Alternately denoising and reconstructing unoriented point sets

Dong Xiao[a,c], Zuoqiang Shi[b,d], Bin Wang[a,c,*]

[a]*School of Software, Tsinghua University, Beijing, China*
[b]*Yau Mathematical Sciences Center, Tsinghua University, Beijing, China*
[c]*Beijing National Research Center for Information Science and Technology, Beijing, China*
[d]*Yanqi Lake Beijing Institute of Mathematical Sciences and Applications, Beijing, China*

ABSTRACT

We propose a new strategy to bridge point cloud denoising and surface reconstruction by alternately updating the denoised point clouds and the reconstructed surfaces. In Poisson surface reconstruction, the implicit function is generated by a set of smooth basis functions centered at the octnodes. When the octree depth is properly selected, the reconstructed surface is a good smooth approximation of the noisy point set. Our method projects the noisy points onto the surface and alternately reconstructs and projects the point set. We use the iterative Poisson surface reconstruction (iPSR) to support unoriented surface reconstruction. Our method iteratively performs iPSR and acts as an outer loop of iPSR. Considering that the octree depth significantly affects the reconstruction results, we propose an adaptive depth selection strategy to ensure an appropriate depth choice. To manage the oversmoothing phenomenon near the sharp features, we propose a λ-projection method, which means to project the noisy points onto the surface with an individual control coefficient λ_i for each point. The coefficients are determined through a Voronoi-based feature detection method. Experimental results show that our method achieves high performance in point cloud denoising and unoriented surface reconstruction within different noise scales, and exhibits well-rounded performance in various types of inputs. The source code is available at https://github.com/Submanifold/AlterUpdate.

1. Introduction

Point clouds are widely applied in a wide range of geometric applications. However, the real scanned point clouds obtained by the sensing technologies typically contain a certain amount of noise and outliers, which significantly reduces their shape representation capacities. When the input point clouds are unoriented (i.e., no consistently oriented normals are given), applying them for surface reconstruction becomes a challenging task. Point cloud denoising is a traditional solution for handling low-quality inputs, which has been extensively studied for more than two decades. The denoising techniques can be classified into several categories. A typical idea is to project the noisy point set onto the estimated local surfaces. However, existing projection-based methods either require consistently oriented normals for robust shape fitting (e.g., MLS-projection [1, 2]) or only considers the local shape properties within a specific neighbor scale (e.g., jet smoothing [3, 4]).

In recent years, researchers have made remarkable progress in unoriented surface reconstruction such as VIPSS [5], PGR [6] and iPSR [7]. Among them, iPSR iteratively performs screened Poisson surface reconstruction [8] and updates the point normals from the surface generated by the previous iteration. The algorithm terminates when approximating the iterative fixed point or reaching the maximum number of iterations. iPSR achieves high performance for clean inputs. However, the sample positions are fixed during the iterative process, reducing its performance in point clouds with large noise. Therefore, we wonder if the point positions can also be updated to achieve a higher reconstruction quality.

In this work, we bridge point cloud denoising and surface reconstruction by alternately updating the denoised point clouds

---

*Corresponding author at: School of Software, Tsinghua University, Beijing, China

e-mail: xiaod18@mails.tsinghua.edu.cn (Dong Xiao), zqshi@tsinghua.edu.cn (Zuoqiang Shi), wangbins@tsinghua.edu.cn (Bin Wang)

and the reconstructed surfaces. Our method acts as an outer loop of iPSR. Specifically, we project the noisy point set onto the surface generated by iPSR in each iteration. This idea is based on the following observations: in Poisson surface reconstruction, the reconstructed surface is generated by the isosurfacing of an implicit function spanned by a series of smooth basis centered at the octnodes. When a proper octree depth is selected, the reconstructed surface can be regarded as a good smooth approximation of the noisy point cloud. Furthermore, we notice that the depth selection will significantly affect the convergence of iPSR. Accordingly, we propose an adaptive depth selection method based on the convergence situation of iPSR, which is judged by the normal variations in the algorithm. To address the oversmoothing phenomenon in the sharp edges, we propose a λ-projection method by estimating the sharpness ratio and assigning an individual control coefficient for each point. The coefficients are determined using a Voronoi-based feature estimation algorithm [9].

From the point cloud denoising viewpoint, our method belongs to the projection-based category and iteratively projects the noisy point set onto the surface generated by iPSR with an adaptive depth selection strategy and a λ-projection method. From the unoriented surface reconstruction viewpoint, our method iteratively updates the point positions in the algorithm (instead of updating the normals only) and achieves considerable improvements for noisy inputs.

We qualitatively and quantitatively examine the efficacy of our method in point-wise noise, structured noise, outliers and real scanned noise. The experimental results indicate that our method achieves high performance in point cloud denoising and unoriented surface reconstruction tasks, and shows well-rounded performance within different shapes and noise scales. We also verify that our approach can not be substituted by simply adjusting some parameters in iPSR or other denoising and reconstructing approaches.

2. Related works

In this section, we provide a brief review of point cloud denoising and unoriented surface reconstruction techniques and focus on the works that are most relevant to our method.

2.1. Projection-based point cloud denoising

Point cloud denoising has been extensively studied for more than two decades due to its broad applications. Han et al. [10] provide a systematic review involving a wide variety of traditional approaches. A recent survey paper [11] introduces more up-to-date techniques including many interesting learning-based methods. Projection-based denoising is the most relevant category of our method with the main idea to project the noisy point cloud onto the local surfaces approximated by adjacent points. A commonly used local fitting technique is the Moving Least Squares (MLS) surfaces. The seminar work [1] defines a C^2 smooth surface through the projection procedure. However, the sharp features may be oversmoothened due to the smoothness property. Fleishman et al. [12] approximate the point set by the feature-preserving piecewise smooth surfaces.

Each piece fits the point shape on one side of the sharp edge. APSS [13] utilizes algebraic spheres to define the MLS surfaces, which significantly improves the quality near the high curvature regions. RIMLS [2] applies the robust local kernel regression to achieve feature preserving surface approximation. Majority of the above-mentioned techniques require consistently oriented normals as inputs. APSS supports both unoriented and oriented point fitting. However, normals are necessary for achieving the high-fidelity quality. Except for MLS projection, Cazals et al. [3, 4] estimate local shape properties through the Taylor expansion of the local height field. Then, a denoising technique can be realized by projecting the noisy points onto the approximated local surface. The aforementioned method is named as "jet smoothing", which does not require normals as inputs. However, the local fitting is only performed within a user specific neighbor scale.

Except for local surface fitting, some other approaches carry out point projection through an optimization manner. Lipman et al. [14] propose a parameterization-free locally optimal projection (LOP) operator, which generates a set of uniform distributed points to approximate the original shape. This technique is further improved by Huang et al. [15] through a robust repulsion term and the adaptive density weights. Liu et al. [16] suggest performing WLOP in an iterative manner. Preiner et al. [17] propose a continuous WLOP formulation based on the Gaussian mixture technique. This work significantly reduces the time resources of the WLOP. However, the above-mentioned methods may lack the feature preserving mechanisms and result in oversmoothing near the sharp edges. To address this issue, Huang et al. [18] propose to detect the sharp regions and carry out a point resampling operation near the sharp edges. Lu et al. [19] propose a GMM-inspired anisotropic projection strategy, which preserves the sharp features by considering filtered normals in the objective function.

2.2. Other point cloud denoising techniques

Projection is not the only means to carry out point cloud denoising. Digne et al. [20] and Zhang et al. [21] introduce the bilateral filters for 3D point set filtering. Their methods are mainly inspired by image denoising techniques. Avron et al. [22] and Sun et al. [23] propose the L1 and L0 optimization formulation for point set filtering based on the observation that many surfaces are piecewise smooth. Accordingly, the sparsity of the first order information can be applied to clean the point set. Agathos et al. [24] extend the Taubin smoothing techniques from the mesh denoising to point clouds, and implement a fast GPU version to support large scale data. Salman et al. [25] detect sharp features by analyzing the covariance matrices of Voronoi cells. Digne et al. [26] utilize an error metric based on optimal transport to achieve the feature-capturing reconstruction and simplification. Wang et al. [27] present a feature-preserving unoriented reconstruction pipeline and separate the entire task into several processes, including noise scale estimation, tangent plane detection, outlier removal, feature detection and noise smoothing. Liu et al. [28] propose a feature detection technique by the bi-tensor voting scheme. Chen et al. [29] propose to incorporate a repulsion term with the data

term and take into account both point distribution and feature preservation. RFEPS [30] introduces a novel feature-line detection and resample approach for CAD models.

The application of deep neural networks to geometric processing has achieved significant success in recent years due to the rapid advancement of 3D deep learning. Some interesting works learn the local shape properties through point-based 3D networks and obtain considerable results in point cloud denoising and unoriented surface reconstruction [31, 32, 33, 34, 35, 36, 37, 38]. However, the above-mentioned techniques typically require abundant training data to learn the shape priors.

2.3. Surface reconstruction from unoriented point sets

Despite a long research history with a considerable number of excellent studies [39, 40, 41, 5], surface reconstruction from unoriented point clouds remains a challenging problem. The main reason is that obtaining a globally consistent normal orientation is always a non-trivial task. Some remarkable studies have emerged in recent years and achieve high-fidelity reconstruction results for clean inputs [6, 7]. However, the point positions are fixed during the algorithm, limiting their effects on point clouds with a certain amount of noise. This is why we propose to fill this gap by updating the point positions iteratively. In contrast with these methods, DPSR [42] proposes a differentiable Poisson solver and updates the source point cloud through the backpropagation of the Chamfer loss. However, the computational process of DPSR relies on the GPU, which results in the high resource demand and the restricted grid resolution.

3. Method

3.1. Overview

Given an input 3D point cloud $P^{(0)} = \{p_1, p_2, ..., p_n\}$ with noise, our method iteratively updates the reconstructed surface and the denoised point cloud. The two main operators in our algorithm are the reconstruction operator and the projection operator. We apply the iterative Poisson surface reconstruction (iPSR) [7] to carry out unoriented surface reconstruction, which takes a raw point cloud P and a user-specific octree depth d as inputs, and generates a smooth surface S directly from the point positions. This operation can be denoted as follows:

$$S = f_{ipsr}(P, d). \tag{1}$$

Then, the noisy points are projected onto the surface. Instead of directly projecting the points onto S, we propose a λ-projection operator f_{proj}, which takes a point cloud P and a surface S as inputs and projects each point $p_i \in P$ on S with a control coefficient λ_i. If q_i is the closest point to p_i on S, then the projected point p'_i satisfies:

$$p'_i = (1 - \lambda_i)p_i + \lambda_i q_i. \tag{2}$$

This projection operation is denoted as follows:

$$P' = f_{proj}(P, S). \tag{3}$$

Generally, the alternative updating process of our method can be expressed as follows:

$$S^{(k)} = f_{ipsr}(P^{(k)}, d^{(k)}), \qquad P^{(k+1)} = f_{proj}(P^{(k)}, S^{(k)}). \tag{4}$$

Our method acts as an outer loop of iPSR. Fig. 1 shows the input point cloud $P^{(0)}$ and the reconstructed surfaces generated by our method from $S^{(0)}$(iPSR) to $S^{(4)}$. It can be seen that the mesh quality is improved and the noise is cleaned through the iteration.

In section 3.2, we will introduce the adaptive octree depth selection strategy to determine the depth $d^{(k)}$ in Equation 4. In section 3.3, we will introduce how the projection coefficient λ_i is selected for each point to handle the oversmoothing phenomenon near the sharp edges.

3.2. Adaptive depth selection strategy

When carrying out accurate and high-fidelity reconstruction by the traditional Poisson surface reconstruction approach [43, 8], the octree depth is always set to a large value for supporting detailed isosurfacing. In the original conception, a larger octree depth would lead to a better mesh quality, until reaching the maximum depth required for the geometric complexity. However, this situation is not the case when doing unoriented reconstruction for noisy point clouds using iPSR. Firstly, we will provide a brief review of the normal iteration process of iPSR as preliminaries.

iPSR. Given an unoriented point cloud P and a user specific octree depth d, iPSR firstly constructs an octree from P and generates a new sample set $\mathcal{S} = \{s_1, s_2, ..., s_m\}$ based on the octree nodes. Each sample s_i is assigned with an initialized normal n_i. The normals can be randomly initialized in the first iteration. We denote the set of normals as $\mathcal{N} = \{n_1, n_2, ..., n_m\}$. iPSR iteratively updates the sample normals $\mathcal{N}$ from the surface generated by the last iteration. Specifically, an intermediate surface is generated by inputting $\mathcal{S}$ and $\mathcal{N}$ to the screened Poisson surface reconstruction [8]. Then, the normal n_i of the sample s_i is updated by the average normal of the adjacent face list near s_i in the intermediate surface. The algorithm terminates when the iterative fixed point is approximated or the maximum iteration is reached. Then, a consistent and high quality triangular mesh can be obtained from the last iteration.

In the original work of iPSR, the octree depth is manually set to a fixed integer 10 in most experiments. However, we notice that choosing an appropriate depth for large noise inputs can bring about considerable advantages for this algorithm. When the octree depth is considerably large, the convergence of iPSR may encounter some challenges for large noise inputs and produce "lattice" surfaces, such as the middle column of Fig. 2. Reducing the octree depth can make the convergence easier and produce smoother surfaces. However, an extremely small depth will result in the loss of the shape details, as shown in the teapot mouth of Fig. 5 when doing reconstruction of depth 6. Therefore, an appropriate depth is critical for achieving a decent performance.

In this work, we propose an adaptive depth selection strategy based on the convergence situation (i.e., good or bad) of iPSR. Recall that iPSR updates the sample normals in an iterative manner. In their algorithm, the average of the top 0.1% normal variations v is calculated for each iteration. This value can be applied to judge if the algorithm has well converged. The iteration process of iPSR stops when $v < 0.175$, or the maximum

3

| Input | $S^{(0)}$(iPSR) | $S^{(1)}$ | $S^{(2)}$ | $S^{(3)}$ | $S^{(4)}$ |

Fig. 1. Given the input point cloud $P^{(0)}$, our method iteratively performs reconstruction and projection in every step. We show the reconstructed surfaces from $S^{(0)}$(iPSR) to $S^{(4)}$ of our method. The surface quality is improved through the iteration.

iteration 30 is reached. Accordingly, we also apply the normal variations to estimate the convergence situation of iPSR in the first reconstruction of our method. Specifically, we set the candidate depths to be $[d_{min}, d_{max}]$ and first perform iPSR by depth $d = d_{max}$. If iPSR converges and stops before reaching the maximum normal iteration 30, or the average normal variations of the last five iterations are less than 0.7, then iPSR converges well at this depth and $d^{(0)} = d$ is set. Otherwise, $d = d - 1$ and iPSR is performed again, until converges well or reaches the minimum depth $d = d_{min}$. This approach ensures that a proper depth is chosen at the first reconstruction.

The depth $d^{(0)}$ of the first reconstruction accounts for a critical role in our algorithm, and has been determined through the above-mentioned strategy. In the subsequent reconstructions, we can set an incremental depth list for each $d^{(0)} \in [d_{min}, d_{max}]$. For instance, the depth list is set to [6, 6, 7, 7, 8] for $d^{(0)} = 6$. The depth increases in the following stages because the noise magnitude becomes smaller through the alternative denoising and reconstructing process. Additional details will be specified in the experimental section. Once d_{min} and d_{max} are determined, and the depth list for each intermediate depth $d \in [d_{min}, d_{max}]$ is set, we can run the dataset in batches without manually adjusting the depth for each shape or reconstruction in Equation 4. Therefore, the depth selection strategy of our method is adaptive.

3.3. λ-projection method

In each iteration of our method with the input point cloud $P^{(k)}$ and the reconstructed surface $S^{(k)} = f_{ipsr}(P^{(k)}, d^{(k)})$, we project each point $p_i^{(k)} \in P^{(k)}$ onto the reconstructed surface $S^{(k)}$. However, if we simply project all the points onto the surface, then the sharp edges will be smoothed into rounded corners after several iterations. Accordingly, a λ-projection method is proposed to combat the issue of oversmoothing near the sharp edges. The main idea is to assign a projection coefficient $\lambda_i^{(k)}$ for each point $p_i^{(k)}$. If $q_i^{(k)}$ is the nearest point of $p_i^{(k)}$ in the surface $S^{(k)}$, then

the projected points $p_i^{(k+1)}$ satisfy:

$$p_i^{(k+1)} = (1 - \lambda_i^{(k)})p_i^{(k)} + \lambda_i^{(k)}q_i^{(k)}. \tag{5}$$

The coefficient $\lambda_i^{(k)}$ is different for each point and mainly depended on its sharpness degree. In this work, a Voronoi-based feature estimation method [9] is used to examine the sharpness degree of all the points. Given the point cloud $P^{(k)} = \{p_i^{(k)}\}_{i=1}^n$, [9] firstly computes the convolved Voronoi covariance measures from the neighborhood of each $p_i^{(k)}$. Then, the sharpness ratio $r_i^{(k)}$ can be computed from the eigenvalues of the covariance measure. The point whose sharpness ratio is larger than a certain threshold is considered as a feature point in the shape.

Once the sharpness ratios of the point set are calculated, the projection coefficient $\lambda_i^{(k)}$ can be determined from the sharpness ratio $r_i^{(k)}$. We set $\lambda_i^{(k)}$ to be a continuous function of $r_i^{(k)}$:

$$\lambda_i^{(k)} = 0.1 + 0.9 \times e^{-g(r_i^{(k)})}, \tag{6}$$

where

$$g(r_i^{(k)}) = \frac{(max\{r_i^{(k)} - c, 0\})^2}{\sigma^2}. \tag{7}$$

Here, c and σ are both user specific parameters, which represent the threshold and the standard deviation, respectively. Equation 6 and 7 demonstrate that $\lambda_i^{(k)} \in (0.1, 1.0]$. If $r_i^{(k)} < c$, then $p_i^{(k)}$ is not regarded as a feature point, $max\{r_i^{(k)} - c, 0\} = 0$, and $\lambda_i^{(k)} = 1.0$, indicating that point $p_i^{(k)}$ is projected onto the surface at $q_i^{(k)}$ ($p_i^{(k+1)} = q_i^{(k)}$). If $r_i^{(k)} > c$, then $p_i^{(k)}$ is regarded as a feature point with a large degree of sharpness. As the sharpness ratio $r_i^{(k)}$ increases, $\lambda_i^{(k)}$ will decrease, but not less than the basic offset 0.1. σ controls the decreasing speed of $\lambda_i^{(k)}$.

It should be noticed that the sharp edges are only considered when the noise scale is not that large. In section 3.2, we propose an adaptive depth selection method. Large noise will result in a low $d^{(0)}$. Here, we choose a depth d_{sharp} as threshold. When $d^{(k)} < d_{sharp}$, the sharp edges are not taken into account in this

Fig. 2. **The effectiveness of the adaptive depth selection strategy in large noise examples. If the depth is simply set to 8, as shown in the middle column, then the resulting meshes contain "lattice" structures. By contrast, utilizing our adaptive depth selection method brings about significant improvements in these examples.**

iteration because of the following reasons: 1) the input point cloud contains large noise; and 2) it is still in the early stage of the algorithm and the top priority of this iteration is to reduce the noise amplitude of the point cloud. When $d^{(k)} < d_{sharp}$, we set $\lambda_i^{(k)} = 0.5$ for all points, which not only filters the noise but also maintains the original shape. When $d^{(k)} \geq d_{sharp}$, Equation 6 and 7 are utilized to calculate the projection coefficient $\lambda_i^{(k)}$ for each point.

In Fig. 3, we show the reconstructed surface of a noisy *fandisk* model with and without the λ-projection method. The results indicate that the proposed λ-projection method is helpful in ameliorating the oversmoothing phenomenon near the sharp edges. For further improvements to this issue, we can carry out a point resample process near the sharp edges similar to EAR [18] and RFEPS [30].

The normal inputs of iPSR can be either random or manually initialized. Therefore, in the subsequent reconstructions of our method, we initialize the point normal of $p_i^{(k+1)}$ as the surface normal at $q_i^{(k)}$. In this way, the convergence speed of iPSR is significantly improved.

| Without λ-projection | With λ-projection |

Fig. 3. **Reconstructed mesh of a noisy *fandisk* model with and without the λ-projection strategy. The results indicate that the proposed λ-projection method can ameliorate the oversmoothing phenomenon near the sharp edges.**

4. Experiments

4.1. Overview

In the experiments, the minimum octree depth d_{min} is set to 6 and the maximum depth d_{max} is set to 8 in the adaptive depth selection strategy of our approach. We perform five alternative updating processes. Specifically, we use $P^{(5)}$ in Equation 4 as the denoised point cloud and $S^{(5)}$ as the reconstructed surface. If $d^{(0)}$ is calculated to be 6 based on the normal variations, then the depths in the subsequent five reconstructions $S^{(1)}$ to $S^{(5)}$ are set to $[6, 6, 7, 7, 8]$. If $d^{(0)} = 7$, then the subsequent depths are $[7, 7, 8, 8, 8]$. If $d^{(0)} = 8$, all the iterations are carried out within depth 8. The threshold d_{sharp} in the λ-projection is set to 8. Specifically, we only consider the sharp features when $d = 8$.

We choose the following non-data-driven and data-driven approaches as baselines and conduct qualitative and quantitative comparisons with these methods. These approaches belong to different categories.

RIMLS [2]: RIMLS applies the robust local kernels to the MLS surfaces. This approach requires consistently oriented normals as inputs. Here, the normals are estimated by PCA [44] and oriented by Hoppe et al. [45]. The RIMLS implementation of MeshLab [46] is used to conduct experiments.

WLOP [15]: WLOP is a locally optimal projection method for point cloud consolidation, which generates an evenly distributed particle set of the original point cloud. We use the implementation of WLOP in CGAL [47].

Jet smoothing [3, 4]: Osculating jet is an efficient polynomial fitting technique to estimate the local surface properties. This technique can be applied to denoise the point cloud through projecting the points onto the local surface shape. We use the "jet smoothing" implementation of CGAL.

Bilateral smoothing [47]: Bilateral filter is a useful tool for reducing the noise in point sets. We apply the CGAL implementation of the "bilateral smoothing" filter. It is worth noticing that the CGAL implementation of bilateral smoothing is combined with an edge preserving module based on the philosophy of EAR [18], which is introduced in the official documentation of CGAL. The bilateral smoothing of CGAL also requires normals as inputs. We estimate the normals by PCA and Hoppe et

Fig. 4. Reconstruction of our method in point weight 1.0 and reconstructions of the iPSR in different smaller point weights. The results indicate that our method cannot be substituted by simply adjusting the point weight of iPSR.

Fig. 5. Comparisons of iPSR at different depths with our method. The results indicate that our method cannot be substituted by simply adjusting the depth of iPSR.

al. [45] similar to RIMLS.

PointCleanNet [32]: PointCleanNet is a representative supervised learning approach for point cloud filtering. The network contains an outlier detector branch and a denoiser branch both based on PointNet [48]. The trained model provided by the authors are directly applied to conduct the qualitative and quantitative comparisons.

4.2. Ablation study with iPSR and PGR

Our method performs iPSR [7] in an iterative manner. Therefore, it is necessary to validate that our method cannot be substituted by simply adjusting some parameters in iPSR. The reconstruction results of the iPSR and ours in a noisy horse model are illustrated in Fig. 4. In Poisson surface reconstruction, "point weight" is an important parameter to control the degree of fitting the sample points. In our method, the point weight is set to 1.0 during the alternative updating process. If the point weight is reduced to 0.5, and iPSR is carried out only once, then the reconstructed mesh remains to be noisy. We also show the reconstruction of iPSR in zero point weight. Setting the point weight to zero directly in the iPSR may cause difficulties in algorithm convergence. Accordingly, we manually initialize the normals with a correct orientation. It can be seen that the zero weight iPSR cannot achieve an on par result with our method. Moreover, the ears of the horse are oversmoothened in the zero weight iPSR.

Fig. 5 shows the reconstruction of our method and iPSR within different octree depths of a noisy teapot model. In our method, the adaptive depth selection strategy is applied, and the depth $d^{(0)}$ is calculated as 8. We also show the iPSR reconstructions from octree depth 6 to 8. Although reducing the octree depth can make the reconstruction of iPSR smoother, some details are lost due to the low depth setting, for instance, the mouth of the teapot. Therefore, our method can not be substituted by simply adjusting the octree depth of iPSR.

Additionally, we compare our method with another recent unoriented reconstruction approach PGR [6]. Similar to iPSR, the point positions are fixed during the optimization in PGR. We set the parameter k_w in PGR to 64 and conduct experiments with four values of the parameter α: 2.0, 4.0, 8.0 and 16.0. The purpose of adjusting α is to manage the noisy inputs. Fig. 6 shows the experiment results where we add randomized Gaussian noise with a standard deviation of 1.0×10^{-2} in the input. We also colorize the error from the reconstructed mesh to the ground truth. It can be observed that increasing the value of α improves the noise adaptability of PGR and the smoothness of the generated surface. However, it also results in an increase in the distance between the generated surface and the ground truth mesh. On the other hand, our method iteratively updates the point positions, resulting in good visual effects and low distance errors between the generated surface and the ground truth.

4.3. Comparison with other denoising approaches

In this section, we compare our method with the other point cloud denoising techniques mentioned in Section 4.1 to examine the efficacy of our method. We use the *famous* dataset complied by a recent study Points2Surf [33], which contains 22 well-known shapes such as Armadillo, Stanford Bunny and Utah teapot. These shapes are normalized with the maximum axis length to be 1.0, and 10K to 100K points are randomly sampled from each shape. Then, the randomized Gaussian noise is added to each sample point of the shape. The standard deviations (STD) are selected within five different levels from 0.5×10^{-2} to 2.5×10^{-2}.

Parameter settings are typically crucial for non-data-driven denoising approaches. During the experiments, we set the parameter "filter scale" for RIMLS to 7. In WLOP, the "representative particle number" is set to 95% of the original point set, and the "neighbor radius" is set to 0.04. Jet smoothing has only one parameter "neighbor size". "Jet-small" is used to

Fig. 6. Comparisons of PGR in different α values with our method. The results indicate that our method produces good visual effects and has a low distance error between the generated surface and the ground truth.

Table 1. Quantitative comparisons on the famous dataset within different noise standard deviations (STD). We report the root mean square distance-to-surface (RMSD) of each method. The RMSD values are multiplied by 10^3.

Method/STD	0.5×10^{-2}	1.0×10^{-2}	1.5×10^{-2}	2.0×10^{-2}	2.5×10^{-2}
RIMLS	2.456	3.121	4.378	6.469	9.813
WLOP	3.433	3.669	4.032	4.799	6.420
Bilateral-small	2.064	2.895	4.298	6.202	8.548
Bilateral-large	3.630	4.061	4.626	5.305	6.447
Jet-small	1.994	2.752	3.837	5.350	7.248
Jet-large	2.807	3.351	4.012	4.892	6.072
PointCleanNet	2.518	3.362	4.267	5.516	7.210
Ours	**1.975**	**2.245**	**2.592**	**3.459**	**4.253**

represent the parameter setting 48 to the neighbor size. Meanwhile, "jet-large" is used to represent the parameter setting 96. Three parameters, namely, "neighbor size", "sharpness angle" and "iters", exist in bilateral smoothing. Increasing the sharpness angle will reduce the sharpness of the results. "Bilateral-small" is used to represent the parameter setting with neighbor size 24, sharpness angle 25 and iters 5, whilst "bilateral-large" is used to denote neighbour size 48, sharpness angle 50 and iters 5. Only one parameter "iters" exists in PointCleanNet, which is set to 5 during the experiments. In our method, the Poisson point weight is set to 1.0 in the datasets with a standard deviation less than 2.0×10^{-2}, and 0.5 otherwise. Parameters c and σ in Equation 7 are determined as follows. Firstly, the sharpness ratios of all points in a shape is sorted. Then, c is set to the value of the 90% position of the sorted array, and σ is set to $c/2$.

Firstly, we quantitatively compare the quality of the denoised point cloud in terms of the root mean square distance-to-surface (RMSD). If P is the denoised point cloud and P' is a densely sampled point cloud of the ground truth mesh. Then, the RMSD value can be calculated as follows:

$$RMSD(P, P') = \sqrt{\frac{1}{N} \sum_{p_i \in P} \min_{p_j \in P'} \|p_i - p_j\|_2^2}. \quad (8)$$

The quantitative results are shown in Table 1. The RMSD values are multiplied by 10^3. The results indicate that our method exhibits high performance in all the five noise scales from 0.5×10^{-2} to 2.5×10^{-2}. The qualitative comparisons are shown in Fig. 7. For jet smoothing and bilateral smoothing, we show the better performing from the small parameter setting and the large parameter setting of each shape. We annotate the RMSD value (also multiplied by 10^3) in the bottom of each point set and colorize the point to surface distance from the denoised point cloud to the ground truth surface. The five examples belong to the dataset of different noise scales. The results show that the generated point clouds of our method perform the lowest error. To make the comparisons more comprehensive, we also conduct an experiment in terms of the mean absolute distance-to-surface (MADS) of the denoised point cloud in Table 3, which is located in the supplementary material. The formulation for calculating the MADS are given as follows:

$$MADS(P, P') = \frac{1}{N} \sum_{p_i \in P} \min_{p_j \in P'} \|p_i - p_j\|_2. \quad (9)$$

Furthermore, we compare the quality of the reconstructed mesh amongst different approaches. For WLOP, bilateral

Fig. 7. Qualitative comparisons of the denoised point clouds. We annotate the RMSD value (multiplied by 10^3) at the bottom of each example and colorize the point to surface distance from the denoised point cloud to the ground truth mesh.

smoothing, jet smoothing and PointCleanNet, we use iPSR to generate the reconstructed surfaces from the denoised point clouds provided by their algorithms. The reason is that no consistently oriented normals are required for iPSR. The L1 Chamfer distance (CD) and the normal consistency (NC) are used to measure the mesh quality. Here, L1 CD means to apply the L1 sum toward all sample points, rather than utilizing the L1 distance of point positions. The formula for calculating the CD value is presented as follows:

$$CD(P, P') = \frac{1}{|P|} \sum_{p_i \in P} \min_{p_j \in P'} \|p_i - p_j\|_2 + \frac{1}{|P'|} \sum_{p_j \in P'} \min_{p_i \in P} \|p_j - p_i\|_2. \quad (10)$$

Where P and P' are point clouds uniformly sampled from the reconstructed mesh and the ground truth mesh. Normal consistency (NC) can also be named as the mesh cosine similarity, which calculates the average absolute normal dot product between the sample of the ground truth mesh and the nearest point in the sample of the reconstructed mesh.

Table 2 quantitatively compares the mesh quality in terms of CD and NC amongst different approaches. We show the model names, noise scales, CD and NC values in the table.

The CD values are also multiplied by 10^3. In each row, the red number represents the best value amongst all methods, and the blue number represents the second best. Our method achieves well-rounded performance. The meshes generated by bilateral smoothing show high NC values. However, the CD values are typically large, indicating that the filtered point positions are far from the ground truth. The results of jet smoothing exhibit low CD values. However, the NC values of the generated surfaces are not that satisfactory. This phenomenon is also shown in Fig. 15, located in the supplementary material, where we present the qualitative results of rows 3,5 and 9 in Table 2. We also colorize the error from the reconstructed mesh to the ground truth mesh with a color bar. The results of our method exhibits low error and high normal consistency. PointCleanNet achieves the lowest CD value in the "Liberty" model. But in reality, this shape is in the training set of PointCleanNet. To make the comparisons more comprehensive, we also provide an evaluation based on the F-score of the reconstructed mesh in Table 4, which is also located in the supplementary material. The F-score is calculated as follows: Firstly, we sample a point cloud P from the

Table 2. Quantitative comparisons of the reconstructed mesh quality. We show the Chamfer distance (CD) and the normal consistency (NC) of each method. The CD values are multiplied by 10^3. The red color is used to represent the best value amongst all approaches, and the blue is used to denote the second best. The results indicate that our method exhibits well-rounded performance.

Model	Noise	RIMLS		WLOP		Bilateral-small		Bilateral-large		Jet-small		Jet-large		PointCleanNet		Ours	
		CD↓	NC↑	CD↓	NC↑	CD↓	NC↑	CD↓	NC↑	CD↓	NC↑	CD↓	NC↑	CD↓	NC↑	CD↓	NC↑
tortuga	0.5×10^{-2}	3.98	0.986	4.33	0.983	4.04	0.986	5.40	0.984	3.48	0.986	3.51	0.986	3.47	0.983	3.47	0.987
Utah_teapot	0.5×10^{-2}	3.76	0.978	4.52	0.973	3.68	0.980	4.60	0.974	3.57	0.975	3.65	0.975	3.61	0.971	3.46	0.978
horse	1.0×10^{-2}	4.81	0.977	6.07	0.973	3.72	0.981	5.07	0.983	3.56	0.977	3.72	0.981	3.56	0.976	3.27	0.985
angle	1.0×10^{-2}	5.79	0.940	7.78	0.908	4.07	0.945	6.15	0.935	3.90	0.937	4.42	0.935	4.22	0.924	3.47	0.948
Armadillo	1.5×10^{-2}	9.23	0.932	6.84	0.930	6.27	0.941	9.40	0.932	5.91	0.935	6.56	0.932	5.87	0.929	5.31	0.943
xyzrgb_dragon	1.5×10^{-2}	14.97	0.829	9.75	0.814	5.92	0.860	8.72	0.839	5.96	0.852	7.20	0.844	6.59	0.837	5.30	0.864
hand	2.0×10^{-2}	28.94	0.793	12.23	0.875	7.56	0.856	10.75	0.872	6.54	0.868	8.38	0.880	8.12	0.860	7.01	0.879
serapis	2.0×10^{-2}	9.27	0.957	6.97	0.956	6.40	0.957	7.22	0.958	6.55	0.946	6.77	0.954	6.59	0.950	6.19	0.964
Liberty	2.5×10^{-2}	43.42	0.613	8.81	0.801	12.39	0.713	7.75	0.791	10.30	0.715	7.29	0.778	6.11	0.805	6.42	0.813
galera	2.5×10^{-2}	19.64	0.908	7.52	0.937	7.42	0.923	8.49	0.934	7.09	0.913	7.44	0.927	7.14	0.927	6.63	0.941

Input   Ours

Fig. 8. Denoised results of our method in misalignment point clouds. The results indicate that our approach effectively addresses this situation.

Input   RIMLS   WLOP

Bilateral smoothing   Jet smoothing   Ours

Fig. 9. The results amongst different methods of a point cloud contains both misalignment artifacts and outliers. Approximately 1K points are randomly added in the unit cube as outliers.

reconstructed mesh and a point cloud P' from the ground truth mesh. For each point p_i in P, we measure the distance between p_i and its nearest point p'_i in P'. We then measure the proportion of points in P that have a nearest distance below a threshold of 5×10^{-3}. This proportion is referred as the "precision". Swapping P and P' allows us to assess the "recall". The F-score, denoted as $F1$, is calculated as:

$$F1 = \frac{2 \times precision \times recall}{precision + recall}. \quad (11)$$

4.4. Managing different situations

In this section, we examine the ability of our method in managing various situations including misalignment, outliers, highly inconsistent sampling densities, noisy CAD-like models and real scanned point clouds.

Misalignment Misalignment is a typical noise category, especially for the point clouds obtained with a scanner. Here, we use the misalignment dataset provided by a recent benchmark [49] to examine the ability of our method to manage this type of structured noise. The point clouds are obtained with the Blensor simulator [50] by adding some perturbations to the camera extrinsics. Fig. 8 shows the misalignment inputs and the denoised point clouds generated by our method. The results indicate that the misalignment situation is efficiently managed by our method.

Outliers and highly inconsistent sampling densities Fig. 9 shows a point cloud including both misalignment artifacts and outliers. In this example, 1K outliers are randomly sampled within a unit cube and added to a point set with approximately 160K points. In Poisson surface reconstruction, a few outliers may not have significant influence on the global implicit function. Accordingly, our method performs a certain degree of robustness to outliers, whilst the other traditional approaches encounter some challenges to manage this situation. Jet smoothing and bilateral smoothing can filter out more outliers if a large neighbor size is utilized. For instance, setting the neighbor size of jet smoothing to 1024 can filter out most of the outliers in this case. However, the original shape will also be severely oversmoothed by such a large neighbor scale. Actually, this situation can also be managed by applying iPSR to the denoised point clouds provided by jet and bilateral smoothing, and then

Input   Ours

Fig. 10. Reconstructions of our method in point clouds with both noise and highly inconsistent sampling densities. Our approach effectively addresses this situation.

projecting the noisy points onto the surface. However, this operation aligns with the underlying philosophy of our method, which further confirms the validity of our concept.

We also conduct experiments to evaluate the performance of our method on point clouds with highly inconsistent sampling densities. In Fig. 10, the input point clouds contain randomized Gaussian noise with inconsistent sampling. Notably, the sampling density of one half of the shape is only 1/5 of the other half. The results demonstrate the effectiveness of our method in addressing this challenging issue.

Noisy CAD-like inputs The CAD-like inputs always include rich sharp features. We have demonstrated in Section 3.3 and Fig. 3 that the λ-projection method we proposed is helpful for alleviating the oversmoothing phenomenon near the sharp edges. In this section, we mainly focus on the comparisons. We set the parameters c and σ in Equation 7 to be 0.11 and 0.05, respectively. Fig. 11 presents the qualitative comparisons of our method with WLOP and jet smoothing in two CAD-like models. The results illustrate that our method has a fundamental edge reconstruction capability compared with traditional denoising approaches with the help of the λ-projection.

To enhance the preservation of sharp edges or feature lines, we can incorporate a point resampling process in the future. A recent method RFEPS [30] specifically addresses this issue in CAD-like input, with a particular focus on edge preservation. In Figure 12, we provide qualitative comparisons between our method and RFEPS. We utilize the source code provided by the authors and keep the default parameters. While RFEPS demonstrates favorable edge preservation, its applicability to general shapes may be limited. Furthermore, we observe that the edge point detection and sampling process of RFEPS could potentially be integrated into our alternative updating process in the

WLOP   Jet smoothing   Ours

Fig. 11. Qualitative comparisons of our method with WLOP and jet smoothing in the CAD-like models.

future. Enhancing the preservation of sharp features is one of our ongoing research directions.

Real scanned data In Fig. 13, we examine the ability of our approach in handling the real scanned point clouds and compare our method with iPSR (run only once) and jet smoothing. The data are provided by [49]. The surfaces of jet smoothing are generated by feeding the denoised point clouds to iPSR. Our method achieves decent performance. Jet smoothing even aggravates the noise near the thin structures, resulting in the failure of the iPSR to converge to the correct surface for the bowl model.

5. Conclusion

In this work, we propose an alternative denoising and reconstructing approach for unoriented point sets and performs iPSR in an iterative manner. An adaptive depth selection strategy is proposed to ensure that the reconstruction is carried out within an appropriate octree depth of iPSR. Moreover, we present a λ-projection method to handle the challenge of oversmoothing near the sharp edges during the iterative process. The experimental results show that our method exhibits high performance in point cloud denoising and surface reconstruction tasks and manages various situations.

The main drawback of our method is the high time consumption compared with traditional denoising techniques. Our method performs iPSR several times and acts as an outer loop of iPSR. iPSR itself is an iterative method for the screened Poisson surface reconstruction. Accordingly, our method requires about 8.5min on AMD Ryzen 5 5600H CPU @ 3.3GHz to denoise and reconstruct the bowl model of Fig. 13 with approximately 0.2M points. However, one time iPSR reconstruction is also required for other denoising approaches to generate a reliable surface when no consistently oriented normals are provided. For instance, combining jet smoothing and iPSR also takes about 105s for this model. Our method is faster than the learning-based PointCleanNet, which requires 27min in the RTX 2080Ti

Fig. 12. Qualitative comparisons of our method with RFEPS. While RFEPS performs well in CAD models, its applicability to general shapes may be limited.

GPU with 5 iters. Furthermore, the reconstruction complexity of screened Poisson surface reconstruction is a linear function of the point number due to the application of the conforming cascade Poisson solver. Therefore, the time complexity of our method is not large relative to the point number. Fig. 14 demonstrates our reconstruction of a noisy point cloud with 1M points. The denoising process is carried out within octree depth 10. Our method can manage this example in about one hour.

In addition to high time consumption, our current method is limited to a fixed number of iterations in the outer loop. In the future, we can enhance the adaptiveness of our method by terminating the denoising process dynamically with an evaluation mechanism. To further improve the sharp feature preservation, we can integrate the feature-line detection and resampling techniques to our method.

References

[1] Alexa, M, Behr, J, Cohen-Or, D, Fleishman, S, Levin, D, Silva, CT. Computing and rendering point set surfaces. IEEE Trans Vis Comput Graph 2003;9(1):3–15.

[2] Öztireli, AC, Guennebaud, G, Gross, MH. Feature preserving point set surfaces based on non-linear kernel regression. Comput Graph Forum 2009;28(2):493–501.

[3] Cazals, F, Pouget, M. Estimating differential quantities using polynomial fitting of osculating jets. Comput Aided Geom Des 2005;22:121–146.

[4] Cazals, F, Pouget, M. Jet_fitting_3: a generic C++ package for estimating the differential properties on sampled surfaces via polynomial fitting. ACM Trans Math Softw 2008;35(3):24:1–24:20.

[5] Huang, Z, Carr, N, Ju, T. Variational implicit point set surfaces. ACM Trans Graph 2019;38(4):124:1–124:13.

[6] Lin, S, Xiao, D, Shi, Z, Wang, B. Surface reconstruction from point clouds without normals by parametrizing the Gauss formula. ACM Trans Graph 2022;42(2):14:1–14:19.

[7] Hou, F, Wang, C, Wang, W, Qin, H, Qian, C, He, Y. Iterative Poisson surface reconstruction (iPSR) for unoriented points. ACM Trans Graph 2022;41(4):128:1–128:13.

[8] Kazhdan, MM, Hoppe, H. Screened Poisson surface reconstruction. ACM Trans Graph 2013;32(3):29:1–29:13.

[9] Mérigot, Q, Ovsjanikov, M, Guibas, LJ. Voronoi-based curvature and feature estimation from point clouds. IEEE Trans Vis Comput Graph 2011;17(6):743–756.

[10] Han, X, Jin, JS, Wang, M, Jiang, W, Gao, L, Xiao, L. A review of algorithms for filtering the 3D point cloud. Signal Process Image Commun 2017;57:103–112.

[11] Zhou, L, Sun, G, Li, Y, Li, W, Su, Z. Point cloud denoising review: from classical to deep learning-based approaches. Graph Model 2022;121:101140.

[12] Fleishman, S, Cohen-Or, D, Silva, CT. Robust moving least-squares fitting with sharp features. ACM Trans Graph 2005;24(3):544–552.

[13] Guennebaud, G, Gross, MH. Algebraic point set surfaces. ACM Trans Graph 2007;26(3):23.

[14] Lipman, Y, Cohen-Or, D, Levin, D, Tal-Ezer, H. Parameterization-free projection for geometry reconstruction. ACM Trans Graph 2007;26(3):22.

[15] Huang, H, Li, D, Zhang, H, Ascher, UM, Cohen-Or, D. Consolidation of unorganized point clouds for surface reconstruction. ACM Trans Graph 2009;28(5):176.

[16] Liu, S, Chan, K, Wang, CCL. Iterative consolidation of unorganized point clouds. IEEE Computer Graphics and Applications 2012;32(3):70–83.

[17] Preiner, R, Mattausch, O, Arikan, M, Pajarola, R, Wimmer, M. Continuous projection for fast L_1 reconstruction. ACM Trans Graph 2014;33(4):47:1–47:13.

[18] Huang, H, Wu, S, Gong, M, Cohen-Or, D, Ascher, UM, Zhang, HR. Edge-aware point set resampling. ACM Trans Graph 2013;32(1):9:1–9:12.

[19] Lu, X, Wu, S, Chen, H, Yeung, S, Chen, W, Zwicker, M. GPF: GMM-inspired feature-preserving point set filtering. IEEE Trans Vis Comput Graph 2018;24(8):2315–2326.

[20] Digne, J, de Franchis, C. The bilateral filter for point clouds. Image Process Line 2017;7:278–287.

[21] Zhang, F, Zhang, C, Yang, H, Zhao, L. Point cloud denoising with principal component analysis and a novel bilateral filter. Traitement du Signal 2019;36(5):393–398.

[22] 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.

[23] Sun, Y, Schaefer, S, Wang, W. Denoising point sets via L_0 minimization. Comput Aided Geom Des 2015;35-36:2–15.

[24] Agathos, A, Azariadis, PN, Kyratzi, S. Elliptic Gabriel Taubin smoothing of point clouds. Comput Graph 2022;106:20–32.

[25] Salman, N, Yvinec, M, Mérigot, Q. Feature preserving mesh generation from 3d point clouds. Comput Graph Forum 2010;29(5):1623–1632.

[26] Digne, J, Cohen-Steiner, D, Alliez, P, de Goes, F, Desbrun, M. Feature-preserving surface reconstruction and simplification from defect-laden point sets. J Math Imaging Vis 2014;48(2):369–382.

[27] Wang, J, Yu, Z, Zhu, W, Cao, J. Feature-preserving surface reconstruction from unoriented, noisy point data. Comput Graph Forum 2013;32(1):164–176.

[28] Liu, Z, Xiao, X, Zhong, S, Wang, W, Li, Y, Zhang, L, et al. A feature-preserving framework for point cloud denoising. Comput Aided Des 2020;127:102857.

[29] Chen, S, Wang, J, Pan, W, Gao, S, Wang, M, Lu, X. Towards uniform point distribution in feature-preserving point cloud filtering. Comput Vis Media 2023;9(2):249–263.

[30] Xu, R, Wang, Z, Dou, Z, Zong, C, Xin, S, Jiang, M, et al. RFEPS: reconstructing feature-line equipped polygonal surface. ACM Trans Graph 2022;41(6):228:1–228:15.

[31] Yu, L, Li, X, Fu, C, Cohen-Or, D, Heng, P. EC-Net: an edge-aware point set consolidation network. In: Computer Vision - ECCV 2018 - 15th European Conference; vol. 11211 of *Lecture Notes in Computer Science*. Springer; 2018, p. 398–414.

[32] Rakotosaona, M, Barbera, VL, Guerrero, P, Mitra, NJ, Ovsjanikov, M. PointCleanNet: learning to denoise and remove outliers from dense point clouds. Comput Graph Forum 2020;39(1):185–203.

[33] Erler, P, Guerrero, P, Ohrhallinger, S, Mitra, NJ, Wimmer, M.

Fig. 13. Reconstruction results on the real scanned point clouds. We compare our method with iPSR and jet smoothing. Our method effectively manages the real scanned noise.

Fig. 14. Our reconstruction of a noisy point cloud with 1M points.

Points2Surf: learning implicit surfaces from point clouds. In: Computer Vision - ECCV 2020 - 16th European Conference; vol. 12350 of *Lecture Notes in Computer Science*. Springer; 2020, p. 108–124.

[34] Boulch, A. ConvPoint: continuous convolutions for point cloud processing. Comput Graph 2020;88:24–34.

[35] Zhang, D, Lu, X, Qin, H, He, Y. Pointfilter: point cloud filtering via encoder-decoder modeling. IEEE Trans Vis Comput Graph 2021;27(3):2015–2027.

[36] Huang, A, Xie, Q, Wang, Z, Lu, D, Wei, M, Wang, J. MODNet: multi-offset point cloud denoising network customized for multi-scale patches. Comput Graph Forum 2022;41(7):109–119.

[37] Liu, Z, Zhan, S, Zhao, Y, Liu, Y, Chen, R, He, Y. PCDNF: revisiting learning-based point cloud denoising via joint normal filtering. CoRR 2022;abs/2209.00798. arXiv:2209.00798.

[38] de Silva Edirimuni, D, Lu, X, Shao, Z, Li, G, Robles-Kelly, A, He, Y. IterativePFN: true iterative point cloud filtering. CoRR 2023;abs/2304.01529. arXiv:2304.01529.

[39] Alliez, P, Cohen-Steiner, D, Tong, Y, Desbrun, M. Voronoi-based variational reconstruction of unoriented point sets. In: Proceedings of the Fifth Eurographics Symposium on Geometry Processing; vol. 257 of *ACM International Conference Proceeding Series*. Eurographics Association; 2007, p. 39–48.

[40] Mullen, P, de Goes, F, Desbrun, M, Cohen-Steiner, D, Alliez, P. Signing the unsigned: robust surface reconstruction from raw pointsets. Comput Graph Forum 2010;29(5):1733–1741.

[41] Giraudot, S, Cohen-Steiner, D, Alliez, P. Noise-adaptive shape reconstruction from raw point sets. Comput Graph Forum 2013;32(5):229–238.

[42] Peng, S, Jiang, C, Liao, Y, Niemeyer, M, Pollefeys, M, Geiger, A. Shape as points: a differentiable Poisson solver. In: Advances in Neural Information Processing Systems. 2021, p. 13032–13044.

[43] Kazhdan, MM, Bolitho, M, Hoppe, H. Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing; vol. 256 of *ACM International Conference Proceeding Series*. Eurographics Association; 2006, p. 61–70.

[44] Pearson, K. On lines and planes of closest fit to systems of points in space. Philosophical Magazine 1901;2(11):559–572.

[45] Hoppe, H, DeRose, T, Duchamp, T, McDonald, JA, Stuetzle, W. Surface reconstruction from unorganized points. In: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques. ACM; 1992, p. 71–78.

[46] Cignoni, P, Callieri, M, Corsini, M, Dellepiane, M, Ganovelli, F, Ranzuglia, G. MeshLab: an open-source mesh processing tool. In: Eurographics Italian Chapter Conference 2008, Salerno, Italy, 2008. Eurographics; 2008, p. 129–136.

[47] CGAL. 2023. URL: https://www.cgal.org/.

[48] Qi, CR, Su, H, Mo, K, Guibas, LJ. PointNet: deep learning on point sets for 3d classification and segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society; 2017, p. 77–85.

[49] Huang, Z, Wen, Y, Wang, Z, Ren, J, Jia, K. Surface reconstruction from point clouds: a survey and a benchmark. CoRR 2022;abs/2205.02413. arXiv:2205.02413.

[50] Gschwandtner, M, Kwitt, R, Uhl, A, Pree, W. Blensor: blender sensor simulation toolbox. In: Advances in Visual Computing - 7th International Symposium; vol. 6939 of *Lecture Notes in Computer Science*. Springer; 2011, p. 199–208.

Supplementary material of "Alternately denoising and reconstructing unoriented point sets"

Table 3. Quantitative comparisons on the famous dataset based on the mean absolute distance-to-surface (MADS) of each method. The MADS values are multiplied by 10^3.

Method/STD	0.5×10^{-2}	1.0×10^{-2}	1.5×10^{-2}	2.0×10^{-2}	2.5×10^{-2}
RIMLS	2.015	2.480	3.350	4.804	7.129
WLOP	2.684	2.874	3.126	3.508	4.162
Bilateral-small	1.721	2.214	2.974	4.059	5.517
Bilateral-large	2.954	3.269	3.634	3.999	4.590
Jet-small	**1.592**	2.159	2.855	3.799	5.011
Jet-large	2.126	2.599	3.081	3.636	4.365
PointCleanNet	1.979	2.592	3.108	3.633	4.196
Ours	1.624	**1.840**	**2.074**	**2.655**	**3.275**

Table 4. Quantitative comparisons of the reconstructed mesh quality based on the F-score with the threshold 5×10^{-3}. The red color is used to represent the best value amongst all approaches, and the blue is used to denote the second best.

Model	Noise	RIMLS	WLOP	Bilateral-small	Bilateral-large	Jet-small	Jet-large	PointCleanNet	Ours
tortuga	0.5×10^{-2}	0.996	0.971	0.997	0.919	0.999	0.999	0.999	0.999
Utah_teapot	0.5×10^{-2}	0.991	0.931	0.995	0.961	0.987	0.979	0.981	0.989
horse	1.0×10^{-2}	0.964	0.846	0.999	0.939	0.997	0.993	0.993	0.998
angle	1.0×10^{-2}	0.921	0.757	0.990	0.866	0.983	0.954	0.951	0.990
Armadillo	1.5×10^{-2}	0.732	0.855	0.886	0.616	0.909	0.859	0.908	0.949
xyzrgb_dragon	1.5×10^{-2}	0.550	0.678	0.883	0.660	0.875	0.775	0.833	0.914
hand	2.0×10^{-2}	0.206	0.413	0.761	0.522	0.835	0.678	0.727	0.803
serapis	2.0×10^{-2}	0.759	0.841	0.901	0.828	0.887	0.861	0.879	0.916
Liberty	2.5×10^{-2}	0.120	0.652	0.606	0.769	0.681	0.817	0.854	0.840
galera	2.5×10^{-2}	0.428	0.793	0.813	0.698	0.833	0.797	0.825	0.869

Fig. 15. Qualitative comparisons of the reconstructed mesh amongst different methods. The results indicate that our approach exhibits well-rounded performance.