Computers & Graphics 116 (2023) 64–72

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on CAD/Graphics 2023

Random screening-based feature aggregation for point cloud denoising

Weijia Wang [a], Wei Pan [b], Xiao Liu [a], Kui Su [c,*], Bernard Rolfe [a], Xuequan Lu [d,*]

[a] Deakin University, Australia
[b] OPT Machine Vision Tech Co., Ltd, Tokyo, Japan
[c] School of Computer and Computing Science, Hangzhou City University, Hangzhou, China
[d] La Trobe University, Australia

ARTICLE INFO

Article history:
Received 12 May 2023
Received in revised form 11 July 2023
Accepted 5 August 2023
Available online 9 August 2023

Keywords:
3D point cloud denoising
Geometry processing
Computer-aided design
Computer graphics
Deep learning

ABSTRACT

Raw point clouds captured by sensing devices are often contaminated with noise, which perturbs the fidelity of the original geometric information. Point cloud denoising is therefore an inseparable post-processing step, aiming to remove the noise in the point clouds. Existing point cloud denoising approaches are typically trained on datasets that have uniform point distributions and densities, making them unsuitable for effectively denoising point clouds with severe noise or irregular point distributions. In this paper, we introduce a novel random screening-based feature aggregation method for point cloud denoising. Our key insight is that merging features of dense and sparse points assists with enhancing the quality of point cloud denoising results. In specific, our approach involves randomly screening the features of local point patches and fusing richer geometric information of denser points into sparser point representations. Comprehensive experiments demonstrate that our method achieves state-of-the-art performance in the point cloud denoising task on both synthetic and real-world datasets.

1. Introduction

Point clouds are used in a vast range of applications, such as 3D scanning, computer-aided modelling, autonomous driving and object tracking. Nonetheless, raw point clouds are often contaminated by noise and outliers due to various factors, including the inherent noise from the environment and precision limitation of the sensing devices. As a result, the fidelity of the original geometric information is corrupted, which prevents point clouds from being effectively utilised. To tackle this issue, point cloud denoising is an indispensable processing step for the aforementioned applications.

Due to point clouds' irregular, unordered and discrete characteristics, point cloud denoising has been a non-trivial task over the decades. Traditional optimisation-based point cloud denoising methods such as [1–7] involve complex parameter tuning, which brings unnecessary burdens to users. Also, it might be difficult for such methods to achieve noise reduction and feature preservation at the same time. Moreover, some optimisation-based methods may become less robust to point clouds contaminated with severe noise [8]. In recent years, the development of deep learning has significantly advanced data-driven point cloud denoising [9–16]. Learning-based point cloud denoising methods have significantly reduced the burden of parameter tuning, and have demonstrated improved capability in feature preservation during the denoising process. Nonetheless, existing learning-based point cloud denoising methods may exhibit reduced robustness when dealing with point clouds that are contaminated with severe noise or have varying point distribution densities.

In this paper, we propose a novel learning-based point cloud denoising method which effectively addresses the aforementioned limitations. Our key insight is that utilising a random screening-based feature aggregation technique can significantly enhance the robustness of point cloud denoising performance. We design an encoder–decoder architecture for our network, and perform point cloud denoising in a local-based manner. For each noisy point, we query its local neighbouring points within a specific ball radius (i.e., as a local patch), and feed the patch into our graph-based point encoder to obtain their features. We then input the features into screening-based feature aggregation modules, in which we fuse the global features of each denser patch into the features of its screened (i.e., randomly sampled), sparser sub-patch. By doing so, we incorporate richer geometric information into sparser points, which helps with improving the network's robustness during denoising. We perform screening-based feature fusion twice to obtain the final aggregated patch

* Corresponding author.
E-mail addresses: wangweijia@deakin.edu.au (W. Wang), vpan@foxmail.com (W. Pan), xiao.liu@deakin.edu.au (X. Liu), suk@hzcu.edu.cn (K. Su), bernard.rolfe@deakin.edu.au (B. Rolfe), b.lu@latrobe.edu.au (X. Lu).

https://doi.org/10.1016/j.cag.2023.08.013

features. Then, we regress point-wise weights using the aggregated features and utilise them to predict the final displacement vector, representing the predicted position of the corresponding clean point. We train our network using a surface proximity loss in conjunction with a weighting loss to remove noise and minimise the negative impacts from irrelevant points, respectively. Experiments on synthetic and real-world datasets demonstrate that our method achieves state-of-the-art performance on point cloud denoising tasks. In short, our contributions are summarised as follows:

- We propose a novel learning-based point cloud denoising technique with a key insight of random feature screening.
- We design a feature aggregation module to incorporate geometric information of denser point patches into sparser ones, which assists with improving the network's robustness against noise and irregular point distributions.
- We conduct comprehensive experiments and demonstrate that our method shows competitive denoising performance on both synthetic and real-world datasets.

2. Related work

2.1. Optimisation-based point cloud denoising

Conventional point cloud denoising approaches are typically optimisation-based, with the aim of minimising objective functions that represent different geometric constraints. An early work by Alexa et al. [1] exploits moving least squares (MLS), which approximates clean surfaces from local point neighbourhoods to denoise point clouds. While the approach is effective, it tends to be fragile to more severe noise and output sub-optimal results. In order to improve the robustness and accuracy of the MLS method, Öztireli et al. [5] propose robust iterative moving least squares (RIMLS), which iteratively updates neighbour fitting weights to enhance the surface fitting process step by step. There are also other variants of MLS [2–4,17,18] in addition to the aforementioned work. From a similar perspective, Cazals et al. [19] propose a jet-fitting framework to approximate the surfaces that fit onto the noisy point surfaces. Nevertheless, the aforementioned methods require users to input proper parameters to perform effective optimisation, where parameter-tuning is a difficult and non-trivial task especially for users without a professional background.

In order to alleviate the pain in parameter tuning, Lipman et al. [20] propose locally optimal projection (LOP), which achieves denoising by projecting noisy points onto the local optimal tangent plane without the need for parameter-tuning. Huang et al. [21] extend the idea and propose a weighted local optimal projection (WLOP) method, which considers contribution during the iterative projection process. Preiner et al. [22] improve WLOP by introducing a Gaussian mixture that describes the density of input points, which also significantly improves the calculation efficiency. However, the aforementioned LOP-based methods may tend to blur sharp features during the denoising process. In order to address the issue, [23,24] are proposed with the aim of preserving sharp features, such as the edge or corner regions in the input shapes. In recent years, other types of point cloud denoising methods have emerged. For instance, Mattei et al. [25] propose moving robust principal component analysis (MRPCA), a sparsity-based method to denoise point clouds. Zeng et al. [26] propose graph Laplacian regularisation (GLR), a graph-based framework that employs the Laplacian regularisation approach to remove point cloud noise.

2.2. Learning-based point cloud denoising

The introduction of point cloud learning backbones, including PointNet [27], PointNet++ [28] and DGCNN [29], significantly accelerates the development of learning-based point cloud denoising. An early work is PCNet [9], which exploits a PointNet-based backbone to predict a displacement vector for each noisy point based on its local neighbourhood structure. The displacement vector is added to each noisy point as its predicted clean position. While PCNet is simple yet effective, its point-based convolutional backbone does not consider the relationships among local neighbouring points, which makes it less context-aware and less robust to severe noise. Later, Pistilli et al. [30] propose GPDNet, which utilises graph convolutional networks and demonstrate improved denoising results on point clouds contaminated with higher levels of noise. Hermosilla et al. [10] propose TotalDenoise, an unsupervised framework for point cloud noise removal. While the aforementioned methods demonstrate capabilities in denoising, they are likely to result in shape shrinkage. To alleviate the issue, Luo et al. [12] propose a differentiable manifold reconstruction method (denoted as DMR), which achieves point cloud denoising by downsampling noisy points to obtain a differentiable manifold and then resampling clean points. However, such a downsampling–resampling process may inevitably discard geometric details in the input point clouds. In order to enhance geometric feature preservation, Zhang et al. [13] propose PointFilter that utilises normals to help with sharp feature preservation during the point cloud denoising process. Similarly, Lu et al. [31] achieve feature-preserving point cloud denoising via a 2-step approach, including a feature classification step and a denoising step, respectively. Recently, Luo et al. [14] propose score-based denoising (abbreviated as Score) which exploits gradient ascending to remove the noise component in the input point clouds, and Mao et al. [32] utilise normalising flows to achieve denoising. To exploit the advantages of multi-scale patches, Huang et al. [15] propose MoDNet that samples patches in multiple sizes and regresses multiple offset vectors to guide point cloud denoising. While such an approach extracts richer geometric information about point local neighbourhoods, it may still become less robust to severe noise or irregular sampling density.

3. Method

Given a point cloud contaminated with noise, we aim to remove the noise component and restore the clean point positions. We define a noisy point cloud $\hat{\mathcal{P}}$ as

$$\hat{\mathcal{P}} = \mathcal{P} + \mathcal{N}, \tag{1}$$

where $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_i \mid i = 1, 2, \ldots, T\}$, T is the total number of the points within the noisy point cloud $\hat{\mathcal{P}}$, $\mathcal{P}$ is the corresponding clean point cloud that represents the ground-truth surface information, and $\mathcal{N}$ is the noise component. We aim to remove noise $\mathcal{N}$ in a patch-based manner, which indicates that each denoised point is dependent on its nearby points within a local neighbourhood. In this paper, we regard the noise component $\mathcal{N}$ as a set of displacement vectors that is applied to each point within the noisy point cloud $\hat{\mathcal{P}}$, and we aim to regress such displacement vectors to restore the positions of the original clean points.

The overview of our network architecture is shown in Fig. 1. In Section 3.1, we introduce the preliminaries for the denoising task, followed by a description of our network structure in Section 3.2. Finally, in Section 3.3, we introduce the loss functions for network training.

3.1. Preliminaries

For any point $\hat{\mathbf{p}}_i$ in a noisy point cloud $\hat{\mathcal{P}}$, we define a local patch $\hat{\mathcal{P}}_i$ that is centred at point $\hat{\mathbf{p}}_i$ as

$$\hat{\mathcal{P}}_i = \{\hat{\mathbf{p}}_j \mid \left\|\hat{\mathbf{p}}_j - \hat{\mathbf{p}}_i\right\| < r,\ \hat{\mathbf{p}}_j \in \hat{\mathcal{P}}\}, \tag{2}$$

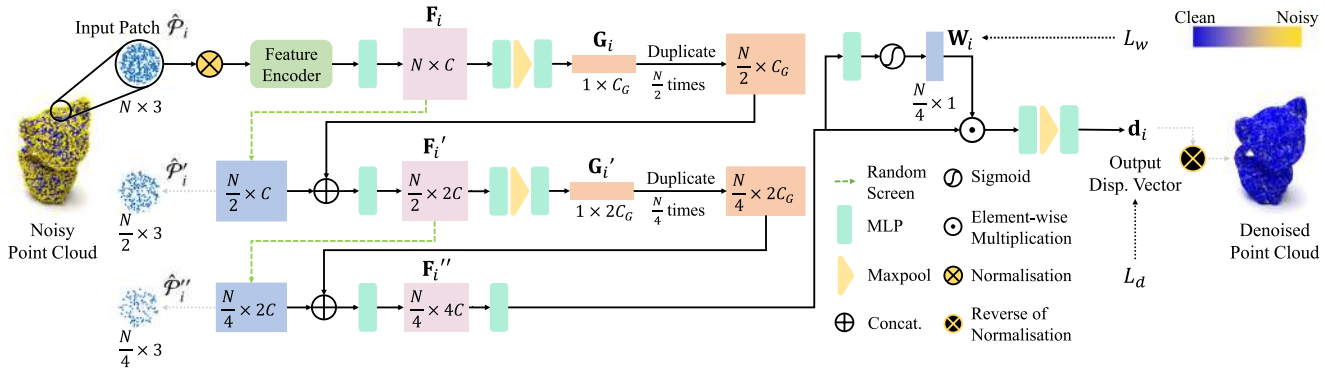

Fig. 1. The pipeline of our approach. For each input patch, we first normalise it and extract its features using a graph-based feature encoder. Then, we aggregate the features using our random screening strategy. Specifically, we fuse the global feature of the denser point patch into the features of the sparser patch. Finally, we regress point-wise weights from the aggregated point features and utilise them for clean point prediction (as displacement vectors), and we utilise our surface proximity loss L_d and weight loss L_w to train our network. The predicted points go through a reversed normalisation process and form the final denoised point cloud during testing.

where r represents the radius of a ball centred at point $\hat{\mathbf{p}}_i$, and $\hat{\mathbf{p}}_j$ is any neighbouring point within the specific radius r. r is calculated based on the bounding box diagonal of the noisy point cloud $\hat{\mathcal{P}}$. During training, we also require the corresponding ground-truth patch $\mathcal{P}_i$ in the clean point cloud $\mathcal{P}$, which is defined as

$$\mathcal{P}_i = \{\mathbf{p}_j \mid \left\|\mathbf{p}_j - \hat{\mathbf{p}}_i\right\| < r,\ \mathbf{p}_j \in \mathcal{P}\}, \tag{3}$$

where we query the ground-truth patch points using the noisy point $\hat{\mathbf{p}}_i$ and the radius r. Nonetheless, it is worth noting that raw patches could be of various sizes and contain arbitrary degrees of freedom (i.e., rigid transformations, including translations and rotations), making it infeasible for neural networks to effectively learn features from the patches [13,33]. In order to alleviate the issue, we follow the patch normalisation process in prior work [9,13,33]. In specific, we first translate the noisy patch to its origin and normalise its size to a unit sphere, i.e., the process is defined as $\hat{\mathcal{P}}_i = (\hat{\mathcal{P}}_i - \hat{\mathbf{p}}_i)/r$. Similarly, we perform the same operation for the clean patch, where the process is denoted as $\mathcal{P}_i = (\mathcal{P}_i - \hat{\mathbf{p}}_i)/r$. Then, for each noisy patch $\hat{\mathcal{P}}_i$, we compute a 3D rotation matrix $\mathbf{R}$ by performing principal component analysis (PCA) decomposition on it. We multiply $\hat{\mathcal{P}}_i$ by $\mathbf{R}$ in order to align it to the canonical space and apply the same rotation to the corresponding ground-truth patch $\mathcal{P}_i$. As a result, we remove the unnecessary degrees of freedom in the patches, making them stay invariant under arbitrary rigid transformations. For the rotation matrix $\mathbf{R}$, we use its inverse matrix $\mathbf{R}^{-1}$ to map the denoised point back to its original position during the testing phase.

As the patches are queried within a specific radius, the number of points within each patch may be inconsistent with each other, preventing them from being formed into batches and fed into the neural network. An effective way to address this issue is performing point sampling for the input patches [9,13,15] using a pre-defined patch size N. We pad extra points for patches containing fewer points than N, and downsample patches with more than N points. In both cases, we shuffle the points within each sampled patch such that the order of the points is randomised. Following prior work [13], we empirically set $N = 500$ and $r = 5\%$ of the input shape's bounding box diagonal during our experiments.

3.2. Random screening-based feature aggregation network

Our network adopts an encoder–decoder structure, which takes in pre-processed local patches and encodes them as feature matrices. Then, it performs random screening-based feature aggregation and finally decodes the features as displacement vectors, which represent the corresponding positions of predicted clean points.

Point Encoding. Prior work such as [9,13] typically exploit point convolutional network based on PointNet [27], which ignores the relationships between each point and its local neighbouring points that potentially contribute to the feature encoding process [30]. Inspired by this, we adopt the technique in [34], where we aggregate the local neighbourhood's feature for each input point in $\hat{\mathcal{P}}_i$ and refine the features via a series of Dense Block modules [35]. Finally, we perform max-pooling to obtain a point-wise feature matrix for each patch $\hat{\mathcal{P}}_i$. We denote this matrix as $\mathbf{F}_i$, an $N \times C$ matrix where C represents the number of encoding dimensions.

Random Screening-based Feature Aggregation. In order to fully exploit point position information, a common approach is fusing the features of larger patches into smaller sub-patches, which is typically achieved by tensor operations such as concatenation [34,36]. However, this technique is typically adopted when the training data has a uniform point resolution, which may cause the network to become less robust on points with varying sampling densities. Inspired by this, we propose a novel feature aggregation technique where we fuse the global feature of a denser patch with the features of its randomly-sampled sub-patch.

Given the input feature matrix $\mathbf{F}_i$, we extract its global permutation-invariant feature and denote it as $\mathbf{G}_i$, which is a $1 \times C_G$ vector with C_G dimensions. Meanwhile, we randomly screen the patch feature $\mathbf{F}_i$ and keep half of the point features in it, which forms an $\frac{N}{2} \times C$ matrix. As the corresponding patch of $\mathbf{F}_i$ is $\hat{\mathcal{P}}_i$, we denote the screened feature's corresponding patch as $\hat{\mathcal{P}}'_i$, which contains $\frac{N}{2}$ points and $\hat{\mathcal{P}}'_i \subset \hat{\mathcal{P}}_i$ (i.e., all points in $\hat{\mathcal{P}}'_i$ are included in $\hat{\mathcal{P}}_i$). We duplicate the global feature vector $\mathbf{G}_i$ for $\frac{N}{2}$ times and obtain an $\frac{N}{2} \times C_G$ matrix. Subsequently, we concatenate the duplicated global feature and the sub-patch feature together and feed the concatenated feature matrix into a multi-layer perceptron (MLP) layer, from which we obtain the output feature $\mathbf{F}'_i$, an $\frac{N}{2} \times 2C$ matrix where $2C$ is the encoding dimension.

Then, we repeat the aforementioned random screening procedure for $\mathbf{F}'_i$, for which we denote the corresponding screened sub-patch as $\hat{\mathcal{P}}''_i$ with $\frac{N}{4}$ points. We also extract the global feature $\mathbf{G}'_i$ of matrix $\mathbf{F}'_i$, duplicate it for $\frac{N}{4}$ times and concatenate it with the screened matrix to form $\mathbf{F}''_i$, which is an $\frac{N}{4} \times 4C$ feature matrix. As a result, the output matrix $\mathbf{F}''_i$ incorporates richer geometric information from denser patches in fewer point features. Such a cascaded feature aggregation structure is also more effective

than a single feature aggregation layer, which is discussed in our ablation study in Section 4.4.

Output Module. Prior work typically performs uniform regression from each patch's feature matrix to obtain a displacement vector [9,13]. Nevertheless, not all points within a patch contribute equally to the denoised result. In order to minimise the impacts from irrelevant points, we regress point-wise weights from the input patch feature and exploit them for displacement regression. Inspired by [37], we let points that are closer to the tangent plane of $\hat{\mathcal{P}}_i$'s central point gain higher weights. The tangent plane can be derived from the ground-truth normal $\mathbf{n}_p$, which belongs to the closest clean point to the noisy query point $\hat{\mathbf{p}}_i$. As shown in Fig. 1, we denote the regressed point-wise weights as $\mathbf{W}_i = \{w_i \mid i = 1, \ldots, \frac{N}{4}\}$, which is calculated based on the sub-patch $\hat{\mathcal{P}}_i''$. We then weigh the feature matrix $\mathbf{F}_i''$ and conduct max-pooling, from which we regress the displacement vector $\mathbf{d}_i$ for input point $\hat{\mathbf{p}}_i$. The process is defined as:

$$\mathbf{d}_i = \phi(MAX(\varphi(\mathbf{W}_i \odot \mathbf{F}_i''))), \tag{4}$$

where ϕ and φ are MLPs, $\odot$ represents element-wise multiplication, and *MAX* denotes the max-pooling operation. The displacement vector $\mathbf{d}_i$ is added to the input noisy point $\hat{\mathbf{p}}_i$ in order to approximate the clean point's position, where we denote the predicted denoised point as $\bar{\mathbf{p}}_i$.

3.3. Loss functions

The denoised point should be as close to the underlying surface as possible. Meanwhile, it needs to be located as close to the input patch's centre as possible, in order to avoid the clustering issue [9,13]. Following PCNet [9], we formulate our surface proximity loss function L_d with two terms: a distance-to-surface term and a repulsion term, which correspond to the aforementioned two objectives, respectively. L_d is defined as

$$L_d = \alpha \min_{\mathbf{p}_j \in \mathcal{P}_i} \left\| \bar{\mathbf{p}}_i - \mathbf{p}_j \right\|_2^2 + (1-\alpha) \max_{\mathbf{p}_j \in \mathcal{P}_i} \left\| \bar{\mathbf{p}}_i - \mathbf{p}_j \right\|_2^2, \tag{5}$$

where the $\min \|\cdot\|_2^2$ term attempts to minimise the distance of the denoised point to the clean surface, while the $\max \|\cdot\|_2^2$ term forces the denoised point to stay at the centre of the clean patch. $\mathbf{p}_j$ represents any clean point belonging to the ground-truth patch $\mathcal{P}_i$. Both terms are controlled by a trade-off factor α. Following [9], we empirically set the factor α to 0.99.

We also introduce a weighting loss to train the weight regression component in our network. Following the definitions in prior literature [34,37], we first define the distance of each point in patch $\hat{\mathcal{P}}_i''$ to the patch's tangent plane as

$$d_m = |\mathbf{p}_m \cdot \mathbf{n}_p|, \tag{6}$$

where $\mathbf{p}_m \in \hat{\mathcal{P}}_i''$, $\mathbf{n}_p$ is the aforementioned ground-truth normal. The weight loss L_w is then defined as

$$L_w = \frac{1}{M} \sum_{m=1}^{M} (\hat{w}_m - w_m)^2, \tag{7}$$

where w_m is the predicted point-wise weight. $\hat{w}_m = \exp(-{d_m}^2/\sigma)$, $\sigma = \max(0.0025, \beta * (\frac{1}{M}\sum_{m=1}^{M} {d_m}^2))$, β is empirically set to 0.3, and $M = \frac{N}{4}$ which is equivalent to the size of the screened patch $\hat{\mathcal{P}}_i''$. By doing so, points near the tangent plane of the patch's centre gain higher weights and thus contribute more to the denoised result.

Our overall loss function combines Eqs. (5) and (7) and is defined as

$$L = L_d + \lambda L_w, \tag{8}$$

where we empirically set λ to 1.0.

4. Experiments and results

4.1. Implementation and training details

We implement our network using PyTorch and conduct the experiments on a single NVIDIA RTX 3080 GPU with 10 GB memory. We set the training batch to 64, and use a single Adam optimiser during training. The initial learning rate is 0.0001 and we multiply the learning rate with a 0.2 factor every 20,000 batch iterations.

For the training phase, we follow the prior work [14] and adopt PUNet's [38] training dataset. The dataset contains 40 different mesh shapes, as well as the corresponding ground-truth point clouds in 3 resolutions (10,000, 30,000 and 50,000 points per shape). All points are sampled from the surfaces of the mesh shapes, such that the points depict the ground-truth geometric information. In addition, as our weight loss L_w requires ground-truth normals, we query each point's corresponding ground-truth normal on the mesh surface, and assign the normal to the point. During the training process, we follow [14] and add noise to the clean points on the fly. In specific, we randomly perturb each point cloud with Gaussian noise, which comes with a standard deviation from 0.5% to 2.0% of the shape's bounding sphere radius. Then, we randomly select a point on each noisy shape as the seed point, which is used for querying the noisy and clean patches.

4.2. Quantitative results

PUNet Dataset. We first demonstrate quantitative performance on PUNet's test dataset. It contains point clouds sampled from 20 mesh shapes at two resolutions settings: *sparse*, where each shape contains 10,000 points, and *dense*, where each shape contains 50,000 points. Each resolution has three levels of Gaussian noise augmentation, with a standard deviation of 1%, 2% and 3% of the bounding sphere's radius for each shape, respectively. Following prior work [14], we denoise for 1 iteration for shapes contaminated with 1% and 2% noise, and denoise for 2 iterations for shapes contaminated with 3% noise.

We evaluate the results on two metrics: Chamfer distance (CD) and Point-to-mesh distance (P2M). CD measures the distances of the points in the denoised point cloud to the ground-truths, and P2M measures the accuracy of the points describing the underlying surface [32]. The CD error metric L_{CD} is formulated as

$$L_{CD}(\bar{\mathcal{P}}, \mathcal{P}) = \frac{1}{|\bar{\mathcal{P}}|} \sum_{\bar{\mathbf{p}} \in \bar{\mathcal{P}}} \min_{\mathbf{p} \in \mathcal{P}} \|\bar{\mathbf{p}} - \mathbf{p}\| + \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \min_{\bar{\mathbf{p}} \in \bar{\mathcal{P}}} \|\mathbf{p} - \bar{\mathbf{p}}\|, \tag{9}$$

where $\bar{\mathcal{P}}$ represents for the denoised point cloud, $\bar{\mathbf{p}}$ represents each denoised point in it, and $\mathcal{P}$ stands for the ground-truth point cloud. The P2M error metric L_{P2M} is defined as

$$L_{P2M}(\bar{\mathcal{P}}, \mathcal{M}) = \frac{1}{|\bar{\mathcal{P}}|} \sum_{\bar{\mathbf{p}} \in \bar{\mathcal{P}}} \min_{f \in \mathcal{M}} d(\bar{\mathbf{p}}, f) + \frac{1}{|\mathcal{M}|} \sum_{f \in M} \min_{\bar{\mathbf{p}} \in \bar{\mathcal{P}}} d(\bar{\mathbf{p}}, f), \tag{10}$$

where $\mathcal{M}$ is the corresponding mesh of the ground-truth point cloud $\mathcal{P}$, and f represents each triangular face in mesh $\mathcal{M}$ that is the nearest to each denoised point $\bar{\mathbf{p}}$.

The quantitative results are demonstrated in Table 1. Our method achieves the minimum CD and P2M results compared with other methods in most cases, and achieves the minimum CD and P2M values on average. Fig. 2 shows the visualised point-to-mesh proximity and P2M error of the denoising results of 2 noisy shapes, Kitten and Horse, where each shape contains 50k points and is contaminated by 2% and 3% noise. Points in yellow denote that they are further from the ground-truth surface, and dark blue points are closer to the surface. The results demonstrate that

Table 1
Quantitative performance on PUNet dataset. The CD, P2M results are both multiplied by 10^4. The best results are marked in bold, and the second best results are marked with underline.

Density	10k						50k						Average	
Noise level	1%		2%		3%		1%		2%		3%			
Method	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Jet [19]	3.47	1.20	4.83	1.89	6.15	2.86	0.82	0.19	2.38	1.35	5.64	4.16	3.88	1.94
GPF [24]	3.28	1.17	4.18	1.54	5.37	2.75	0.76	0.23	2.04	0.94	3.82	2.87	3.24	1.58
MRPCA [25]	3.14	1.01	3.87	1.26	5.13	2.03	0.70	0.12	2.11	1.06	5.64	3.97	3.43	1.58
GLR [26]	2.79	0.92	3.66	1.14	4.84	2.08	0.71	0.18	1.61	0.85	3.74	2.67	2.89	1.31
PCNet [9]	3.57	1.15	7.54	3.92	13.0	8.92	0.95	0.27	1.56	0.62	2.32	1.32	4.82	2.70
GPDNet [30]	3.75	1.33	8.00	4.50	13.4	9.33	1.97	1.09	5.08	3.84	9.65	8.14	6.98	4.71
PointFilter [13]	2.86	0.75	3.97	1.30	4.94	2.14	0.82	0.24	1.46	0.77	2.25	1.44	2.72	1.11
DMR [12]	4.54	1.70	5.04	2.13	5.87	2.86	1.17	0.46	1.58	0.81	2.45	1.54	3.44	1.58
Score [14]	2.52	0.46	3.68	1.08	4.69	1.94	0.71	0.15	1.28	0.57	1.92	1.05	2.47	0.88
MoDNet [15]	**2.37**	**0.35**	3.71	1.00	5.11	2.00	**0.69**	**0.11**	1.04	0.34	1.48	0.67	2.40	0.75
Ours	2.38	0.38	**3.55**	**0.90**	**4.48**	**1.59**	0.70	0.12	**1.00**	**0.31**	**1.39**	**0.61**	**2.25**	**0.65**

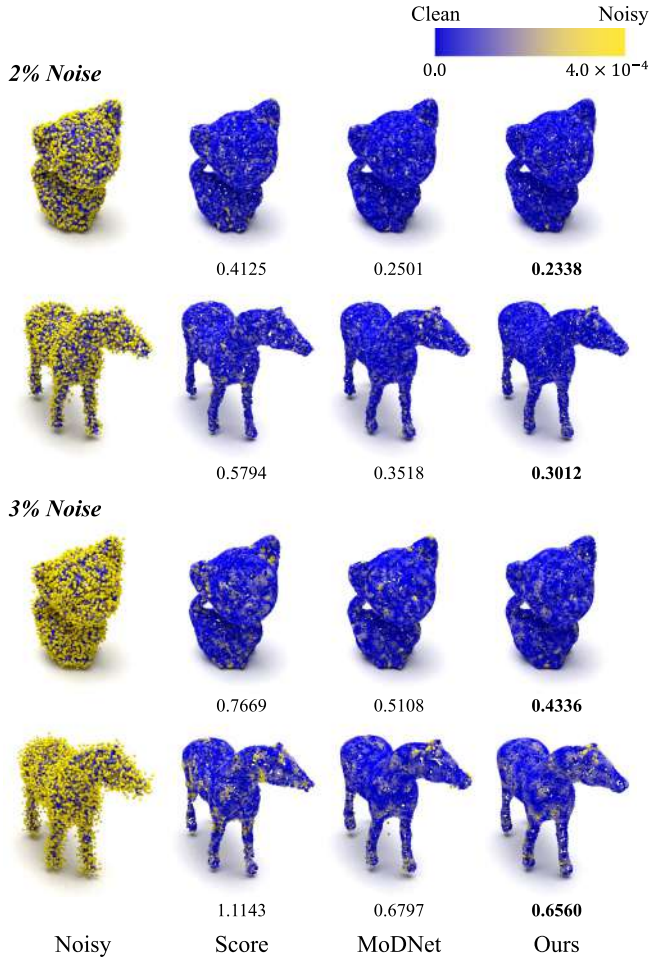

Fig. 2. Denoising performance on PUNet's 50k-resolution point clouds with 2% and 3% noise, where we visualise the point-to-mesh proximity and display the P2M results. The displayed P2M values are multiplied by 10^4.

Table 2
Quantitative performance on the ABC Dataset, where the best results are shown in bold. The CD, P2M results are both multiplied by 10^4.

Method	CD	P2M
Score [14]	2.719	1.302
MoDNet [15]	2.511	1.104
Ours	**2.299**	**0.920**

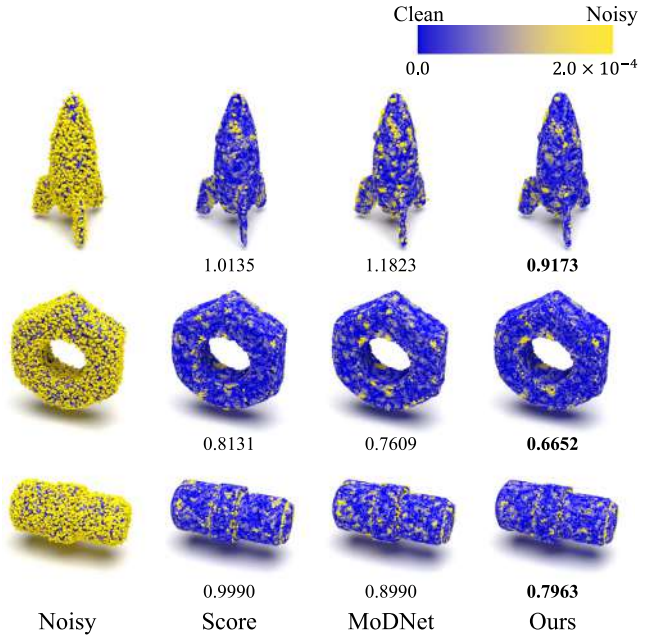

Fig. 3. Denoising performance on the ABC Dataset, where we visualise the point-to-mesh proximity and display the P2M results. The displayed P2M values are multiplied by 10^4.

our method achieves a smaller P2M error, and achieves desirable feature preservation on small details.

ABC Dataset. We show the performance on the ABC Dataset [39], which consists of a large number of computer-aided design (CAD) shapes. We evaluate on 1,792 shapes in total, where we add Gaussian noise on the ground-truth point clouds and evaluate the denoising performance of the state-of-the-art methods, Score [14] and MoDNet [15], as well as our proposed approach. As shown in Table 2, our method outperforms Score and MoDNet in terms of average CD and P2M errors. We also visualise the denoising results of several shapes in Fig. 3.

Kinect Fusion Dataset. We also test on Kinect Fusion [40] dataset, where the 3D geometric information is captured by depth-cameras. The point clouds' geometry is contaminated with noise due to the sensors' limited precision as well as potential environmental noise factors, which is significantly different from Gaussian noise. We denoise the noisy point clouds and measure their point-to-mesh distances using their corresponding ground-truth meshes, and the results are demonstrated in Fig. 4. Despite the challenge that the noise type is different from Gaussian noise,

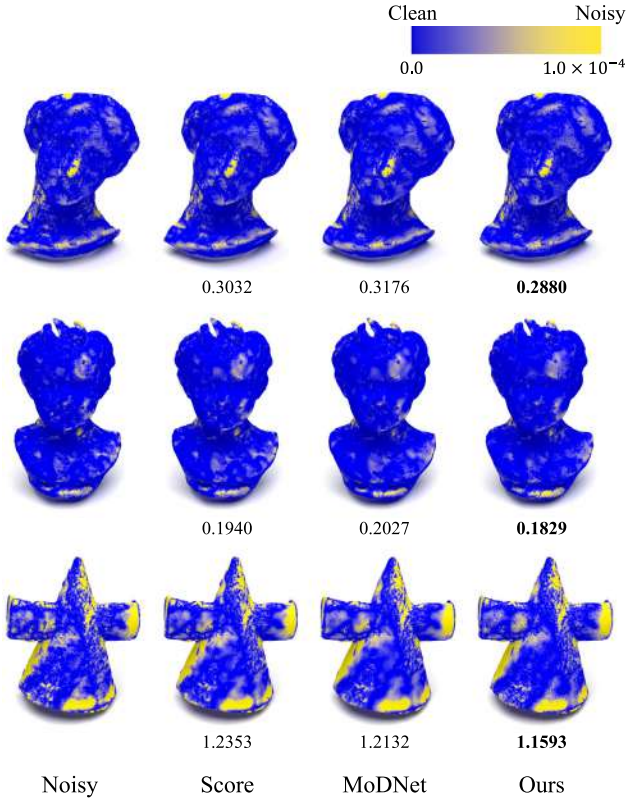

Fig. 4. Denoising performance on Kinect Fusion dataset, where we visualise the point-to-mesh proximity and display the P2M results. The displayed P2M values are multiplied by 10^4.

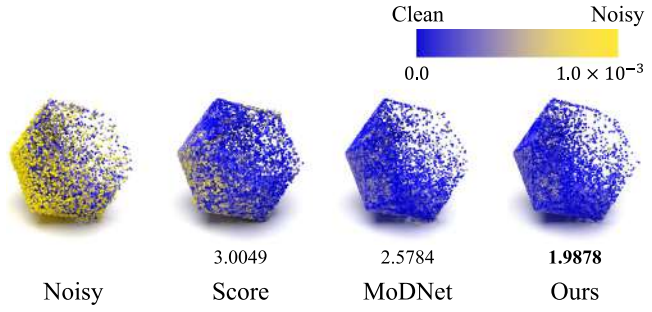

Fig. 5. Denoising performance on an irregular Icosahedron shape, where we visualise the point-to-mesh proximity and display the P2M results. The displayed P2M values are multiplied by 10^4.

our method can also handle such type of noise and outperforms other denoising methods.

Irregular Point Distribution. In addition, we demonstrate the denoising performance on uneven point distributions. As shown in Fig. 5, the Icosahedron shape has irregular point distribution and is corrupted with noise, which turns noise removal into a challenging task. We visualise the point-to-mesh proximity and the P2M errors in Fig. 5, where our method outperforms the state-of-the-art methods, Score [14] and MoDNet [15] in terms of the P2M metric.

4.3. Qualitative results

We demonstrate the denoising results on Paris-rue-Madame [41] dataset. It contains point clouds depicting the street scenes in Paris, which are captured by mobile laser 3D scanners. Due to environmental factors such as lighting conditions and occlusion, the points are contaminated with significant noise and their distribution is much more irregular. The ground-truth points are unknown, such that we only display the visual effects of the denoised points. As demonstrated in Fig. 6, our method effectively restores the geometric information near the A-pillar and the car window area, compared with the state-of-the-art methods Score [14] and MoDNet [15].

Table 3
Ablation study on our network structure and training configurations. The CD, P2M results are both multiplied by 10^4.

No.	$\hat{\mathcal{P}}_i$	$\hat{\mathcal{P}}_i'$	$\hat{\mathcal{P}}_i''$	Aggregation	L_w	CD	P2M
1	✓				✓	3.556	1.149
2	✓	✓		✓	✓	3.321	0.975
3	✓		✓	✓	✓	3.324	0.977
4		✓	✓	✓	✓	3.321	0.975
5	✓	✓	✓	✓		4.054	1.407
6	✓	✓	✓	✓	✓	**3.297**	**0.961**

4.4. Ablation study

Feature Aggregation Methods. In order to evaluate the effectiveness of our feature aggregation layers, we present ablation study regarding different configurations for our module. Following [14,32], we evaluate the performance on PUNet's validation dataset, which contains training point clouds with 10,000 points per shape and is augmented by 1.5% Gaussian noise based on each shape's bounding sphere radius. We list the results of our ablation study by outlining CD and P2M errors of different configurations in Table 3. The configurations we have evaluated include:

1. without any feature aggregation operations, where we directly extract the feature of the original N-point patch $\hat{\mathcal{P}}_i$, and pass the feature into simple MLP layers for subsequent operations (i.e., denoised point prediction and weight regression). We do not perform any feature sampling or aggregation;
2. single feature aggregation, where we concatenate the global feature of $\hat{\mathcal{P}}_i$ with the local feature of $\hat{\mathcal{P}}_i'$ (i.e., concatenate the N-point patch's global feature with its $\frac{N}{2}$-point sub-patch);
3. single feature aggregation, where we directly concatenate the global feature of $\hat{\mathcal{P}}_i$ with the local feature of $\hat{\mathcal{P}}_i''$ (i.e., concatenate the N-point patch's global feature with its $\frac{N}{4}$-point sub-patch);
4. downsample $\hat{\mathcal{P}}_i$ to $\hat{\mathcal{P}}_i'$, then concatenate the global feature of $\hat{\mathcal{P}}_i'$ with the local feature of $\hat{\mathcal{P}}_i''$ (i.e., concatenate the $\frac{N}{2}$-point patch's global feature with its $\frac{N}{4}$-point sub-patch);
5. follow the full feature aggregation pipeline in Fig. 1, without the use of the weighting loss function L_w, and
6. following the full feature aggregation pipeline in Fig. 1 and use all loss terms.

The results demonstrate that utilising the full screening-based feature aggregation pipeline achieves more desirable denoising outcomes than other settings. Also, the weighting loss is effective in eliminating the impacts from irrelevant points, resulting in lower CD and P2M errors.

Point Feature Sampling Ratio. We also evaluate different sampling ratios other than $\frac{1}{2}$ on our validation dataset. As demonstrated in Table 4, our $\frac{1}{2}$ sampling ratio achieves the minimum P2M error compared with others (i.e., $\frac{1}{3}$, $\frac{2}{3}$ and $\frac{3}{4}$).

Comparison with Cascaded Scale Feature Aggregation. We also compare with cascaded scale feature aggregation, which is a

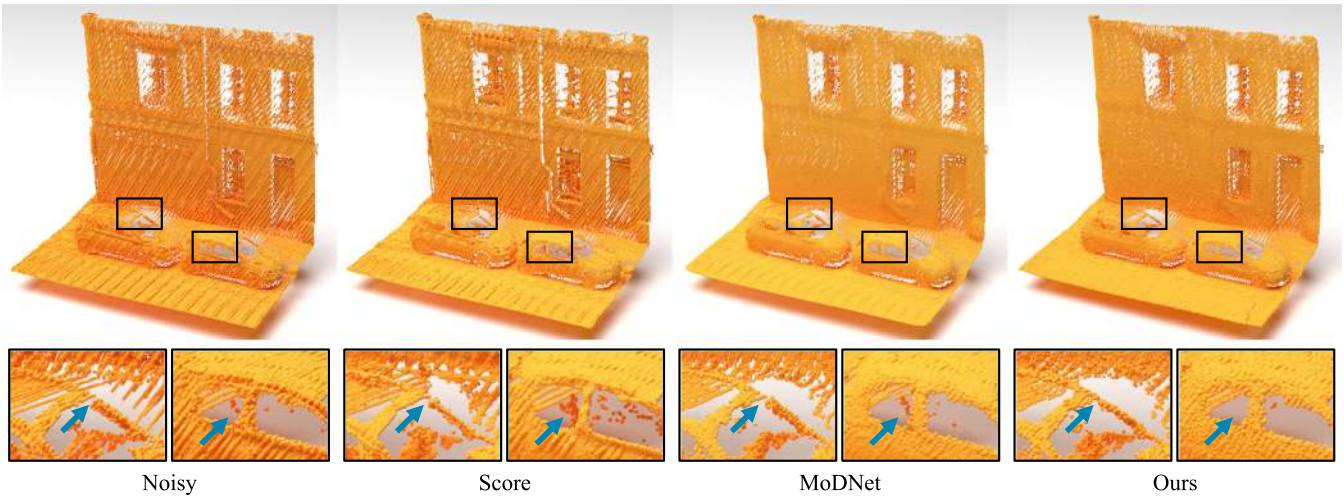

Fig. 6. Point cloud denoising results on Paris-rue-Madame dataset.

Table 4
Sampling ratios and the corresponding P2M errors, which are multiplied by 10^4.

Sampling ratio	1/3	1/2	2/3	3/4
P2M error	1.270	**0.961**	1.254	1.251

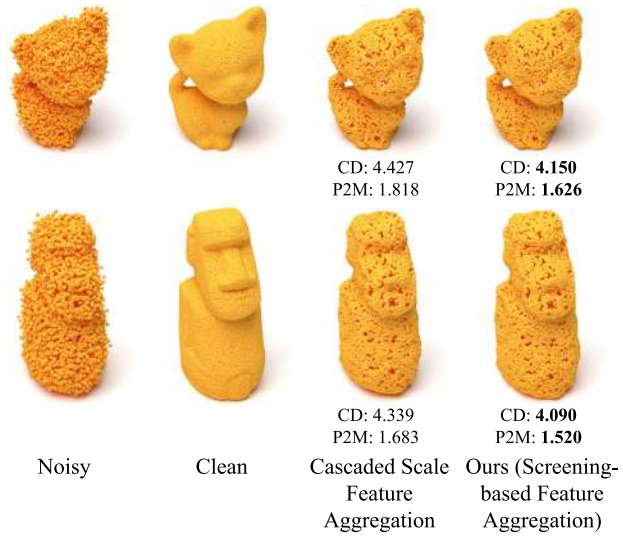

Fig. 7. Comparison with cascaded scale feature aggregation technique. The CD and P2M are multiplied by 10^4, respectively.

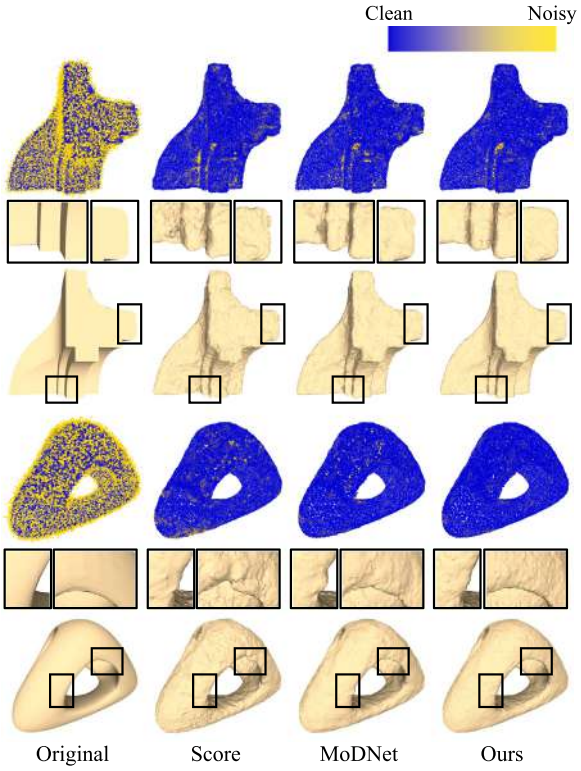

Fig. 8. The mesh surface reconstruction results and the corresponding point clouds.

technique adopted by prior work [34,36]. For example, in [36], the features of large-scale patches are fused into small-scale patches. We observe that this technique achieve similar performance compared with our method on lower noise levels. Nonetheless, our method becomes superior in terms of geometry preservation for shapes contaminated by severe noise. Examples in Fig. 7 demonstrate the denoising results on point clouds with 10,000 points and 3% Gaussian noise, where our method outperforms the scale-based feature aggregation method on CD and P2M metrics.

5. Application

Denoised point clouds can be used for mesh surface reconstruction, where more accurate filtered points lead to better visual results for the reconstructed mesh shapes [42]. We adopt Poisson reconstruction [43], a common backbone utilised by prior work such as [13]. As Poisson reconstruction requires point normals as its input, we use the PCA normal estimation method [44] to estimate normals on the denoised point clouds, where we set the k-nearest neighbours to 100. We show examples of denoised point clouds as well as their reconstructed mesh surfaces in Fig. 8, where our method delivers more accurate reconstruction results on edges and complicated regions, as shown in the close-up pictures.

6. Conclusion

In this paper, we proposed a novel random screening-based feature aggregation technique for point cloud denoising. Our

network removes noise of point clouds in a patch-based manner, where we randomly screen the features of local patches, and fuse the features of denser patches into sparser sub-patches. Richer geometric representations are thus incorporated into fewer points, which improves the network's robustness on severe noise and irregular point distributions. Comprehensive experiments on synthetic and real-world datasets demonstrate that our method delivers competitive point cloud denoising performance, achieving state-of-the-art results.

CRediT authorship contribution statement

Weijia Wang: Conceptualization, Methodology, Formal analysis, Software, Draft. **Wei Pan:** Discussion meeting, Review & editing. **Xiao Liu:** Supervision, Formal analysis, Writing – review & editing. **Kui Su:** Review & editing, Project administration, Discussion. **Bernard Rolfe:** Review & editing, Providing insightful advice. **Xuequan Lu:** Supervision, Formal analysis, Writing – review & editing, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D, Silva C. Point set surfaces. In: Proceedings visualization, 2001. VIS '01. 2001, p. 21–9. http://dx.doi.org/10.1109/VISUAL.2001.964489, 537.

[2] Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D, Silva C. Computing and rendering point set surfaces. IEEE Trans Vis Comput Graphics 2003;9(1):3–15. http://dx.doi.org/10.1109/TVCG.2003.1175093.

[3] Amenta N, Kil YJ. Defining point-set surfaces. ACM Trans Graph 2004;23(3):264–70. http://dx.doi.org/10.1145/1015706.1015713.

[4] Fleishman S, Cohen-Or D, Silva CT. Robust moving least-squares fitting with sharp features. ACM Trans Graph 2005;24(3):544–52. http://dx.doi.org/10.1145/1073204.1073227.

[5] Öztireli AC, Guennebaud G, Gross M. Feature preserving point set surfaces based on non-linear kernel regression. Comput Graph Forum 2009;28(2):493–501. http://dx.doi.org/10.1111/j.1467-8659.2009.01388.x.

[6] Agathos A, Azariadis P, Kyratzi S. Elliptic gabriel taubin smoothing of point clouds. Comput Graph 2022;106:20–32. http://dx.doi.org/10.1016/j.cag.2022.05.009, URL: https://www.sciencedirect.com/science/article/pii/S0097849322000814.

[7] 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):249–63. http://dx.doi.org/10.1007/s41095-022-0278-4.

[8] Zheng Y, Li G, Wu S, Liu Y, Gao Y. Guided point cloud denoising via sharp feature skeletons. Vis Comput 2017;33(6–8):857–67. http://dx.doi.org/10.1007/s00371-017-1391-8.

[9] Rakotosaona M-J, La Barbera V, Guerrero P, Mitra NJ, Ovsjanikov M. Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In: Computer graphics forum, vol. 39. Wiley Online Library; 2020, p. 185–203.

[10] Casajus P, Ritschel T, Ropinski T. Total denoising: Unsupervised learning of 3D point cloud cleaning. 2019, p. 52–60. http://dx.doi.org/10.1109/ICCV.2019.00014.

[11] Boulch A. ConvPoint: Continuous convolutions for point cloud processing. Comput Graph 2020;88:24–34. http://dx.doi.org/10.1016/j.cag.2020.02.005, URL: https://www.sciencedirect.com/science/article/pii/S0097849320300224.

[12] Luo S, Hu W. Differentiable manifold reconstruction for point cloud denoising. In: Proceedings of the 28th ACM international conference on multimedia. 2020.

[13] Zhang D, Lu X, Qin H, He Y. Pointfilter: Point cloud filtering via encoder-decoder modeling. IEEE Trans Vis Comput Graphics 2020.

[14] Luo S, Hu W. Score-based point cloud denoising. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV). 2021, p. 4583–92.

[15] 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. http://dx.doi.org/10.1111/cgf.14661.

[16] De Silva Edirimuni D, Lu X, Shao Z, Li G, Robles-Kelly A, He Y. IterativePFN: True iterative point cloud filtering. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR). 2023, p. 13530–9.

[17] Levin D. The approximation power of moving least-squares. Math Comp 1998;67(224):1517–31. http://dx.doi.org/10.1090/S0025-5718-98-00974-0.

[18] Levin D. Mesh-independent surface interpolation. In: Brunnett G, Hamann B, Müller H, Linsen L, editors. Geometric modeling for scientific visualization. Berlin, Heidelberg: Springer Berlin Heidelberg; 2004, p. 37–49.

[19] Cazals F, Pouget M. Estimating differential quantities using polynomial fitting of osculating jets. Comput Aided Geom Design 2005;22(2):121–46. http://dx.doi.org/10.1016/j.cagd.2004.09.004, URL: https://www.sciencedirect.com/science/article/pii/S016783960400113X.

[20] Lipman Y, Cohen-Or D, Levin D, Tal-Ezer H. Parameterization-free projection for geometry reconstruction. ACM Trans Graph 2007;26(3):22–es. http://dx.doi.org/10.1145/1276377.1276405.

[21] Huang H, Li D, Zhang H, Ascher U, Cohen-Or D. Consolidation of unorganized point clouds for surface reconstruction. ACM Trans Graph (Proc. SIGGRAPH Asia 2009) 2009;28:176:1–7.

[22] Preiner R, Mattausch O, Arikan M, Pajarola R, Wimmer M. Continuous projection for fast L1 reconstruction. ACM Trans Graph 2014;33(4). http://dx.doi.org/10.1145/2601097.2601172.

[23] 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.

[24] Lu X, Wu S, Chen H, Yeung S-K, Chen W, Zwicker M. GPF: GMM-inspired feature-preserving point set filtering. IEEE Trans Vis Comput Graphics 2018;24(8):2315–26. http://dx.doi.org/10.1109/TVCG.2017.2725948.

[25] Mattei E, Castrodad A. Point cloud denoising via moving RPCA. Comput Graph Forum 2017;36(8):123–37. http://dx.doi.org/10.1111/cgf.13068.

[26] Zeng J, Cheung G, Ng M, Pang J, Yang C. 3D point cloud denoising using graph laplacian regularization of a low dimensional manifold model. 2018, arXiv preprint arXiv:1803.07252.

[27] Charles RQ, Su H, Kaichun M, Guibas LJ. PointNet: Deep learning on point sets for 3D classification and segmentation. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR). 2017, p. 77–85. http://dx.doi.org/10.1109/CVPR.2017.16.

[28] Qi CR, Yi L, Su H, Guibas LJ. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Proceedings of the 31st international conference on neural information processing systems. NIPS '17, Red Hook, NY, USA: Curran Associates Inc.; 2017, p. 5105–14.

[29] Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM. Dynamic graph CNN for learning on point clouds. ACM Trans Graph 2019.

[30] Pistilli F, Fracastoro G, Valsesia D, Magli E. Learning graph-convolutional representationsfor point cloud denoising. In: The European conference on computer vision (ECCV). 2020.

[31] Lu D, Lu X, Sun Y, Wang J. Deep feature-preserving normal estimation for point cloud filtering. Comput Aided Des 2020;102860.

[32] Mao A, Du Z, Wen Y-H, Xuan J, Liu Y-J. PD-flow: A point cloud denoising framework with normalizing flows. In: The European conference on computer vision (ECCV). 2022.

[33] Wang W, Lu X, De Silva Edirimuni D, Liu X, Robles-Kelly A. Deep point cloud normal estimation via triplet learning. In: 2022 IEEE international conference on multimedia and expo (ICME). 2022, p. 1–6. http://dx.doi.org/10.1109/ICME52920.2022.9859844.

[34] Li Q, Liu Y-S, Cheng J-S, Wang C, Fang Y, Han Z. HSurf-Net: Normal estimation for 3D point clouds by learning hyper surfaces. In: Advances in neural information processing systems (NeurIPS). 2022.

[35] Huang G, Liu Z, van der Maaten L, Weinberger KQ. Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). 2017.

[36] Zhu R, Liu Y, Dong Z, Wang Y, Jiang T, Wang W, Yang B. AdaFit: Rethinking learning-based normal estimation on point clouds. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV). 2021, p. 6118–27.

[37] Zhang J, Cao J-J, Zhu H-R, Yan D-M, Liu X-P. Geometry guided deep surface normal estimation. Comput Aided Des 2022;142:103119. http://dx.doi.org/

10.1016/j.cad.2021.103119, URL: https://www.sciencedirect.com/science/article/pii/S0010448521001305.

[38] Yu L, Li X, Fu C-W, Cohen-Or D, Heng P-A. PU-net: Point cloud upsampling network. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR). 2018.

[39] Koch S, Matveev A, Jiang Z, Williams F, Artemov A, Burnaev E, Alexa M, Zorin D, Panozzo D. ABC: A big CAD model dataset for geometric deep learning. In: The IEEE conference on computer vision and pattern recognition (CVPR). 2019.

[40] Izadi S, Kim D, Hilliges O, Molyneaux D, Newcombe R, Kohli P, Shotton J, Hodges S, Freeman D, Davison A, Fitzgibbon A. KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera. In: UIST '11 Proceedings of the 24th annual ACM symposium on user interface software and technology. ACM; 2011, p. 559–68.

[41] Serna A, Marcotegui B, Goulette F, Deschaud J-E. Paris-rue-Madame database: a 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. In: 4th international conference on pattern recognition, applications and methods ICPRAM 2014. Angers, France; 2014.

[42] Lu X, Schaefer S, Luo J, Ma L, He Y. Low rank matrix approximation for 3D geometry filtering. IEEE Trans Vis Comput Graphics 2022;28(04):1835–47. http://dx.doi.org/10.1109/TVCG.2020.3026785.

[43] Kazhdan M, Bolitho M, Hoppe H. Poisson surface reconstruction. In: Proceedings of the fourth eurographics symposium on geometry processing. SGP '06, Goslar, DEU: Eurographics Association; 2006, p. 61–70.

[44] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. SIGGRAPH Comput Graph 1992;26(2):71–8.