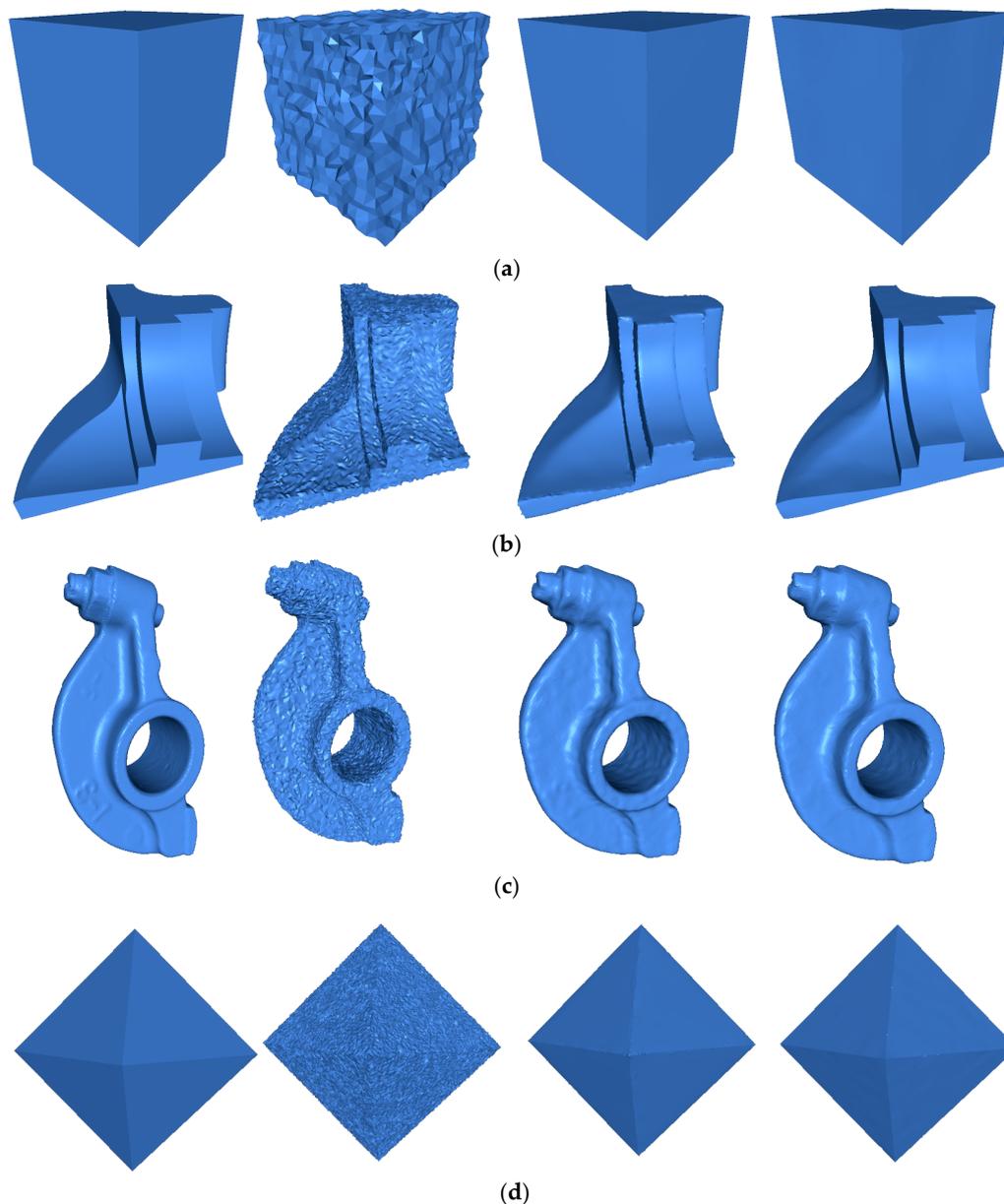


Figure 3. The first column shows the model from the original data. The second model is corrupted by Gaussian noise. The third column shows most accurate result of the comparison methods using MSE metric. The fourth column shows the result of correcting by the proposed method. The Gaussian noise levels and the comparison method are (a) Cube $\sigma = 0.3$ h method CVN, (b) Fandisk $\sigma = 0.28$ h method RN, (c) Rocker Arm $\sigma = 0.3$ h, and (d) Octahedron $\sigma = 0.3$ h.

4.3. Qualitative Analysis

For visual comparison, we used the ball pivoting algorithm (BPA) [35] to reconstruct the mesh from the smoothed point cloud. The point clouds were contaminated with Gaussian noise along random directions and normal directions, and impulsive noise along random direction.

Irregular Point Clouds

The Cube and the Fandisk objects had non-uniform density points corrupted by Gaussian noise in random directions ($\sigma = 0.28 h$, $\sigma = 0.3 h$, and $0.3 h$, respectively). Figure 4, shows that our method could preserve the sharp features and not produce bumps in flat areas like the APSS [8], RIMLS [9], and EAR [12] methods. GLR [19], MRPCA [3], RN [24], and GN [22] methods cleaned the noise effectively over flat regions but produced over-smoothing in the corners and edges. CNV [25] properly reconstructed the sharp features and cleaned the flat areas, but small artifacts appeared in some corners. As seen in Figure 5, our method could reconstruct sharp features and shallow features. APSS smoothed around the sharp features and did not remove the noise correctly. RIMLS and EAR preserved sharp features but produced some bump features in the resulting models. MRPCA removed the noise and preserved some sharp features but smoothed shallow areas around flat regions. GLR removed noise effectively but over smoothed the sharp features and shallow areas. GN, RN, and CVN produced a similar output to our method, but there were some artifacts on the borders and in corners.

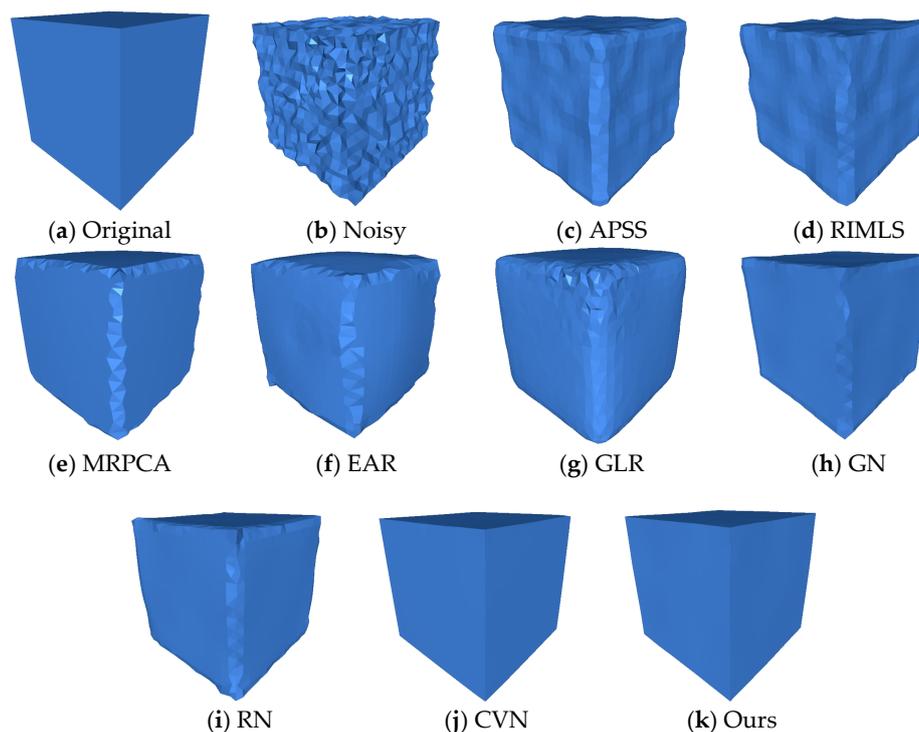


Figure 4. The Cube model with non-uniform distribution of points, corrupted by Gaussian noise ($\sigma = 0.3 h$) along all directions, where h is the average distance between the points of the point cloud. We can see that the proposed method was able to preserve sharp features effectively when compared to the state-of-the-art methods. The surface was reconstructed using the ball pivoting algorithm.

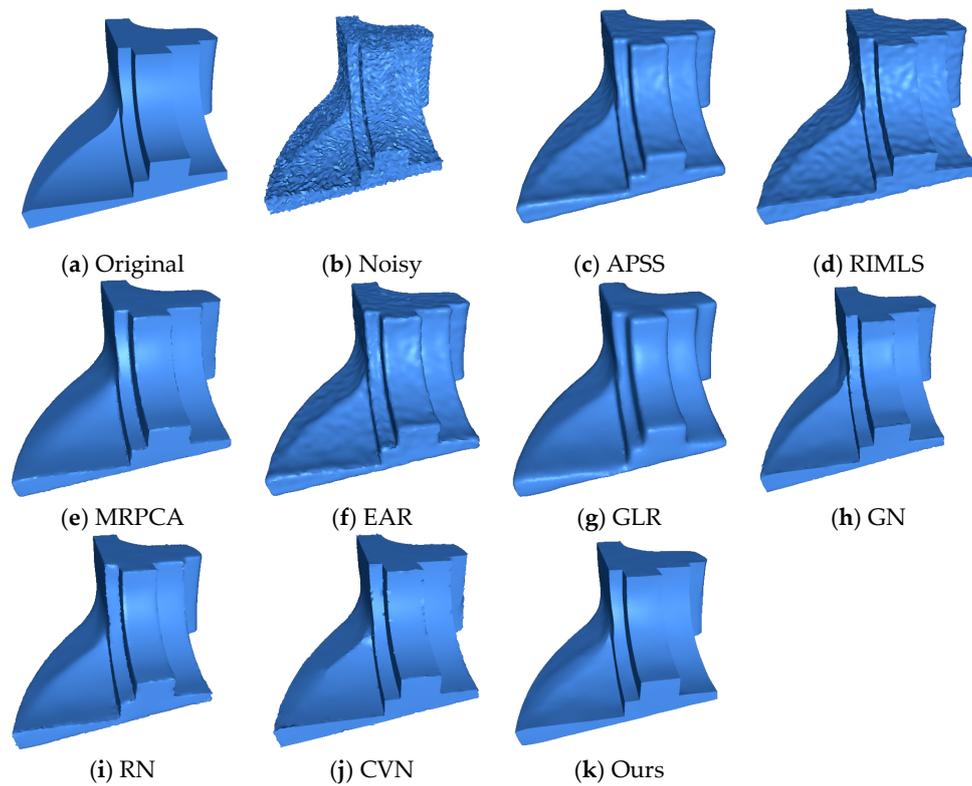


Figure 5. The Fandisk model with non-uniform distribution of points, corrupted by Gaussian noise ($\sigma = 0.28$ h). We can see that the proposed method was able to preserve sharp features effectively when compared to the state-of-the-art methods.

4.4. Impulsive Noise

Figure 6 shows the results of handling a corrupted point cloud adding an impulsive noise of $\sigma = 0.5$ h along the normal direction. The Twelve model was smoothed by the proposed method and its edges were preserved. RIMLS, APSS, and EAR methods were not able to smooth the noise properly and reconstruct the edges.

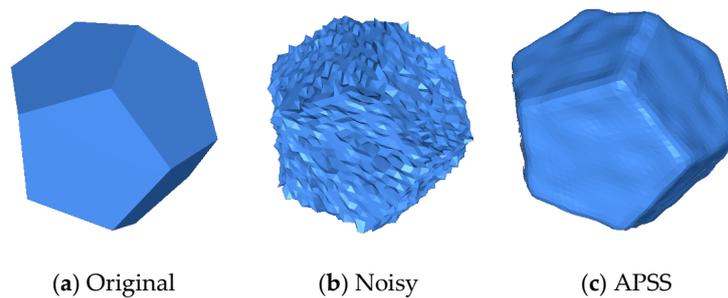


Figure 6. Cont.

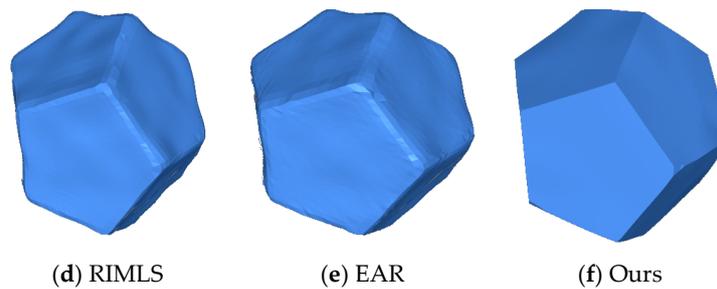


Figure 6. The Twelve model corrupted by impulsive noise ($\sigma = 0.5$ h) in the normal direction. We can see that the proposed method was able to preserve sharp features effectively when compared to the state-of-the-art methods. The surface was reconstructed using the ball pivoting algorithm.

4.5. Natural Noise of 3D Scan Objects

We also compared these approaches using real scanned data of free form objects. Figure 7 shows the results of different methods applied to raw data scans. From the Rabbit object, it seems like our method effectively removed the noise while preserving features when compared to APSS, RIMLS, and EAR.

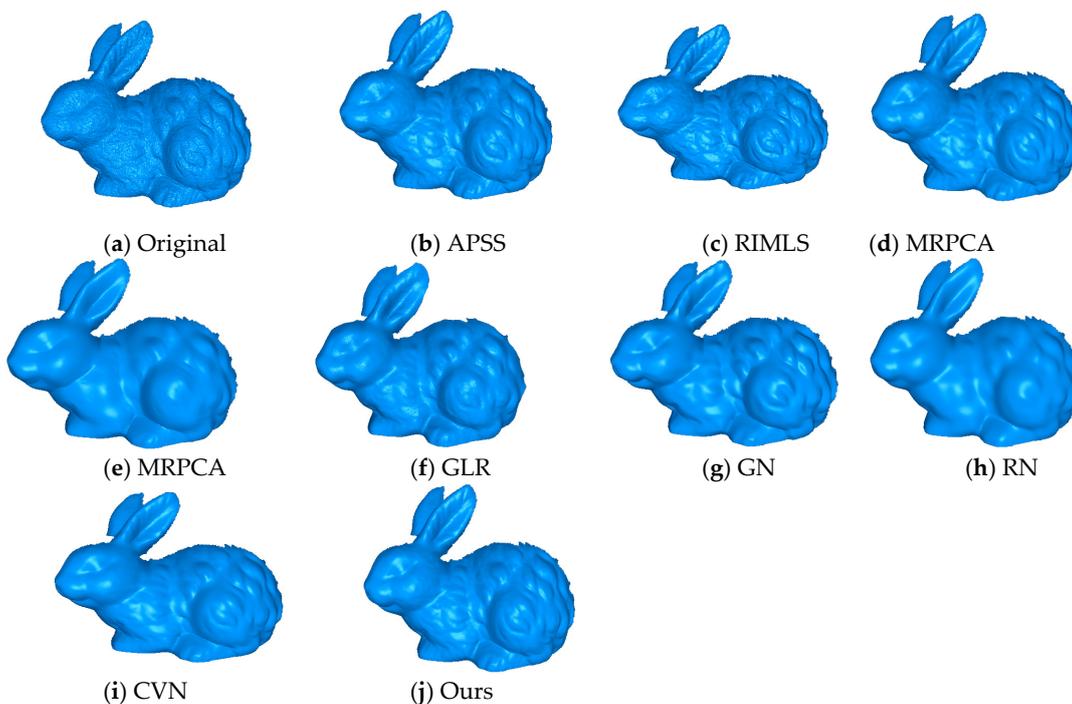


Figure 7. The rabbit model, with natural noise. We can see that the proposed method was able to preserve sharp features effectively when compared to the state-of-the-art methods. The surface was reconstructed using the ball pivoting algorithm.

GLR and RN preserved features, but lost some fine details as the eye and grooves in the ear were lost. GN and CVN preserved more detail than any of the other methods but they lost details in the eye and nose. MRPCA and the proposed method produced very similar results. Figure 8 shows the ball joint medical data. We can see that our method removed the noise, while details and sharp features were preserved, and the spherical shape was effectively smoothed. The APSS and RIMLS methods were not able to smooth the noise properly, and the resulting surfaces presented bumps. MRPCA, GN, RN, and GLR effectively removed the noise component but smoothed the sharp features. The EAR and CVN methods produced noise-free results, preserving the sharp features and smoothing the surfaces.

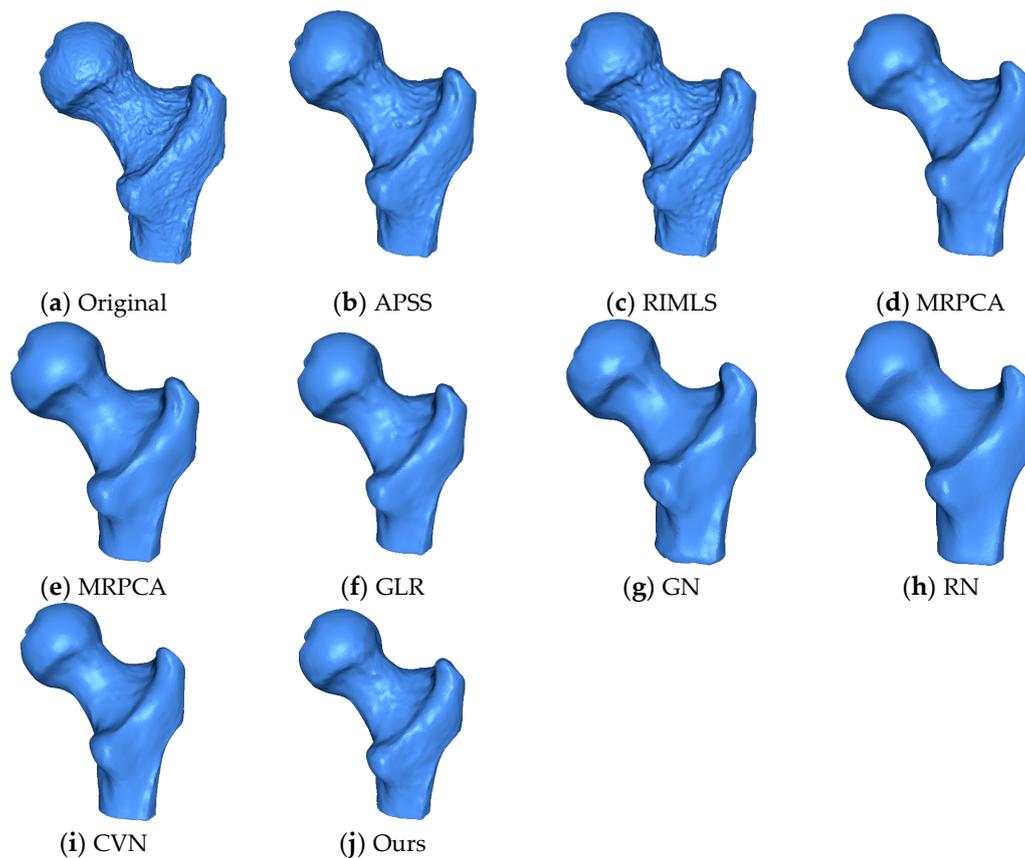


Figure 8. The ball joint model. We can see that the proposed method was able to preserve sharp features effectively when compared to the state-of-the-art methods. The surface was reconstructed using the ball pivoting algorithm.

5. Conclusions

In this paper, we proposed combining the L1 median filter and the L1 norm regularization for a point cloud-based denoising algorithm that preserves sharp features. The algorithm uses double sparsity modeling both in the fitting term and in the regularization term. The L1 median is insensitive to outliers and noise, while the L1 norm preserves the sharp features and smooths the surface. The combined L1-median and L1-norm cost function was optimized with an alternating minimization strategy using a proximal gradient and a descent iterative schema, allowing the implementation of a simple algorithm. The proposed method can handle models contaminated with Gaussian and impulse noise. High noise levels can produce erroneous results as they affect the normals estimation. Another issue is the concern with irregular point sampling models. While the irregular sampling remains low, the output of our algorithm produces good results; but when the point cloud density is highly irregular, the output quality decreases. To recover the sharp features, we introduce a border correction procedure that helps to correct edges and corners, preserving the models' original sharp features.

Experimental results reveal that our proposal can preserve sharp features when compared to previous point cloud denoising methods, and the algorithm is robust in denoising both synthetic and raw point scans. The method depends on some empirical parameters, σ_h and σ_d , defined by the user, and which we tuned manually to obtain the desired results. How to determine these parameters continues to be a challenge and is a direction we are going to investigate in our future research. The implementation of a global solution for the cost function is another issue to be examined in future work.

Author Contributions: All authors have read and agree to the published version of the manuscript. Conceptualization, E.L.; methodology, E.L, G.S.-T., and J.W.B.; software, E.L.; validation, E.L., G.S.-T., and

J.B.; formal analysis, E.L.; investigation, E.L.; writing—original draft preparation, E.L.; writing—review and editing, G.S.-T.; supervision, G.S.-T. and J.W.B.

Funding: This research was supported by the Administrative Department of Science and Technology of Colombia-COLCIENCIAS under the doctoral scholarship program COLCIENCIAS 2015-727 and by The Universidad del Magdalena under research fund FONCIENCIAS.

Acknowledgments: We would like to thank Sunil Kumar Yadav for supplying part of the 3D models used in this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sun, Y.; Schaefer, S.; Wang, W. Denoising point sets via L0 minimization. *Comput. Aided Geom. Des.* **2015**, *35–36*, 2–15. [[CrossRef](#)]
2. Avron, H.; Sharf, A.; Greif, C.; Cohen-Or, D. ℓ_1 Sparse reconstruction of sharp point set surfaces. *ACM Trans. Graph.* **2010**, *29*, 135:1–135:12. [[CrossRef](#)]
3. Mattei, E.; Castrodad, A. Point cloud denoising via moving RPCA. *Comput. Graph. Forum* **2017**, *36*, 123–137. [[CrossRef](#)]
4. Sarkar, K.; Bernard, F.; Varanasi, K.; Theobalt, C.; Stricker, D. Structured low-rank matrix factorization for point-cloud denoising. In Proceedings of the 2018 International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; pp. 444–453. [[CrossRef](#)]
5. Donoho, D.L. Compressed sensing. *IEEE Trans. Inf. Theor.* **2006**, *52*, 1289–1306. [[CrossRef](#)]
6. Candes, E.J.; Romberg, J.; Tao, T. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inf. Theory* **2006**, *52*, 489–509. [[CrossRef](#)]
7. Alexa, M.; Behr, J.; Cohen-Or, D.; Fleishman, S.; Levin, D.; Silva, C.T. Computing and rendering point set surfaces. *IEEE Trans. Vis. Comput. Graph.* **2003**, *9*, 3–15. [[CrossRef](#)]
8. Guennebaud, G.; Gross, M. Algebraic point set surfaces. *ACM Trans. Graph.* **2007**, *26*, 23-es. [[CrossRef](#)]
9. Öztireli, A.C.; Guennebaud, G.; Gross, M. Feature preserving point set surfaces based on non-linear kernel regression. *Comput. Graph. Forum* **2009**, *28*, 493–501. [[CrossRef](#)]
10. Lipman, Y.; Cohen-Or, D.; Levin, D.; Tal-Ezer, H. Parameterization-free projection for geometry reconstruction. In *ACM SIGGRAPH 2007 Papers*; ACM: New York, NY, USA, 2007. [[CrossRef](#)]
11. Huang, H.; Li, D.; Zhang, H.; Ascher, U.; Cohen-Or, D. Consolidation of unorganized point clouds for surface reconstruction. *ACM Trans. Graph.* **2009**, *28*, 1–7. [[CrossRef](#)]
12. Huang, H.; Wu, S.; Gong, M.; Cohen-Or, D.; Ascher, U.; Zhang, H. Edge-aware point set resampling. *ACM Trans. Graph.* **2013**, *32*, 9:1–9:12. [[CrossRef](#)]
13. Buades, A.; Coll, B.; Morel, J.-M. A non-local algorithm for image denoising. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 2, pp. 60–65. [[CrossRef](#)]
14. Dabov, K.; Foi, A.; Katkovnik, V.; Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans. Image Process.* **2007**, *16*, 2080–2095. [[CrossRef](#)] [[PubMed](#)]
15. Deschaud, J.-E.; Goulette, F. Point cloud non local denoising using local surface descriptor similarity. In *IAPRS, Vol. XXXVIII, Part 3A–Saint-Mandé, France, 1–3 September 2010*; Paparoditis, N., Pierrot-Deseilligny, M., Mallet, C., Tournaire, O., Eds.; Institut Geographique National, Laboratoire MATIS: Saint-Mandé, France, 2010; pp. 109–114.
16. Rosman, G.; Dubrovina, A.; Kimmel, R. Patch-collaborative spectral point-cloud denoising: Patch-collaborative spectral point-cloud denoising. *Comput. Graph. Forum* **2013**, *32*, 1–12. [[CrossRef](#)]
17. Schoenenberger, Y.; Paratte, J.; Vanderghynst, P. Graph-based denoising for time-varying point clouds. In Proceedings of the 2015 3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON), Lisbon, Portugal, 8–10 July 2015; pp. 1–4. [[CrossRef](#)]
18. Dinesh, C.; Cheung, G.; Bajic, I.V.; Yang, C. Fast 3D point cloud denoising via bipartite graph approximation & total variation. *arXiv* **2018**, arXiv:1804.10831.
19. Zeng, J.; Cheung, G.; Ng, M.; Pang, J.; Yang, C. 3D point cloud denoising using graph laplacian regularization of a low dimensional manifold model. *IEEE Trans. Image Process.* **2020**, *29*, 3474–3489. [[CrossRef](#)] [[PubMed](#)]

20. Duan, C.; Chen, S.; Kovacevic, J. Weighted multi-projection: 3d point cloud denoising with tangent planes. In Proceedings of the 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Anaheim, CA, USA, 26–29 November 2018; pp. 725–729. [\[CrossRef\]](#)
21. Leal, E.; Leal, N. Point cloud denoising using robust principal component analysis. In Proceedings of the First International Conference on Computer Graphics Theory and Applications, Setúbal, Portugal, 25–28 February 2006; pp. 51–58. [\[CrossRef\]](#)
22. Zheng, Y.; Li, G.; Wu, S.; Liu, Y.; Gao, Y. Guided point cloud denoising via sharp feature skeletons. *Vis. Comput.* **2017**, *33*, 857–867. [\[CrossRef\]](#)
23. Huang, H.; Wu, S.; Cohen-Or, D.; Gong, M.; Zhang, H.; Li, G.; Chen, B. L1-medial skeleton of point cloud. *ACM Trans. Graph.* **2013**, *32*, 65:1–65:8. [\[CrossRef\]](#)
24. Zheng, Y.; Li, G.; Xu, X.; Wu, S.; Nie, Y. Rolling normal filtering for point clouds. *Comput. Aided Geom. Des.* **2018**, *62*, 16–28. [\[CrossRef\]](#)
25. Yadav, S.K.; Reitebuch, U.; Skrodzki, M.; Zimmermann, E.; Polthier, K. Constraint-based point set denoising using normal voting tensor and restricted quadratic error metrics. *Comput. Graph.* **2018**, *74*, 234–243. [\[CrossRef\]](#)
26. Xu, L.; Wang, R.; Zhang, J.; Yang, Z.; Deng, J.; Chen, F.; Liu, L. Survey on sparsity in geometric modeling and processing. *Graph. Models* **2015**, *82*, 160–180. [\[CrossRef\]](#)
27. Ng, A.Y. Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance. In *Proceedings of the Twenty-First International Conference on Machine Learning*; ACM: New York, NY, USA, 2004; p. 78. [\[CrossRef\]](#)
28. Liu, Q.; Wang, J. L1-minimization algorithms for sparse signal reconstruction based on a projection neural network. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *27*, 698–707. [\[CrossRef\]](#)
29. Elad, M.; Starck, J.-L.; Querre, P.; Donoho, D.L. Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA). *Appl. Comput. Harmon. Anal.* **2005**, *19*, 340–358. [\[CrossRef\]](#)
30. Chen, S.; Donoho, D. Basis pursuit. In Proceedings of the 1994 28th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 31 October–2 November 1994; Volume 1, pp. 41–44. [\[CrossRef\]](#)
31. Donoho, D.L.; Elad, M.; Temlyakov, V.N. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Trans. Inf. Theory* **2006**, *52*, 6–18. [\[CrossRef\]](#)
32. Rudin, L.I.; Osher, S.; Fatemi, E. Nonlinear total variation based noise removal algorithms. *Physica D* **1992**, *60*, 259–268. [\[CrossRef\]](#)
33. Parikh, N.; Boyd, S. Proximal Algorithms. *Found. Trends Optim.* **2014**, *1*, 127–239. [\[CrossRef\]](#)
34. Mederos, B.; Velho, L.; de Figueiredo, L.H. Robust smoothing of noisy point clouds. In *Proc. SIAM Conference on Geometric Design and Computing*; SIAM: Philadelphia, PA, USA, 2003.
35. Bernardini, F.; Mittleman, J.; Rushmeier, H.; Silva, C.; Taubin, G. The ball-pivoting algorithm for surface reconstruction. *IEEE Trans. Vis. Comput. Graph.* **1999**, *5*, 349–359. [\[CrossRef\]](#)

