



Original article

Confidence-guided progressive point cloud denoising via MSAA-EdgeConv

Moh Moh Thet Aung^a, JaeJun Yoo^b, Ida Bagus Krishna Yoga Utama^a, Su Mon Ko^a,
Muhammad Alfi Aldolio^a, May Thu^a, Muhammad Fairuz Mummtaz^a, Yeong Min Jang^{a,*}

^a Wireless Communication and Artificial Intelligence Lab, Kookmin University, 77 Jeongneung-ro, Seoul, 02707, Republic of Korea

^b DNA+ Drone Platform Research Center, Electronics and Telecommunications Research Institute (ETRI), Daejeon, 34129, Republic of Korea

ARTICLE INFO

Keywords:

Point cloud denoising
Progressive refinement
Confidence-guided
MSAA-EdgeConv
Multi-scale attention augmented edge convolution

ABSTRACT

Point cloud denoising is essential for improving the quality and reliability of noisy 3D point clouds, yet existing reconstruction-, distribution-, and displacement-based methods may suffer from geometric information loss, high computational complexity, or limited adaptation to spatially varying noise. To address these limitations, we propose a confidence-guided progressive refinement point cloud denoising network that refines noisy point clouds through stage-dependent coarse-to-fine geometric learning. At each stage, the network dynamically re-extracts local features using multi-scale attention augmented edge convolution (MSAA-EdgeConv) and jointly predicts point displacements and confidence scores. The confidence scores act as learned per-point weights that adaptively scale displacement updates, while stage-specific scale factors adjust the correction magnitude throughout refinement. Extensive experiments on synthetic benchmarks (PUNet and PCNet) and real-world datasets (Kinect v1, Kinect v2, and 3DCSR) demonstrate strong denoising performance and generalization across diverse noise conditions. We obtained 18.15 for CD and 6.83 for HD on the 10 K-point PU-Net dataset, showing better results than the 3DMambaIPF and C2AENet methods, which are current SOTA methods. On PUNet, the proposed models achieve the best CD across all tested noise levels and point densities, while on Kinect v1 and Kinect v2 they obtain the lowest CD without dataset-specific fine-tuning.

1. Introduction

With the rapid advancement of 3D acquisition technologies, such as LiDAR sensors and depth cameras, 3D point clouds have become a predominant data format for representing real-world objects and scenes. However, raw point clouds acquired from these devices are inevitably corrupted by noise and outliers due to inherent sensor limitations and environmental interference. Consequently, point cloud denoising has emerged as a fundamental yet challenging problem in 3D computer vision.

The difficulty of point cloud denoising stems from two competing objectives: aggressive noise removal risks erasing fine geometric details and sharp features, while conservative filtering leaves residual noise in high-variance regions. Furthermore, real-world noise is spatially non-uniform, different regions of the same point cloud may require fundamentally different levels of correction depending on local geometry and noise severity. These properties make the denoising problem significantly harder than its 2D image counterpart, where spatial regularity can be exploited.

Classical optimization relies on geometric priors, such as low-rank matrix approximation [1,2], but requires extensive parameter tuning

and often fails to balance noise reduction with feature preservation. Conversely, deep learning models including encoder-decoders [3,4] and score-based gradient fields [5–7] learn robust priors that significantly outperform classical methods. Despite this progress, current frameworks, including single-pass predictors [8–10], flow-based models [11], and self-supervised methods [12,13] largely treat denoising as a holistic, single-stage operation. This paradigm forces networks to resolve coarse corrections and fine details simultaneously, which is problematic under high noise.

Existing approaches exhibit category-specific limitations: reconstruction-based methods (e.g., DMRDenoise [14], TDNet [15]) risk geometric information loss from 2D projection or aggressive down-sampling; distribution-based methods (e.g., ScoreDenoise [6], PDFlow [11], P2P-Bridge [16]) face high computational complexity and potential instability in gradient-based sampling; and displacement-based methods (e.g., PointCleanNet [3], Pointfilter [4], IterativePFN [17]) typically apply corrections without an explicit per-point reliability signal. Our confidence-guided progressive refinement directly targets the last limitation, while MSAA-EdgeConv's dynamic re-extraction addresses the staleness of features noted in single-pass reconstruction- and distribution-based methods.

* Corresponding author.

E-mail address: yjang@kookmin.ac.kr (Y.M. Jang).

<https://doi.org/10.1016/j.aej.2026.09.007>

Received 25 March 2026; Received in revised form 29 July 2026; Accepted 13 September 2026

Available online 21 September 2026

1110-0168/© 2026 The Authors. Published by Elsevier B.V. on behalf of Faculty of Engineering, Alexandria University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

While iterative strategies [17,18] exist, they typically utilize fixed passes without adaptive correction or reliability assessment, often leading to error accumulation in ambiguous regions. Displacement-based methods [3,17,19] typically apply corrections without an explicit per-point reliability signal.

To address these limitations, we propose a confidence-guided denoising framework centered on the novel MSAA-EdgeConv. Our framework decomposes denoising into a cascade of lightweight refinement stages, each operating on the most recent estimate of the point cloud. We introduce two key architectural innovations. First, dynamic feature re-extraction utilizes MSAA-EdgeConv to rebuild the k -NN graph at each stage, ensuring features reflect the evolving geometric structure rather than the initial noise. Second, each stage outputs a confidence-weighted displacement; these learned per-point confidence scores modulate update magnitudes, enforcing conservative predictions in uncertain regions to prevent inter-stage error accumulation. For scenarios requiring stronger denoising, we further introduce geometry-aware attentive edge convolution (GAA-EdgeConv), which augments feature extraction with a local point density encoder to produce noise-robust representations.

The main contributions of this paper are summarized as follows:

1. **Confidence-Guided Progressive Refinement:** we propose a multi-stage cascaded architecture in which each stage jointly predicts a displacement vector and a per-point confidence score, allowing the network to make conservative corrections in uncertain regions and progressively refine point positions from coarse to fine without over-committing at any single stage.
2. **MSAA-EdgeConv:** we propose a novel graph convolution operator that augments standard edge convolution with channel-wise attention re-weighting and a grouped depthwise convolution path, enabling richer and more discriminative local feature extraction for dynamic graph-based denoising.
3. **GAA-EdgeConv:** for the heavy model variant, we introduce a geometry-aware operator that encodes local point density through a distance-based branch, fused with attention-weighted edge features, producing noise-robust representations that adapt to varying point distributions.

Extensive experiments on the synthetic benchmarks, as well as real-world scans, demonstrate that our method achieves competitive or superior denoising performance compared to SOTA approaches across diverse noise types and magnitudes.

2. Related work

2.1. Optimization-based denoising

Traditional denoising typically treats the task as a constrained optimization problem. Surface-fitting methods, such as Moving Least Squares (MLS) [20], approximate local geometry via polynomials but often struggle with sharp features. Projection-based approaches (e.g., LOP [21] and its variants [22,23]) project points onto objective-minimizing sets, though they are prone to over-smoothing. Other techniques utilize sparsity-based regularization [24] or graph-based Laplacian smoothing [25]. While theoretically grounded, these methods require extensive parameter tuning and struggle to preserve intricate details under high-noise conditions.

2.2. Deep learning-based denoising

Deep learning approaches have surpassed traditional methods by learning geometric priors from large datasets. These can be broadly categorized into supervised and unsupervised methods. Recent image restoration methods, such as Omni-Kernel Network for Image Restoration [26], Revitalizing Convolutional Network for Image Restoration

[27], and Focal Network for Image Restoration [28], have demonstrated the effectiveness of adaptive receptive fields, convolutional revitalization, and focal modulation for degradation removal.

Although these methods are designed for 2D images, they provide useful context for the broader restoration literature and motivate the use of multi-scale and context-aware feature modeling in our geometry-aware point-cloud denoising framework. In 3D point cloud denoising, these concepts can be adapted to model multi-scale local neighborhoods and geometric dependencies for effective noise removal while preserving surface details.

Recent multi-dimensional uncertainty estimation methods [29] based on MIMO system-identification frameworks have explored the characterization of unknown structures and uncertainty in complex systems; although distinct from 3D geometric processing, they provide broader context for our per-point confidence mechanism, which regulates displacement refinement under sensor-noise and geometry-related uncertainties.

2.2.1. Supervised learning methods

Supervised methods leverage paired noisy and clean data to learn mappings or distributions. We categorize them based on their core modeling strategy:

Reconstruction-based Approaches: Instead of noise regression, these methods reconstruct clean point sets from underlying geometry. PointProNet [30] projects 3D points onto 2D height maps to leverage CNN-based surface consolidation. DMRDenoise [14] utilizes a downsample-upsample architecture to recover manifolds, while Transformer-based TDNet [15] captures global dependencies for distribution reconstruction. However, these approaches often suffer from geometric information loss; 2D projections introduce distortions, and aggressive downsampling frequently discards fine structural details.

Distribution-based Approaches: These methods model point cloud distributions to guide denoising. ScoreDenoise [6] and DeepPSR [5] utilize score function estimation and global gradient fields, respectively, to shift points via gradient ascent. PDFlow [11] employs normalizing flows for latent space disentanglement, while P2P-Bridge [16] treats denoising as a Schrödinger bridge optimal transport problem. Despite their theoretical rigor, these methods face high computational complexity in modeling high-dimensional distributions. More recent methods further extend this direction through DPCD [31], FGRNN [32], UDF-guided [33] implicit fields to improve restoration stability, enhance geometric representation, and reduce surface drift noise. Moreover, gradient-based sampling often suffers from instability and convergence issues, particularly in high-noise scenarios.

Filter-based Approaches: Rather than direct coordinate regression, these methods learn to guide traditional filtering rules. GeoDual-CNN [34] and PCDNF [35] utilize multi-task frameworks to jointly estimate surface normals and denoising updates, while PointFilterNet [19] learns local filtering coefficients for outlier recognition and progressive refinement. However, these approaches are highly dependent on the accuracy of intermediate features; for example, erroneous normal estimations propagate directly to position updates, often inducing geometric artifacts.

Classification-based Approaches: These methods focus on identifying and removing specific types of noise (weather artifacts or outliers) rather than displacing points. WeatherNet [36] and 4DenoiseNet [37], are designed for LiDAR data to segment and remove points caused by rain or snow. RocketPCD [38] applies attention mechanisms to identify noise in industrial scans. These approaches are limited by their binary nature; they simply remove points rather than correcting them, which can lead to data sparsity in the surface rather than a reconstructed geometry. The local neighborhood features method [39] is constrained by high parameter sensitivity, risking structural degradation if real points are misidentified as noise and removed.

Displacement-based Approaches: Most recent methods predict a displacement vector for each noisy point to shift it toward the

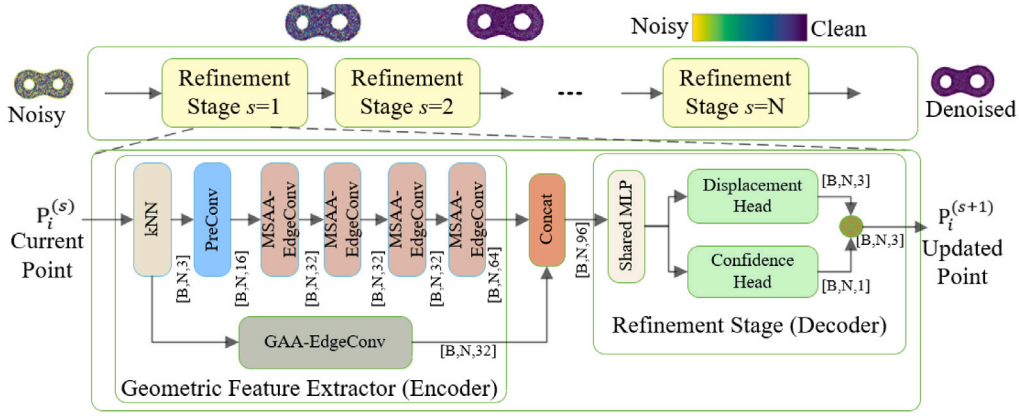


Fig. 1. Overall architecture of confidence-guided progressive point cloud denoising method. The network iteratively refines noisy point clouds through confidence-weighted geometric displacements computed from dynamically extracted local features across s stages.

underlying surface. PointCleanNet [3] pioneered a two-stage outlier classification and correction process, while Pointfilter [4] uses an encoder–decoder architecture with sharp-feature-sensitive loss. Graph-based methods like GPDNet [40] leverage convolutional networks for irregular structures. Recent advancements emphasize iterative refinement: RePCD-Net [18] employs recurrent networks for multi-scale noise removal, and IterativePFN [17] models filtering via stacked encoder–decoders. PathNet [41] introduces dynamic routing based on local complexity, and 3DMambaPF [42] utilizes Selective State Space Models for efficiency. HeteroDenoise [43] further addresses spatially varying noise by using intensity-guided regression to adapt displacement prediction under heterogeneous noise conditions. BDNNet [44] is limited by lower time efficiency and potential structural gaps, with GPU memory usage increasing significantly as attention heads are added. FCNet [45] targets feature-domain noise through non-local self-similarity and teacher-student learning, yet it remains vulnerable to over-regularization and geometric information loss.

Unlike these methods, which often apply uniform corrections, our approach introduces a confidence-weighted mechanism to adaptively scale displacements progressively, addressing the robustness limitations of existing iterative frameworks in regions with varying noise magnitudes.

2.2.2. Unsupervised learning methods

Unsupervised methods bypass ground-truth requirements by leveraging structural priors. TotalDenoising [46] utilizes spatial density clustering, while Noise4Denoise [47] and U-CAN [13] adopt self-supervised displacement prediction and “Noise-to-Noise” consistency, respectively. Despite their utility, these approaches often struggle when statistical priors fail; for example, density-based assumptions frequently misclassify sparse geometric features as noise, leading to over-smoothing.

3. Method

3.1. Overview

As illustrated in Fig. 1, our framework progressively recovers clean surfaces by decomposing denoising into a coarse-to-fine sequence of s refinement stages. In this work, “progressive” refers to the stage-dependent coarse-to-fine nature of this process. Specifically, each stage operates on the point cloud refined by the previous stage, dynamically re-extracts geometric features, and applies a stage-specific correction. Thus, the denoising process is not merely repeated with identical behavior; it progressively transitions from coarse corrections in early stages to fine geometric adjustments in later stages.

Given a noisy input $P_{\text{noisy}} \in \mathbb{R}^{N \times 3}$, a shared Geometric Feature Extractor (GFE) utilizing hierarchical MSAA-EdgeConv captures local surface characteristics to produce features $F^{\text{geo}} = \{f_i \in \mathbb{R}^{64}\}_{i=1}^N$. At each

Algorithm 1 Confidence-Guided Progressive Point Cloud Denoising

Require: Noisy point cloud $P^{(0)} = P_{\text{noisy}} \in \mathbb{R}^{N \times 3}$; number of stages S , where $S = 5$ for the light variant and $S = 3$ for the heavy variant; variant $v \in \{\text{light}, \text{heavy}\}$

Ensure: Denoised point cloud $P^{(S)} \in \mathbb{R}^{N \times 3}$

- 1: **for** $s = 0, 1, \dots, S - 1$ **do**
- 2: $F_s^{(\text{geo})} \leftarrow \text{GFE}_{\text{MSAA}}(P^{(s)})$
- 3: **if** $v = \text{heavy}$ **then**
- 4: $F_s^{(\text{gaa})} \leftarrow \text{GFE}_{\text{GAA}}(P^{(s)})$
- 5: $F_s^{(\text{fused})} \leftarrow \text{FF}([F_s^{(\text{geo})} \parallel F_s^{(\text{gaa})}])$
- 6: **else**
- 7: $F_s^{(\text{fused})} \leftarrow F_s^{(\text{geo})}$
- 8: **end if**
- 9: $\Delta^{(s)} \leftarrow \alpha_s \psi_{\text{disp}}(F_s^{(\text{fused})})$
- 10: $C^{(s)} \leftarrow \sigma(\psi_{\text{conf}}(F_s^{(\text{fused})}))$
- 11: $P^{(s+1)} \leftarrow P^{(s)} + C^{(s)} \odot \Delta^{(s)}$
- 12: **end for**
- 13: **return** $P^{(S)}$

stage, the Refinement Stage (RS) jointly estimates confidence weights $C^{(s)} \in \mathbb{R}^{N \times 1}$ and displacement vectors $\Delta^{(s)} \in \mathbb{R}^{N \times 3}$ from dynamically re-extracted features. The point positions are updated via: $P^{(s+1)} = P^{(s)} + C^{(s)} \odot \Delta^{(s)}$. This gating mechanism ensures conservative updates in ambiguous regions while driving structural correction in certain ones. For extreme noise, a high-capacity variant employs GAA-EdgeConv to extract attentive features $F^{\text{gaa}} = \{f_i \in \mathbb{R}^{32}\}_{i=1}^N$. These are integrated via a fusion module: $F^{\text{fused}} = \text{FF}([F^{\text{geo}} \parallel F^{\text{gaa}}]) \in \mathbb{R}^{96}$, where $\parallel$ denotes concatenation. This multi-modal representation explicitly distinguishes geometric structures from noise artifacts, enhancing robustness under severe corruption. The overall of the proposed algorithm is shown in Algorithm 1.

3.2. Geometric feature extraction (Encoder)

The GFE serves as the foundation of our progressive denoising architecture, capturing multi-scale local surface characteristics through a hierarchical MSAA-EdgeConv framework. Given a point cloud $P = \{p_i \in \mathbb{R}^3\}_{i=1}^N$ at any refinement stage, the GFE produces discriminative geometric features that encode local surface normals, curvature patterns, and neighborhood distributions. The architecture consists of designed layers that progressively accumulate features through concatenation, creating a rich multi-scale representation suitable for displacement prediction.

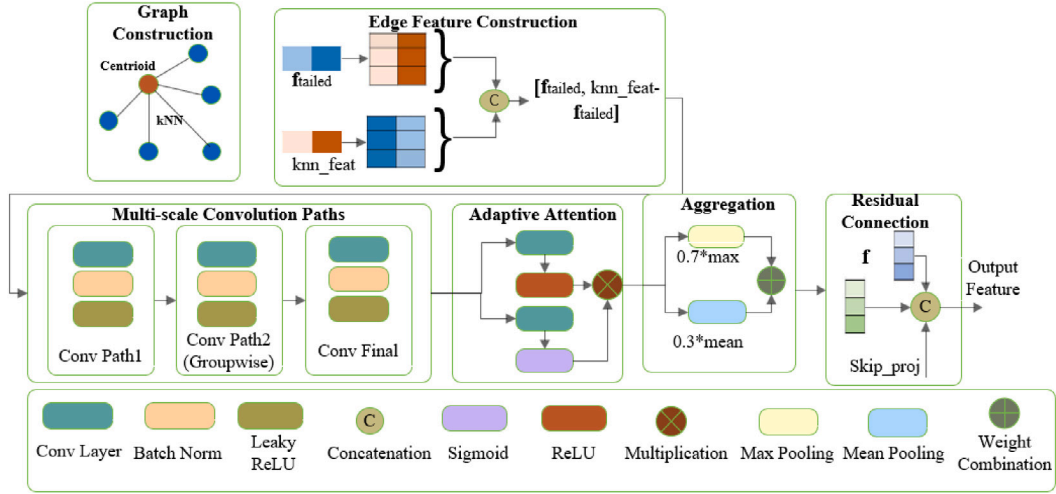


Fig. 2. Architecture of MSAA-EdgeConv module. The module constructs local edge features from k-NN graphs, processes them through a three-layer convolution cascade with groupwise convolution, applies channel-wise attention, and performs dual aggregation (max+mean pooling). Skip connections enable residual learning for enhanced feature extraction.

3.2.1. k-nearest neighbor graph construction

The first step in GFE is establishing local neighborhood structures through k-nearest neighbors (k-NN) graph construction. This fundamental operation transforms an unstructured point cloud into a structured graph representation that encodes spatial relationships between points, enabling subsequent geometric feature learning through local neighborhood aggregation [17,48].

Following [48], we construct a directed k-NN graph $G = (V, E)$ with $k = 32$ neighbors per point based on Euclidean distance, consistent with prior graph-based denoising approaches [17,40]. Crucially, this graph is rebuilt at every refinement stage rather than computed once from the initial noisy input. This dynamic reconstruction ensures that neighborhood relationships reflect the progressively refined geometry at each stage, allowing feature extraction quality to improve as denoising advances [17,48].

3.2.2. Pre-convolution layer for feature initialization

To initialize features from raw 3D coordinates, a preliminary convolution (PreConv) generates a 16-dimensional representation via relative edge convolution [48]:

$$\mathbf{F}_i^{(0)} = \max_{j \in \mathcal{N}(p_i)} \text{MLP}([p_i \parallel (p_j - p_i)]). \quad (1)$$

This lightweight initialization provides stable low-level surface cues for the subsequent hierarchical MSAA-EdgeConv layers, accelerating training convergence [48].

3.2.3. Hierarchical MSAA-edge convolution with feature accumulation

The GFE employs a hierarchical architecture that constructs a multi-scale feature pyramid through progressive concatenation. By accumulating features from all preceding layers, the design captures multi-level geometric abstractions and scale-invariant representations simultaneously. Let $F^{(l)} \in \mathbb{R}^{N \times d_l}$ denote the feature representation at layer l , where $l \in \{0, 1, 2, 3, 4\}$ represents the layer index and d_l is the feature dimension at that layer. Our hierarchical MSAA-EdgeConv with feature accumulation follows the general recursive formulation:

$$F^{(l)} = ([F^{(0)} \parallel \dots \parallel F^{(l-1)}]) : \mathbb{R}^{D_{in}^{(l)}} \rightarrow \mathbb{R}^{d_l}. \quad (2)$$

where: $[F^{(0)} \parallel F^{(1)} \parallel \dots \parallel F^{(l-1)}]$ denotes the concatenation of feature representations from layers 0 to $l-1$ of MSAA-EdgeConv. The input feature dimension grows progressively as $D_{in}^{(l)} = \sum_{i=0}^{l-1} d_i$. The output feature dimension d_l is layer-specific. This acts as a feature bottleneck, distilling multi-scale geometric information into a compact, discriminative representation. By accumulating features, each layer accesses

both low-level geometric details and high-level structural information, preserving fine-scale features while enabling abstract geometric reasoning.

As shown in Fig. 2, for each layer l of MSAA-EdgeConv, local geometric relationships are explicitly modeled through edge feature construction. For each point, the edge feature is defined as $\mathbf{e}_{i,j}^{(l)}$. This formulation jointly encodes absolute multi-scale context and relative geometric variation, enabling the network to capture surface orientation changes, sharp edges, and local curvature patterns. The resulting edge features are transformed using a shared MLP implemented via 1×1 convolutions: $\mathbf{h}_{i,j}^{(l)}$.

$$\begin{aligned} \mathbf{h}_{i,j}^{(l)} &= \sigma \left(\text{BN} \left(\mathbf{W}_1^{(l)} * \mathbf{e}_{i,j}^{(l)} \right) \right), \\ \mathbf{h}_{i,j}^{(l,2)} &= \sigma \left(\text{BN} \left(\mathbf{W}_2^{(l)} \mathbf{h}_{i,j}^{(l)} \right) \right), \\ \mathbf{h}_{i,j}^{(l,3)} &= \sigma \left(\text{BN} \left(\mathbf{W}_3^{(l)} \mathbf{h}_{i,j}^{(l,2)} \right) \right). \end{aligned} \quad (3)$$

where $\mathbf{W}_i^{(l)}$ denotes learnable weights, $\text{BN}(\cdot)$ represents batch normalization, and $\sigma(\cdot)$ is a LeakyReLU activation. Since the MLP is shared across all edges, the operation remains permutation invariant with respect to neighbor ordering with $\mathbf{h}_{i,j}^{(l,2)}$, followed by a projection layer that maps features to the target dimensionality $\mathbf{h}_{i,j}^{(l,3)}$. An attention mechanism, $\mathbf{a}_{i,j}^{(l)}$ is introduced to adaptively reweight channel responses. The attended edge features are aggregated across the neighborhood using a weighted combination of max and mean pooling.

$$\begin{aligned} \mathbf{a}_{i,j}^{(l)} &= \text{Sigmoid} \left(\mathbf{W}_a^{(2)} * \text{ReLU} \left(\mathbf{W}_a^{(1)} * \mathbf{h}_{i,j}^{(l,3)} \right) \right), \\ \mathbf{F}_i^{(l)} &= 0.7 \max_{j \in \mathcal{N}(p_i)} \left(\mathbf{a}_{i,j}^{(l)} \odot \mathbf{h}_{i,j}^{(l,3)} \right) + 0.3 \text{mean}_{j \in \mathcal{N}(p_i)} \left(\mathbf{a}_{i,j}^{(l)} \odot \mathbf{h}_{i,j}^{(l,3)} \right). \end{aligned} \quad (4)$$

where $\odot$ denotes element-wise multiplication. This aggregation strategy preserves sharp geometric responses through max pooling while maintaining robustness to noise via mean pooling. The resulting features $\mathbf{F}_i^{(l)}$ are concatenated with previous layers according to Eq. (2). In Eq. (4), the coefficients 0.7 and 0.3 are empirically selected hyperparameters for feature aggregation. Specifically, we use a max-dominant convex combination to balance geometric detail preservation (max pooling) and noise robustness (mean pooling). These values were chosen based on preliminary validation experiments as a stable trade-off across tested noise settings, rather than from a closed-form analytical derivation.

3.3. Refinement stage (Decoder)

The core innovation of our method is a RS that iteratively refines point positions via learned displacement predictions [3,17]. Unlike

Iterative-PFN [17], which applies uniform iterative corrections, our method introduces dual-modulation: each displacement is jointly gated by a learnable stage-wise scale parameter and a per-point confidence score. The scale parameters enforce progressive attenuation across stages, while the confidence scores adaptively suppress unreliable predictions at the point level, together enabling stable, fine-grained denoising of heavily corrupted point clouds without over-smoothing.

Each refinement stage $s \in \{1, 2, \dots, S\}$ implements a specialized neural network that predicts how each point should move to better approximate the underlying clean surface. The stage network consists of three mechanisms: stage-wise displacement prediction, a confidence-weighted mechanism, and dynamic feature re-extraction and adaptation.

3.3.1. Stage-wise displacement prediction

The point-wise feature at stage s is denoted $h^{(s)} = \phi^{(s)} f^{(s)} : \mathbb{R}^{64} \rightarrow \mathbb{R}^{128}$, where $\phi^{(s)}$ is a two-layer 1D convolutional network with batch normalization and ReLU.

At each refinement stage s , the displacement head predicts a residual 3D displacement vector modulated by a learnable stage-specific scaling factor α_s , enabling coarse-to-fine magnitude control. The displacement function is a two-layer MLP with a bottleneck structure: the first layer projects to a lower-dimensional representation and the second maps it to a 3D vector. The displacement prediction is defined as

$$\begin{aligned} \Delta_i^{(s)} &= \alpha_s \cdot \psi_{\text{disp}}^{(s)} \left(\mathbf{h}_i^{(s)} \right), \\ \psi_{\text{disp}}^{(s)} &= \mathbf{W}_{\text{disp2}}^{(s)} \cdot \text{ReLU} \left(\mathbf{W}_{\text{disp1}}^{(s)} \mathbf{h}_i^{(s)} \right), \\ \alpha_s^{(0)} &= 0.1 \times \left(1 - \frac{s}{S} \right) \end{aligned} \quad (5)$$

where $\mathbf{h}_i^{(s)} \in \mathbb{R}^{128}$ denotes the point-wise feature at stage s , $\mathbf{W}_{\text{disp1}}^{(s)} \in \mathbb{R}^{64 \times 128}$ projects the input features to a 64-dimensional latent space, and $\mathbf{W}_{\text{disp2}}^{(s)} \in \mathbb{R}^{3 \times 64}$ maps the compressed representation to the final 3D displacement vector. The scale parameter α_s is a learnable scalar initialized as which progressively decreases the initial displacement magnitude for deeper refinement stages. This initialization embodies the coarse-to-fine principle: early stages have larger scales and make aggressive corrections to remove prominent noise, while later stages have smaller scales and perform fine adjustments to recover geometric details.

3.3.2. Confidence-weighted update mechanism

A critical innovation in our approach is the confidence-weighted update mechanism, which enables the network to selectively apply displacements based on local geometric certainty. The confidence mechanism allows the network to be aggressive in high-confidence regions and conservative in uncertain areas. For each point, we predict a scalar confidence score:

$$\begin{aligned} c_i^{(s)} &= \sigma \left(\psi_{\text{conf}}^{(s)} \left(\mathbf{h}_i^{(s)} \right) \right) \in [0, 1], \\ \psi_{\text{conf}}^{(s)} &= \mathbf{W}_{\text{conf2}}^{(s)} \cdot \text{ReLU} \left(\mathbf{W}_{\text{conf1}}^{(s)} \mathbf{h}_i^{(s)} \right) \end{aligned} \quad (6)$$

where $\sigma(\cdot)$ is the sigmoid function ensuring outputs in $[0, 1]$ and $\psi_{\text{conf}}^{(s)}$ is a two-layer MLP. $\mathbf{W}_{\text{conf1}}^{(s)} \in \mathbb{R}^{32 \times 128}$ projects the input features to a 32-dimensional latent space, and $\mathbf{W}_{\text{conf2}}^{(s)} \in \mathbb{R}^{1 \times 32}$ maps the reduced representation to a scalar logit. The confidence branch shares the processed features $\mathbf{h}_i^{(s)}$ with the displacement prediction head, ensuring computational efficiency and promoting coherent, mutually consistent predictions. This mechanism provides several desirable properties. First, it enables *adaptive step sizes*, where high confidence values ($c_i \approx 1$) apply the full predicted displacement, while low confidence values ($c_i \approx 0$) result in minimal updates, effectively realizing point-wise adaptive learning rates. Second, it ensures *stability*, since $0 \leq c_i \leq 1$ guarantees that the applied displacement never exceeds the predicted offset, preventing oscillations or divergence that may arise from unmodulated

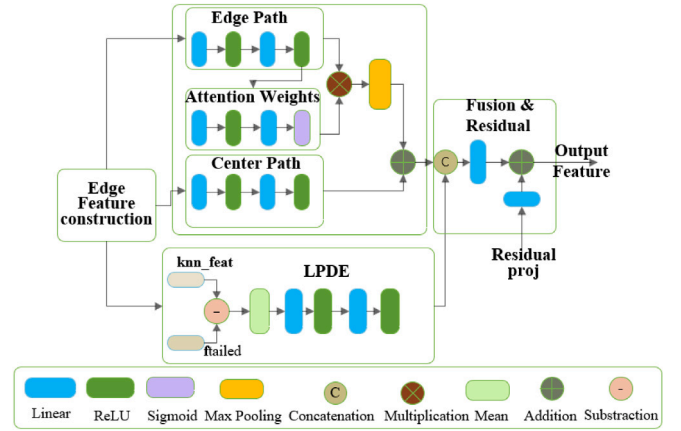


Fig. 3. GAA-EdgeConv feature extraction of heavy network.

updates. Finally, the confidence term acts as an *implicit regularizer*, encouraging conservative corrections in geometrically ambiguous regions and thereby improving robustness and generalization.

The point position is updated according to:

$$\mathbf{p}_i^{(s+1)} = \mathbf{p}_i^{(s)} + c_i^{(s)} \odot \Delta_i^{(s)} \quad (7)$$

where $\odot$ denotes element-wise multiplication. This update rule has several desirable properties.

Point cloud denoising involves uncertainty from sensor noise and ambiguous local geometry. Sensor uncertainty arises from unknown Gaussian, Laplace, non-isotropic, or uniform-ball perturbations, while geometric uncertainty occurs near sharp edges, thin structures, and highly curved surfaces. Uncertain points at each refinement stage are identified using the learned confidence score. For interpretability, the confidence-derived uncertainty is defined as: $u_i^{(s)} = 1 - c_i^{(s)}$. Thus, lower confidence corresponds to higher update uncertainty. Through the confidence-weighted update in Eq. (7), low-confidence points receive smaller effective displacements and are refined more conservatively. This uncertainty represents a learned update-confidence measure rather than a fixed geometric-region label or a direct estimate of P2M error.

3.3.3. Dynamic feature re-extraction and adaptation

After each refinement stage updates the point positions, geometric features are re-extracted from the updated point cloud using the shared GFE: $f_i^{(s+1)} = \Phi(\mathbf{p}^{(s+1)})$, where $\Phi(\cdot)$ denotes the feature extraction process described in Section 3.2. This allows feature representations to evolve together with the progressively refined geometry. To support effective information transfer between stages, a lightweight stage-specific linear transformation with batch normalization and ReLU adapts the re-extracted features before they are passed to the next refinement module.

3.4. Heavy network

Heavy network provides explicit noise intelligence to compensate for having fewer refinement passes, making each stage more effective through informed feature fusion.

3.4.1. GAA-EdgeConv feature extraction

The proposed module enhances standard graph operations by integrating explicit geometric density encoding with an attentive aggregation mechanism, as shown in Fig. 3. For each point $\mathbf{p}_i$ and its corresponding k -nearest neighbors $\mathbf{p}_{i,j} \in \mathcal{N}(\mathbf{p}_i)$, we define the edge features $\mathbf{e}_{i,j}^{(l)} = \text{Concat}(\mathbf{p}_{i,j}, \mathbf{p}_{i,j} - \mathbf{p}_i)$ to capture both absolute and relative information [48]. To explicitly model local sparsity and improve robustness against outliers, a Local Point Density Encoder

(LPDE) computes the average Euclidean distance in the neighborhood, $d_{avg} = \frac{1}{k} \sum_{j=1}^k \|\mathbf{p}_{i,j} - \mathbf{p}_i\|_2$, and projects this metric into a geometric embedding vector $\mathbf{g}_i$ via a multi-layer perceptron. Simultaneously, to filter irrelevant or noisy edges, an adaptive attention branch calculates scalar importance weights $\alpha_{i,j} = \sigma(\Phi_{att}(\mathbf{h}_{i,j}))$, where $\mathbf{h}_{i,j}$ represents the transformed edge features derived from $\mathbf{e}_{i,j}^{(l)}$. These weights modulate the aggregation process via attention-weighted max-pooling: $\mathbf{x}_i^{agg} = \max_{j \in \{1, \dots, k\}} (\alpha_{i,j} \cdot \mathbf{h}_{i,j})$. The final noise aware feature representation is synthesized by concatenating the aggregated context $\mathbf{x}_i^{agg}$, transformed central features $\mathbf{x}_i^{center}$, and the geometric embedding $\mathbf{g}_i$, followed by a linear fusion and a residual update:

$$\mathbf{F}_i^{(gaa)} = \Phi_{final}(\text{Concat}(\mathbf{x}_i^{agg} + \mathbf{x}_i^{center}, \mathbf{g}_i)) + \mathcal{R}(\mathbf{p}_i). \quad (8)$$

where $\mathcal{R}$ ensures dimensional alignment for the skip connection.

3.4.2. Feature fusion

The heavy network integrates the geometric features and attentive features are concatenated to form a combined representation:

$$\mathbf{F}^{fused} = \text{FF}([F^{geo} \parallel F^{gaa}]) \in \mathbb{R}^{64+32} = \mathbb{R}^{96}. \quad (9)$$

The fused features undergo a 1D convolutional transformation with batch normalization and ReLU activation, yielding 96 dimensional point-wise representations. This expanded feature space enhances representational capacity over the 64 dimensional baseline, enabling robust handling of noise.

3.5. Loss function

The network is trained using a multi-stage supervised learning strategy, where the output at each refinement stage is explicitly supervised to enable coarse-to-fine noise removal. To clarify the optimization problem, the total multi-stage training loss in Eq. (10) is used as the cost function minimized during training. Let γ denote all trainable network parameters, and let P_{noisy}, P_{gt} denote the input noisy point cloud and ground-truth clean point cloud, respectively. For refinement stage s , P_s denotes the predicted point cloud and T_s denotes the stage-wise supervision target. The objective is to learn γ by minimizing the weighted sum of stage-wise denoising losses.

Let $\mathbf{P}_{noisy}, \mathbf{P}_{gt} \in \mathbb{R}^{N \times 3}$ denote the input noisy and ground truth clean point and $\mathbf{P}_s \in \mathbb{R}^{N \times 3}$ the predicted output at the refinement stage $s \in \{1, \dots, S\}$. The total training loss is defined as a weighted sum of stage-wise losses,

$$\mathcal{L}_{total} = \sum_{s=1}^S w_s \mathcal{L}_{EMD}(\mathbf{P}_s, \mathbf{T}_s), \quad (10)$$

$$w_s = \frac{s^\gamma}{\sum_{j=1}^S j^\gamma}, \quad \gamma = 1.5$$

which assigns larger weights to later refinement stages. We set $\gamma = 1.5$ as a fixed design choice so that supervision weight increases monotonically with stage index, ensuring early stages receive sufficient gradient signal for coarse correction while later stages are more strongly supervised toward the final clean geometry. The supervision target at stage s is defined as $T_s = P_{gt} + \xi_s(P_{noisy} - P_{gt})$, where $\xi_s = (S - s)/S$. This stage-dependent coefficient gradually decreases as refinement progresses, so early stages emphasize coarse noise suppression, while the final stage enforces full denoising toward the ground-truth point cloud. Each stage loss $\mathcal{L}_{EMD}$ is computed using the Earth Mover's Distance (EMD) [49].

$$\mathcal{L}_{EMD}(\mathbf{P}_s, \mathbf{T}_s) = \min_{\phi \in \Pi} \sum_{i=1}^N \|\mathbf{p}_i^s - \mathbf{t}_{\phi(i)}^s\|_2. \quad (11)$$

where Π denotes the set of bijective point correspondences. EMD is adopted because point clouds are unordered sets, and it provides permutation-invariant matching between predicted and target points. The stage-wise weights emphasize later refinement stages while still

supervising intermediate outputs, supporting the proposed coarse-to-fine denoising process. For consistency across refinement stages, the correspondence is computed once between the noisy input and the stage target and reused during loss evaluation.

4. Experiments

4.1. Datasets

We evaluate the proposed method against SOTA approaches using two synthetic datasets (PU-Net [50] and PCNet [3]) and two real-world scanned datasets (Kinect v1/v2 [51] and 3DCSR [52]). Following the standard protocol in PUNet [50], models are trained on PU-Net meshes sampled at 10 K, 30 K, and 50 K densities with varying Gaussian noise levels. Real-world performance is further validated on Kinect and PCNet, with 3DCSR used for qualitative visualization of sensor-specific noise mitigation.

4.2. Evaluation metrics

We assess denoising quality using three standard metrics computed on unit sphere-normalized point clouds. The Chamfer Distance (CD) [49] measures the bidirectional average nearest-neighbor discrepancy between the predicted and ground-truth point sets:

$$\mathcal{L}_{CD}(\mathbf{P}, \mathbf{P}_{gt}) = \frac{1}{N} \sum_{i=1}^N \min_{1 \leq j \leq M} \|\mathbf{p}_i - \mathbf{p}_j^{gt}\|_2^2 + \frac{1}{M} \sum_{j=1}^M \min_{1 \leq i \leq N} \|\mathbf{p}_j^{gt} - \mathbf{p}_i\|_2^2 \quad (12)$$

This metric evaluates the overall geometric similarity between two point clouds.

The Point-to-Mesh (P2M) distance [53] measures how closely the predicted points adhere to the underlying surface. Let $\mathcal{F}$ denote the set of triangular faces of the ground-truth mesh $\mathcal{M}$. The P2M metric is defined as:

$$\mathcal{L}_{P2M}(\mathbf{P}, \mathcal{M}) = \frac{1}{N} \sum_{i=1}^N \min_{f \in \mathcal{F}} d_\Delta^2(\mathbf{p}_i, f). \quad (13)$$

where $d_\Delta^2(\mathbf{p}_i, f)$ is the squared Euclidean distance from point $\mathbf{p}_i$ to its nearest triangular face f . Unlike CD, P2M is strictly one-directional and directly penalizes surface deviation, making it the most sensitive indicator of whether confidence-weighted displacements correctly place points on the true surface manifold.

The Hausdorff Distance (HD) [54] captures the maximum deviation in both directions, identifying worst-case errors and localized artifacts.

$$\mathcal{L}_{HD}(\mathbf{P}, \mathbf{P}_{gt}) = \max \left\{ \max_{1 \leq i \leq N} \min_{1 \leq j \leq M} \|\mathbf{p}_i - \mathbf{p}_j^{gt}\|_2^2, \max_{1 \leq j \leq M} \min_{1 \leq i \leq N} \|\mathbf{p}_j^{gt} - \mathbf{p}_i\|_2^2 \right\} \quad (14)$$

All metrics are implemented using PyTorch3D [54] with unit bounding sphere normalization applied to ensure fair comparison across different models. Lower values indicate better performance for all three metrics. In this paper, all of the CD and P2M are multiplied by 10^5 while HD is multiplied by 10^4 .

4.3. Implementation details

The input noisy point cloud consists of 1024 points per patch. The proposed progressive denoising network refines the geometry over multiple stages, using 5 stages for the light model and 3 stages for the heavy model. The Light model employs five refinement stages with a feature dimension of 64 and a hidden dimension of 128. The Heavy model employs three refinement stages with a feature dimension

Table 1

Quantitative comparison on the PUNet test set.

Method	10K (Light)									50K (Dense)								
	1% noise			2% noise			3% noise			1% noise			2% noise			3% noise		
	CD	P2M	HD	CD	P2M	HD	CD	P2M	HD	CD	P2M	HD	CD	P2M	HD	CD	P2M	HD
ScoreDenoise	25.21	4.63	17.21	36.86	10.74	52.27	47.08	19.42	65.84	7.16	<i>1.50</i>	18.61	12.88	5.66	45.62	19.28	10.41	87.83
PDFlow	21.20	3.80	10.66	32.50	10.20	37.10	44.50	20.90	64.45	6.52	4.17	11.13	14.27	10.56	38.14	39.11	32.45	77.76
PCDNF	25.57	7.73	12.26	37.50	13.47	30.70	47.84	22.12	64.69	6.71	3.48	6.17	10.17	5.77	25.25	15.83	10.02	53.70
IterativePFN	20.56	5.01	9.29	30.43	8.45	16.66	42.42	16.89	25.87	6.05	3.02	<i>3.74</i>	8.03	4.36	10.80	19.70	13.02	29.86
PathNet	31.84	13.07	16.80	41.07	17.07	32.20	48.60	22.57	55.60	7.29	3.86	8.49	11.60	6.85	30.22	26.12	18.42	83.38
SVCNet	24.12	25.37	12.32	34.32	28.19	30.70	48.64	36.26	64.69	6.26	3.02	6.01	8.47	5.24	16.68	13.63	10.00	23.12
P2P-Bridge	22.83	3.90	50.12	32.00	8.10	41.56	39.90	14.20	45.65	5.90	0.90	26.19	9.02	3.20	69.64	15.60	8.40	88.63
C2AENet	19.60	4.59	11.86	30.09	8.00	14.51	43.70	27.37	34.33	5.81	2.83	5.78	7.58	4.05	14.35	14.40	9.12	22.88
3DMambaIPF	19.89	4.77	8.07	29.95	<i>8.03</i>	15.83	37.19	13.23	23.85	5.89	2.91	3.75	7.55	4.05	<i>8.80</i>	14.55	9.24	22.94
Our (Heavy s = 3)	<i>18.25</i>	5.04	<i>7.26</i>	<i>26.65</i>	9.02	<i>13.21</i>	<i>33.60</i>	<i>12.07</i>	26.60	<i>4.87</i>	1.95	3.24	<i>7.68</i>	<i>2.79</i>	8.72	11.97	5.52	<i>22.05</i>
Our (Light s = 5)	18.15	4.99	6.83	26.21	8.80	13.19	33.31	12.06	28.37	4.83	1.55	3.78	7.25	2.01	9.34	12.70	5.05	22.51

CD and P2M are multiplied by 10^5 while HD is multiplied by 10^4 . **Bold** indicates the best result and *italic* indicates the second-best result.

of 96 and a hidden dimension of 256, together with the additional GAA-EdgeConv module for enhanced geometric feature extraction.

All models are implemented and trained end-to-end using PyTorch [55] with the Adam optimizer [56], using $\beta_1 = 0.9$, $\beta_2 = 0.999$, an initial learning rate of 0.001, and a weight decay of 1×10^{-5} . A ReduceLROnPlateau scheduler adjusts the learning rate when validation performance plateaus. Unless otherwise specified, the models are trained for 150 epochs with a batch size of 8. A smaller batch size is used for the Heavy model when required by GPU memory limitations. Data augmentation includes random rotations around the three coordinate axes and uniform scaling within the range [0.8, 1.2].

The learning law is implemented by back-propagating the gradient of the total cost function through all refinement stages and updating all trainable parameters using Adam. At each training iteration, the network predicts the progressively refined point clouds, computes the multi-stage cost in Eq. (10), and updates the parameters end-to-end. All experiments are conducted on an NVIDIA RTX 4090 GPU using PyTorch 2.7.1, CUDA 12.9, and Python 3.13.9.

For the PU-Net and PCNet benchmarks, baseline results are taken directly from the original publications when the corresponding studies follow the standard protocol of training on 40 meshes and applying Gaussian noise of the unit-sphere bounding radius. When a specific metric or noise level is not reported, the corresponding method is re-evaluated using our unified pipeline. For Kinect v1 and Kinect v2, published results may rely on different preprocessing procedures. Therefore, all compared methods are re-evaluated using official pre-trained checkpoints released by the original authors under the same evaluation pipeline. No baseline model is retrained. For all checkpoint-based evaluations, the same procedure is applied consistently. Each point cloud is first centered at the midpoint of its bounding box and scaled so that the maximum distance from any point to the center is 1. After normalization, Gaussian noise is independently added to each point. Input patches containing 1024 points are then extracted using Farthest Point Sampling. Finally, CD, HD, and P2M are computed using PyTorch3D [55] on the unit-sphere-normalized outputs, with CD and P2M multiplied by 1×10^{-5} and HD multiplied by 1×10^{-4} for reporting.

4.4. Result on PUNet dataset

As shown in Table 1, our 5-stage configuration achieves the best CD across all PUNet settings; HD is best in the majority of configurations, while P2M is strongest at higher noise levels. The performance gap is most pronounced at high noise levels (3%), where baselines like PDFlow and P2P-Bridge degrade significantly while our model maintains stronger global structural integrity as measured by CD. Although P2P-Bridge and C2AENet [57] occasionally yield lower P2M errors, our density-aware global features ensure higher reconstruction accuracy and reduced outlier sensitivity. Qualitative results on a 50 K-point sample (Fig. 4) further validate these findings. While competitors such

as PathNet, SVCNet [58], and PDFlow exhibit ghosting artifacts or local clustering, and models like 3DMambaIPF over-smooth sharp boundaries, our method preserves fine details with a point distribution most consistent with GT. This confirms that MSAA-EdgeConv captures local geometric priors more robustly than standard EdgeConv or generative architectures, effectively preventing high-density artifacts common in iterative refinement.

4.5. Result on kinect dataset

As shown in Table 2, our model demonstrates superior generalization on the Kinect dataset using weights pretrained only on PUNet. The proposed architecture effectively mitigates characteristic Time-of-Flight (ToF) noise, which significantly degrades the performance of competitors like PCDNF. While specialized priors in SVCNet and C2AENet yield slightly higher P2M and HD scores on Kinect v1, our approach prioritizes global structural integrity, resulting in more robust reconstructions across diverse sensor types. Figs. 5 and 6 demonstrate visual comparisons on Kinect v1 and v2 test sets that competing methods struggle to balance noise removal with feature preservation. Most models fail to eliminate high-frequency surface bumps, resulting in lumpy textures inconsistent with the GT. Our method effectively removes structured depth layers to restore a clean, continuous surface while maintaining sharp edges and planar integrity. These results confirm our strategy's efficacy in handling complex, real-world sensor degradation.

4.6. Result on PCNet dataset

As summarized in Table 3, our proposed Heavy and Light variants achieve the best CD in the majority of PCNet configurations; at low noise on 10 K points, competing methods achieve marginally lower CD while our method maintains competitive P2M. In the 10 K category, the Heavy variant demonstrates superior robustness against high stochastic noise compared to P2P-Bridge and PDFlow. This high-precision reconstruction scales effectively to dense sets, where our Light network achieves the leading CD and P2M error. Overall, our models maintain a more stable performance profile, suggesting a superior ability to preserve global geometric structures and local surface manifolds under challenging perturbations. Visual analysis on the PCNet dataset (Fig. 7) highlights distinct denoising capabilities. While existing SOTA models reduce heavy input noise, they often introduce artifacts or struggle to preserve complex geometric features. In contrast, our proposed approach achieves superior visual fidelity, effectively eliminating outliers while maintaining sharp edges and clean boundaries that most closely align with GT.

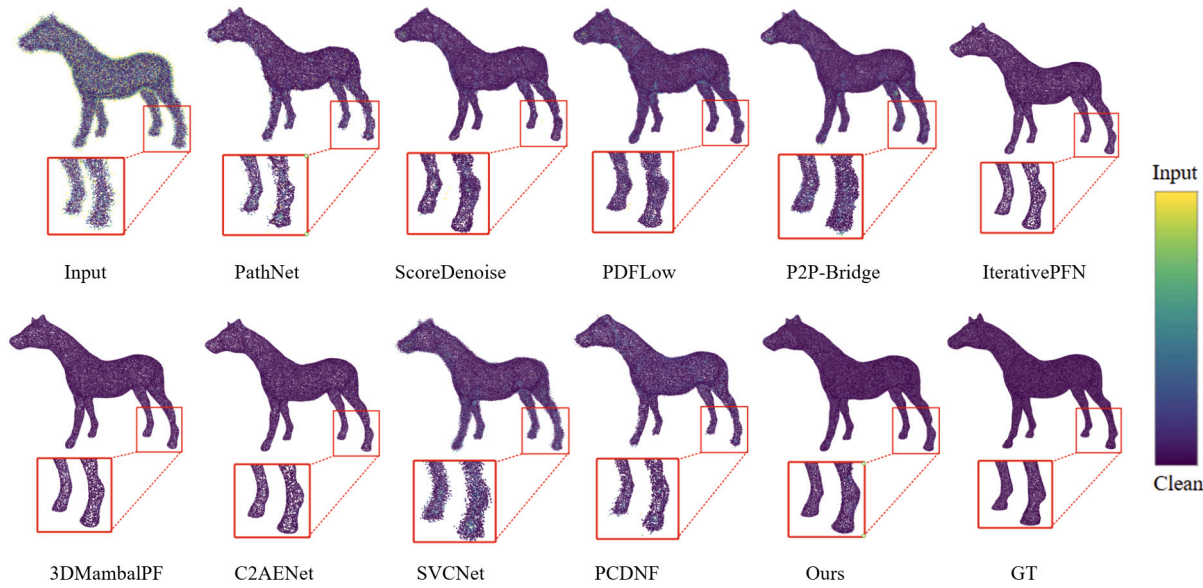


Fig. 4. Visual comparison of denoising results on the PUNet test set (50K points, 20% noise). Rectangles indicate zoomed-in regions of the back leg, highlighting our method’s ability to maintain sharp features and surface smoothness compared to existing baselines.

Table 2
Quantitative comparison on the Kinect test set.

Method	Kinect v1			Kinect v2		
	CD	P2M	HD	CD	P2M	HD
PathNet	18.82	12.32	21.04	31.59	19.92	28.74
ScoreDenoise	13.22	8.18	33.45	19.66	11.08	40.79
PDFLow	13.34	8.68	15.06	19.87	11.69	24.97
P2P-Bridge	11.47	7.64	23.32	18.72	11.41	31.02
IterativePFN	13.20	8.43	15.93	18.69	10.92	22.16
3DMambalPF	13.09	9.09	15.41	18.96	11.98	22.81
C2AENet	13.09	8.29	12.87	18.92	10.83	22.18
SVCNet	15.31	6.01	22.99	18.99	12.16	33.89
PCDNF	12.30	7.96	36.02	21.26	12.75	45.52
Our (Heavy s = 3)	11.06	7.63	15.02	15.99	9.80	21.68
Our (Light s = 5)	10.85	7.38	16.32	16.29	10.05	21.54

CD and P2M are multiplied by 10^5 while HD is multiplied by 10^4 .

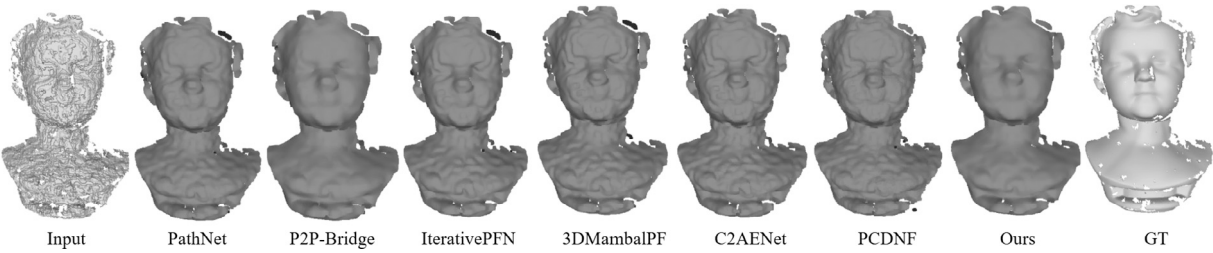


Fig. 5. Visual comparison of denoising methods on reconstructed meshes of the Kinect v1 test set.

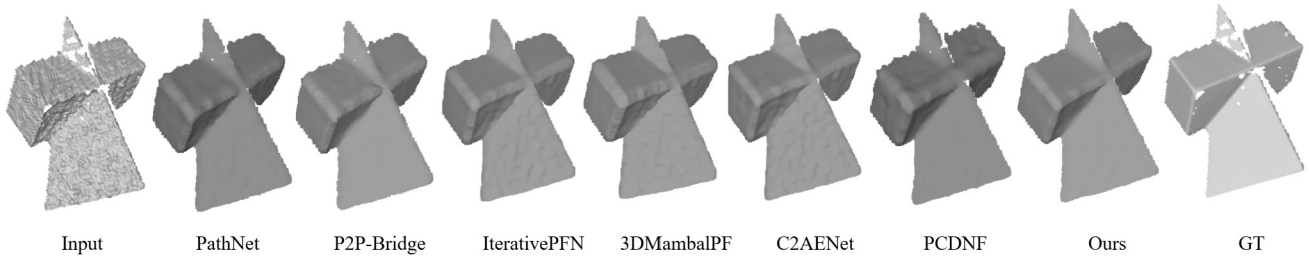


Fig. 6. Visual comparison of denoising methods on reconstructed meshes of the Kinect v2 test set.

Table 3
Quantitative comparison on the PCNet test set.

Method	10K (Light)						50K (Dense)					
	1% noise		2% noise		3% noise		1% noise		2% noise		3% noise	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
PathNet	30.76	7.78	46.31	17.43	74.86	40.14	9.02	2.53	31.04	20.43	88.04	7.09
ScoreDenoise	33.69	8.30	51.32	11.95	67.76	19.41	10.66	1.77	16.59	3.54	24.94	6.57
PDFlow	32.43	6.06	45.45	9.66	59.34	14.41	9.69	1.52	16.46	4.24	24.50	5.69
P2P-Bridge	28.80	6.30	44.70	8.90	55.80	12.90	9.20	1.20	13.50	2.40	21.20	4.90
IterativePFN	26.21	6.98	44.39	10.11	60.26	15.60	9.13	1.39	12.51	2.38	25.29	7.16
3DMambalPF	25.62	13.69	44.47	20.64	56.22	28.65	9.18	4.71	13.15	7.89	19.57	12.84
C2AENet	25.01	12.70	44.16	20.04	57.04	29.08	8.85	4.42	12.41	7.29	25.88	17.50
SVCNet	40.06	18.69	50.87	24.47	71.23	41.80	17.12	6.91	20.82	15.09	34.40	27.43
PCDNF	98.79	48.04	117.44	53.69	133.29	59.02	11.94	2.79	18.29	5.40	25.91	9.08
Our (Heavy $s = 3$)	26.69	6.35	39.90	9.53	50.07	12.77	7.77	1.97	11.55	2.33	16.82	5.69
Our ($s = 5$)	27.14	6.18	39.73	9.60	49.73	12.41	7.69	1.93	11.26	2.26	17.41	5.43

CD and P2M are multiplied by 10^5 .

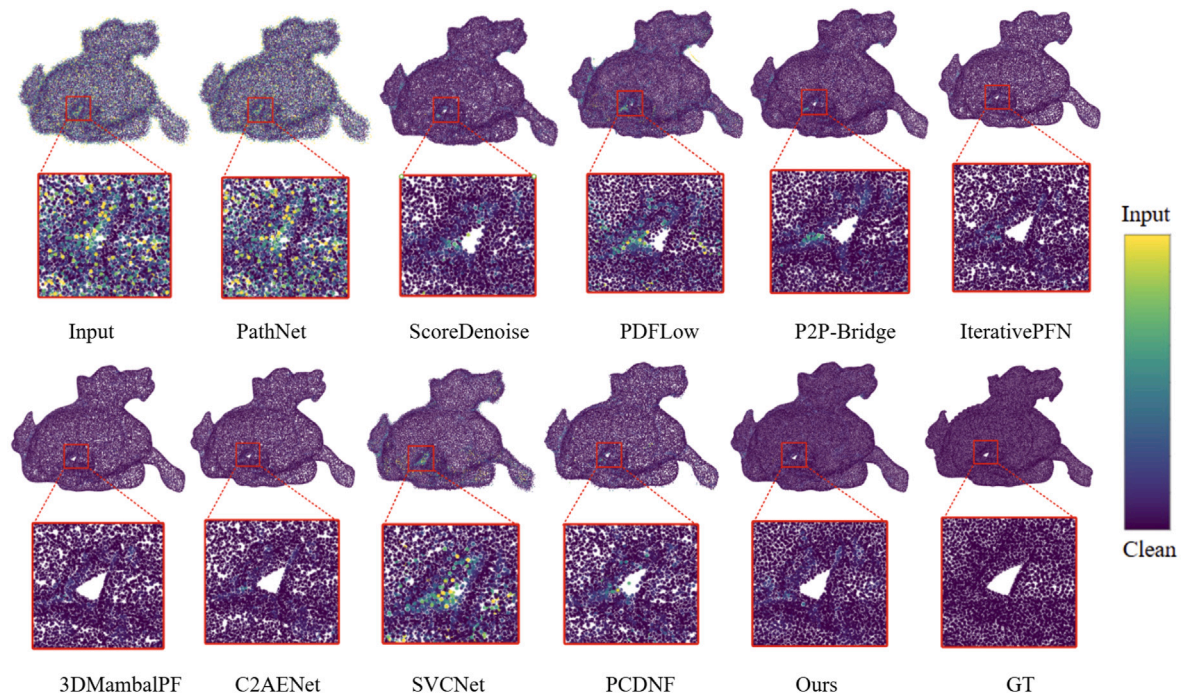


Fig. 7. Visual comparison of denoising results on the PCNet test set (50K points, 20% noise). Rectangles indicate zoom-in region.

Table 4
Denoising performance under different noise distributions on PCNet Dataset of 10K 30%noise.

Methods	Non-isotropic CD	Non-isotropic P2M	Uniform ball CD	Uniform ball P2M	Laplace CD	Laplace P2M
PathNet	78.58	42.76	53.2	27.36	88.84	49.46
ScoreDenoise	71.53	20.6	47.87	12.65	86.06	27.2
PDFlow	62.29	15.35	42.17	9.82	70.43	17.76
P2P-Bridge	58.57	13.74	39.65	8.79	66.22	15.9
IterativePFN	62.46	16.89	36.68	8.2	65.55	17.29
3DMambalPF	59.01	30.52	39.95	19.53	66.72	35.3
C2AENet	59.87	30.98	40.53	19.82	67.7	35.83
SVCNet	74.77	44.53	50.62	28.5	84.54	51.51
PCDNF	140.88	62.1	108.86	51.22	160.29	70.05
Our (Heavy $s = 3$)	54.56	13.61	51.10	13.01	59.43	15.74
Our ($s = 5$)	54.20	13.22	50.56	13.09	59.02	15.29

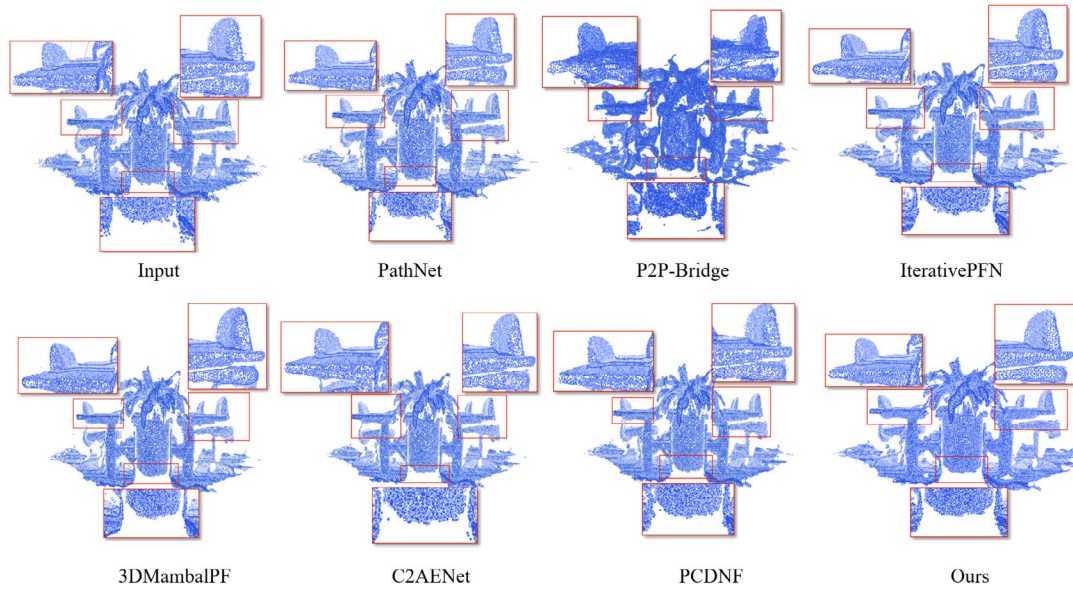


Fig. 8. Visual comparison on 3DCSR dataset. Rectangles indicate zoom-in region.

Table 5

Ablation study of various stages on the PUNet test set (10K input, 1% noise).

PUNet	10K			
	1% noise			
	CD	P2M	HD	
Light $s = 1$	31.15	11.97	12.37	
Light $s = 2$	25.67	8.68	9.62	
Light $s = 3$	21.33	6.34	8.07	
Light $s = 4$	18.91	5.28	7.00	
Light $s = 5$	18.16	4.99	6.83	
Heavy $s = 1$	26.80	9.39	10.16	
Heavy $s = 2$	20.13	5.78	7.83	
Heavy $s = 3$	18.25	5.04	7.26	

4.6.1. Robustness to non-Gaussian noise distributions

Table 4 evaluates robustness under three non-Gaussian noise distributions such as non-isotropic, uniform ball, and Laplace on the PCNet dataset at 10 K points with a 30% noise level, where all models were trained only on Gaussian noise. Our Light variants achieve CD and P2M values of 54.20 and 13.22 under non-isotropic and 59.02 of CD and 15.29 on Laplace noise, respectively, demonstrating that the confidence-guided progressive refinement and dynamic graph reconstruction generalize effectively beyond the Gaussian training distribution.

4.7. Visualization results on 3DCSR dataset

Fig. 8 compares qualitative results on the 3DCSR dataset, characterized by real-world cross-source noise and irregular densities. The input displays severe outliers and blurred boundaries, particularly on thin structures like chair legs and foliage. While P2P-Bridge amplifies distortions and often degrades quality, other methods such as PathNet, IterativePFN, and C2AENet fail to suppress outliers effectively. Conversely, our method successfully eliminates high-magnitude noise while preserving sharp structural details, yielding a clean reconstruction that closely aligns with the underlying geometry.

4.8. Ablation study

4.8.1. Ablation study on various stages analysis

Table 5 evaluates the impact of progressive stages on the PUNet dataset by testing models trained with multiple stages but utilizing only

a subset during inference. For the Light model, increasing the number of inference stages yields a significant percentage improvement in P2M distance and a substantial reduction in HD. Notably, the Heavy model achieves comparable performance (CD: 18.25, P2M: 5.04) to the 5-stage Light model (CD: 18.16, P2M: 4.99) using only 3 stages, corresponding to a 40% reduction in refinement stages. This result indicates that GAA-EdgeConv improves per-stage feature extraction efficiency, allowing the Heavy model to reach similar denoising quality with fewer refinement steps. The consistent downward trend in P2M and HD metrics across both configurations confirms that our coarse-to-fine refinement strategy effectively handles noise across multiple geometric scales.

4.8.2. Ablation study on key model components

To validate the effectiveness of our key design choices, we conduct an ablation study by systematically removing individual components from the full model. We evaluate five variants:

1. *w/o confidence*, which removes the adaptive confidence weighting mechanism and replaces it with uniform weights;
2. *w/o learnable α_s* , where the stage-specific scaling factors are replaced by a fixed refinement schedule;
3. *w/o feat. adapt.*, which disables the inter-stage feature adaptation layers;
4. *w/o dynamic graph reconstruction*, where the k -NN graph is constructed once from the initial noisy input and reused across all refinement stages;
5. *w/o stage-wise loss weighting*, where the exponential stage weights are replaced with uniform weights;

As Table 6 and Fig. 9 illustrate, removing any individual component consistently degrades performance across all metrics. Specifically, disabling confidence weighting leads to significant error increases in uncertain regions (Fig. 9a), while replacing learnable scaling with fixed values yields higher CD and P2M errors, confirming that adaptive stage-wise scaling is critical for coarse-to-fine refinement (Fig. 9b). Furthermore, the absence of feature adaptation results in poor geometric preservation due to insufficient inter-stage interaction (Fig. 9c). Removing dynamic graph reconstruction causes the largest performance degradation, indicating that rebuilding the graph at each stage is essential for avoiding stale geometric neighborhoods (Fig. 9d). Uniform stage-loss weighting also degrades all metrics, confirming the importance of stronger supervision at later refinement stages (Fig. 9e).

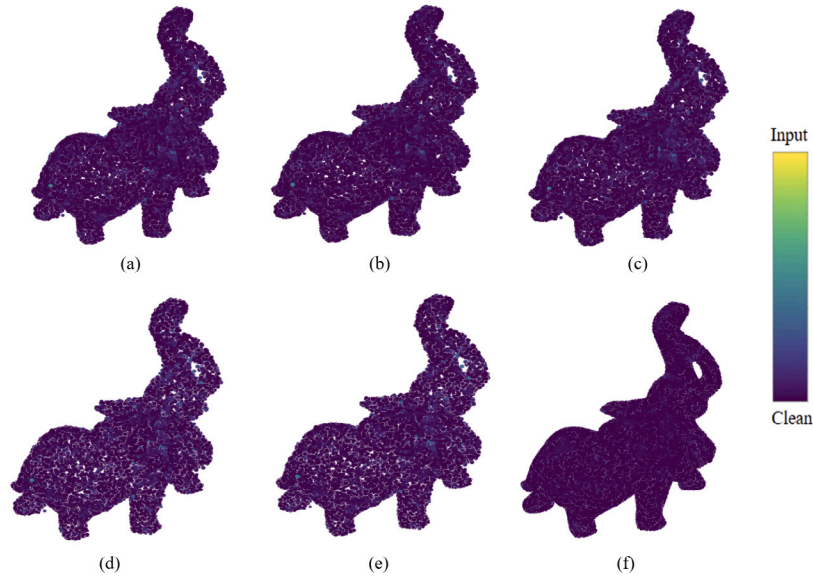


Fig. 9. Visual comparison without key model components on the testing set (10 K points, 1% noise) of PUNet. (a) without confidence, (b) without learnable scale, (c) without feature adaptation, (d) without dynamic graph reconstruction, (e) without stage-wise loss weighting, and (f) full model.

Table 6

Ablation study without key model components on the PUNet test set (10K input, 1% noise).

Key mechanism	CD	P2M	HD
w/o confidence weighting	26.44	8.12	11.73
w/o learnable α_s	25.52	7.42	11.62
w/o feature adaptation	26.02	7.83	11.48
w/o dynamic graph reconstruction	26.80	8.43	12.18
w/o stage-wise loss weighting	26.30	8.05	11.82
Full model (ours)	18.16	4.99	6.83

In contrast, the full model achieves the highest visual coherence and lowest reconstruction error (Fig. 9f), validating the synergy between confidence estimation, learnable scaling, and feature adaptation in achieving optimal denoising quality.

4.8.3. Ablation study on complexity comparisons

We evaluate computational efficiency using a 10 K point input, reporting parameter counts, FLOPs, and inference time (Table 7). To ensure a fair comparison across architectures, FLOPs are normalized to a 1000-point output scale. For Patch-to-Patch methods (like 3DMambaIPF, P2P-Bridge, IterativePFN and StraightPCF [59]), we compute FLOPs for a single forward pass of the 1000-point patch. For Patch-to-Point methods (like PathNet, PCDNF, MODNet [60]), which predict only a center point, the single-point FLOPs are multiplied by 1000. Inference time represents the average forward-pass latency for 1000 points, measured with a batch size of 1 on a single GPU, excluding dataloading and pre/post-processing. All methods are evaluated in a single-pass setting to maintain consistency.

For our proposed multi-stage progressive method, inference time is measured over one complete end-to-end forward pass through the full cascaded architecture with batch size 1 on a standardized 1000-point patch. This runtime includes all operations executed during inference, including all refinement stages, dynamic k -NN graph construction inside the model, feature extraction and re-extraction, displacement prediction, confidence prediction, and feature fusion in the heavy model. The light model uses 5 refinement stages, while the heavy model uses 3 refinement stages with an additional noise-aware GAA-EdgeConv branch. FLOPs are computed for the complete forward model under the same standardized input setting using the adopted profiling tool. Here,

Table 7

Comparison of network complexity.

Method	Params (M) (whole model)	FLOPs (G) (per patch)	Time (ms) (per patch)
P2P-Bridge	498.76	116.41	17.84
PathNet	77.85	90.00	636.59
MODNet	16.60	528.00	512.62
PCDNF	5.45	378.00	966.67
IterativePFN	3.20	67.26	107.01
StraightPCF	0.53	17.34	63.10
3DMambaIPF	0.81	75.07	128.60
Ours (Heavy $s = 3$)	0.53	6.82	42.88
Ours ($s = 5$)	0.28	10.51	63.21

FLOPs and time are measured per 1000-point patch. Params indicates the total number of model parameters. $M = 10^6$, $G = 10^9$, and ms denotes milliseconds.

Table 8

Stage-wise comparison of confidence scores and P2M errors in flat and non-flat regions of coverrear_Lp in PUNet.

Stage	Flat Conf.	Non-flat Conf.	Flat P2M	Non-flat P2M
Stage 1	0.125217	0.163282	7.02	7.72
Stage 2	0.336071	0.342366	4.61	6.21
Stage 3	0.213730	0.252628	2.81	4.71
Stage 4	0.379377	0.389776	1.90	4.00
Stage 5	0.419816	0.425028	1.66	3.84

“single-pass” refers to one complete forward call of the entire cascaded model, rather than one individual refinement stage or repeated external iterations.

4.8.4. Ablation study on the confidence mechanism

To quantitatively evaluate the confidence mechanism, we analyze per-point confidence and P2M error across five refinement stages in flat and non-flat regions. The regions are separated using PCA-based surface variation, calculated as $\lambda_3/(\lambda_1 + \lambda_2 + \lambda_3)$ over a $k = 15$ neighborhood.

As shown in Table 8, confidence generally increases from 0.125217 to 0.419816 in flat regions and from 0.163282 to 0.425028 in non-flat regions, while P2M decreases from 7.02 to 1.66 and from 7.72 to 3.84, respectively. The temporary confidence decrease at Stage 3 and the higher confidence but larger P2M in non-flat regions indicate that confidence does not directly represent error. Instead, it acts as a learned weighting signal for displacement updates based on local geometry.

Table 9

Ablation study of the feature extraction components on the PU-Net dataset with 1% Gaussian noise and 10K input points.

Stage	Feature extractor	Grouped Conv	Channel attention	Dual pool	GAA	CD ($\times 10^{-5}$)	P2M ($\times 10^{-5}$)	HD ($\times 10^{-4}$)
Light ($s = 5$)	Standard EdgeConv	×	×	×	×	18.48	5.13	6.93
Light ($s = 5$)	+ Grouped Depthwise Conv	✓	×	×	×	18.38	5.10	7.19
Light ($s = 5$)	+ Channel Attention	✓	✓	×	×	18.40	5.08	6.70
Light ($s = 5$)	+ Dual Aggregation (Full MSAA, Light)	✓	✓	✓	×	18.15	4.99	6.83
Heavy ($s = 3$)	Full MSAA, No GAA	✓	✓	✓	×	18.33	5.07	7.55
Heavy ($s = 3$)	+ GAA Base (Edge + Center, Max-Pool)	✓	✓	✓	Core	18.35	5.08	7.48
Heavy ($s = 3$)	+ GAA + Attention Gate	✓	✓	✓	+ Attention	18.34	5.04	7.52
Heavy ($s = 3$)	+ Full GAA-EdgeConv	✓	✓	✓	+ LPDE	18.25	5.03	7.26

4.8.5. Ablation study of feature extraction components

According to Table 9, all variants are trained from scratch under identical settings on PU-Net with 1% Gaussian noise and 10 K points. Rows 1 to 4 show that grouped depthwise convolution improves CD and P2M, channel attention mainly improves P2M and HD, and dual aggregation achieves the best overall MSAA-EdgeConv result. Row 5 introduces the heavy three-stage Full MSAA baseline without GAA for a direct comparison with the GAA variants. Rows 6 to 8 evaluate the GAA-EdgeConv branch using the same full MSAA-EdgeConv extractor. The attention gate improves P2M, while LPDE provides the best CD and HD among the GAA-EdgeConv variants, demonstrating their complementary contributions to surface alignment and geometric preservation.

5. Conclusion

In this paper, we propose a progressive point cloud denoising network that efficiently removes noise through multi-stage iterative refinement. By leveraging MSAA-EdgeConv and GAA-EdgeConv based feature extraction with adaptive local feature aggregation and confidence-weighted denoising networks.

Moreover, the proposed framework offers both Light and Heavy variants with exceptionally low computational overhead, achieving substantial parameter reduction and significantly lower computational costs compared to existing learning-based methods. Our extensive experiments on synthetic benchmarks (PUNet, PCNet), real-scene scanned datasets (Kinect v1, v2) and 3DCSR dataset demonstrate that our approach achieves competitive or superior denoising quality with dramatically improved efficiency and faster inference times. The substantial reduction in model complexity, coupled with high denoising accuracy, facilitates deployment in resource-constrained environments.

Overall, the proposed framework achieves strong denoising accuracy on synthetic and real-world datasets while maintaining low computational complexity. The proposed models achieve leading Chamfer Distance performance on the PU-Net and Kinect v1/v2 datasets, while the light variant uses only 0.28M parameters and 10.51 GFLOPs. In particular, on the 10 K-point PU-Net dataset, the proposed model achieves a CD of 18.15 and an HD of 6.83, outperforming the recent state-of-the-art methods 3DMambaIPF and C2AENet.

5.1. Limitation and future work

Although the proposed framework achieves strong performance on both synthetic and real-world benchmarks, several aspects remain for further improvement. First, while the Light and Heavy variants are efficient compared with existing learning-based methods, inference cost increases with the number of refinement stages due to repeated dynamic k-nearest-neighbor graph reconstruction. Second, the current supervised training setting requires paired clean and noisy point clouds, which may be difficult to obtain in some real-world scenarios. Third, despite strong cross-dataset generalization from PUNet to Kinect, performance may still be affected by extreme out-of-distribution noise or

sensor artifacts not covered by the training data. Finally, the current displacement-based design focuses on refining existing point positions; extending it to explicit outlier removal, completion, and semantic-aware denoising would further broaden its practical applicability.

Future work will focus on extending the proposed framework toward real-time and multi-sensor point cloud denoising. Complementary information from LiDAR, cameras, radar, and IMUs could be incorporated to improve robustness against sparse, non-uniform, and sensor-specific noise. More efficient graph construction, neighbor reuse, feature reuse, and lightweight network designs will also be investigated to reduce inference latency and computational cost. In addition, unsupervised or self-supervised learning will be explored to reduce dependence on paired clean and noisy training data. Finally, the framework will be extended toward joint denoising, outlier removal, and point cloud completion for practical applications in mobile sensing platforms.

CRediT authorship contribution statement

Moh Moh Thet Aung: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **JaeJun Yoo:** Supervision, Funding acquisition. **Ida Bagus Krishna Yoga Utama:** Writing – review & editing, Software, Investigation, Formal analysis. **Su Mon Ko:** Writing – review & editing. **Muhammad Alfi Aldolio:** Visualization, Investigation. **May Thu:** Visualization, Data curation. **Muhammad Fairuz Mumtaz:** Investigation. **Yeong Min Jang:** Writing – review & editing, Supervision.

Statement

During the preparation of this work, the authors did not use any generative models and take full responsibility for the content of the publication.

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.

Acknowledgments

This work is supported by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Grant RS-2022-00141819).

References

- [1] X. Lu, S. Schaefer, J. Luo, L. Ma, Y. He, Low rank matrix approximation for 3D geometry filtering, *IEEE Trans. Vis. Comput. Graphics* 28 (4) (2022) 1835–1847, <http://dx.doi.org/10.1109/TVCG.2020.3026785>.
- [2] D. Zhu, H. Chen, W. Wang, H. Xie, G. Cheng, M. Wei, J. Wang, F.L. Wang, Nonlocal Low-Rank point cloud denoising for 3-D measurement surfaces, *IEEE Trans. Instrum. Meas.* 71 (2022) 1–14, <http://dx.doi.org/10.1109/TIM.2021.3139686>.
- [3] M.-J. Rakotosaona, V. La Barbera, P. Guerrero, N.J. Mitra, M. Ovsjanikov, PointCleanNet: Learning to denoise and remove outliers from dense point clouds, *Comput. Graph. Forum* 39 (1) (2020) 185–203, <http://dx.doi.org/10.1111/cgf.13753>.
- [4] D. Zhang, X. Lu, H. Qin, Y. He, Pointfilter: Point cloud filtering via Encoder-Decoder modeling, *IEEE Trans. Vis. Comput. Graphics* 27 (03) (2021) 2015–2027, <http://dx.doi.org/10.1109/TVCG.2020.3027069>.
- [5] H. Chen, B. Du, S. Luo, W. Hu, Deep point set resampling via gradient fields, *IEEE Trans. Pattern Anal. Mach. Intell.* 45 (3) (2023) 2913–2930, <http://dx.doi.org/10.1109/TPAMI.2022.3175183>.
- [6] S. Luo, W. Hu, Score-Based point cloud denoising, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 4563–4572, <http://dx.doi.org/10.1109/ICCV48922.2021.00454>.
- [7] Y. Zhao, H. Zheng, Z. Wang, J. Luo, E.Y. Lam, Point cloud denoising via momentum ascent in gradient fields, in: 2023 IEEE International Conference on Image Processing, ICIP, IEEE, 2023, pp. 161–165.
- [8] D.d. Edirimuni, X. Lu, G. Li, A. Robles-Kelly, Contrastive learning for joint normal estimation and point cloud filtering, *IEEE Trans. Vis. Comput. Graphics* 30 (8) (2024) 4527–4541, <http://dx.doi.org/10.1109/TVCG.2023.3263866>.
- [9] D. de Silva Edirimuni, X. Lu, G. Li, L. Wei, A. Robles-Kelly, H. Li, StraightPCF: Straight point cloud filtering, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2024, pp. 20721–20730, <http://dx.doi.org/10.1109/CVPR52733.2024.01958>.
- [10] J. Wang, X. Lu, M. Wang, F. Hou, Y. He, Learning implicit fields for point cloud filtering, *IEEE Trans. Vis. Comput. Graphics* 31 (9) (2025) 5408–5420, <http://dx.doi.org/10.1109/TVCG.2024.3450699>.
- [11] A. Mao, Z. Du, Y.-H. Wen, J. Xuan, Y.-J. Liu, PD-Flow: A point cloud denoising framework with normalizing flows, in: S. Avidan, G. Brostow, M. Cissé, G.M. Farinella, T. Hassner (Eds.), *Computer Vision – ECCV 2022*, Springer Nature Switzerland, Cham, 2022, pp. 398–415.
- [12] Z. Su, C. Wang, K. Jiang, K. Jiang, W. Li, SITF: A Self-Supervised iterative training framework for point cloud denoising, *Comput.-Aided Des.* 179 (2025) 103812, <http://dx.doi.org/10.1016/j.cad.2024.103812>.
- [13] J. Zhou, X. Shi, H. Song, Y. Fang, Y.-S. Liu, Z. Han, U-CAN: Unsupervised point cloud denoising with Consistency-Aware Noise2Noise matching, 2025, arXiv:2510.25210.
- [14] S. Luo, W. Hu, Differentiable manifold reconstruction for point cloud denoising, in: *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 1330–1338.
- [15] X. Xu, G. Geng, X. Cao, K. Li, M. Zhou, TDNet: Transformer-based network for point cloud denoising, *Appl. Opt.* 61 (6) (2021) C80–C88.
- [16] M. Vogel, K. Tateno, M. Pollefeys, F. Tombari, M.-J. Rakotosaona, F. Engelmann, P2p-bridge: Diffusion bridges for 3d point cloud denoising, in: *European Conference on Computer Vision*, Springer, 2024, pp. 184–201.
- [17] D. de Silva Edirimuni, X. Lu, Z. Shao, G. Li, A. Robles-Kelly, Y. He, IterativePFN: True iterative point cloud filtering, in: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2023, pp. 13530–13539, <http://dx.doi.org/10.1109/CVPR52729.2023.01300>.
- [18] H. Chen, Z. Wei, X. Li, Y. Xu, M. Wei, J. Wang, RePCD-net: Feature-aware recurrent point cloud denoising network, *Int. J. Comput. Vis.* 130 (3) (2022) 615–629.
- [19] X. Wang, X. Fan, D. Zhao, PointFilterNet: A filtering network for point cloud denoising, *IEEE Trans. Circuits Syst. Video Technol.* 33 (3) (2023) 1276–1290, <http://dx.doi.org/10.1109/TCSVT.2022.3207789>.
- [20] D. Levin, The approximation power of moving least-squares, *Math. Comp.* 67 (224) (1998) 1517–1531.
- [21] Y. Lipman, D. Cohen-Or, D. Levin, H. Tal-Ezer, Parameterization-free projection for geometry reconstruction, *ACM Trans. Graph. (ToG)* 26 (3) (2007) 22–es.
- [22] H. Huang, D. Li, H. Zhang, U. Ascher, D. Cohen-Or, Consolidation of unorganized point clouds for surface reconstruction, *ACM Trans. Graph.* 28 (5) (2009) 1–7.
- [23] R. Preiner, O. Mattausch, M. Arikian, R. Pajarola, M. Wimmer, Continuous projection for fast L1 reconstruction, *ACM Trans. Graph.* 33 (4) (2014) 47–1.
- [24] L. Xu, R. Wang, J. Zhang, Z. Yang, J. Deng, F. Chen, L. Liu, Survey on sparsity in geometric modeling and processing, *Graph. Model.* 82 (2015) 160–180.
- [25] W. Hu, X. Gao, G. Cheung, Z. Guo, Feature graph learning for 3D point cloud denoising, *IEEE Trans. Signal Process.* 68 (2020) 2841–2856, <http://dx.doi.org/10.1109/TSP.2020.2978617>.
- [26] Y. Cui, W. Ren, A. Knoll, Omni-Kernel network for image restoration, *Proc. AAAI Conf. Artif. Intell.* 38 (2) (2024) 1426–1434, <http://dx.doi.org/10.1609/aaai.v38i2.27907>, URL: <https://ojs.aaai.org/index.php/AAAI/article/view/27907>.
- [27] Y. Cui, W. Ren, X. Cao, A. Knoll, Revitalizing convolutional network for image restoration, *IEEE Trans. Pattern Anal. Mach. Intell.* 46 (12) (2024) 9423–9438, <http://dx.doi.org/10.1109/TPAMI.2024.3419007>.
- [28] Y. Cui, W. Ren, X. Cao, A. Knoll, Focal network for image restoration, in: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 13001–13011.
- [29] O. Tutsoy, K. Balıkcı, N.F. Ozdil, Unknown uncertainties in the COVID-19 pandemic: Multi-dimensional identification and mathematical modelling for the analysis and estimation of the casualties, *Digit. Signal Process.* 114 (2021) 103058, <http://dx.doi.org/10.1016/j.dsp.2021.103058>, URL: <https://www.sciencedirect.com/science/article/pii/S105120042100097X>.
- [30] R. Roveri, A.C. Öztireli, I. Pande, M. Gross, Pointprons: Consolidation of point clouds with convolutional neural networks, in: *Computer Graphics Forum*, Vol. 37, Wiley Online Library, 2018, pp. 87–99.
- [31] Z. Liu, Z. Huang, M. Pan, Y. He, Deterministic point cloud diffusion for denoising, *IEEE Trans. Vis. Comput. Graphics* 32 (2) (2026) 1822–1834, <http://dx.doi.org/10.1109/TVCG.2025.3621633>.
- [32] H. Wang, C. Yang, L. Fang, H. Wei, Feature-Guided reversible network for point cloud denoising, *IEEE Sens. J.* 26 (11) (2026) 17356–17368, <http://dx.doi.org/10.1109/JSEN.2026.3684364>.
- [33] P. Du, X. Wang, H. Zhao, Z. Wu, Progressive point cloud denoising guided by unsigned distance function, *IEEE Trans. Geosci. Remote Sens.* 64 (2026) 5702512, <http://dx.doi.org/10.1109/TGRS.2026.3696434>.
- [34] M. Wei, H. Chen, Y. Zhang, H. Xie, Y. Guo, J. Wang, GeoDualCNN: Geometry-supporting dual convolutional neural network for noisy point clouds, *IEEE Trans. Vis. Comput. Graphics* 29 (2) (2023) 1357–1370, <http://dx.doi.org/10.1109/TVCG.2021.3113463>.
- [35] Z. Liu, Y. Zhao, S. Zhan, Y. Liu, R. Chen, Y. He, PCDNF: Revisiting Learning-Based point cloud denoising via joint normal filtering, *IEEE Trans. Vis. Comput. Graphics* 30 (8) (2024) 5419–5436, <http://dx.doi.org/10.1109/TVCG.2023.3292464>.
- [36] R. Heinzler, F. Piewak, P. Schindler, W. Stork, CNN-Based Lidar point cloud Denoising in adverse weather, *IEEE Robot. Autom. Lett.* 5 (2) (2020) 2514–2521, <http://dx.doi.org/10.1109/LRA.2020.2972865>.
- [37] A. Seppänen, R. Ojala, K. Tammi, 4DenoiseNet: Adverse weather denoising from adjacent point clouds, *IEEE Robot. Autom. Lett.* 8 (1) (2023) 456–463, <http://dx.doi.org/10.1109/LRA.2022.3227863>.
- [38] L.L. Zuo, J. Zhang, S.L. Ding, Y.L. Lv, Attention Mechanism-Based deep learning denoising of scanned point cloud for rocket tank panel, in: 2023 IEEE International Conference on Industrial Engineering and Engineering Management, IEEM, 2023, pp. 1682–1686, <http://dx.doi.org/10.1109/IEEM58616.2023.10407020>.
- [39] G. Xi, C. Wang, X. Liu, B. Xiao, Y. Ding, Point cloud denoising method based on local neighborhood features, *Opt. Laser Technol.* 193 (2026) 114144, <http://dx.doi.org/10.1016/j.optlastec.2025.114144>, URL: <https://www.sciencedirect.com/science/article/pii/S0030399225017359>.
- [40] F. Pistilli, G. Fracastoro, D. Valsesia, E. Magli, Learning robust graph-convolutional representations for point cloud denoising, *IEEE J. Sel. Top. Signal Process.* 15 (2) (2020) 402–414.
- [41] Z. Wei, H. Chen, L. Nan, J. Wang, J. Qin, M. Wei, PathNet: Path-Selective point cloud denoising, *IEEE Trans. Pattern Anal. Mach. Intell.* 46 (6) (2024) 4426–4442, <http://dx.doi.org/10.1109/TPAMI.2024.3355988>.
- [42] Q. Zhou, W. Yang, B. Fei, J. Xu, R. Zhang, K. Liu, Y. Luo, Y. He, 3Dmambaipf: A state space model for iterative point cloud filtering via differentiable rendering, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39, 2025, pp. 10843–10851.
- [43] Y. Tian, M. Chen, Z. Wang, L. Chen, J. Zhou, J. Yin, HeteroDenoise: Intensity-Guided point cloud denoising under heterogeneous noise, *IEEE Trans. Instrum. Meas.* 75 (2026) 1–16, <http://dx.doi.org/10.1109/TIM.2026.3666016>.
- [44] H. Shi, Z. Chen, J. Chen, D. Liu, H. Yang, BDNet: a deep learning-based point cloud denoising network for Brassica rapa, *Inform. Sci.* 735 (2026) 123072, <http://dx.doi.org/10.1016/j.ins.2026.123072>, URL: <https://www.sciencedirect.com/science/article/pii/S0020025526000034>.
- [45] X. Wang, W. Cui, R. Xiong, X. Fan, D. Zhao, FCNet: Learning Noise-Free features for point cloud denoising, *IEEE Trans. Circuits Syst. Video Technol.* 33 (11) (2023) 6288–6301, <http://dx.doi.org/10.1109/TCSVT.2023.3266458>.
- [46] P. Hermosilla, T. Ritschel, T. Ropinski, Total denoising: Unsupervised learning of 3D point cloud cleaning, in: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 52–60.
- [47] W. Wang, X. Liu, H. Zhou, L. Wei, Z. Deng, M. Murshed, X. Lu, Noise4Denoise: Leveraging noise for unsupervised point cloud denoising, *Comput. Vis. Media* 10 (4) (2024) 659–669.
- [48] Y. Wang, Y. Sun, Z. Liu, S.E. Sarma, M.M. Bronstein, J.M. Solomon, Dynamic graph cnn for learning on point clouds, *ACM Trans. Graph. (Tog)* 38 (5) (2019) 1–12.
- [49] H. Fan, H. Su, L.J. Guibas, A point set generation network for 3d object reconstruction from a single image, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 605–613.
- [50] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, P.-A. Heng, PU-Net: Point cloud up-sampling network, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 2790–2799, <http://dx.doi.org/10.1109/CVPR.2018.00295>.

- [51] P.-S. Wang, Y. Liu, X. Tong, Mesh denoising via cascaded normal regression, *ACM Trans. Graph.* 35 (6) (2016) 232–1.
- [52] X. Huang, G. Mei, J. Zhang, R. Abbas, A comprehensive survey on point cloud registration, 2021, arXiv preprint [arXiv:2103.02690](https://arxiv.org/abs/2103.02690).
- [53] R. Hanocka, G. Metzer, R. Giryes, D. Cohen-Or, Point2mesh: A self-prior for deformable meshes, 2020, arXiv preprint [arXiv:2005.11084](https://arxiv.org/abs/2005.11084).
- [54] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, G. Gkioxari, Accelerating 3d deep learning with pytorch3d, 2020, arXiv preprint [arXiv:2007.08501](https://arxiv.org/abs/2007.08501).
- [55] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimeshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, *Adv. Neural Inf. Process. Syst.* 32 (2019).
- [56] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980).
- [57] W. Chen, H. Fan, Q. Jiang, C. Huang, Y. Yang, Progressive point cloud denoising with cross-stage cross-coder adaptive edge graph convolution network, in: *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 6578–6587.
- [58] T. Zhao, P. Gao, T. Tian, J. Ma, J. Tian, From noise addition to denoising: A Self-Variation capture network for point cloud optimization, *IEEE Trans. Vis. Comput. Graphics* 30 (7) (2024) 3413–3426, <http://dx.doi.org/10.1109/TVCG.2022.3231680>.
- [59] D.d. Edirimuni, X. Lu, G. Li, L. Wei, A. Robles-Kelly, H. Li, StraightPCF: Straight point cloud filtering, 2024, [arXiv:2405.08322](https://arxiv.org/abs/2405.08322) URL: <https://arxiv.org/abs/2405.08322>.
- [60] A. Huang, Q. Xie, Z. Wang, D. Lu, M. Wei, J. Wang, MODNet: Multi-offset point cloud denoising network customized for Multi-scale patches, *Comput. Graph. Forum* 41 (7) (2022) 109–119, <http://dx.doi.org/10.1111/cgf.14661>.