Computational Visual Media
https://doi.org/10.1007/s41095-024-0423-3 Vol. 10, No. 4, August 2024, 659–669

Research Article

Noise4Denoise: Leveraging noise for unsupervised point cloud denoising

Weijia Wang[1], Xiao Liu[1], Hailing Zhou[2], Lei Wei[1], Zhigang Deng[3], Manzur Murshed[1], and Xuequan Lu[4] (✉)

© The Author(s) 2024.

Abstract Existing deep learning-based point cloud denoising methods are generally trained in a supervised manner that requires clean data as ground-truth labels. However, in practice, it is not always feasible to obtain clean point clouds. In this paper, we introduce a novel unsupervised point cloud denoising method that eliminates the need to use clean point clouds as ground-truth labels during training. We demonstrate that it is feasible for neural networks to only take noisy point clouds as input, and learn to approximate and restore their clean versions. In particular, we generate two noise levels for the original point clouds, requiring the second noise level to be twice the amount of the first noise level. With this, we can deduce the relationship between the displacement information that recovers the clean surfaces across the two levels of noise, and thus learn the displacement of each noisy point in order to recover the corresponding clean point. Comprehensive experiments demonstrate that our method achieves outstanding denoising results across various datasets with synthetic and real-world noise, obtaining better performance than previous unsupervised methods and competitive performance to current supervised methods.

1 Deakin University, Geelong, VIC 3216, Australia. E-mail: W. Wang, wangweijia@deakin.edu.au; X. Liu, xiao.liu@deakin.edu.au; L. Wei, lei.wei@deakin.edu.au; M. Murshed, m.murshed@deakin.edu.au.

2 Department of Mechanical Engineering and Product Design Engineering, Swinburne University of Technology, Melbourne, VIC 3122, Australia. E-mail: hailing.zhou@hotmail.com.

3 Department of Computer Science, University of Houston, Houston, TX 77204-3027, USA. E-mail: zdeng4@central.uh.edu.

4 Department of Computer Science and Information Technology, La Trobe University, Melbourne, VIC 3086, Australia. E-mail: b.lu@latrobe.edu.au (✉).

Manuscript received: 2024-01-24; accepted: 2024-03-01

Keywords point clouds; denoising; geometry processing; deep learning

1 Introduction

Point clouds are used in a wide range of scenarios, such as autonomous driving, immersive reality, robotics, and remote sensing. Despite their notable advantages such as intuitive representation and lightweight characteristics, raw point clouds from sensors are usually corrupted with noise which can degrade their positional accuracy. Such noise comes in the form of displacement vectors that perturb the points and prevent them from accurately representing the objects' surfaces. Noise contamination is often caused by the sensors' quality and precision limitations or by factors from the environment, including material reflectivity and lighting conditions.

As such, point cloud denoising is an essential and fundamental research problem. However, since point clouds are unordered and lack connectivity information [1, 2], removing noise from them has been a long-standing challenge. Traditional point cloud denoising methods, based on optimisation techniques [3–5], often need non-trivial parameter tuning and thus can be burdensome for users. Recently, deep learning-based denoising methods [6–8] have alleviated the burden of parameter tuning and enhanced the robustness of denoising. Most existing learning-based methods require pairs of noisy data and the corresponding ground-truth (i.e., clean) data to guide training. However, ground-truth point clouds are not always available, as it is infeasible to obtain noise-free point clouds in some scenarios due to scanner precision limitations or environmental constraints. In recent years, unsupervised denoising

methods for images, including Refs. [9–11], have demonstrated desirable results for noise removal without training the networks using clean data. Motivated by this, we aim to design a new neural network to denoise 3D point clouds without requiring ground-truth data as training labels.

In this paper, we propose a novel unsupervised point cloud denoising framework. The primary insight of our method is that neural networks can learn to remove noise without requiring clean points as labels. During training, we generate two versions of point clouds: one with standard noise and the other with double noise. Then, we deduce the relationship between the displacement information in these two noisy versions, enabling us to design an effective loss term for minimisation. Our network leverages an encoder–decoder architecture which takes in local neighbourhoods from the doubly-noisy point clouds, learns their features, and subsequently outputs displacement vectors to restore clean point positions. Notably, our training is accomplished using only the noisy point clouds, without reference to clean data. Extensive experiments on different datasets demonstrate that our method can be generalised to both synthetic and real-world noise, showing comparable performance to supervised methods.

In summary, our contributions are outlined as follows:

- we propose an unsupervised point cloud denoising network that eliminates the need for ground-truth point clouds as labels during training;
- our network is designed to learn noise residuals using merely noisy data, without additional inputs beyond point clouds during training and testing, and
- our approach can achieve outstanding denoising results on synthetic and real-world noise com-pared to existing unsupervised and supervised methods.

2 Related work

2.1 Point cloud denoising

2.1.1 Optimisation-based methods

Traditional point cloud denoising methods are typically optimisation-based, removing noise by optimising a set of geometrical constraints. One main category of existing methods is based on *moving least squares* (*MLS*) which aims to approximate the clean underlying surfaces of the input shapes. Based on the seminal work [12], Alexa et al. [3, 13] proposed an MLS-based method to approximate the clean surfaces of the input noisy point clouds. Several other relevant variants were proposed later, including Point Set Surfaces [14], Robust MLS (RMLS) [15], and Robust Implicit MLS (RIMLS) [16]. Another category of existing methods is based on *locally optimal projection* (*LOP*) which projects noisy points onto the optimal surfaces. The early work, proposed by Lipman et al. [4], eliminates the burden of parameterisation. Subsequently, further works were proposed, including Weighted LOP (WLOP) [17], Edge Aware Resampling (EAR) [18], and Continuous LOP (CLOP) [19]. Other denoising methods include jet fitting (Jet) [20], bilateral filtering (Bilateral) [21], feature-preserving methods [22, 23], and a method based on aligning non-local similar patches (referred to later as Non-local) [5].

2.1.2 Supervised learning-based methods

As deep learning techniques have been increasingly used to handle point clouds [1, 24], data-driven point cloud denoising methods, most of which are supervised, have demonstrated their robustness and generalisability. One of the early works, PointCleanNet (PCNet) [6], requires pairs of noisy and clean point clouds for training. Luo and Hu [7] proposed Differentiable Manifold Reconstruction (DMR), an autoencoder-like network that reconstructs manifolds and re-samples denoised points. However, the resampling process may blur shape details. To address this limitation, Score-based Denoising (Score) [8] utilises gradient ascent to guide the denoising process. In recent years, other supervised methods have also been proposed [25–28].

2.1.3 Unsupervised learning-based methods

While supervised methods have demonstrated superior capabilities, ground-truth labels are not always available during training. To address the issue, unsupervised point cloud denoising methods have emerged over the years. Casajus et al. [29] proposed Total Denoising (TotalDn) which achieves noise removal by converging noisy points to their unique modes. However, it usually requires multiple denoising iterations to effectively remove noise. In the aforementioned DMR denoising method [7], Luo and Hu observed that points with denser neighbourhoods are typically closer to the ground-truth surfaces. Thus, these points can be treated as denoising

objectives. With that in mind, they designed an unsupervised version of DMR which trains the network using an altered unsupervised loss function. Similarly, Score [8] exploits an unsupervised loss that shares the same training objective as the supervised one. However, the unsupervised loss tends to cluster points, resulting in unevenly distributed denoised point clouds.

2.2 Unsupervised image denoising

Conventional image denoising operations, such as Non-local Means [30] and Bilateral Filtering [31], as well as supervised methods [32, 33], have demonstrated promising capabilities and have been widely adopted. However, noise-free images are not always available as ground-truth labels during the training process. Over the years, a range of unsupervised image denoising methods have been proposed, demonstrating results comparable to those of conventional and supervised methods. For instance, the Deep Image Prior method [34] can learn image priors and produce high-quality images without needing ground-truth data as input. Lehtinen et al. [9] proposed Noise2Noise where the network only observes noisy images and learns to restore their clean versions. Krull et al. [10] proposed Noise2Void where the objective pixels' values are masked, and the network attempts to predict those values from neighboring patches. Similarly, Batson and Royer [35] proposed Noise2Self which aims to predict residual noise from noisy images alone. Moran et al. [11] proposed Noisier2Noise to restore clean images from additional noise that is added to noisy images. In summary, the aforementioned unsupervised methods showcase superior noise removal capabilities on images.

3 Our method

3.1 Problem formulation

In this section, we formulate the relationship of displacement information across different versions of noisy point clouds, and establish a structured approach to approximate the clean points that is inspired by unsupervised image denoising [11]. We start with the basic definitions: a clean point cloud $\mathcal{P}$ with T points can be defined as a set of 3D coordinates as

$$\mathcal{P} = \{\boldsymbol{p}_i \mid i = 1, \cdots, T\} \tag{1}$$

where $\boldsymbol{p}_i = (x_i, y_i, z_i)$ stands for each point in $\mathcal{P}$. We also define the noisy version of $\mathcal{P}$ (denoted as $\hat{\mathcal{P}}$) as

$$\hat{\mathcal{P}} = \{\hat{\boldsymbol{p}}_i \mid i = 1, \cdots, T\} = \mathcal{P} + \mathcal{N} \tag{2}$$

where $\hat{\boldsymbol{p}}_i$ stands for a corresponding noisy point in $\hat{\mathcal{P}}$, and $\mathcal{N}$ is the noise component that perturbs each point. $\mathcal{N} \sim \mathcal{A}$ where $\mathcal{A}$ is the noise distribution.

We then define another noise component $\mathcal{M}$ sampled from the same noise distribution, such that $\mathcal{M} \sim \mathcal{A}$. *Note that the two noise components $\mathcal{M}$ and $\mathcal{N}$ are independent and identically distributed (i.i.d.), but are not identical to each other.* We add $\mathcal{M}$ to $\hat{\mathcal{P}}$ to obtain another noisier point cloud $\hat{\mathcal{P}}'$ where

$$\begin{aligned}\hat{\mathcal{P}}' = \{\hat{\boldsymbol{p}}'_i \mid i = 1, \cdots, T\} &= \hat{\mathcal{P}} + \mathcal{M} \\ &= \mathcal{P} + \mathcal{N} + \mathcal{M}\end{aligned} \tag{3}$$

such that $\hat{\mathcal{P}}'$ is doubly-noisy and contains twice the amount of noise in $\hat{\mathcal{P}}$. The concepts of clean, noisy, and doubly-noisy point clouds are shown in Fig. 1. *Our network has no access to the ground-truth point clouds during training.*

We argue that it is feasible to estimate the clean point cloud $\mathcal{P}$ based on the noisy inputs. Given $\hat{\mathcal{P}}'$ as the prior, which represents the observed doubly-noisy point cloud, we denote the estimation of the overall surface of the noisy point cloud $\hat{\mathcal{P}}$ via an expected value $\mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}')$. Based on Eq. (2), we have

$$\begin{aligned}\mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}') &= \mathbb{E}(\mathcal{P} + \mathcal{N}|\hat{\mathcal{P}}') \\ &= \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{N}|\hat{\mathcal{P}}')\end{aligned} \tag{4}$$

Since $\mathcal{M}$ and $\mathcal{N}$ are i.i.d. and are both sampled from the distribution $\mathcal{A}$, their expectation values are equivalent, i.e., $\mathbb{E}(\mathcal{M}|\hat{\mathcal{P}}') = \mathbb{E}(\mathcal{N}|\hat{\mathcal{P}}')$. With that established, we can derive Eq. (5):

$$\begin{aligned}2\mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}') &= 2\mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + 2\mathbb{E}(\mathcal{N}|\hat{\mathcal{P}}') \\ &= \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{N}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{N}|\hat{\mathcal{P}}') \\ &= \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{N}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{M}|\hat{\mathcal{P}}') \\ &= \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\mathcal{P} + \mathcal{N} + \mathcal{M}|\hat{\mathcal{P}}') \\ &= \mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') + \mathbb{E}(\hat{\mathcal{P}}'|\hat{\mathcal{P}}')\end{aligned} \tag{5}$$

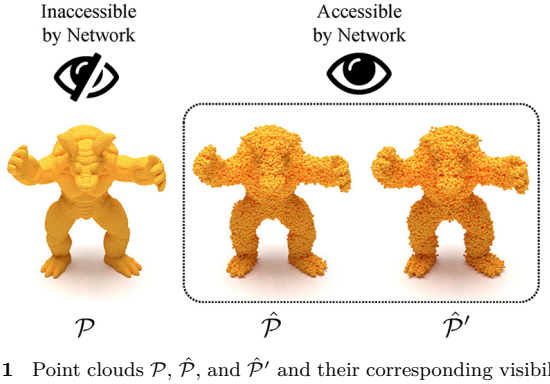


Fig. 1 Point clouds $\mathcal{P}$, $\hat{\mathcal{P}}$, and $\hat{\mathcal{P}}'$ and their corresponding visibilities to the network. Our network can only access the noisy versions of the point clouds.

which thus leads to

$$\mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') = 2\mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}') - \mathbb{E}(\hat{\mathcal{P}}'|\hat{\mathcal{P}}')$$
$$\mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') - \mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}') = \mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}') - \mathbb{E}(\hat{\mathcal{P}}'|\hat{\mathcal{P}}') \quad (6)$$

Here, we denote $\mathbb{E}(\mathcal{P}|\hat{\mathcal{P}}') = \bar{\mathcal{P}}$, $\mathbb{E}(\hat{\mathcal{P}}|\hat{\mathcal{P}}') = \bar{\hat{\mathcal{P}}}$, and $\mathbb{E}(\hat{\mathcal{P}}'|\hat{\mathcal{P}}') = \bar{\hat{\mathcal{P}}}'$, where $\bar{\mathcal{P}}$, $\bar{\hat{\mathcal{P}}}$, and $\bar{\hat{\mathcal{P}}}'$ stand for the predictions of $\mathcal{P}$, $\hat{\mathcal{P}}$, and $\hat{\mathcal{P}}'$, respectively. We substitute the notations back into Eq. (6) to give

$$\bar{\mathcal{P}} - \bar{\hat{\mathcal{P}}} = \bar{\hat{\mathcal{P}}} - \bar{\hat{\mathcal{P}}}' \quad (7)$$

It is worth noting that $\bar{\hat{\mathcal{P}}}' = \mathbb{E}(\hat{\mathcal{P}}'|\hat{\mathcal{P}}')$ is equivalent to $\hat{\mathcal{P}}'$ itself based on the rules of conditional expectation and point clouds' discrete characteristics. We also assume that the points in $\bar{\mathcal{P}}$, $\bar{\hat{\mathcal{P}}}$, and $\hat{\mathcal{P}}'$ have one-to-one correspondences. Based on this assumption, we can estimate a noise displacement vector for each point in $\hat{\mathcal{P}}'$, and use these vectors to build the approximated clean point cloud $\bar{\mathcal{P}}$. Here, we denote the relationship as

$$\bar{\mathcal{P}} = \{\bar{\boldsymbol{p}}_i \mid i = 1, \cdots, T\} = \{\hat{\boldsymbol{p}}'_i + \bar{\boldsymbol{d}}'_i \mid i = 1, \cdots, T\} \quad (8)$$

where $\bar{\boldsymbol{d}}'_i$ represents the predicted displacement vector for point $\hat{\boldsymbol{p}}'_i$; this directly estimates the position of the corresponding clean point.

The prediction $\bar{\hat{\mathcal{P}}}$ can also be obtained by adding another set of displacement vectors to $\hat{\mathcal{P}}'$, as Eq. (9):

$$\bar{\hat{\mathcal{P}}} = \{\bar{\hat{\boldsymbol{p}}}_i \mid i = 1, \cdots, T\} = \{\hat{\boldsymbol{p}}'_i + \bar{\boldsymbol{d}}_i \mid i = 1, \cdots, T\} \quad (9)$$

Based on our one-to-one correspondence assumption, we finally demonstrate that the direct estimate $\bar{\boldsymbol{d}}'_i$ is equivalent to twice $\bar{\boldsymbol{d}}_i$ by substituting the values back into Eq. (7):

$$\begin{aligned} \bar{\boldsymbol{p}}_i - \bar{\hat{\boldsymbol{p}}}_i &= \bar{\hat{\boldsymbol{p}}}_i - \hat{\boldsymbol{p}}'_i \\ (\hat{\boldsymbol{p}}'_i + \bar{\boldsymbol{d}}'_i) - (\hat{\boldsymbol{p}}'_i + \bar{\boldsymbol{d}}_i) &= (\hat{\boldsymbol{p}}'_i + \bar{\boldsymbol{d}}_i) - \hat{\boldsymbol{p}}'_i \\ \bar{\boldsymbol{d}}'_i - \bar{\boldsymbol{d}}_i &= \bar{\boldsymbol{d}}_i \\ \bar{\boldsymbol{d}}'_i &= 2\bar{\boldsymbol{d}}_i \end{aligned} \quad (10)$$

which guides the design of our loss function for training.

3.2 Denoising framework

Our network architecture is shown in Fig. 2. Following previous works [6, 25, 27], we design a point cloud denoising pipeline based on local neighbourhoods as patches. The noisier point cloud $\hat{\mathcal{P}}'$ is the input source for our network, and we define the local patch $\hat{\mathcal{P}}'_i$ around each query point $\hat{\boldsymbol{p}}'_i$ as

$$\hat{\mathcal{P}}'_i = \{\hat{\boldsymbol{p}}'_j \mid \left\|\hat{\boldsymbol{p}}'_j - \hat{\boldsymbol{p}}'_i\right\|_2 < r,\ \hat{\boldsymbol{p}}'_j \in \hat{\mathcal{P}}'\} \quad (11)$$

where r is the query ball radius, $\|\cdot\|_2$ represents L2-norm, and $\hat{\boldsymbol{p}}'_j$ stands for neighbouring points within the radius r. During training, we empirically set r to be equivalent to 5% of the bounding box diagonal length of $\hat{\mathcal{P}}'$, a common setting in prior literature [25, 27]. Meanwhile, we also obtain $\hat{\boldsymbol{p}}_i$, the point with the corresponding index i in point cloud $\hat{\mathcal{P}}$, for training purposes.

We then perform normalisation to minimise the arbitrary degrees of freedom in the queried patches and facilitate effective training. We first centre $\hat{\mathcal{P}}'_i$ by translating it to its query point and normalise its size using r, so that $\hat{\mathcal{P}}'_i = (\hat{\mathcal{P}}'_i - \hat{\boldsymbol{p}}'_i)/r$. Then, we align $\hat{\mathcal{P}}'_i$ with the canonical space using the rotation matrix $\boldsymbol{R}$ obtained via principal component analysis (PCA) decomposition on $\hat{\mathcal{P}}'_i$. We also process $\hat{\boldsymbol{p}}_i$ following the same transformation procedure for $\hat{\mathcal{P}}'_i$. Next, as the number of points may be inconsistent between raw patches, we set an empirical threshold $N = 500$ to regularise the patch size for $\hat{\mathcal{P}}'_i$ to ensure they can be grouped into mini-batches. Specifically, following previous works, we downsample patches with more than N points, and add extra points for patches with fewer than N points. Note that during the testing phase, the inverse of the aforementioned

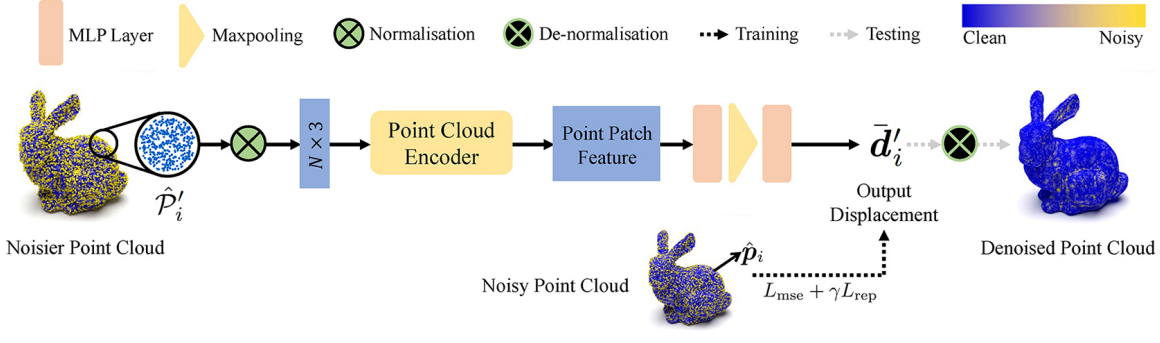


Fig. 2 Network architecture of our method. We employ normalisation before inputting the patch into our encoder, and do de-normalisation during the testing phase.

transformations (rotation, scaling, and translation) are applied to map the denoised point back to its original position.

We input the normalised patch $\hat{\mathcal{P}}'_i$ (as an $N \times 3$ matrix) into our point cloud encoder. It uses graph-based Dense Block modules [36] as a backbone and encodes each point with its neighbours' information. To further enhance the network's generalisation performance, we adopt the Global Shift module from Ref. [37] to fuse patch features into fewer points, in order to improve the robustness of our network and increase inferencing efficiency. The output of our encoder is a feature matrix which goes through our output module to form the final predicted displacement $\bar{\boldsymbol{d}}'_i$, a 1×3 vector. During training, we utilise the noisy point $\hat{\boldsymbol{p}}_i$ to train the estimation of the displacement vector.

3.3 Loss functions

We exploit $\hat{\boldsymbol{d}}_i = \hat{\boldsymbol{p}}_i - \hat{\boldsymbol{p}}'_i$, the actual value for prediction $\bar{\boldsymbol{d}}_i$ in Eq. (10), to guide the estimation of displacement $\bar{\boldsymbol{d}}'_i$. We design a mean squared error (MSE) loss function which is defined as

$$L_{\text{mse}} = \left\| \bar{\boldsymbol{d}}'_i - 2\hat{\boldsymbol{d}}_i \right\|_2^2 \tag{12}$$

and we further discuss our choice regarding noise prediction in Section 5.

We also require our denoised point cloud to be well-spaced to avoid potential clustering issues. To achieve this, we design a repulsion loss inspired by Refs. [6, 25, 27] to assist with the distribution of the denoised points. First of all, we define a pseudo-clean point cloud $\tilde{\mathcal{P}}$ as

$$\tilde{\mathcal{P}} = \{\hat{\boldsymbol{p}}'_i + 2\hat{\boldsymbol{d}}_i \mid i = 1, \cdots, T\} \tag{13}$$

where $\tilde{\mathcal{P}}$ does not actually have a clean appearance. Next, we query pseudo-clean patches from it using the process defined by

$$\tilde{\mathcal{P}}_i = \{\tilde{\boldsymbol{p}}_j \mid \|\tilde{\boldsymbol{p}}_j - \hat{\boldsymbol{p}}_i\|_2 < r,\ \tilde{\boldsymbol{p}}_j \in \tilde{\mathcal{P}}\} \tag{14}$$

Here, we use $\hat{\boldsymbol{p}}_i \in \hat{\mathcal{P}}$ to perform queries for $\tilde{\mathcal{P}}_i$ as we assume $\hat{\boldsymbol{p}}_i$ is less impacted by noise, and we normalise patch $\tilde{\mathcal{P}}_i$ using the same normalisation process introduced in Section 3.2. Then, we use these patches to formulate our repulsion loss as

$$L_{\text{rep}} = \max_{\tilde{\boldsymbol{p}}_j \in \tilde{\mathcal{P}}_i} \left\| (\hat{\boldsymbol{p}}'_i + \bar{\boldsymbol{d}}'_i) - \tilde{\boldsymbol{p}}_j \right\|_2^2 \tag{15}$$

Our overall loss L is thus formulated as

$$L = L_{\text{mse}} + \gamma L_{\text{rep}} \tag{16}$$

where γ is the factor controlling the repulsion loss and is set to 0.0005. We discuss the setting of γ in Section 5.

4 Experimental results

4.1 Training configuration

We carried out our experiments on an NVIDIA RTX 3080 GPU with 10 GB memory. We implemented our network model using PyTorch, and set the batch size for training to 128. We trained the network for 200 epochs with a single Adam optimiser. The learning rate was set to 0.0001 and was multiplied by a factor of 0.1 in the 40th, 80th, 120th, 160th, and 180th epochs.

4.2 Training dataset

We adopted PUNet [38] for training, which contains 40 clean triangular mesh shapes and corresponding noise-free point clouds at three different resolutions (10k, 30k, and 50k). All clean point clouds were taken from the surfaces of the mesh shapes, and we sampled 1000 patches per point cloud in each epoch. During training, we followed the noise addition method in Refs. [8, 26], i.e., the additive noise is zero-mean Gaussian noise with a standard deviation ranging from 0.5% to 2.0% of each shape's bounding sphere radius. To do so, the clean point cloud must first be normalised into a unit sphere, and then be added with noise $\mathcal{Z} \sim \mathcal{D}(\mu_{\mathcal{D}}, \sigma_{\mathcal{D}}^2)$ where $\mathcal{D}$ is the objective noise distribution, $\mu_{\mathcal{D}} = 0$, and $\sigma_{\mathcal{D}} \in [0.005, 0.02]$.

We define the distribution of each level of the additive noise during training as $\mathcal{A}(\mu_{\mathcal{A}}, \sigma_{\mathcal{A}}^2)$, and recall that $\mathcal{N} \sim \mathcal{A}$ and $\mathcal{M} \sim \mathcal{A}$. To ensure $\mathcal{N} + \mathcal{M} = \mathcal{Z}$ holds, we need $2\mu_{\mathcal{A}} = \mu_{\mathcal{D}}$ and $2\sigma_{\mathcal{A}}^2 = \sigma_{\mathcal{D}}^2$. Thus, we set $\mu_{\mathcal{A}} = 0$ and $\sigma_{\mathcal{A}} \in \left[\frac{0.005}{\sqrt{2}}, \frac{0.02}{\sqrt{2}}\right]$ for noise sampling during our training process.

4.3 Quantitative results

4.3.1 PUNet dataset

First, we evaluated the performance of our method on the PUNet test dataset. It contains point clouds sampled from 20 mesh shapes at 2 resolutions (sparse, 10k points per shape; and dense, 50k points per shape), and has 3 Gaussian noise levels (1%, 2%, and 3% of the bounding sphere's radius) for each resolution. Following prior works [8, 26], we ran one denoising iteration for 1% and 2% noise and ran two denoising iterations for 3% noise. We employed

two metrics, Chamfer Distance (CD) and Point-to-mesh Distance (P2M), to measure the quality of the denoised point clouds. For both metrics, smaller values indicate more accurate results. We denote the unsupervised versions of DMR and Score using DMR-U and Score-U, respectively. The denoising results are shown in Table 1 where our method outperforms all others for each setting.

4.3.2 Kinect dataset

We also tested our method on real-world scanned data captured by Kinect v1 and v2 [39], which is inherently noisy. The datasets provide reconstructed clean data for quantitative measurement. We present the results in Table 2 where our method outperforms other unsupervised denoising techniques as well as the supervised method PCNet and the optimisation-based method Non-local in terms of CD and P2M errors.

4.4 Qualitative results

4.4.1 PUNet dataset

We first demonstrate the visual denoising effects on the PUNet dataset. We use a colour scale to indicate the distances of the denoised points from the ground-truth mesh surfaces, where blue indicates more accurate positions and yellow indicates inaccurate positions. The results are shown in Fig. 3 for both sparse and dense resolutions. As illustrated, the downsampling–resampling strategy of DMR-U does not effectively remove the residual noise, leading to many noisy points that are marked in yellow. Score-U tends to cluster the points together, resulting in uneven distribution. Other methods cannot effectively preserve small details, resulting in more yellow points in those areas. By contrast, our method achieves more accurate results overall on both sparse and dense configuration settings.

Table 1 Quantitative results on PUNet's test set. Best results are marked in bold. The abbreviations Op., Sup., and Unsup. represent optimisation-based, supervised, and unsupervised denoising methods, respectively. CD and P2M are both multiplied by 10^4

Type	Density	10k						50k					
	Noise level	1%		2%		3%		1%		2%		3%	
	Method name	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Op.	Bilateral	3.646	1.342	5.007	2.018	6.998	3.557	0.877	0.234	2.376	1.389	6.304	4.730
	Jet	2.712	0.613	4.155	1.347	6.262	2.921	0.851	0.207	2.432	1.403	5.788	4.267
	Non-local	3.506	1.331	4.720	1.985	7.405	4.489	0.966	0.337	2.240	1.317	3.623	2.566
Sup.	PCNet	3.515	1.148	7.467	3.965	13.067	8.737	1.049	0.346	1.447	0.608	2.289	1.285
	DMR	4.482	1.722	4.982	2.115	5.892	2.846	1.162	0.469	1.566	0.800	2.432	1.528
	Score	2.521	0.463	3.686	1.074	4.708	1.942	0.716	0.150	1.288	0.566	1.928	1.041
Unsup.	TotalDn	3.390	0.826	7.251	3.485	13.385	8.740	1.024	0.314	2.722	1.567	7.474	5.729
	DMR-U	5.313	2.522	6.455	3.317	8.134	4.647	1.226	0.521	2.138	1.251	2.496	1.520
	Score-U	3.107	0.888	4.675	1.829	7.225	3.762	0.918	0.265	2.439	1.411	5.303	3.841
	Ours	**2.204**	**0.354**	**3.449**	**0.930**	**4.153**	**1.554**	**0.700**	**0.134**	**1.056**	**0.374**	**1.282**	**0.563**

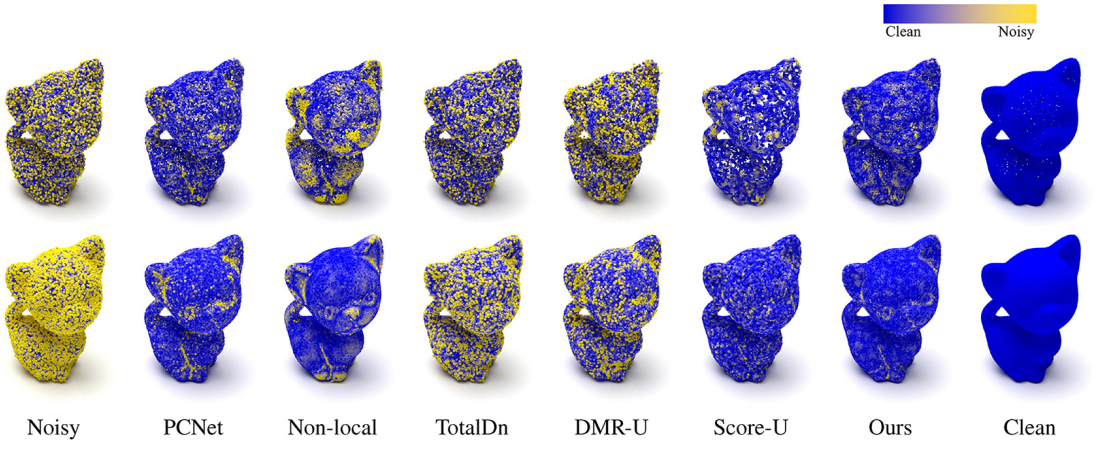


Fig. 3 A visual comparison of results on a shape from the PUNet dataset (1% noise) where yellow points are further from the ground-truth mesh surfaces. The top row shows the sparse (10k) configuration and the bottom row shows the dense (50k) configuration.

Table 2 Results on Kinect dataset. CD and P2M are both multiplied by 10^4

Method name	Kinect v1		Kinect v2	
	CD	P2M	CD	P2M
PCNet	1.540	0.780	2.304	1.234
Non-local	1.442	0.750	2.264	1.305
TotalDn	1.356	0.703	2.094	1.200
DMR-U	1.668	0.743	2.997	1.494
Score-U	1.664	0.738	2.547	1.092
Ours	**1.213**	**0.623**	**1.918**	**1.088**

4.4.2 Kinect dataset

We also present the visual denoising effects on the Kinect dataset in Fig. 4 where the denoised shapes are taken from the results presented in Table 2. It demonstrates that our method can produce smoother point cloud surfaces compared to other methods.

4.4.3 Paris-rue-Madame dataset

We show denoising results on Paris-rue-Madame [40], a street scene point cloud dataset collected by laser scanners. It is contaminated with severe noise due to various outdoor environmental factors. No ground-truth data is associated with this dataset, so we compare the visual denoising effects in terms of smoothness and feature preservation. The results are shown in Fig. 5 where our method outputs smoother point clouds and effectively restores small details, as illustrated in the close-up windows.

5 Ablation study

We performed our ablation study on the validation dataset following the settings in Ref. [8] and measured CD and P2M metrics. In this section, we explore the effect of repulsion loss and our displacement regression technique.

5.1 Repulsion loss

We first considered the effects with and without the repulsion term L_{rep}, and additionally evaluated different γ values. The results are shown in configurations 1–6 in Table 3, where setting $\gamma = 0.0005$ provides the best results.

5.2 Displacement regression technique

An alternative training technique is to regress the displacement for noise $\mathcal{M}$ only and double it during the evaluation phase. Consequently, we should regress $\bar{\boldsymbol{d}}_i$ in this setting and the altered training objective becomes

$$L' = L'_{\text{mse}} + \gamma L'_{\text{rep}} \tag{17}$$

where we define L'_{mse} as

$$L'_{\text{mse}} = \left\| \bar{\boldsymbol{d}}_i - \hat{\boldsymbol{d}}_i \right\|_2^2 \tag{18}$$

Table 3 Ablation study on validation set with different training configurations. CD and P2M are both multiplied by 10^4

Config.	Loss	γ	CD	P2M
1	L	0.0	2.923	0.619
2	L	0.0001	2.916	0.615
3	L	0.0005	**2.880**	**0.606**
4	L	0.001	2.903	0.610
5	L	0.01	2.909	0.617
6	L	0.05	2.977	0.673
7	L'	0.0005	3.056	0.668

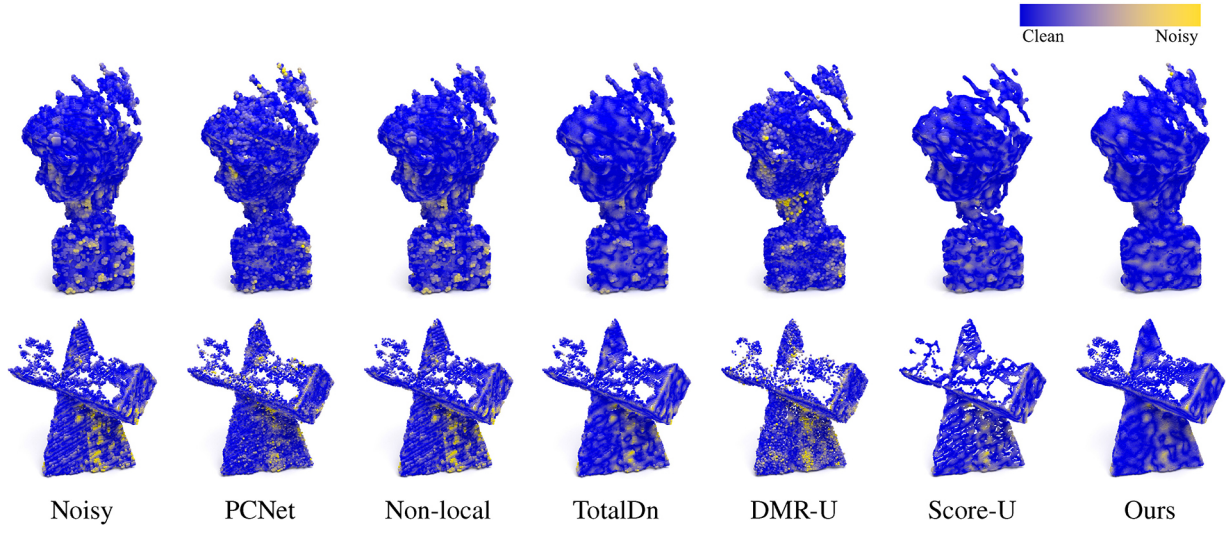


Fig. 4 Visual comparison of denoising effects on the Kinect dataset. Yellow points are further from the ground-truth mesh surfaces.

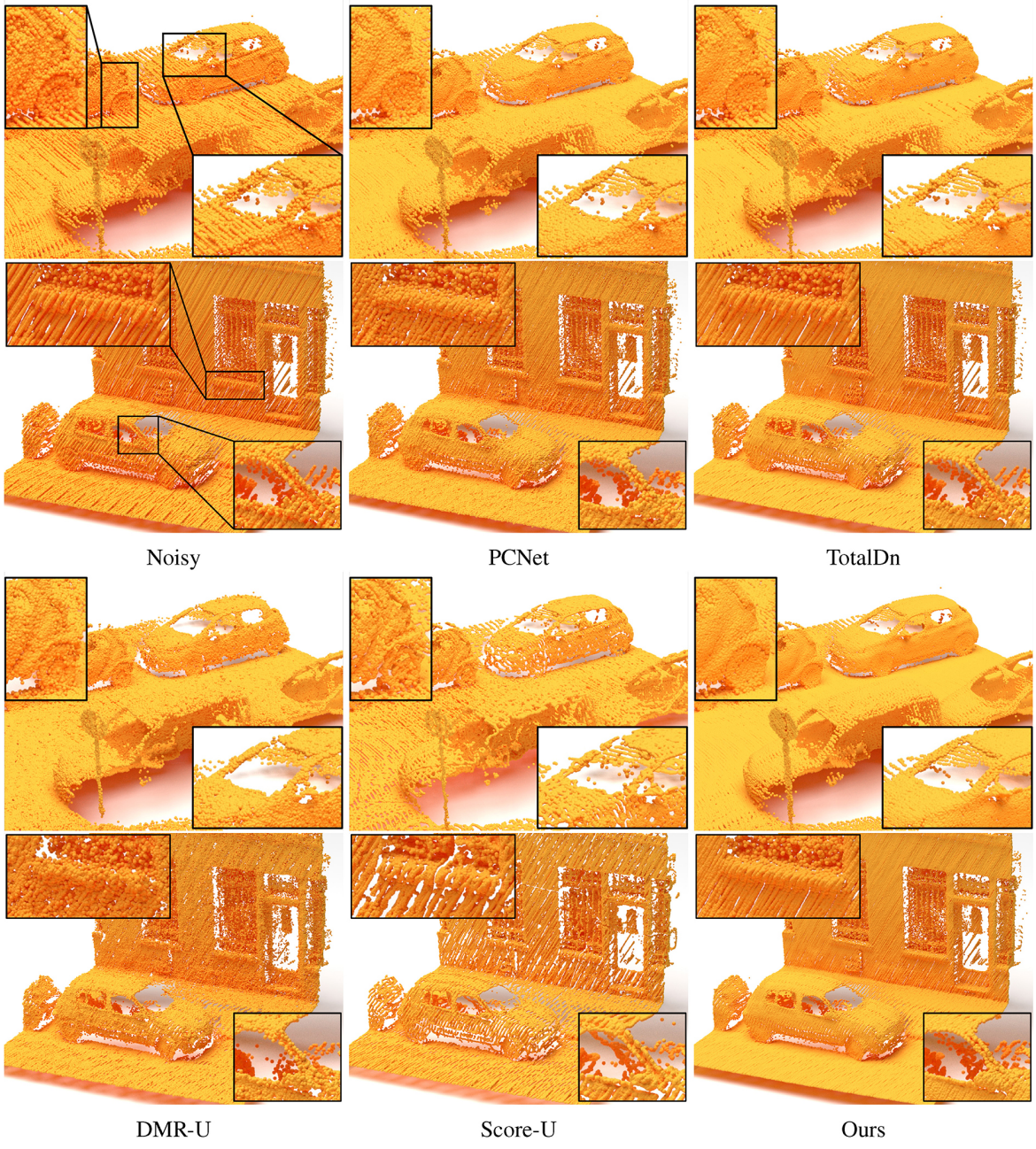


Fig. 5 Comparison of the denoising results on Paris-rue-Madame dataset.

and define L'_{rep} as

$$L'_{\text{rep}} = \max_{\tilde{\boldsymbol{p}}_j \in \tilde{\mathcal{P}}_i} \left\| (\hat{\boldsymbol{p}}'_i + 2\bar{\boldsymbol{d}}_i) - \tilde{\boldsymbol{p}}_j \right\|_2^2 \tag{19}$$

We set γ to be consistent with our experiments. However, this training technique, displayed as configuration 7 in Table 3, cannot achieve comparable results on the validation dataset.

6 Application

Denoised point clouds can be utilised for mesh reconstruction. To achieve this, we first exploited HSurf-Net [37] to estimate normals for the denoised point clouds produced by different methods, and then performed Poisson surface reconstruction [41]. The denoised point clouds and the reconstructed meshes are shown in Fig. 6 where our method achieves smoother results on the mesh surface.

7 Discussion and conclusions

In this paper, we have presented a novel unsupervised

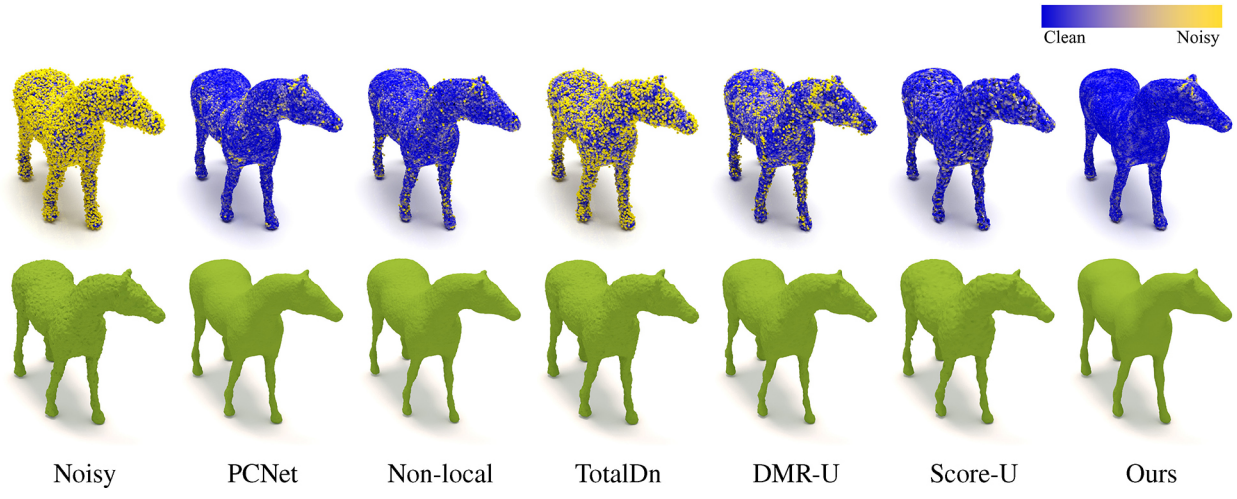


Fig. 6 Mesh reconstruction results on the denoised point clouds.

point cloud denoising framework that learns to restore clean point clouds from noisy inputs only. During training, we leverage two levels of noise, where the second noise level is twice that of the first, and train our network to predict clean points without using ground-truth data as labels. Extensive experimental results demonstrate that our method generalises well across synthetic and real-world noise, outperforming state-of-the-art unsupervised methods and achieving comparable performance to supervised methods.

Our method unavoidably demonstrates some limitations during denoising. For instance, since our method is trained using point-based loss functions only, it may occasionally blur sharp features and produce smooth edges and corners. In order to enhance the capability to preserve sharp features, it is necessary to utilise normal information as demonstrated in prior literature [25, 42]. In our future work, we aim to explore further in terms of sharp feature preservation by incorporating normal information in our denoising process.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Charles, R. Q.; Hao, S.; Mo, K.; Guibas, L. J. PointNet: Deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 77–85, 2017.

[2] Guerrero, P.; Kleiman, Y.; Ovsjanikov, M.; Mitra, N. J. PCPNet learning local shape properties from raw point clouds. *Computer Graphics Forum* Vol. 37, No. 2, 75–85, 2018.

[3] Alexa, M.; Behr, J.; Cohen-Or, D.; Fleishman, S.; Levin, D.; Silva, C. T. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics* Vol. 9, No. 1, 3–15, 2003.

[4] Lipman, Y.; Cohen-Or, D.; Levin, D.; Tal-Ezer, H. Parameterization-free projection for geometry reconstruction. *ACM Transactions on Graphics* Vol. 26, No. 3, 22–es, 2007.

[5] Wang, J.; Jiang, J.; Lu, X.; Wang, M. Rethinking point cloud filtering: A non-local position based approach. *Computer-Aided Design* Vol. 144, 103162, 2022.

[6] Rakotosaona, M. J.; La Barbera, V.; Guerrero, P.; Mitra, N. J.; Ovsjanikov, M. PointCleanNet: Learning to denoise and remove outliers from dense point clouds. *Computer Graphics Forum* Vol. 39, No. 1, 185–203, 2020.

[7] Luo, S.; Hu, W. Differentiable manifold reconstruction for point cloud denoising. In: Proceedings of the 28th ACM International Conference on Multimedia, 1330–1338, 2020.

[8] Luo, S.; Hu, W. Score-based point cloud denoising. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 4563–4572, 2021.

[9] Lehtinen, J.; Munkberg, J.; Hasselgren, J.; Laine, S.; Karras, T.; Aittala, M.; Aila, T. Noise2Noise: Learning image restoration without clean data. In: Proceedings of the 35th International Conference on Machine Learning, 2965–2974, 2018.

[10] Krull, A.; Buchholz, T. O.; Jug, F. Noise2Void - Learning denoising from single noisy images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2124–2132, 2019.

[11] Moran, N.; Schmidt, D.; Zhong, Y.; Coady, P. Noisier2Noise: Learning to denoise from unpaired noisy data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 12061–12069, 2020.

[12] Levin, D. The approximation power of moving least-squares. *Mathematics of Computation* Vol. 67, No. 224, 1517–1531, 1998.

[13] Alexa, M.; Behr, J.; Cohen-Or, D.; Fleishman, S.; Levin, D.; Silva, C. T. Point set surfaces. In: Proceedings of the Visualization, 21–29, 2001.

[14] Amenta, N.; Kil, Y. J. Defining point-set surfaces. *ACM Transactions on Graphics* Vol. 23, No. 3, 264–270, 2004.

[15] Fleishman, S.; Cohen-Or, D.; Silva, C. T. Robust moving least-squares fitting with sharp features. *ACM Transactions on Graphics* Vol. 24, No. 3, 544–552, 2005.

[16] Öztireli, A. C.; Guennebaud, G.; Gross, M. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum* Vol. 28, No. 2, 493–501, 2009.

[17] Huang, H.; Li, D.; Zhang, H.; Ascher, U.; Cohen-Or, D. Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics* Vol. 28, No. 5, 1–7, 2009.

[18] Huang, H.; Wu, S.; Gong, M.; Cohen-Or, D.; Ascher, U.; Zhang, H. Edge-aware point set resampling. *ACM Transactions on Graphics* Vol. 32, No. 1, Article No. 9, 2013.

[19] Preiner, R.; Mattausch, O.; Arikan, M.; Pajarola, R.; Wimmer, M. Continuous projection for fast L_1 reconstruction. *ACM Transactions on Graphics* Vol. 33, No. 4, Article No. 47, 2014.

[20] Cazals, F.; Pouget, M. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design* Vol. 22, No. 2, 121–146, 2005.

[21] Fleishman, S.; Drori, I.; Cohen-Or, D. Bilateral mesh denoising. *ACM Transactions on Graphics* Vol. 22, No. 3, 950–953, 2003.

[22] Lu, X.; Wu, S.; Chen, H.; Yeung, S. K.; Chen, W.; Zwicker, M. GPF: GMM-inspired feature-preserving point set filtering. *IEEE Transactions on Visualization and Computer Graphics* Vol. 24, No. 8, 2315–2326, 2018.

[23] Chen, S.; Wang, J.; Pan, W.; Gao, S.; Wang, M.; Lu, X. Towards uniform point distribution in feature-preserving point cloud filtering. *Computational Visual Media* Vol. 9, No. 2, 249–263, 2023.

[24] Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; Solomon, J. M. Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics* Vol. 38, No. 5, Article No. 146, 2019.

[25] Zhang, D.; Lu, X.; Qin, H.; He, Y. Pointfilter: Point cloud filtering via encoder-decoder modeling. *IEEE Transactions on Visualization and Computer Graphics* Vol. 27, No. 3, 2015–2027, 2021.

[26] Mao, A.; Du, Z.; Wen, Y. H.; Xuan, J.; Liu, Y. J. PD-flow: A point cloud denoising framework with normalizing flows. In: *Computer Vision – ECCV 2022. Lecture Notes in Computer Science, Vol. 13663*. Avidan, S.; Brostow, G.; Cissé, M.; Farinella, G. M.; Hassner, T. Eds. Springer Cham, 398–415, 2022.

[27] Wang, W.; Pan, W.; Liu, X.; Su, K.; Rolfe, B.; Lu, X. Random screening-based feature aggregation for point cloud denoising. *Computers & Graphics* Vol. 116, 64–72, 2023.

[28] 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, 13530–13539, 2023.

[29] Casajus, P. H.; Ritschel, T.; Ropinski, T. Total denoising: Unsupervised learning of 3D point cloud cleaning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 52–60, 2019.

[30] Buades, A.; Coll, B.; Morel, J.-M. A non-local algorithm for image denoising. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 60–65, 2005.

[31] Tomasi, C.; Manduchi, R. Bilateral filtering for gray and color images. In: Proceedings of the 6th International Conference on Computer Vision, 839–846, 1998.

[32] Burger, H. C.; Schuler, C. J.; Harmeling, S. Image denoising: Can plain neural networks compete with BM3D? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2392–2399, 2012.

[33] Zhang, K.; Zuo, W.; Chen, Y.; Meng, D.; Zhang, L. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing* Vol. 26, No. 7, 3142–3155, 2017.

[34] Lempitsky, V.; Vedaldi, A.; Ulyanov, D. Deep image prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9446–9454, 2018.

[35] Batson, J.; Royer, L. Noise2Self: Blind denoising by self-supervision. In: Proceedings of the 36th International Conference on Machine Learning, 524–533, 2019.

[36] Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K. Q. Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2261–2269, 2017.

[37] 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: Proceedings of the 36th International Conference on Neural Information Processing Systems, Article No. 305, 4218–4230, 2022.

[38] Yu, L.; Li, X.; Fu, C. W.; Cohen-Or, D.; Heng, P. A. PU-net: Point cloud upsampling network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2790–2799, 2018.

[39] Wang, P. S.; Liu, Y.; Tong, X. Mesh denoising via cascaded normal regression. *ACM Transactions on Graphics* Vol. 35, No. 6, Article No. 232, 2016.

[40] Serna, A.; Marcotegui, B.; Goulette, F.; Deschaud, J. Paris-rue-Madame database - A 3D mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. In: Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods, Vol. 1, 819–824, 2014.

[41] Kazhdan, M.; Bolitho, M.; Hoppe, H. Poisson surface reconstruction. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing, 61–70, 2006.

[42] Lu, D.; Lu, X.; Sun, Y.; Wang, J. Deep feature-preserving normal estimation for point cloud filtering. *Computer-Aided Design* Vol. 125, 102860, 2020.



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



Xiao Liu is currently an associate professor and director for the Software Engineering Innovation Lab at the School of Information Technology, Deakin University. His main research areas include software engineering and distributed/service computing.



Hailing Zhou is an associate professor at Swinburne University of Technology in Australia. Her research areas include multi-modal AI, 3D vision, and vision techniques for robotics.



Lei Wei received his Ph.D. degree in July 2011 from Nanyang Technological University. His research fields include haptic rendering, interaction and collaboration, collision detection, HCI, function-based and physics-based modelling, as well as web visualisation.



Zhigang Deng is Moores Professor of Computer Science and the director of graduate studies at the Computer Science Department of the University of Houston. His research interests are in the broad areas of computer graphics, computer animation, etc.



Manzur Murshed is currently a professor of computer science in the School of Information Technology at Deakin University. His research interests include video technology, machine learning, big data analytics, and computational mathematics.



Xuequan Lu is a senior lecturer at La Trobe University. He received his Ph.D. degree from the College of Computer Science and Technology at Zhejiang University. His research interests include geometry modelling, processing, and analysis.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made.

The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

To view a copy of this licence, visit `http://creativecommons.org/licenses/by/4.0/`.

Other papers from this open access journal are available free of charge from http://www.springer.com/journal/41095. To submit a manuscript, please go to https://www.editorialmanager.com/cvmj.