Clean-Prior Distillation and Noise-Adaptive Structural Restoration for Point Cloud Denoising

Anonymous authors
Paper under double-blind review

Abstract

Point cloud denoising must remove measurement noise without erasing fine geometric structure. Structural priors extracted from noisy features may mix these signals, while uniform feature fusion and coordinate updates may move points that are already reliable. We address the two problems separately. Hierarchical Geometry-Aware Clean-Prior Distillation (HGCPD) transfers clean structural knowledge from a gradient-frozen teacher, updated by an exponential moving average, to a masked noisy student. Its targets include clean-code assignments and multi-scale feature and geometry cues. Noise-Adaptive Structural Restoration (NASR) regulates decoder skip features and pointwise residuals through a Noise-Aware Structural Refinement Module (NSRM) and an Update-Gated Residual Head (UGRH). The teacher, masked branch, and clean codebook are used only during training, so inference retains the complete student restoration path. This combination addresses over- and under-denoising while preserving geometric detail. Across benchmarks, point densities, Gaussian noise levels, and unseen noise distributions, the model achieves lower mean denoising errors, with the clearest gains under dense sampling and strong corruption. Component studies indicate that clean-prior distillation and noise-adaptive restoration make complementary contributions. Project page: https://nasr-hgcpd.github.io/

1 Introduction

Point clouds directly represent three-dimensional surfaces, but sensor noise, quantization, and registration errors perturb their coordinates and sampling density. An observed displacement may reflect either noise or fine geometry. Denoising must therefore suppress perturbations without smoothing sharp edges and thin structures.

Existing methods use direct regression, scores or flows, progressive filtering, diffusion, and adaptive routing or stopping (Rakotosaona et al., 2020; Luo & Hu, 2021; Mao et al., 2022; 2024; de Silva Edirimuni et al., 2023; 2024; Vogel et al., 2024; Guo et al., 2025; Liu et al., 2026; Du et al., 2026; Cheng et al., 2026). Despite their different update paths, these methods face two related challenges: obtaining reliable structural guidance from corrupted observations and deciding how strongly to update each feature and point.

Structural-prior denoisers quantize intermediate features into geometric primitives (Guo et al., 2026), but prototypes learned from noisy features may retain corruption. A prior also does not determine how much skip information to retain or how far each point should move. Both decisions affect the balance between residual noise and excessive correction.

We use distillation to separate clean structural supervision from corrupted observations, rather than to compress the denoiser. Figure 1 contrasts this restoration objective with a representative compression-oriented KD setup.

Our clean-prior-guided framework addresses supervision and restoration with two complementary components. Hierarchical Geometry-Aware Clean-Prior Distillation (HGCPD) transfers clean code assignments, features, and geometry from an EMA teacher to a masked noisy student (van den Oord et al., 2017; Tarvainen & Valpola, 2017; Pang et al., 2024;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

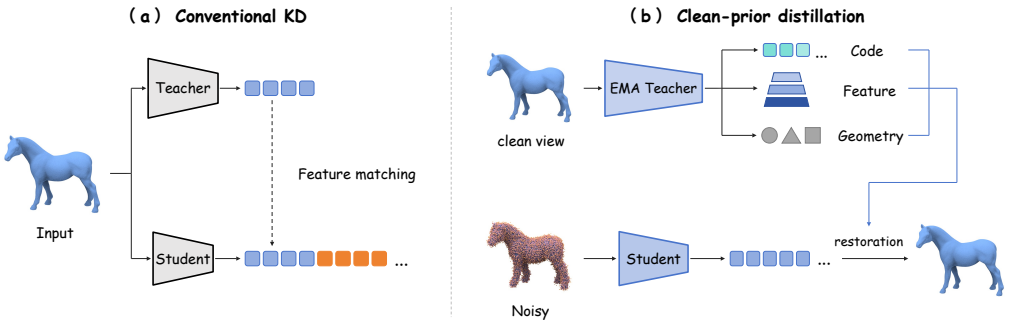

Figure 1: Conventional and clean-prior distillation. Left: A representative compression-oriented KD setup aligns one teacher and student feature representation from a shared input. Right: Our restoration-oriented setup separates the views: an EMA teacher observes clean geometry and transfers code, hierarchical feature, and geometric targets to a noisy student. The resulting supervision provides a clean structural reference for denoising.

Zhang et al., 2022; Li et al., 2025; Weinmann et al., 2015). Noise-Adaptive Structural Restoration (NASR) then regulates skip features with NSRM and scales pointwise residuals with UGRH. Training-only teacher components are removed at inference.

Experiments cover PUNet at two densities, direct transfer to PC-Net, and unseen noise distributions. Shape-level analyses show the clearest gains under dense sampling and stronger corruption, while retaining uncertainty for small P2M differences. Our contributions are:

- HGCPD transfers clean structural targets to a noisy encoder through masked, hierarchical distillation.
- NASR jointly regulates skip refinement and pointwise displacement.
- Multi-density, cross-dataset, unseen-noise, shape-level uncertainty, and fixed-checkpoint diagnostics test accuracy, transfer, robustness, and the behavior of the learned restoration controls.

2 Related Work

2.1 Deep Learning for Point Cloud Denoising

Single-step denoisers directly predict clean coordinates or displacements using local regression, latent separation, noisy-only scores, structural priors, or adaptive stopping (Rakotosaona et al., 2020; Zhang et al., 2021; Liu et al., 2024; Mao et al., 2022; 2024; Wei et al., 2025; Guo et al., 2026; 2025). Multi-step methods instead use score following, cascaded filtering, diffusion bridges, or adaptive routing (Luo & Hu, 2021; de Silva Edirimuni et al., 2023; 2024; Vogel et al., 2024; Liu et al., 2026; Du et al., 2026; Cheng et al., 2026). The former preserve correspondence efficiently; the latter control large displacements through progressive updates. In both cases, guidance is usually inferred from the noisy or partially restored state, allowing corruption to enter latent priors or propagate across stages. Repeated evaluation passes do not change where this guidance originates.

2.2 Masked Point Modeling and Clean-Prior Distillation

Vector quantization supplies discrete structural prototypes (van den Oord et al., 2017), while masked point models learn context-aware geometry through token prediction or reconstruction (Yu et al., 2022; Pang et al., 2024; Zhang et al., 2022). PointGAC further combines an online codebook with teacher–student learning (Li et al., 2025). These methods primarily learn representations from incomplete but clean inputs. HGCPD instead grounds masked prediction in paired clean geometry: an EMA teacher (Tarvainen & Valpola, 2017)

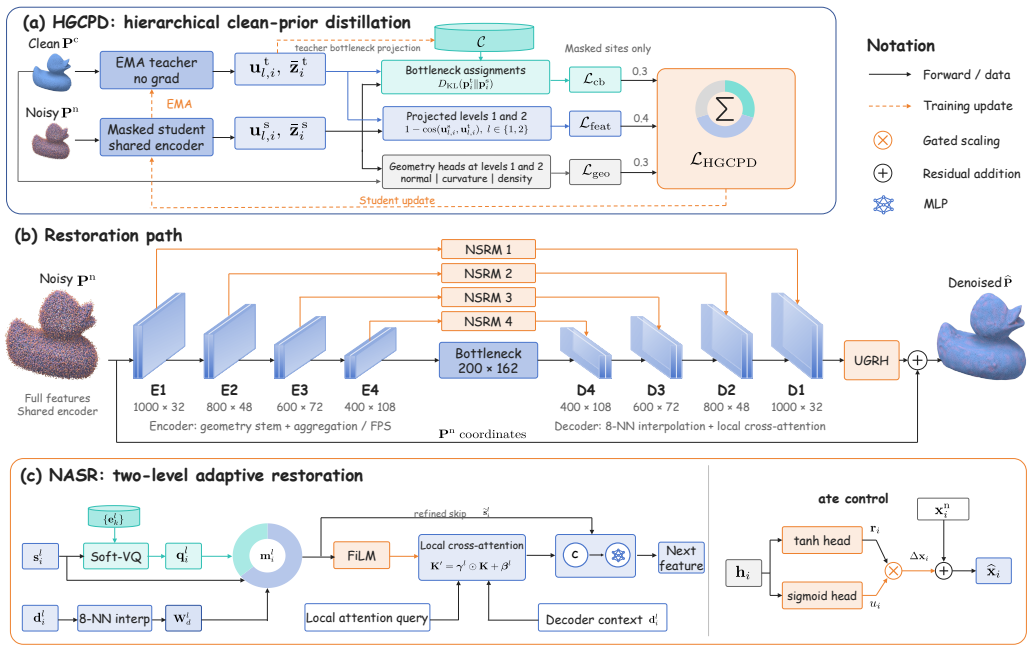

Figure 2: Overall framework of the proposed point cloud denoising method. (a) During training, HGCPD compares a masked noisy student with an exponential-moving-average clean teacher through bottleneck code assignments and projected multi-scale features; student geometry heads are supervised by clean normal, curvature, and density targets. The weighted auxiliary objective updates the shared student, which in turn updates the gradient-frozen teacher by EMA. (b) The complete noisy path retains full features, routes four encoder skips through NSRM, and uses UGRH before adding the predicted displacement to the noisy coordinates. (c) NSRM controls skip refinement and conditioned local cross-attention, whereas UGRH controls pointwise residual magnitude. At inference, the teacher, masked branch, and clean codebook are removed, leaving the complete student restoration path.

provides clean code, feature, and PCA-derived geometry targets (Weinmann et al., 2015), while inference remains student-only.

2.3 Adaptive Feature and Residual Modulation

Attention gates, residual learning, and uncertainty-guided restoration regulate information or update strength (Oktay et al., 2018; Zhang et al., 2017; Yasarla & Patel, 2019); point-cloud variants act through stopping, trajectories, alignment, or routing (de Silva Edirimuni et al., 2024; Guo et al., 2025; Du et al., 2026; Cheng et al., 2026). NASR combines both controls in the student restoration path: NSRM interpolates between skip features and structural prototypes, and UGRH gates bounded coordinate residuals. Figure 2 summarizes the combined training and inference paths.

3 Method

3.1 Backbone and Overview

Let $\mathbf{X}^n \in \mathbb{R}^{N\times 3}$ and $\mathbf{X}^c$ be paired noisy and clean clouds. A seed and its $n = 1{,}000$ nearest noisy points define $\mathbf{P}^n$; the same indices form $\mathbf{P}^c$, and both are centered at the noisy seed. The network predicts pointwise residuals:

$$\hat{\mathbf{P}} = \mathbf{P}^n + \Delta\mathbf{P}, \qquad \Delta\mathbf{P} = \mathcal{F}_\theta(\mathbf{P}^n). \tag{1}$$

A U-shaped encoder–decoder with structural-prior-conditioned skip fusion implements this mapping (Guo et al., 2026). GMEM initializes the multi-scale feature pyramid, NSRM refines decoder skips, and UGRH bounds the final displacement. During training, a masked noisy branch shares the student encoder, while a gradient-frozen EMA teacher processes the clean patch. These HGCPD branches are removed at inference.

The three paths serve different roles. The complete noisy branch alone predicts coordinate residuals. The masked branch shares its encoder but does not supply the decoder output; instead, its features are compared with clean-teacher targets at aligned points. This keeps restoration tied to the observed cloud while paired clean geometry regularizes intermediate features. At test time, denoising a patch requires neither clean coordinates nor the teacher and masked branches.

Geometry multi-modal encoding. GMEM uses the $k = 16$ nearest neighbors (including the query). Define $\bar{\mathbf{x}}_i = k^{-1} \sum_{j \in \mathcal{N}_k(i)} \mathbf{x}_j$ and $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2$. We compute local covariance as:

$$\mathbf{C}_i = \frac{1}{k-1} \sum_{\mathbf{x}_j \in \mathcal{N}_k(i)} (\mathbf{x}_j - \bar{\mathbf{x}}_i)(\mathbf{x}_j - \bar{\mathbf{x}}_i)^{\mathsf{T}}. \tag{2}$$

Let $\lambda_{i,1} \le \lambda_{i,2} \le \lambda_{i,3}$ be the eigenvalues, and let $\mathbf{n}_i$ be the eigenvector of $\lambda_{i,1}$. Following standard local geometric analysis (Weinmann et al., 2015), we form the descriptor:

$$\mathbf{g}_i = [|\mathbf{n}_i|,\, \kappa_i,\, \rho_i,\, \ell_i,\, \pi_i,\, s_i]. \tag{3}$$

Here, $\kappa_i = \lambda_{i,1} / \sum_j \lambda_{i,j}$ is curvature and the log-compressed inverse density is defined as:

$$\rho_i = \log\left(1 + \frac{1}{\mu_i}\right), \qquad \mu_i = \max\left(10^{-6}, \frac{1}{k-1} \sum_{j \in \mathcal{N}_k(i) \setminus \{i\}} \sqrt{d_{ij}}\right). \tag{4}$$

Because d_{ij} is a Euclidean distance, its square root further compresses the neighborhood distance range before averaging and inversion. The remaining entries are linearity $\ell_i = (\lambda_{i,3} - \lambda_{i,2})/\lambda_{i,3}$, planarity $\pi_i = (\lambda_{i,2} - \lambda_{i,1})/\lambda_{i,3}$, and scattering $s_i = \lambda_{i,1}/\lambda_{i,3}$. An MLP embeds $\mathbf{g}_i$ and concatenates it with coordinates; exact clamps, feature widths, point counts, and interpolation settings appear in Appendix D.

3.2 HGCPD

HGCPD supplies clean training targets without changing the inference graph.

Decoupled masked student and clean teacher. The complete noisy branch supplies decoder features, while a shared masked branch replaces 30% of initial point features and retains coordinates. For a 1,000-point patch, we sample $\lceil 300/16 \rceil = 19$ spatial centers and mask the 300 points closest to any center; a learned token replaces their features. The same FPS indices align the complete branch, masked student, and clean teacher. The gradient-frozen teacher is an EMA copy of the student and processes the clean patch. At batch step t in zero-indexed epoch e, its parameters are updated by EMA (Tarvainen & Valpola, 2017):

$$\bar{\theta}^{(t)} = m_e \bar{\theta}^{(t-1)} + (1 - m_e)\theta^{(t)}. \tag{5}$$

The momentum follows $m_e = 0.9999 - 0.0039[1 + \cos(\pi e/799)]/2$, starting at 0.996 in the configured 800-epoch schedule. The clean-codebook objective starts at epoch 20; the exact center sampling and batch-normalization rules are in Appendix D.

Clean codebook assignment. Student and teacher bottleneck projections are compared against a training-only clean codebook $\mathcal{C}$ of 128 normalized prototypes, distinct from the decoder codebooks (van den Oord et al., 2017; Guo et al., 2026). Clean-teacher features update $\mathcal{C}$ by online spherical clustering. Their soft assignments are defined as:

$$\begin{cases} p^{\mathrm{t}}_{ik} = \dfrac{\exp((\bar{\mathbf{z}}^{\mathrm{t}}_i)^{\mathsf{T}} \bar{\mathbf{c}}_k / \tau_{\mathrm{t}})}{\sum_j \exp((\bar{\mathbf{z}}^{\mathrm{t}}_i)^{\mathsf{T}} \bar{\mathbf{c}}_j / \tau_{\mathrm{t}})}, \\ p^{\mathrm{s}}_{ik} = \dfrac{\exp((\bar{\mathbf{z}}^{\mathrm{s}}_i)^{\mathsf{T}} \bar{\mathbf{c}}_k / \tau_{\mathrm{s}})}{\sum_j \exp((\bar{\mathbf{z}}^{\mathrm{s}}_i)^{\mathsf{T}} \bar{\mathbf{c}}_j / \tau_{\mathrm{s}})}. \end{cases} \tag{6}$$

Bars denote ℓ_2 normalization; $\tau_{\mathrm{t}} = 0.04$ and $\tau_{\mathrm{s}} = 0.10$. The codebook loss over masked tokens is given by:

$$\mathcal{L}_{\mathrm{cb}} = h_{\mathcal{C}} \frac{\sum_{i \in \mathcal{M}} w_i D_{\mathrm{KL}}(\mathbf{p}_i^{\mathrm{t}} \| \mathbf{p}_i^{\mathrm{s}})}{\sum_{i \in \mathcal{M}} w_i}, \qquad w_i = \left(\max_k p_{ik}^{\mathrm{t}} \right)^2. \tag{7}$$

For current-batch hard-assignment utilization u, the health multiplier is $h_{\mathcal{C}} = \mathrm{clip}((u - 0.50)/0.25, 0, 1)$: assignment supervision vanishes below 50% utilization and reaches full weight at 75%. The 128×64 clean codebook uses EMA momentum 0.99; centers unused for 100 active batches are eligible for revival, capped at 16 per batch. Initialization and replacement rules are specified in the appendix.

Hierarchical clean targets. We also align masked student and teacher features at encoder levels 1 and 2:

$$\mathcal{L}_{\mathrm{feat}} = \frac{1}{\sum_{l \in \mathcal{S}} \omega_l} \sum_{l \in \mathcal{S}} \frac{\omega_l}{|\mathcal{M}_l|} \sum_{i \in \mathcal{M}_l} \left[1 - \cos(\mathbf{u}_{l,i}^{\mathrm{s}}, \mathbf{u}_{l,i}^{\mathrm{t}}) \right], \quad \mathcal{S} = \{1, 2\}. \tag{8}$$

We use $(\omega_1, \omega_2) = (4, 3)$. A geometry head predicts normal, curvature, and density computed deterministically by GMEM on the clean coordinates in the teacher pass:

$$\mathcal{L}_{\mathrm{geo}} = \frac{1}{3} \left(\mathcal{L}_{\mathrm{normal}} + \mathcal{L}_{\mathrm{curvature}} + \mathcal{L}_{\mathrm{density}} \right). \tag{9}$$

These geometric labels are computed from clean coordinates, not predicted by the teacher network. Shared sampling indices align them with the clean feature targets; direct computation on the clean patch with the same GMEM rules yields the same labels. The normal term is sign invariant, and scalar targets use smooth ℓ_1. The combined HGCPD objective is:

$$\mathcal{L}_{\mathrm{HGCPD}} = 0.30\mathcal{L}_{\mathrm{cb}} + 0.40\mathcal{L}_{\mathrm{feat}} + 0.30\mathcal{L}_{\mathrm{geo}}. \tag{10}$$

3.3 NASR

HGCPD regularizes the shared encoder. NASR controls how strongly its features alter the output, using NSRM at the feature level and UGRH at the coordinate level.

NSRM. At decoder level l, let $\mathbf{s}_i^l$ be a dense skip feature and $\mathbf{d}_i^l$ the interpolated decoder context. A stage-specific codebook softly quantizes $\mathbf{s}_i^l$ into a structural prior $\mathbf{q}_i^l$:

$$a_{ik}^l = \mathrm{softmax}_k \left(\frac{\cos(\mathbf{s}_i^l, \mathbf{e}_k^l)}{\tau_d} \right), \qquad \mathbf{q}_i^l = \sum_k a_{ik}^l \mathbf{e}_k^l. \tag{11}$$

Here, $\tau_d = 0.1$. The four decoder codebooks have 512, 384, 256, and 192 entries and are updated from noisy skip features with EMA momentum 0.99 only in training mode. At inference, their checkpointed vectors are read without updating counts, sums, or entries; NSRM still uses their soft assignments to predict a point- and channel-wise interpolation gate:

$$\begin{cases} \mathbf{m}_i^l = \sigma\big(\mathrm{MLP}_l[\mathbf{s}_i^l, \mathbf{q}_i^l, \mathbf{W}_d^l \mathbf{d}_i^l]\big), \\ \widetilde{\mathbf{s}}_i^l = (1 - \mathbf{m}_i^l) \odot \mathbf{s}_i^l + \mathbf{m}_i^l \odot \mathbf{q}_i^l. \end{cases} \tag{12}$$

The gate interpolates between detailed skips and regularized prototypes. The result modulates cross-attention keys by FiLM (Perez et al., 2018):

$$(\boldsymbol{\gamma}^l, \boldsymbol{\beta}^l) = \mathrm{FiLM}(\widetilde{\mathbf{S}}^l), \qquad \mathbf{K}' = \boldsymbol{\gamma}^l \odot \mathbf{K} + \boldsymbol{\beta}^l. \tag{13}$$

The modulated attention output and $\widetilde{\mathbf{S}}^l$ form the next decoder feature.

UGRH. From the final decoder representation $\mathbf{h}_i$, UGRH predicts a bounded residual and scalar correction-demand gate:

$$\mathbf{r}_i = \tanh(\mathbf{W}_r \mathbf{h}_i), \qquad u_i = \sigma(\mathrm{MLP}_u(\mathbf{h}_i)), \qquad \Delta\mathbf{x}_i = u_i \mathbf{r}_i. \tag{14}$$

The paired displacement magnitude defines the gate target:

$$u_i^* = \mathrm{clip}\left(\frac{\|\mathbf{x}_i^{\mathrm{c}} - \mathbf{x}_i^{\mathrm{n}}\|_2}{3\sigma}, 0, 1\right), \qquad \mathcal{L}_{\mathrm{gate}} = \mathrm{SmoothL1}(u_i, u_i^*). \tag{15}$$

The target assigns lower correction demand to lightly corrupted points; u_i is a deterministic gate, not a probabilistic uncertainty estimate.

3.4 Training Objective and Patch-Based Inference

The restoration path uses a symmetric InfoCD-style nearest-neighbor loss $\mathcal{L}_{\mathrm{rec}}$ (Guo et al., 2026; Lin et al., 2023). Clean normals additionally define a Charbonnier point-to-plane loss:

$$\mathcal{L}_{\mathrm{surf}} = \frac{1}{n}\sum_{i=1}^{n}\left[\sqrt{((\widehat{\mathbf{x}}_i - \mathbf{x}_i^{\mathrm{c}})^{\mathsf{T}}\mathbf{n}_i^{\mathrm{c}})^2 + \epsilon_s^2} - \epsilon_s\right], \qquad \epsilon_s = 10^{-4}. \tag{16}$$

To limit conflicts with restoration, auxiliary gradients are projected against the reconstruction gradient (Yu et al., 2020):

$$\widetilde{\mathbf{g}}_a = \mathbf{g}_a - \min\left(\frac{\mathbf{g}_a^{\mathsf{T}}\mathbf{g}_r}{\|\mathbf{g}_r\|_2^2}, 0\right)\mathbf{g}_r. \tag{17}$$

With batch-mean noise $\bar{\sigma}$ and distillation schedule s_e, the dynamic weights are defined as:

$$\begin{cases} r_d(\bar{\sigma}) = 0.005 + 0.010\ \mathrm{clip}\left(\dfrac{\bar{\sigma} - 0.005}{0.015}, 0, 1\right), \\ \lambda_d = \mathrm{clip}\left(\dfrac{s_e r_d(\bar{\sigma})\|\mathbf{g}_r\|_2}{\max(\|\widetilde{\mathbf{g}}_d\|_2, 10^{-8})}, 0, 0.40\right), \\ \lambda_s = \mathrm{clip}\left(\dfrac{0.01\|\mathbf{h}_r\|_2}{\max(\|\widetilde{\mathbf{h}}_s\|_2, 10^{-8})}, 0, 80\right). \end{cases} \tag{18}$$

Here, $\mathbf{g}$ and $\mathbf{h}$ denote shared-encoder and coordinate gradients, and tildes indicate projection. The complete training objective is given by:

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{rec}} + 0.01\mathcal{L}_{\mathrm{gate}} + \lambda_d \mathcal{L}_{\mathrm{HGCPD}} + \lambda_s \mathcal{L}_{\mathrm{surf}}. \tag{19}$$

Appendix D gives s_e, gradient scopes, codebook updates, and the exact optimization order.

At inference, the student processes overlapping 1,000-point FPS patches. Their fused prediction is given by:

$$w_{mi} = \exp(-\eta \widetilde{d}_{mi}), \qquad \widehat{\mathbf{x}}_i = \frac{\sum_m w_{mi}\widehat{\mathbf{x}}_{mi}}{\sum_m w_{mi}}, \qquad \eta = 4. \tag{20}$$

Here, $\widetilde{d}_{mi}$ is the squared distance to patch seed m divided by the largest squared seed distance within that patch. Before patch extraction, the input cloud is normalized by its bounding-box midpoint and maximum radius. Deterministic FPS selects $6N/1{,}000$ seeds, and each seed defines a centered 1,000-point nearest-neighbor patch. Equation 20 gives greater weight to predictions near patch centers when patches overlap; uncovered points retain their input coordinates. The fused cloud is mapped back to the input coordinate system. The repeated-pass protocol and implementation details are given in Appendix D.7.

4 Experiments

4.1 Experimental Setup

Data and metrics. PUNet (Yu et al., 2018) provides 40 training and 20 evaluation meshes; we use 10K and 50K samples, unit-sphere normalization, online Gaussian training noise

Points sparsity	10K points						50K points					
Noise level	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
Method	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$
PCN	3.686	1.599	7.926	4.759	10.486	6.987	1.103	0.646	1.978	1.370	3.203	2.486
GPDNet	2.310	0.714	4.284	1.855	5.837	3.066	1.049	0.635	3.288	2.503	5.085	4.134
DMR	4.712	2.196	5.085	2.523	5.277	2.669	1.205	0.762	1.443	0.970	1.696	1.190
PointFilter	2.461	0.730	3.534	1.155	4.099	1.505	0.758	0.432	0.907	0.507	1.099	0.629
Score	2.522	0.754	3.683	1.380	4.232	1.904	0.716	0.400	1.289	0.833	1.445	0.958
PDFlow	2.126	0.674	3.246	1.324	3.627	1.702	0.651	0.416	1.270	0.921	1.874	1.426
IterativePFN	2.055	0.501	3.043	0.843	3.353	1.046	0.605	0.302	0.803	0.436	1.015	0.588
StraightPCF	1.904	0.545	2.672	0.929	2.925	1.120	0.620	0.389	0.811	0.551	0.987	0.668
PD-LTS	1.825	0.496	2.567	0.824	2.849	1.034	0.496	0.306	0.706	0.447	0.936	0.605
P2P-Bridge	2.284	0.686	3.202	1.114	3.531	1.386	0.586	0.330	0.902	0.580	1.165	0.803
3DMambaIPF	1.989	0.477	2.995	0.803	3.262	0.992	0.589	0.291	0.755	0.405	0.928	0.531
ASDN	1.836	0.490	2.508	0.776	2.697	0.961	0.498	0.308	0.686	0.448	0.850	0.575
GPD	1.797	0.469	2.430	0.729	2.626	0.904	0.479	0.286	0.615	0.404	0.881	0.604
Ours	1.776	0.463	2.408	0.713	2.619	0.898	0.461	0.287	0.603	0.398	0.750	0.505

Table 1: Quantitative evaluation on PUNet under isotropic Gaussian noise. Competitor rows are literature or previously recorded benchmark aggregates under their original protocols; Ours is evaluated from the reported local checkpoint. The appendix reports the matched local comparisons. CD and P2M use 10^{-4} units; bold and underlining denote the best and second-best results, respectively.

$\sigma \sim \mathcal{U}(0.005, 0.020)$, and test levels of 1%, 2%, and 2.5%. We also test the PUNet-trained model without fine-tuning on ten 50K PC-Net shapes (Rakotosaona et al., 2020) and on four unseen corruption families. We report symmetric CD and bidirectional P2M, multiplied by 10^4 (lower is better). Exact corruption definitions, metric formulas, and software settings are in Appendices D.1 and D.8.

Training and inference. We train on 1,000-point patches with Adam (initial learning rate 5×10^{-4}), batch size 20, mixed precision, and seed 2025 on one RTX 3090. The full feature network and active inference graph contain 1.402M and 1.044M parameters, respectively; the event-log resource record appears in Table 16. At inference, $6N/1{,}000$ FPS-centered patches are fused by Eq. 20. We use two restoration passes at 1% and 2% Gaussian noise and three at 2.5%; PC-Net uses two and three passes at 1% and 2.5%, and non-Gaussian inputs use two. Complete optimization, checkpoint, normalization, and patch settings are in Appendices D.6 and D.7.

Statistical analysis. The PUNet shape is the paired resampling unit ($n = 20$). We report shape-level values, sample SDs, and win counts, and compare Ours with a local GPD reproduction using 100,000 paired bootstrap resamples. For non-Gaussian noise, five realizations per shape (seeds 42–46) enter a hierarchical bootstrap over shapes and within-shape realizations. Appendix A contains the full design, intervals, and per-shape results; these intervals do not estimate training-run variation.

4.2 Comparison with Published Methods

We compare direct-regression, score, flow, progressive, bridge, and adaptive denoisers (Rakotosaona et al., 2020; Pistilli et al., 2020; Luo & Hu, 2020; Zhang et al., 2021; Luo & Hu, 2021; Mao et al., 2022; de Silva Edirimuni et al., 2023; 2024; Mao et al., 2024; Vogel et al., 2024; Zhou et al., 2025; Guo et al., 2025; 2026). Competitor entries in the main comparison are literature or previously recorded benchmark aggregates under their original protocols. We report shape-wise local evaluations with common inputs and metrics in the appendix.

In Table 1, our model is best in 11 of 12 settings and second in one. The largest gains over GPD occur at 50K–2.5%: CD and P2M fall from 0.881/0.604 to 0.750/0.505 (14.9%/16.3%). All 12 paired local mean differences favour Ours; eight bootstrap intervals exclude zero, whereas four small P2M intervals cross zero (Table 5).

Figure 3 shows the 50K–2.5% setting. We selected Cow, Duck, and Horse before comparing methods to span rounded, compact-curved, and thin-extremity shapes with complete renderings. Our method is not best on every selected shape. Appendix A.5 quantifies curvature, sharp-edge, and thin-sheet subsets; further visual comparisons appear in Appendix E.

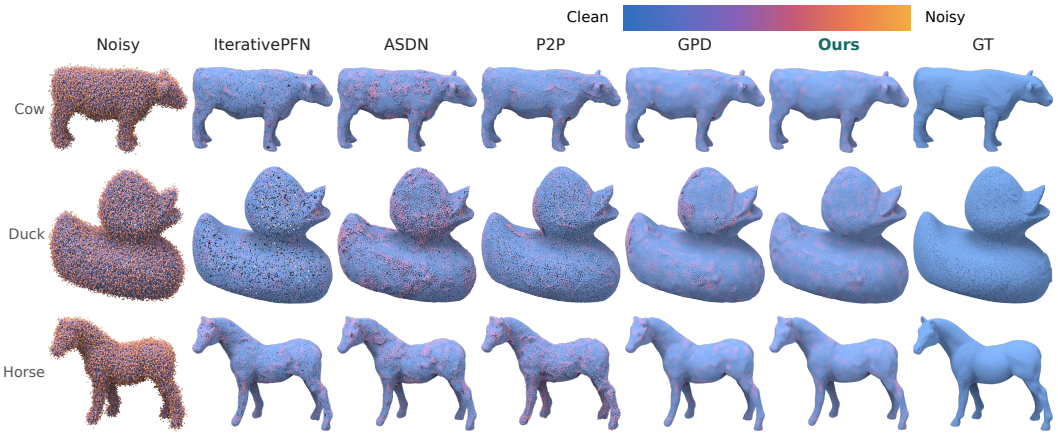

Figure 3: Point-wise P2M distance visualizations for PUNet shapes sampled at 50K points and corrupted by 2.5% isotropic Gaussian noise. Columns show the noisy input, competing methods, our method, and the ground truth (GT) in the final column; blue-to-orange colors indicate lower-to-higher point-wise error. Cow, Duck, and Horse provide rounded, compact-curved, and thin-extremity cases, respectively; all have complete seven-column renderings.

4.3 Cross-Dataset Generalization on PC-Net

We apply the PUNet-trained models to PC-Net without fine-tuning. ASDN and IterativePFN use released checkpoints with the same inputs and evaluator. Our model has the lowest CD/P2M at 1% (0.789/0.320) and 2.5% (1.420/0.876), with a small margin over IterativePFN at 2.5% (1.434/0.880). Matched visual examples appear in Appendix B.3.

Points sparsity	50K points			
Noise level	1% noise		2.5% noise	
Method	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$
ASDN	0.805	0.352	1.719	1.122
IterativePFN	0.902	0.426	<u>1.434</u>	<u>0.880</u>
P2P-Bridge	0.921	0.395	1.673	1.063
GPD	<u>0.793</u>	<u>0.332</u>	1.564	0.994
Ours	0.789	0.320	1.420	0.876

Table 2: Cross-dataset evaluation on PC-Net at 50K points without fine-tuning. The ASDN and IterativePFN entries are local evaluations of their released checkpoints. CD and P2M use 10^{-4} units; bold and underlining denote the best and second-best results, respectively.

4.4 Robustness to Unseen Noise Distributions

Under four unseen distributions (Table 3), Ours is best in 12 of 16 settings and ties once. Its largest gain over GPD is at 2% Laplace: CD/P2M fall from 3.226/1.419 to 2.970/1.182 (7.9%/16.7%).

PD-LTS remains better for 1% uniform noise and for discrete CD. Across five corruption realizations, all 16 paired means favour Ours and 13 hierarchical intervals exclude zero; three small P2M intervals cross zero (Table 6).

Points sparsity		10K points			
Noise level		1% noise		2% noise	
Noise	Method	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$
Ani	StraightPCF	1.923	0.529	2.753	0.999
	PD-LTS	1.833	0.503	2.630	0.878
	P2P-Bridge	2.312	0.708	3.230	1.151
	ASDN	1.847	0.497	2.592	0.853
	GPD	1.806	0.477	2.497	0.786
	Ours	1.787	0.471	2.473	0.770
Lap	StraightPCF	2.249	0.665	3.051	1.241
	PD-LTS	2.158	0.624	3.090	1.232
	P2P-Bridge	2.637	0.855	3.908	1.727
	ASDN	2.145	0.598	3.080	1.284
	GPD	2.098	0.573	3.226	1.419
	Ours	2.072	0.562	2.970	1.182

Points sparsity		10K points			
Noise level		1% noise		2% noise	
Noise	Method	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$
Uni	StraightPCF	0.673	0.389	1.877	0.567
	PD-LTS	0.635	0.368	1.787	0.477
	P2P-Bridge	1.084	0.560	2.293	0.662
	ASDN	0.700	0.385	1.791	0.471
	GPD	0.672	0.376	1.760	0.456
	Ours	0.650	0.370	1.734	0.448
Dis	StraightPCF	0.678	0.391	1.711	0.556
	PD-LTS	0.643	0.371	1.596	0.475
	P2P-Bridge	1.087	0.580	1.982	0.676
	ASDN	0.709	0.386	1.635	0.472
	GPD	0.680	0.375	1.595	0.462
	Ours	0.657	0.371	1.568	0.449

Table 3: Evaluation on PUNet 10K under unseen noise distributions, arranged as two side-by-side blocks: anisotropic Gaussian (Ani), Laplace (Lap), uniform (Uni), and discrete (Dis). Values use 10^{-4} units and the originally recorded seed-42 sets. The 1% Dis P2M values tie at the reported precision.

4.5 Ablation Study

Table 4 compares six locally evaluated variants on identical inputs; its caption specifies the components retained in each variant.

Points sparsity	10K points						50K points					
Noise level	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
Method / variant	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$	$CD\downarrow$	$P2M\downarrow$
GPD (local eval.)	1.798	0.469	2.430	0.730	2.657	0.919	0.477	0.297	0.616	0.405	0.883	0.602
HGCPD only	1.801	0.477	2.438	0.737	2.652	0.924	0.472	0.293	0.627	0.413	0.866	0.594
NSRM (w/o UGRH)	1.790	0.468	2.429	0.729	2.643	0.911	0.468	0.290	0.621	0.406	0.865	0.580
UGRH (w/o NSRM)	1.798	0.468	2.410	0.716	2.651	0.916	0.473	0.294	0.609	0.398	1.086	0.755
NASR only	1.786	0.466	2.410	0.721	2.672	0.920	0.470	0.291	0.612	0.399	0.851	0.570
HGCPD + NASR	1.776	0.463	2.408	0.713	2.619	0.898	0.461	0.287	0.603	0.398	0.750	0.505

Table 4: Component comparison on PUNet with common inputs and metrics. HGCPD only omits NSRM/UGRH; NSRM-only and UGRH-only retain GMEM with one component, and NASR only uses both. The three NASR variants omit clean-prior distillation; single-component rows use nearest-patch fusion. CD and P2M use 10^{-4} units. Bold and underlining denote best and second-best values at reported precision; ties share formatting.

Against GPD, NSRM only and NASR only each improve 10 of 12 metrics, UGRH only nine with one tie, and HGCPD only five. UGRH only helps at 10K–2% and 50K–2%, but worsens the dense 2.5% case (1.086/0.755 versus NSRM-only 0.865/0.580). Combining NSRM and UGRH improves eight metrics over NSRM alone, including 50K–2.5% (0.851/0.570). The full model is best or tied across all 12 reported metrics, reaching 0.750/0.505 at 50K–2.5%.

5 Conclusion

In this paper, we present a clean-prior-guided point cloud denoising framework that combines hierarchical clean-prior distillation with noise-adaptive structural restoration. HGCPD transfers clean structural knowledge to a masked student during training, while NASR adaptively modulates decoder features and point updates to suppress noise while preserving fine geometry. At inference, the model retains a single restoration path without the teacher or clean codebook. Experiments across multiple noise settings, together with ablation studies, demonstrate effective denoising and the complementary contributions of clean-prior learning and adaptive restoration. Although gains on real sensor data are less pronounced, the results remain competitive. Together, these results support the framework's potential across varied noise conditions and motivate further work to improve its generalization to real sensor data.

AI use statement

Generative AI tools were used to assist with experimental-protocol review, statistical analysis, LaTeX code writing, interpretation and organization of experimental results obtained by the authors, and Chinese-to-English translation. The tools were also used to improve the clarity and structure of the manuscript, and to assist with the preparation of tables, figures, references, and submission materials. Generative AI was not used to generate the datasets used in the paper, replace the execution of the proposed method and baseline models for comparison, or derive mathematical proofs. All methodological decisions, experimental conclusions, and final scientific judgments were made by the authors.

Reproducibility statement

Sections 3 and 4 describe the method and evaluation setup. Appendix D specifies data preparation, training, checkpoint selection, inference, and metric definitions; Appendices A and B provide per-shape results and fixed-checkpoint diagnostics.

References

X. Cheng, D. Xiao, H. Li, Z. Liu, and R. Chen. Routing on demand: DSNet for efficient progressive point cloud denoising. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 39111–39120, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Cheng_Routing_on_Demand_DSNet_for_Efficient_Progressive_Point_Cloud_Denoising_CVPR_2026_paper.html.

Dasith de Silva Edirimuni, Xuequan Lu, Zhiwen Shao, Gang Li, Antonio Robles-Kelly, and Ying He. Iterativepfn: True iterative point cloud filtering. In 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 13530–13539, 2023. doi: 10.1109/CVPR52729.2023.01300.

Dasith de Silva Edirimuni, Xuequan Lu, Gang Li, Lei Wei, Antonio Robles-Kelly, and Hongdong Li. Straightpcf: Straight point cloud filtering. In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 20721–20730, 2024. doi: 10.1109/CVPR52733.2024.01958.

Peng Du, Xingce Wang, Zhongke Wu, Xudong Ru, Xavier Granier, and Ying He. Lapda: Latent-space point cloud denoising with adaptivity. IEEE Transactions on Visualization and Computer Graphics, 32(2):1525–1539, 2026. doi: 10.1109/TVCG.2025.3635138.

C. Guo, W. Zhou, Z. Liu, and Y. He. You should learn to stop denoising on point clouds in advance. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pp. 3212–3219, 2025. doi: 10.1609/aaai.v39i3.32331.

C. Guo, Z. Liu, and Y. He. Guiding point cloud denoising with learned structural priors. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 40, pp. 4395–4402, 2026. doi: 10.1609/aaai.v40i6.42437.

Abiao Li, Chenlei Lv, Yuming Fang, Yifan Zuo, Jian Zhang, and Guofeng Mei. Pointgac: Geometric-aware codebook for masked point cloud modeling. In 2025 IEEE/CVF International Conference on Computer Vision (ICCV), 2025. doi: 10.1109/ICCV51701.2025.02317.

Fangzhou Lin, Yun Yue, Ziming Zhang, Songlin Hou, Kazunori Yamada, Vijaya Kolachalama, and Venkatesh Saligrama. Infocd: A contrastive chamfer distance loss for point cloud completion. In Advances in Neural Information Processing Systems, volume 36, pp. 76960–76973, 2023. doi: 10.52202/075280-3364.

Zheng Liu, Yaowu Zhao, Sijing Zhan, Yuanyuan Liu, Renjie Chen, and Ying He. Pcdnf: Revisiting learning-based point cloud denoising via joint normal filtering. IEEE Transactions on Visualization and Computer Graphics, 30(8):5419–5436, 2024. doi: 10.1109/TVCG.2023.3292464.

Zheng Liu, Zhenyu Huang, Maodong Pan, and Ying He. Deterministic point cloud diffusion for denoising. IEEE Transactions on Visualization and Computer Graphics, 32(2):1822–1834, 2026. doi: 10.1109/TVCG.2025.3621633.

Shitong Luo and Wei Hu. Differentiable manifold reconstruction for point cloud denoising. In Proceedings of the 28th ACM International Conference on Multimedia, pp. 1330–1338, 2020. doi: 10.1145/3394171.3413727.

Shitong Luo and Wei Hu. Score-based point cloud denoising. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 4563–4572, 2021. doi: 10.1109/ICCV48922.2021.00454.

A. Mao, Z. Du, Y.-H. Wen, J. Xuan, and Y.-J. Liu. PD-Flow: A point cloud denoising framework with normalizing flows. In Proceedings of the European Conference on Computer Vision, pp. 398–415, 2022. doi: 10.1007/978-3-031-20062-5_23.

Aihua Mao, Biao Yan, Zijing Ma, and Ying He. Denoising point clouds in latent space via graph convolution and invertible neural network. In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5768–5777, 2024. doi: 10.1109/CVPR52733.2024.00551.

Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas. 2018. URL https://arxiv.org/abs/1804.03999.

Yatian Pang, Eng Hock Francis Tay, Li Yuan, and Zhenghua Chen. Masked autoencoders for 3d point cloud self-supervised learning. volume 02, 2024. doi: 10.1142/S2811032324400010.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, pp. 3942–3951, 2018. doi: 10.1609/aaai.v32i1.11671.

Francesca Pistilli, Giulia Fracastoro, Diego Valsesia, and Enrico Magli. Learning graph-convolutional representations for point cloud denoising. In Proceedings of the European Conference on Computer Vision, pp. 103–118, 2020. doi: 10.1007/978-3-030-58565-5_7.

Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J. Mitra, and Maks Ovsjanikov. PointCleanNet: Learning to denoise and remove outliers from dense point clouds. Computer Graphics Forum, 39(1):185–203, 2020. doi: 10.1111/cgf.13753.

A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Advances in Neural Information Processing Systems, volume 32, 2017. URL https://par.nsf.gov/biblio/10195511.

A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In Advances in Neural Information Processing Systems, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

Mathias Vogel, Keisuke Tateno, Marc Pollefeys, Federico Tombari, Marie-Julie Rakotosaona, and Francis Engelmann. P2P-Bridge: Diffusion bridges for 3d point cloud denoising. In Proceedings of the European Conference on Computer Vision, pp. 184–201, 2024. doi: 10.1007/978-3-031-72627-9_11.

Xiangbin Wei, Yuanfeng Wang, Ao Xu, Lingyu Zhu, Dongyong Sun, Keren Li, Yang Li, and Qi Qin. Noise2score3d: Tweedie's approach for unsupervised point cloud denoising. In 2025 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 25993–26003, 2025. doi: 10.1109/ICCV51701.2025.02411.

Martin Weinmann, Boris Jutzi, Stefan Hinz, and Clément Mallet. Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers. ISPRS Journal of Photogrammetry and Remote Sensing, 105:286–304, 2015. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2015.01.016.

Tong Wu, Liang Pan, Junzhe Zhang, Tai Wang, Ziwei Liu, and Dahua Lin. Density-aware chamfer distance as a comprehensive metric for point cloud completion. In Advances in Neural Information Processing Systems, 2021.

Rajeev Yasarla and Vishal M. Patel. Uncertainty guided multi-scale residual learning-using a cycle spinning cnn for single image de-raining. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8397–8406, 2019. doi: 10.1109/CVPR.2019.00860.

Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2790–2799, 2018. doi: 10.1109/CVPR.2018.00295.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In Advances in Neural Information Processing Systems, volume 33, pp. 5824–5836, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3fe78a8acf5fda99de95303940a2420c-Paper.pdf.

Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 19291–19300, 2022. doi: 10.1109/CVPR52688.2022.01871.

Dongbo Zhang, Xuequan Lu, Hong Qin, and Ying He. Pointfilter: Point cloud filtering via encoder-decoder modeling. IEEE Transactions on Visualization and Computer Graphics, 27(3):2015–2027, 2021. doi: 10.1109/TVCG.2020.3027069.

Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing, 26(7):3142–3155, 2017. doi: 10.1109/TIP.2017.2662206.

Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. Point-m2ae: Multi-scale masked autoencoders for hierarchical point cloud pre-training. In Advances in Neural Information Processing Systems, volume 35, pp. 27061–27074, 2022. doi: 10.52202/068431-1962.

Qingyuan Zhou, Weidong Yang, Ben Fei, Jingyi Xu, Rui Zhang, Keyi Liu, Yeqi Luo, and Ying He. 3DMambaIPF: A state space model for iterative point cloud filtering via differentiable rendering. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pp. 10843–10851, 2025. doi: 10.1609/aaai.v39i10.33178.

A Statistical Reliability on PUNet

A.1 Analysis Design and Scope

Analysis unit and pairing. We resample the 20 PUNet evaluation shapes as clusters; none is excluded. Within each table, methods receive the same corrupted cloud and are paired by shape. We define the difference as $\Delta = \text{GPD}_{\text{rep}} - \text{Ours}$, so a positive value means that our method has lower error. CD and P2M are recomputed after unit-sphere normalization with the same PyTorch3D 0.7.8 implementation and are multiplied by 10^4.

Uncertainty estimation. For Gaussian noise, a percentile paired bootstrap resamples the 20 shapes with replacement and recomputes the mean difference. For the non-Gaussian suite, each shape is corrupted independently with seeds 42–46. A hierarchical paired bootstrap first resamples 20 shape clusters and then resamples the five paired corruption realizations within each selected shape. Both analyses use 100,000 resamples and bootstrap seed 20260826. The intervals are pointwise 95% intervals for individual condition–metric comparisons. We do not interpret them as simultaneous family-wise intervals or report multiple unadjusted p-values.

Model and checkpoint scope. The paired tables use one local GPD reproduction, denoted GPD_{rep}, and the complete model. GPD_{rep} uses the epoch-244 checkpoint with its original nearest-patch aggregation. Ours uses the epoch-299 checkpoint and distance-weighted overlap fusion. The crossed control in Appendix B.1 applies both fusion rules to both checkpoints, allowing the network contrast to be read under a shared aggregation rule.

The learning-rate scheduler used the official PUNet test geometries, so this benchmark is not a fully held-out evaluation of geometry. Prior implementations, including GPD and ASDN, used this protocol; we applied it to all locally reproduced methods. The PUNet results support protocol-matched comparisons rather than claims of leakage-free generalization.

A.2 Paired Gaussian Bootstrap

Table 5 gives the effect size, across-shape SD, shape-wise wins, and paired interval for every Gaussian condition. All 12 mean differences are positive. Eight intervals lie entirely above zero. The four intervals that cross zero are P2M at 10K–1%, 10K–2.5%, 50K–1%, and 50K–2%. These small mean differences are therefore not separated from zero by the corresponding pointwise resampling intervals. CD has an entirely positive interval in all six conditions. Removing any one test shape leaves the mean difference positive for every condition–metric pair.

Condition	Metric	GPD_{rep}	Ours	Δ	Reduction	95% paired CI	Wins
10K-1%	CD	1.792 ± 0.455	1.776 ± 0.451	+0.0161	+0.9%	[+0.0116, +0.0207]	18/20
10K-1%	P2M	0.466 ± 0.166	0.463 ± 0.170	+0.0031	+0.7%	[-0.0046, +0.0091]	15/20
10K-2%	CD	2.424 ± 0.903	2.408 ± 0.912	+0.0160	+0.7%	[+0.0064, +0.0254]	16/20
10K-2%	P2M	0.725 ± 0.235	0.713 ± 0.235	+0.0122	+1.7%	[+0.0043, +0.0208]	15/20
10K-2.5%	CD	2.646 ± 0.912	2.619 ± 0.907	+0.0263	+1.0%	[+0.0056, +0.0446]	17/20
10K-2.5%	P2M	0.909 ± 0.276	0.898 ± 0.284	+0.0116	+1.3%	[-0.0068, +0.0274]	16/20
50K-1%	CD	0.465 ± 0.185	0.461 ± 0.183	+0.0048	+1.0%	[+0.0018, +0.0075]	18/20
50K-1%	P2M	0.290 ± 0.152	0.287 ± 0.151	+0.0027	+0.9%	[-0.0002, +0.0051]	18/20
50K-2%	CD	0.616 ± 0.208	0.603 ± 0.201	+0.0132	+2.1%	[+0.0086, +0.0193]	18/20
50K-2%	P2M	0.402 ± 0.158	0.398 ± 0.154	+0.0038	+1.0%	[-0.0019, +0.0108]	14/20
50K-2.5%	CD	0.854 ± 0.227	0.750 ± 0.195	+0.1043	+12.2%	[+0.0717, +0.1464]	20/20
50K-2.5%	P2M	0.573 ± 0.153	0.505 ± 0.154	+0.0675	+11.8%	[+0.0427, +0.0996]	20/20

Table 5: Paired Gaussian reliability analysis against a locally reproduced GPD checkpoint. CD, P2M, Δ, and interval endpoints are in 10^{-4} units. Method entries are mean ± sample SD across 20 scheduler-observed evaluation shapes. $\Delta = \text{GPD}_{\text{rep}} - \text{Ours}$, so positive values favour Ours. Intervals are pointwise 95% percentile paired-bootstrap intervals over shapes (100,000 resamples); wins count positive unrounded shape-level differences.

A.3 Repeated Non-Gaussian Corruptions

Table 6 reports the controlled five-seed suite. All 16 mean differences are positive and 13 hierarchical intervals lie entirely above zero. The intervals cross zero for P2M under 1% uniform, 2% uniform, and 1% discrete noise. The paired mean remains positive after any single shape is omitted in all 16 comparisons. These results address corruption-realization variability for the comparison with GPD_{rep}; the historical external-method table remains a single-seed comparison.

Noise	Level	Metric	GPD_{rep}	Ours	Δ	Reduction	95% hierarchical CI	Wins
Ani	1%	CD	1.807 ± 0.0038	1.788 ± 0.0043	+0.0188	+1.0%	[+0.0131, +0.0254]	19/20
Ani	1%	P2M	0.478 ± 0.0007	0.471 ± 0.0010	+0.0069	+1.4%	[+0.0007, +0.0126]	17/20
Ani	2%	CD	2.480 ± 0.0054	2.456 ± 0.0050	+0.0238	+1.0%	[+0.0137, +0.0338]	19/20
Ani	2%	P2M	0.772 ± 0.0044	0.758 ± 0.0033	+0.0135	+1.7%	[+0.0051, +0.0219]	17/20
Lap	1%	CD	2.089 ± 0.0031	2.069 ± 0.0034	+0.0202	+1.0%	[+0.0141, +0.0273]	19/20
Lap	1%	P2M	0.571 ± 0.0037	0.560 ± 0.0032	+0.0110	+1.9%	[+0.0043, +0.0174]	17/20
Lap	2%	CD	3.156 ± 0.0313	2.993 ± 0.0264	+0.1632	+5.2%	[+0.1231, +0.2052]	20/20
Lap	2%	P2M	1.348 ± 0.0257	1.204 ± 0.0231	+0.1440	+10.7%	[+0.1090, +0.1811]	20/20
Uni	1%	CD	0.668 ± 0.0019	0.653 ± 0.0015	+0.0148	+2.2%	[+0.0123, +0.0174]	20/20
Uni	1%	P2M	0.374 ± 0.0008	0.371 ± 0.0007	+0.0032	+0.8%	[-0.0019, +0.0075]	15/20
Uni	2%	CD	1.757 ± 0.0030	1.734 ± 0.0030	+0.0230	+1.3%	[+0.0168, +0.0294]	19/20
Uni	2%	P2M	0.454 ± 0.0037	0.448 ± 0.0027	+0.0055	+1.2%	[-0.0016, +0.0110]	18/20
Dis	1%	CD	0.671 ± 0.0025	0.655 ± 0.0020	+0.0157	+2.3%	[+0.0130, +0.0185]	20/20
Dis	1%	P2M	0.373 ± 0.0009	0.369 ± 0.0004	+0.0043	+1.1%	[-0.0003, +0.0083]	17/20
Dis	2%	CD	1.574 ± 0.0055	1.564 ± 0.0046	+0.0104	+0.7%	[+0.0034, +0.0186]	14/20
Dis	2%	P2M	0.456 ± 0.0027	0.447 ± 0.0021	+0.0084	+1.8%	[+0.0018, +0.0139]	18/20

Table 6: Reliability analysis over five independently generated non-Gaussian corruption realizations (seeds 42–46). All quantities except percentages are in 10^{-4} units. Method entries are the grand mean ± sample SD of the five seed-level means. The pointwise 95% hierarchical paired-bootstrap interval resamples 20 shapes as clusters and then the five paired realizations within each selected shape; the 100 observations are not treated as independent. Wins are computed from each shape's mean over the five realizations.

A.4 Five-Method Shape-Level Gaussian Results

Purpose. Table 1 reports means over the 20 PUNet evaluation shapes. To expose the variation hidden by those aggregates, the following six tables report every named shape at each density and noise level.

Matched evaluation protocol. We evaluated Ours, GPD, ASDN, P2P-Bridge, and IterativePFN on the same corrupted input cloud for each shape and condition. Each output was matched to the clean reference and mesh by shape name. The four competitor columns are fixed-checkpoint local reproductions, whereas their entries in Table 1 are previously recorded or published aggregates. Small differences between a reproduced mean and its main-table entry can arise when the available checkpoint or inference seed differs. Comparisons within each appendix table remain paired by input and use one metric implementation.

Summary rows. "Across-shape SD" is the sample SD of the 20 shape-level values. "Shape-wise wins" counts the shapes on which a method has the lowest unrounded metric value. A numerical tie is credited to every tied method. Ours has the lowest locally reproduced mean for all 12 condition–metric pairs. Its per-shape win counts range from 13 to 19 for CD and from 12 to 17 for P2M, showing that the aggregate ordering is not attributable to one or two test objects. These six tables contain one fixed corruption realization and one fixed output per method–shape–condition tuple. The preceding paired analyses provide the corresponding resampling intervals and repeated-corruption check.

A.4.1 10K Input Points

Table 7: Shape-level PUNet results at 10K input points and 1% Gaussian noise. All values are recomputed with one common evaluator and reported in $\times 10^{-4}$; lower is better. Bold marks the best local result in each shape–metric pair.

Shape	IterativePFN		P2P-Bridge		ASDN		GPD		Ours	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Icosahedron	2.657	0.349	2.977	0.482	2.486	0.373	2.435	0.349	2.400	0.347
Octahedron	2.336	0.395	2.460	0.476	2.095	0.413	2.077	0.418	2.040	0.386
camel	1.470	0.680	2.014	1.165	1.248	0.575	1.193	0.523	1.186	0.513
casting	2.643	0.618	3.614	1.329	2.519	0.740	2.440	0.667	2.426	0.705
chair	1.752	0.592	2.921	1.630	1.554	0.578	1.424	0.490	1.376	0.454
coverrear_Lp	2.375	0.356	2.594	0.518	2.129	0.352	2.104	0.339	2.052	0.307
cow	1.650	0.595	1.606	0.585	1.368	0.483	1.355	0.478	1.339	0.477
duck	2.356	0.659	2.457	0.716	2.125	0.673	2.089	0.662	2.069	0.647
eight	1.729	0.352	1.658	0.374	1.452	0.351	1.428	0.335	1.423	0.330
elephant	1.625	0.356	1.672	0.447	1.391	0.320	1.362	0.305	1.362	0.309
elk	2.391	0.604	2.586	0.704	2.161	0.577	2.079	0.538	2.055	0.524
fandisk	1.760	0.316	1.806	0.416	1.540	0.325	1.512	0.298	1.507	0.299
genus3	2.015	0.309	2.059	0.375	1.783	0.329	1.759	0.314	1.737	0.305
horse	1.331	0.351	1.338	0.397	1.106	0.300	1.097	0.295	1.091	0.296
kitten	2.056	0.394	2.047	0.442	1.802	0.370	1.776	0.354	1.760	0.355
moai	2.067	0.444	2.072	0.504	1.806	0.420	1.790	0.426	1.771	0.418
pig	1.893	0.772	1.875	0.814	1.637	0.753	1.611	0.746	1.600	0.745
quadric	1.691	0.538	1.827	0.744	1.451	0.480	1.460	0.499	1.434	0.502
sculpt	2.865	0.874	3.557	1.125	2.832	0.924	2.781	0.910	2.728	0.902
star	2.441	0.462	2.553	0.489	2.236	0.466	2.181	0.441	2.160	0.443
Mean	2.055	0.501	2.285	0.687	1.836	0.490	1.798	0.469	1.776	0.463
Across-shape SD	0.433	0.166	0.630	0.356	0.469	0.174	0.460	0.168	0.451	0.170
Shape-wise wins	0	2	0	0	0	1	1	5	19	12

Table 8: Shape-level PUNet results at 10K input points and 2% Gaussian noise. All values are recomputed with one common evaluator and reported in $\times 10^{-4}$; lower is better. Bold marks the best local result in each shape–metric pair.

Shape	IterativePFN		P2P-Bridge		ASDN		GPD		Ours	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Icosahedron	4.492	0.605	4.334	0.728	3.755	0.580	3.712	0.571	3.706	0.562
Octahedron	3.309	0.525	3.145	0.621	2.680	0.517	2.662	0.517	2.614	0.468
camel	2.058	1.247	2.670	1.665	1.773	1.157	1.708	1.050	1.666	1.013
casting	4.357	1.492	7.659	4.049	4.042	1.589	3.493	1.185	3.508	1.227
chair	2.306	1.020	3.089	1.675	2.039	0.978	1.795	0.805	1.755	0.800
coverrear_Lp	3.553	0.759	3.558	0.918	2.829	0.562	2.813	0.566	2.748	0.501
cow	2.190	0.941	2.130	0.979	1.721	0.760	1.734	0.782	1.678	0.737
duck	3.427	0.875	3.210	0.972	2.774	0.863	2.701	0.841	2.705	0.810
eight	2.149	0.479	1.933	0.497	1.657	0.445	1.625	0.424	1.601	0.419
elephant	2.292	0.811	2.443	1.035	1.879	0.705	1.827	0.682	1.818	0.676
elk	3.560	1.037	3.519	1.112	2.891	0.900	2.756	0.823	2.693	0.777
fandisk	2.349	0.601	2.305	0.717	1.917	0.567	1.875	0.521	1.852	0.515
genus3	2.706	0.467	2.596	0.585	2.191	0.480	2.158	0.466	2.120	0.449
horse	1.765	0.677	1.827	0.788	1.410	0.573	1.411	0.577	1.380	0.558
kitten	2.801	0.711	2.649	0.760	2.224	0.581	2.185	0.576	2.192	0.586
moai	2.781	0.748	2.667	0.834	2.223	0.642	2.219	0.658	2.176	0.624
pig	2.540	1.070	2.378	1.093	2.040	1.026	1.990	0.998	2.005	0.991
quadric	2.261	1.039	2.384	1.243	1.844	0.868	1.837	0.887	1.803	0.873
sculpt	6.178	1.129	6.137	1.373	5.299	1.167	5.165	1.119	5.182	1.127
star	3.779	0.636	3.417	0.610	2.988	0.563	2.941	0.544	2.959	0.539
Mean	3.043	0.843	3.203	1.113	2.509	0.776	2.430	0.730	2.408	0.713
Across-shape SD	1.070	0.274	1.427	0.767	0.953	0.293	0.900	0.227	0.912	0.235
Shape-wise wins	0	0	0	0	0	1	6	3	14	16

Table 9: Shape-level PUNet results at 10K input points and 2.5% Gaussian noise. All values are recomputed with one common evaluator and reported in $\times 10^{-4}$; lower is better. Bold marks the best local result in each shape–metric pair.

Shape	IterativePFN		P2P-Bridge		ASDN		GPD		Ours	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Icosahedron	4.856	0.757	4.600	0.878	3.948	0.721	3.918	0.703	3.861	0.675
Octahedron	3.588	0.659	3.303	0.720	2.799	0.640	2.786	0.628	2.689	0.544
camel	2.463	1.660	3.274	2.411	2.189	1.591	2.108	1.421	2.077	1.429
casting	4.909	1.831	9.838	5.599	4.524	1.952	3.739	1.363	3.816	1.471
chair	2.617	1.193	3.489	1.974	2.367	1.241	1.970	0.965	2.083	1.061
coverrear_Lp	3.951	1.105	3.956	1.278	3.042	0.830	3.137	0.913	2.967	0.803
cow	2.463	1.143	2.528	1.322	1.952	0.978	1.999	0.990	1.889	0.901
duck	3.709	1.045	3.396	1.102	2.974	1.023	2.850	0.957	2.860	0.955
eight	2.357	0.631	2.082	0.614	1.779	0.545	1.738	0.516	1.714	0.505
elephant	2.718	1.122	2.808	1.363	2.205	0.980	2.163	0.960	2.107	0.933
elk	3.890	1.259	3.956	1.486	3.001	1.047	2.925	0.993	2.861	0.936
fandisk	2.721	0.895	2.627	0.992	2.118	0.766	2.122	0.742	2.036	0.675
genus3	2.943	0.618	2.857	0.810	2.308	0.613	2.348	0.631	2.282	0.590
horse	2.070	0.883	2.797	1.596	1.613	0.745	1.709	0.798	1.648	0.763
kitten	3.114	0.942	2.913	0.978	2.367	0.754	2.397	0.784	2.376	0.764
moai	3.028	0.919	2.797	0.940	2.364	0.788	2.375	0.791	2.326	0.762
pig	2.830	1.295	2.644	1.295	2.310	1.294	2.335	1.259	2.334	1.250
quadric	2.593	1.320	2.676	1.543	2.083	1.141	2.067	1.098	2.002	1.067
sculpt	6.815	1.287	6.655	1.584	5.627	1.327	5.421	1.203	5.456	1.250
star	3.970	0.734	3.521	0.691	3.088	0.679	3.025	0.657	3.006	0.619
Mean	3.380	1.065	3.636	1.459	2.733	0.983	2.657	0.919	2.620	0.898
Across-shape SD	1.138	0.327	1.760	1.073	0.982	0.359	0.891	0.253	0.907	0.284
Shape-wise wins	0	0	0	0	3	2	4	4	13	14

A.4.2 50K Input Points

Table 10: Shape-level PUNet results at 50K input points and 1% Gaussian noise. All values are recomputed with one common evaluator and reported in $\times 10^{-4}$; lower is better. Bold marks the best local result in each shape–metric pair.

Shape	IterativePFN		P2P-Bridge		ASDN		GPD		Ours	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Icosahedron	0.912	0.175	0.862	0.200	0.751	0.190	0.734	0.183	0.724	0.174
Octahedron	0.671	0.305	0.624	0.327	0.540	0.319	0.539	0.326	0.521	0.310
camel	0.356	0.213	0.337	0.195	0.292	0.205	0.291	0.194	0.277	0.188
casting	0.762	0.327	0.796	0.410	0.643	0.346	0.624	0.332	0.606	0.330
chair	0.809	0.466	0.813	0.490	0.779	0.467	0.448	0.276	0.352	0.228
coverrear_Lp	0.682	0.152	0.658	0.190	0.546	0.159	0.553	0.168	0.531	0.150
cow	0.413	0.316	0.387	0.311	0.340	0.311	0.337	0.307	0.318	0.293
duck	0.689	0.518	0.634	0.529	0.562	0.528	0.547	0.521	0.537	0.516
eight	0.433	0.268	0.393	0.278	0.354	0.278	0.343	0.275	0.336	0.269
elephant	0.400	0.151	0.441	0.217	0.322	0.153	0.317	0.153	0.310	0.147
elk	0.673	0.332	0.673	0.380	0.537	0.328	0.535	0.330	0.541	0.338
fandisk	0.434	0.177	0.426	0.211	0.353	0.185	0.348	0.186	0.336	0.174
genus3	0.533	0.216	0.486	0.230	0.430	0.229	0.423	0.227	0.411	0.216
horse	0.314	0.149	0.426	0.258	0.258	0.151	0.258	0.155	0.246	0.145
kitten	0.543	0.200	0.500	0.209	0.438	0.205	0.430	0.204	0.422	0.196
moai	0.552	0.239	0.510	0.249	0.451	0.242	0.448	0.243	0.434	0.232
pig	0.480	0.610	0.445	0.613	0.392	0.621	0.383	0.620	0.374	0.612
quadric	0.414	0.225	0.400	0.234	0.338	0.221	0.337	0.222	0.328	0.216
sculpt	1.270	0.669	1.224	0.718	1.052	0.682	1.039	0.682	1.022	0.674
star	0.751	0.336	0.684	0.341	0.601	0.344	0.599	0.347	0.585	0.335
Mean	0.605	0.302	0.586	0.330	0.499	0.308	0.477	0.297	0.461	0.287
Across-shape SD	0.229	0.153	0.217	0.151	0.197	0.155	0.183	0.150	0.183	0.151
Shape-wise wins	0	6	0	0	0	1	1	0	19	13

Table 11: Shape-level PUNet results at 50K input points and 2% Gaussian noise. All values are recomputed with one common evaluator and reported in $\times 10^{-4}$; lower is better. Bold marks the best local result in each shape–metric pair.

Shape	IterativePFN		P2P-Bridge		ASDN		GPD		Ours	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Icosahedron	1.067	0.252	0.999	0.288	0.872	0.273	0.844	0.260	0.830	0.249
Octahedron	0.775	0.358	0.728	0.404	0.604	0.360	0.597	0.363	0.573	0.345
camel	0.652	0.486	0.918	0.710	0.587	0.502	0.545	0.456	0.537	0.470
casting	1.518	0.811	1.721	1.142	1.567	1.012	1.108	0.662	1.092	0.650
chair	0.761	0.433	1.992	1.363	0.892	0.548	0.647	0.431	0.609	0.430
coverrear_Lp	0.868	0.270	0.947	0.426	0.670	0.248	0.643	0.234	0.629	0.220
cow	0.592	0.453	0.608	0.489	0.514	0.444	0.496	0.433	0.475	0.423
duck	0.819	0.593	0.765	0.620	0.666	0.600	0.633	0.580	0.628	0.576
eight	0.531	0.328	0.492	0.347	0.441	0.336	0.410	0.317	0.402	0.312
elephant	0.642	0.336	0.952	0.656	0.538	0.319	0.485	0.285	0.477	0.283
elk	1.096	0.592	0.940	0.590	0.926	0.566	0.664	0.425	0.667	0.431
fandisk	0.580	0.271	0.611	0.364	0.487	0.279	0.450	0.255	0.434	0.243
genus3	0.646	0.287	0.626	0.336	0.535	0.299	0.507	0.284	0.489	0.270
horse	0.491	0.281	0.897	0.656	0.396	0.258	0.384	0.256	0.385	0.265
kitten	0.685	0.307	0.657	0.337	0.556	0.298	0.532	0.286	0.522	0.278
moai	0.735	0.367	0.708	0.399	0.612	0.355	0.599	0.350	0.586	0.342
pig	0.606	0.703	0.609	0.733	0.508	0.707	0.486	0.697	0.472	0.687
quadric	0.635	0.446	0.664	0.488	0.531	0.423	0.493	0.396	0.492	0.393
sculpt	1.497	0.752	1.453	0.866	1.165	0.751	1.139	0.744	1.120	0.728
star	0.856	0.387	0.767	0.397	0.665	0.390	0.653	0.384	0.634	0.370
Mean	0.803	0.436	0.903	0.581	0.687	0.448	0.616	0.405	0.603	0.398
Across-shape SD	0.289	0.170	0.391	0.282	0.280	0.200	0.203	0.154	0.201	0.154
Shape-wise wins	0	0	0	0	0	0	2	3	18	17

Table 12: Shape-level PUNet results at 50K input points and 2.5% Gaussian noise. All values are recomputed with one common evaluator and reported in $\times 10^{-4}$; lower is better. Bold marks the best local result in each shape–metric pair.

Shape	IterativePFN		P2P-Bridge		ASDN		GPD		Ours	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Icosahedron	1.177	0.325	1.085	0.346	0.982	0.363	1.495	0.748	0.986	0.358
Octahedron	0.871	0.421	0.804	0.458	0.680	0.420	1.227	0.845	0.695	0.430
camel	1.091	0.837	1.273	1.022	0.915	0.771	0.856	0.727	0.785	0.684
casting	1.861	1.011	2.399	1.621	1.981	1.258	1.378	0.849	1.269	0.778
chair	1.055	0.651	2.466	1.828	1.053	0.664	0.905	0.607	0.752	0.521
coverrear_Lp	1.038	0.398	1.199	0.646	0.807	0.358	0.874	0.395	0.769	0.330
cow	0.868	0.669	0.901	0.742	0.749	0.632	0.708	0.587	0.637	0.540
duck	0.911	0.654	0.885	0.716	0.749	0.666	1.009	0.844	0.770	0.666
eight	0.666	0.414	0.601	0.427	0.542	0.411	0.545	0.409	0.511	0.390
elephant	1.003	0.614	1.462	1.105	0.810	0.524	0.716	0.453	0.686	0.444
elk	1.150	0.635	1.187	0.787	0.957	0.607	0.948	0.608	0.836	0.534
fandisk	0.812	0.436	0.814	0.523	0.648	0.398	0.627	0.384	0.593	0.355
genus3	0.794	0.389	0.744	0.425	0.651	0.391	0.650	0.385	0.599	0.347
horse	0.880	0.577	1.359	1.051	0.586	0.404	0.563	0.387	0.507	0.350
kitten	0.804	0.400	0.786	0.446	0.650	0.385	0.748	0.452	0.670	0.392
moai	0.892	0.478	0.868	0.527	0.724	0.446	0.884	0.554	0.762	0.458
pig	0.736	0.789	0.810	0.879	0.627	0.803	0.647	0.806	0.610	0.781
quadric	0.969	0.694	1.177	0.950	0.766	0.622	0.719	0.575	0.683	0.543
sculpt	1.608	0.817	1.583	0.964	1.234	0.817	1.214	0.809	1.166	0.771
star	0.936	0.440	0.818	0.434	0.719	0.444	0.950	0.612	0.713	0.435
Mean	1.006	0.583	1.161	0.795	0.842	0.569	0.883	0.602	0.750	0.505
Across-shape SD	0.285	0.186	0.508	0.400	0.319	0.223	0.269	0.173	0.195	0.154
Shape-wise wins	0	2	0	1	5	3	0	0	15	14

A.5 Local Geometry Stratification

Global CD and P2M can obscure errors concentrated around local geometric features. We therefore analyze the 50K, 2.5% Gaussian condition using a localized reference-to-output error. For each clean reference point, the error is its squared distance to the nearest denoised output point after the same reference-defined unit-sphere normalization used for CD, multiplied by 10^4. Subset errors are averaged within each shape and then across eligible shapes, so shapes with more feature points do not receive greater weight.

The curvature statistic is the smallest eigenvalue divided by the trace of the 32-neighbor covariance matrix. Low-, mid-, and high-curvature subsets are the within-shape tertiles of this statistic. A mesh edge is marked sharp when its two incident face tangent planes differ by at least 60°; reference points within 0.01 unit-sphere radius of such an edge form the sharp-edge subset. For the thin-sheet proxy, we search the 128 nearest reference points for a second surface sheet whose normal is parallel within 30° and whose connecting direction has absolute cosine at least 0.70 with both normals. A valid separation lies between 0.006 and 0.04 unit-sphere radius. Shape-level sharp-edge and thin-sheet values are retained only when the corresponding subset contains at least 50 points.

Method	Low κ	Mid κ	High κ	Sharp edge	Thin sheet
IterativePFN	0.494±0.161	0.526±0.166	0.653±0.216	0.893±0.301	0.771±0.279
P2P-Bridge	0.424±0.156	0.434±0.156	<u>0.527±0.208</u>	**0.698±0.300**	**0.627±0.210**
ASDN	<u>0.354±0.135</u>	<u>0.402±0.201</u>	0.561±0.276	0.903±0.323	0.896±0.325
GPD	0.369±0.128	0.406±0.127	0.533±0.205	0.788±0.306	0.726±0.334
Ours	**0.310±0.087**	**0.341±0.087**	**0.471±0.167**	<u>0.748±0.289</u>	<u>0.687±0.296</u>

Table 13: Localized reference-to-output error on PUNet 50K at 2.5% Gaussian noise. Entries are shape-balanced mean ± sample SD in 10^{-4} units. Curvature strata include all 20 shapes; sharp-edge and thin-sheet summaries include 18 and 10 shapes, respectively, after requiring at least 50 reference points in a subset. Lower is better; bold and underlining denote the best and second-best means.

Our method has the lowest shape-balanced error in all three curvature strata. It ranks second on the sharp-edge and thin-sheet subsets, where P2P-Bridge has the lowest mean. Thus, the local analysis supports detail retention across curvature levels while distinguishing it from uniform superiority on every geometric subset.

B Inference-Only Controls and Diagnostics

All experiments in this section reuse the reported fixed checkpoints. They change only the inference rule, resample evaluation units, or expose quantities already produced by the network; no parameter is updated.

B.1 Patch-Fusion Control

To separate network and aggregation effects, we evaluate the local GPD checkpoint and our complete checkpoint with both nearest-patch assignment and distance-weighted overlap fusion. For each of the four combinations, patch size, sixfold coverage, input clouds, and external recovery passes are held fixed. The 1% and 2% conditions use two passes, and the 2.5% condition uses three. Table 14 reports the six PUNet conditions and paired shape-bootstrap intervals for the two contrasts that isolate model choice under a common fusion rule.

Condition	Metric	GPD-N	GPD-W	Ours-N	Ours-W	Δ_N	Δ_W	W/L/T$_W$
10K-1%	CD	1.792	1.784	1.784	1.776	+0.0076	+0.0082	16/4/0
10K-1%	P2M	0.466	0.463	0.468	0.463	-0.0016	+0.0001	11/9/0
10K-2%	CD	2.424	2.416	2.421	2.408	+0.0028	+0.0077	13/7/0
10K-2%	P2M	0.725	0.718	0.723	0.713	+0.0022	+0.0051	12/8/0
10K-2.5%	CD	2.646	2.633	2.644	2.619	+0.0019	+0.0132	14/6/0
10K-2.5%	P2M	0.909	0.901	0.914	0.898	-0.0044	+0.0036	12/8/0
50K-1%	CD	0.465	0.465	0.462	0.461	+0.0036	+0.0040	18/2/0
50K-1%	P2M	0.290	0.290	0.288	0.287	+0.0020	+0.0024	18/2/0
50K-2%	CD	0.616	0.610	0.609	0.603	+0.0065	+0.0071	15/5/0
50K-2%	P2M	0.402	0.401	0.400	0.398	+0.0016	+0.0023	15/5/0
50K-2.5%	CD	0.854	0.782	0.801	0.750	+0.0533	+0.0324	19/1/0
50K-2.5%	P2M	0.573	0.526	0.539	0.505	+0.0343	+0.0203	17/3/0

Table 14: Network-by-fusion control on the same PUNet inputs. N and W denote nearest-patch and distance-weighted overlap fusion. Δ_N and Δ_W are GPD minus Ours under the corresponding shared fusion rule; positive values favour Ours. W/L/T$_W$ counts shape-wise wins, losses, and ties for Ours under weighted fusion. CD and P2M use 10^{-4} units.

Weighted fusion lowers the mean error for both checkpoints in all 12 condition–metric pairs. Under a common weighted rule, our checkpoint has the lower mean in all 12 pairs; six paired intervals are entirely positive and six include zero. Under a common nearest rule, it has the lower mean in ten pairs, with the two exceptions being P2M at 10K–1% and 10K–2.5%. The 50K–2.5% advantage remains under either fusion rule. With weighted fusion, the paired differences are +0.0324 for CD and +0.0203 for P2M, with wins on 19 and 17 of the 20 shapes, respectively. Patch weighting therefore contributes to the reported performance, but it does not account for the dense, strong-noise result by itself.

B.2 Paired PC-Net Uncertainty

The ten PC-Net shapes support a paired cross-dataset analysis. At 1% noise, our checkpoint wins on seven shapes for both CD and P2M, while both paired intervals include zero. At 2.5%, it wins on all ten shapes and both intervals are positive. The result separates a clear strong-corruption effect from the less certain mild-noise ordering.

Noise	Metric	GPD	Ours	Δ	95% paired CI	W/L/T
1%	CD	0.793	0.789	+0.0038	[-0.0201, +0.0262]	7/3/0
1%	P2M	0.332	0.320	+0.0113	[-0.0083, +0.0360]	7/3/0
2.5%	CD	1.564	1.420	+0.1435	[+0.0791, +0.2442]	10/0/0
2.5%	P2M	0.994	0.876	+0.1175	[+0.0576, +0.2053]	10/0/0

Table 15: Paired PC-Net comparison over ten shapes. Δ is GPD minus Ours, so positive values favour Ours. Intervals use 100,000 paired shape bootstrap resamples. W/L/T counts shape-wise wins, losses, and ties for Ours. CD and P2M use 10^{-4} units.

B.3 PC-Net Qualitative Results

Figure 4 compares two PC-Net shapes under the same cross-dataset setting as Table 2. The method order and point-wise error colors follow Figure 3. Both examples retain the PC-Net geometry and viewpoint while exposing local residual error around the box openings and the thin ornamental structures of the column.

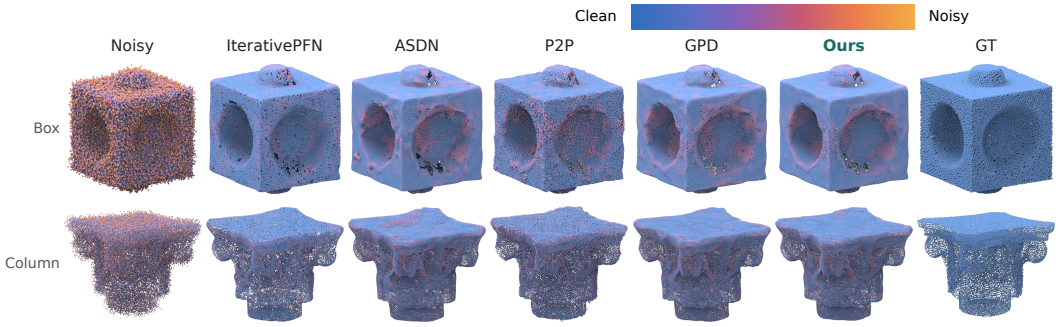

Figure 4: Point-wise P2M distance visualizations on the PC-Net Box and Column shapes from the cross-dataset test set. All models are trained on PUNet and applied without PC-Net fine-tuning. Columns show the noisy input, IterativePFN, ASDN, P2P-Bridge (P2P), GPD, our method, and the ground truth (GT); blue-to-orange colors indicate lower-to-higher point-wise error.

B.4 Training Record and Common-Hardware Inference Cost

Table 16 accounts for the single reported training trajectory from its checkpoint and TensorBoard events. No training was run for this analysis. The 197.1 hours are elapsed time in the event segments that advance the optimizer to step 1.2M, not an estimate from nominal epoch length.

Table 16: Resource record for the single reported training run. Elapsed GPU time is reconstructed from the TensorBoard wall-clock intervals that advance the optimizer trajectory to the epoch-299 checkpoint; a 400-step aborted segment that was immediately re-executed is excluded. This is an accounting record, not an additional run.

Quantity	Value
Completed epochs	300
Optimizer steps	1,200,000
Feature-network parameters	1.402 M
Active inference parameters	1.044 M
Event-log elapsed GPU time	197.1 h
Training devices	1 GPU

Table 17 profiles four fixed checkpoints on the same RTX 3090, the same five predeclared shapes, and the same 2% inputs at each density. Every process receives one warm-up cloud; each subsequent complete cloud is synchronized and timed separately. Ours adds 0.157M active parameters over GPD and takes 0.11/0.22 seconds longer at 10K/50K, while retaining similar allocated-memory peaks. All four methods use two external recovery passes; IterativePFN retains its four learned stages within each pass. The timing therefore measures the complete implemented pipelines, including patch aggregation, rather than an isolated network forward pass.

Table 17: Common-hardware fixed-checkpoint inference profile on an RTX 3090 at 2% Gaussian noise. All methods use two external recovery passes. Full counts all parameters stored in the denoising feature network; Active counts only modules executed at test time. Time is mean ± sample SD over the same five complete point clouds after one unreported warm-up run. Peak is process-local allocated CUDA memory. Disk I/O and unit-sphere normalization are excluded. Ext./native gives external passes and learned stages within each pass; IterativePFN retains four native stages per pass.

Input	Method	Full (M)	Active (M)	Time (s)	Peak (MiB)	Ext./native
10K	IterativePFN	3.196	3.196	21.85 ± 0.09	264	2/4
10K	ASDN	0.819	0.819	4.64 ± 0.07	160	2/1
10K	GPD	0.887	0.887	1.72 ± 0.03	153	2/1
10K	Ours	1.402	1.044	1.83 ± 0.01	155	2/1
50K	IterativePFN	3.196	3.196	36.61 ± 0.60	1365	2/4
50K	ASDN	0.819	0.819	13.17 ± 0.10	799	2/1
50K	GPD	0.887	0.887	5.23 ± 0.05	231	2/1
50K	Ours	1.402	1.044	5.45 ± 0.06	237	2/1

B.5 Interpretation and Protocol Scope

The EMA teacher has no separate clean-reconstruction objective: clean inputs change the view from which targets are extracted, while EMA smooths the student's parameters. Clean/noisy code assignments differ more under stronger corruption (Table 22), but that diagnostic does not establish that the teacher is intrinsically better than an independently trained teacher. The shallow geometry labels are a separate case: GMEM calculates them from clean coordinates along the teacher path, so they would be unchanged if calculated directly with the same neighborhoods and sampling indices. The teacher supplies the aligned clean feature and code targets, not a learned substitute for these geometric measurements.

The published aggregates in Table 1 retain their source protocols. The shape-wise local comparisons and the inference-cost profile instead use shared inputs and evaluation code; the local inference profile also fixes the external pass count at two for all four methods (Table 17). This separates protocol-matched local evidence from cross-paper rankings. The checkpoint was trained once, and the learning-rate scheduler used the official PUNet evaluation geometries (Appendix D.6); the paired intervals therefore quantify shape variation conditional on that training trajectory, not robustness to a disjoint validation split or new training seeds.

B.6 Additional Metrics and Recovery-Pass Sensitivity

We re-evaluate the saved outputs of all five methods on the same 20 PUNet shapes at 50K points and 2.5% Gaussian noise. These are the existing three-pass outputs, not new training runs. CD and P2M use the common evaluator of Appendix D.6. For density-aware Chamfer distance (DCD) (Wu et al., 2021), we normalize both clouds by the clean reference's unit-sphere transform, find nearest neighbors in both directions, and apply the published multiplicity correction with $\alpha = 1000$ and $\lambda = 1$. We retain all 50K points rather than downsampling either method output. The normal metric is the mean unsigned angular error between a 32-neighbor PCA normal at each denoised point and the normal of its closest ground-truth mesh triangle. The absolute dot product makes it invariant to normal orientation. We report this mesh-referenced normal-angle diagnostic separately from Surface Normal Concordance (SNC).

Our saved outputs have the lowest mean in all four columns of Table 18. The DCD ordering is not identical to the CD ordering across the other methods, so the additional metric contributes information rather than simply repeating CD.

We also vary only the number of external recovery passes for the released GPD checkpoint and our fixed checkpoint. The scan uses the five shapes preselected for the common-hardware profile (Icosahedron, camel, cow, duck, and horse), the same noisy 50K–2.5% inputs, seed 2025, patch size 1000, sixfold coverage, and five sampling steps. Each next pass starts from

Table 18: Inference-only metric re-evaluation on the 20 PUNet shapes at 50K points and 2.5% Gaussian noise. All outputs are from fixed checkpoints; lower is better. CD and P2M use the shared evaluation implementation. DCD uses $\alpha = 1000$, $\lambda = 1$ after clean-reference unit-sphere normalization. Normal angle is unsigned PCA-to-mesh error (32 neighbors), not SNC.

Method	CD $\times 10^4$	P2M $\times 10^4$	DCD	Normal angle (°)
IterativePFN	1.006	0.583	0.3748	10.34
P2P-Bridge	1.161	0.795	0.3459	13.09
ASDN	0.842	0.569	0.2204	9.30
GPD	0.883	0.602	0.2219	9.24
Ours	0.750	0.505	0.2055	8.87

Table 19: External-pass sensitivity on five prespecified PUNet 50K/2.5% shapes. The GPD checkpoint uses nearest-patch fusion; ours uses weighted fusion, matching their respective original protocols. Only the pass count varies within each method. Lower is better.

Method	Passes	CD $\times 10^4$	P2M $\times 10^4$	DCD	Normal angle (°)
GPD	1	1.221	0.826	0.3490	11.53
GPD	2	0.990	0.696	0.2563	10.66
GPD	3	0.926	0.658	0.2324	10.41
Ours	1	1.009	0.677	0.3245	10.56
Ours	2	0.776	0.539	0.2289	9.95
Ours	3	0.737	0.519	0.2083	9.82

the previous pass's output. GPD retains nearest-patch aggregation; ours retains weighted overlap aggregation with decay 4. Thus, pass count is controlled within each method, whereas the cross-method comparison retains the original fusion choices. The three-pass scan values come from these continuous one-to-three-pass trajectories; Table 18 instead uses the archived 20-shape outputs. The five-shape scan is a sensitivity check, not a re-ranking of all published baselines or a substitute for multiple training seeds.

For both scanned checkpoints, all four mean errors decrease from one to three passes on these five shapes. The CD gain from the second to the third pass is smaller than that from the first to the second for either checkpoint. Our five-shape mean remains lower at each matched pass count, but this scan does not establish the effect of additional passes for ASDN, P2P-Bridge, or IterativePFN.

B.7 UGRH Gate Diagnostic

Because the synthetic PUNet corruption preserves point correspondence, the UGRH output can be compared directly with its training target u_i^* from Eq. 15. We use four deterministic paired patches per shape and compute correlations within each shape before averaging across the 20 shapes. The positive Pearson and Spearman correlations in Table 20 show that larger gates tend to occur where larger corrections are required. Their moderate magnitude, together with the difference between the mean gate and the target, also shows that the output is better interpreted as a learned relative control signal than as an absolutely calibrated estimate of displacement magnitude.

Figure 5 gives the corresponding quantile-bin curves. Each point first averages a gate quantile within a shape; curves then average shapes, and shaded bands are 95% shape-bootstrap intervals.

Condition	Pearson r	Spearman ρ	Mean gate	Target MAE
10K–1%	0.245 ± 0.079	0.290 ± 0.054	0.059	0.458 ± 0.027
10K–2%	0.294 ± 0.148	0.334 ± 0.067	0.063	0.454 ± 0.027
10K–2.5%	0.298 ± 0.156	0.340 ± 0.081	0.065	0.452 ± 0.027
50K–1%	0.324 ± 0.117	0.344 ± 0.032	0.061	0.457 ± 0.028
50K–2%	0.365 ± 0.136	0.390 ± 0.062	0.064	0.453 ± 0.027
50K–2.5%	0.279 ± 0.127	0.341 ± 0.074	0.066	0.450 ± 0.027

Table 20: UGRH gate diagnostics. Correlations are computed within each PUNet shape and reported as mean ± sample SD over 20 shapes. The target is the clipped pointwise corruption magnitude used to supervise UGRH.

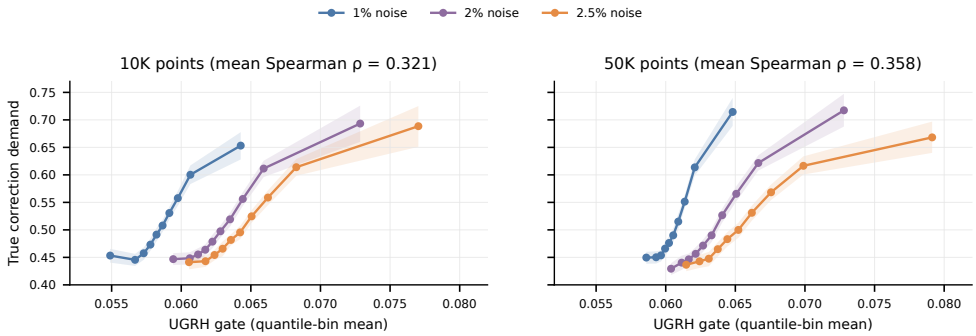

Figure 5: Binned UGRH gate against the known correction-demand target on paired PUNet patches. The axes span the observed bin means so that the within-condition trend is visible; the absolute scale mismatch is reported by the gate mean and target MAE in Table 20.

B.8 NSRM Gate by Geometry and Corruption Region

We hook the four trained NSRM mask predictors at 50K–2.5% and average their point- and channel-wise codebook weights within clean-reference geometry regions. Table 21 shows a stage-dependent controller, not one global noise score. At D4, high-corruption points receive more prior weight than the all-point mean (0.384 versus 0.368), whereas sharp-edge and thin-sheet points receive less (0.352 and 0.355). This ordering changes at shallower stages, and D2 is nearly uniform. The thin-sheet row covers only four eligible shapes and is consequently descriptive. These exposed gate values help characterize the trained checkpoint, but they do not replace a retrained intervention when assessing causal necessity.

Table 21: NSRM structural-prior weights on PUNet 50K at 2.5% noise. Each entry is the shape-balanced mean ± sample SD of the point- and channel-averaged blend gate. Larger values assign more weight to the decoder codebook prior. Geometry strata are defined from the clean reference; high corruption is the upper displacement tertile.

Region	D4	D3	D2	D1
All sampled points	0.368±0.027	0.429±0.009	0.663±0.010	0.351±0.004
Low κ	0.378±0.029	0.431±0.009	0.664±0.008	0.352±0.005
High κ	0.362±0.025	0.430±0.011	0.663±0.010	0.351±0.005
Sharp edge	0.352±0.020	0.433±0.010	0.663±0.014	0.354±0.009
Thin sheet	0.355±0.032	0.452±0.002	0.673±0.012	0.366±0.009
High corruption	0.384±0.029	0.416±0.005	0.663±0.009	0.346±0.005

Eligible shapes: All sampled points=20, Low κ=19, High κ=20, Sharp edge=11, Thin sheet=4, High corruption=20.

B.9 Clean/Noisy Code-Assignment Stability

We pass paired clean and noisy patches through the frozen EMA teacher using identical hierarchical FPS indices. Assignment confidence is the largest soft code probability, and entropy is divided by $\log 128$. Confidence and entropy remain similar between the two

views, but top-1 agreement decreases steadily as corruption grows and is lower at 50K. Thus a sharp assignment is not necessarily a stable assignment: clean supervision changes which code is selected even when the assignment distribution remains confident. This is the quantity transferred from the clean teacher during HGCPD training.

Condition	Clean conf.	Noisy conf.	Clean ent.	Noisy ent.	Top-1 agree.
10K–1%	0.763	0.770	0.136	0.132	0.640
10K–2%	0.759	0.758	0.139	0.139	0.481
10K–2.5%	0.754	0.751	0.141	0.143	0.411
50K–1%	0.688	0.725	0.181	0.159	0.440
50K–2%	0.683	0.713	0.185	0.166	0.331
50K–2.5%	0.684	0.704	0.185	0.173	0.289

Table 22: Frozen-teacher clean/noisy code-assignment diagnostics, averaged over 20 shapes. Entropy is normalized by $\log 128$; confidence is the maximum assignment probability. Clean and noisy views use paired patches and identical hierarchical FPS indices.

B.10 Clean-Codebook Utilization

Table 23 complements assignment confidence with prototype coverage. Each view contributes 800 bottleneck tokens per shape. It selects roughly 24–31% of the 128 prototypes per shape, with 21–25 effective codes; utilization does not collapse as noise rises. The checkpoint buffers provide a longer-horizon check: all 128 clean prototypes have positive cumulative usage and 122 were selected in the final recorded training batch. At the same time, paired top-1 agreement falls from 0.640 at 10K–1% to 0.289 at 50K–2.5%. The codebook is therefore populated, but the selected clean target becomes increasingly different from the noisy-view assignment under stronger corruption and denser sampling.

Table 23: Frozen clean-codebook usage under paired clean and noisy views. Each shape contributes 800 bottleneck tokens. Utilization is the fraction of 128 prototypes selected at least once; effective codes are the perplexity of the mean soft assignment. Entropy is normalized by $\log 128$. Values are shape-balanced means over 20 PUNet shapes.

Condition	Util. (%)		Effective codes		Norm. entropy		Top-1 agree.
	Clean	Noisy	Clean	Noisy	Clean	Noisy	
10K–1%	29.8	29.6	23.0	23.5	0.136	0.132	0.640
10K–2%	29.9	30.7	23.6	24.6	0.139	0.139	0.481
10K–2.5%	31.1	31.0	24.4	24.8	0.141	0.143	0.411
50K–1%	23.8	25.9	21.1	23.8	0.181	0.159	0.440
50K–2%	24.6	26.8	21.6	24.1	0.185	0.166	0.331
50K–2.5%	25.2	27.9	21.9	25.2	0.185	0.173	0.289

C Real-Scan Evaluation on Kinect v2

Protocol. We evaluate ASDN (Guo et al., 2025), GPD (Guo et al., 2026), P2P-Bridge (Vogel et al., 2024), IterativePFN (de Silva Edirimuni et al., 2023), and our complete model on the same 72 registered Kinect v2 scans. All methods receive the same noisy inputs and reference geometry, use exact filename pairing and the same unit-sphere normalization, and are scored with one common implementation of symmetric Chamfer distance (CD) and bidirectional point-to-mesh distance (P2M). We apply every method once externally and use seed 2025. ASDN, GPD, IterativePFN, and our model use 1,000-point patches with sixfold patch coverage. P2P-Bridge retains the 2,048-point patches, threefold coverage, and five internal bridge steps for which its released checkpoint was trained. These bridge steps form one native inference trajectory rather than five external reapplications.

	Kinect v2	
Method	$CD\downarrow$	$P2M\downarrow$
Noisy input	22.63 ± 6.98	13.39 ± 4.44
ASDN	20.11 ± 6.42	11.98 ± 4.15
GPD	20.42 ± 6.49	12.23 ± 4.23
P2P-Bridge	$\underline{20.00 \pm 6.46}$	$\mathbf{11.41 \pm 4.12}$
IterativePFN	$\mathbf{19.68 \pm 6.05}$	$\underline{11.69 \pm 3.99}$
Ours	20.33 ± 6.43	12.16 ± 4.19

Table 24: Quantitative evaluation on 72 Kinect v2 scans under the shared local protocol. Entries are mean ± sample SD across scans. CD and P2M are multiplied by 10^5. Lower is better. Bold and underlining denote the best and second-best denoised results, respectively.

Results. Our model reduces CD and P2M on Kinect v2 by 10.2% and 9.2% relative to the noisy input and slightly outperforms GPD on both metrics (Table 24). IterativePFN obtains the lowest CD, whereas P2P-Bridge obtains the lowest P2M. The gains are smaller than on synthetic corruptions: the model transfers without fine-tuning but remains below the best method for either metric.

Paired scan-level uncertainty. The aggregate means conceal how consistently each ordering holds across the 72 registered scans. Table 25 therefore reports paired scan bootstrap intervals and wins. Ours is lower than GPD on 48 scans for CD and 45 for P2M, with positive paired intervals. The intervals against ASDN, P2P-Bridge, and IterativePFN are negative, consistent with their lower aggregate errors under this protocol. The analysis supports a modest advantage over GPD on these scans, but not a claim of best real-scan performance.

Baseline	Metric	Baseline	Ours	Δ	95% paired CI	W/L/T
ASDN	CD	20.11	20.33	-0.223	[-0.285, -0.154]	10/62/0
	P2M	11.98	12.16	-0.187	[-0.228, -0.144]	6/66/0
GPD	CD	20.42	20.33	+0.088	[+0.037, +0.141]	48/24/0
	P2M	12.23	12.16	+0.063	[+0.017, +0.112]	45/27/0
P2P-Bridge	CD	20.00	20.33	-0.325	[-0.576, -0.067]	26/46/0
	P2M	11.41	12.16	-0.755	[-0.961, -0.554]	14/58/0
IterativePFN	CD	19.68	20.33	-0.647	[-0.786, -0.516]	6/66/0
	P2M	11.69	12.16	-0.471	[-0.581, -0.368]	7/65/0

Table 25: Paired scan-level comparison on Kinect v2. Δ is the baseline minus Ours, so positive values favour Ours. Intervals use 100,000 paired bootstrap resamples over 72 scans. W/L/T counts scan-wise wins, losses, and ties for Ours. CD and P2M use 10^{-5} units.

D Implementation and Reproducibility Details

This appendix records the implementation used by the reported complete model. It distinguishes the training objective from the CD and P2M metrics used for evaluation.

D.1 Evaluation Datasets and Corruptions

PUNet contains 40 training meshes and 20 official evaluation meshes. Each training shape is sampled at 10K and 50K points, yielding 80 clean clouds, and both densities are retained for testing. Isotropic Gaussian training noise is sampled online from 0.5% to 2.0% of the unit-sphere radius; the fixed test levels are 1%, 2%, and 2.5%. The 2.5% condition is outside the training range. Each reported PUNet aggregate averages the 20 evaluation shapes.

For cross-dataset testing, the ten PC-Net shapes are sampled at 50K points and corrupted by 1% and 2.5% isotropic Gaussian noise. The PUNet-trained checkpoint is applied directly, without fine-tuning; normalization, corruption, and metrics otherwise match the main evaluation. Each PC-Net entry averages the ten shapes.

The unseen-noise study uses PUNet 10K clouds. Anisotropic Gaussian noise has normalized axis ratios 1.2:1.0:0.8; zero-mean Laplace noise uses scale equal to the nominal level; uniform perturbations lie within a ball of that radius; and discrete noise moves 60% of points by the nominal distance along one signed coordinate axis. Table 3 uses the historical seed-42 sets. The paired reliability suite independently regenerates all four corruptions with seeds 42–46 for Ours and $\mathrm{GPD}_{\mathrm{rep}}$; its seed-42 inputs are not assumed to be byte-identical to the historical sets, so the two analyses are kept separate.

D.2 Normalization and Paired Patch Construction

For a clean training cloud $\mathbf{Y}$, define the normalization quantities:

$$\mathbf{c}_{\mathbf{Y}} = \frac{\min(\mathbf{Y}) + \max(\mathbf{Y})}{2}, \qquad r_{\mathbf{Y}} = \max_{\mathbf{y}\in\mathbf{Y}} \|\mathbf{y} - \mathbf{c}_{\mathbf{Y}}\|_2, \qquad \widetilde{\mathbf{Y}} = \frac{\mathbf{Y} - \mathbf{c}_{\mathbf{Y}}}{r_{\mathbf{Y}}}. \tag{21}$$

Here, the minimum and maximum are coordinate-wise. Gaussian noise is added after this normalization. Its standard deviation is drawn independently for each training item from $\mathcal{U}(0.005, 0.020)$. The clean and noisy clouds are then rotated together, with independent angles sampled uniformly from $[-180°, 180°]$ about the x, y, and z axes.

To form one paired training patch, we choose one noisy-cloud point uniformly without replacement from the set of candidate indices and take its 1,000 nearest noisy points. The same indices select the clean patch. Both patches are translated by the noisy seed, not by separate centroids. Thus point i in the two patches remains paired. The patch ratio is one. The loader contains 80,000 items per epoch: 1,000 draws per shape–resolution pair on average over 40 shapes and the 10K and 50K resolutions.

D.3 Exact GMEM and Mask Construction

GMEM computes pairwise Euclidean distances in float32 and takes $k = 16$ neighbors, including the query point. It uses the centroid and covariance in Eq. 2, divides the covariance by $k-1$, and clamps each eigenvalue returned by the symmetric eigensolver to at least 10^{-8}. With $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_2$, the implemented density term is exactly Eq. 4; in particular, the square root is applied to the Euclidean distance. The descriptor is ordered as follows:

$$[|n_x|, |n_y|, |n_z|, \kappa, \rho, \ell, \pi, s]. \tag{22}$$

The GMEM projector is Linear$(8, 32)$–BatchNorm–ReLU–Linear$(32, 16)$– BatchNorm–ReLU. The resulting 16-dimensional feature and the coordinates enter the 32-dimensional start block.

For a patch with n points, the masked branch uses:

$$M = \max(1, \min(n-1, \mathrm{round}(0.30n))), \qquad G = \max(1, \lceil M/16 \rceil). \tag{23}$$

We sample G point indices uniformly without replacement. For every point, we measure its Euclidean distance to the closest sampled center and mask the M smallest values. Hence a 1,000-point patch uses $M = 300$ masked points and $G = 19$ centers. The value 16 controls the number of centers; it is not a fixed cardinality assigned to each center. The learned token replaces the 32-dimensional feature after the start block, while all coordinates remain visible. At deeper levels the mask follows the FPS indices. The complete noisy path chooses all FPS indices, which are reused by the masked student and clean teacher.

The main restoration branch updates its BatchNorm running statistics. The masked branch uses current masked-batch statistics without updating those buffers. The EMA teacher likewise uses statistics from the current clean batch without changing its persistent BatchNorm buffers. Teacher parameters and floating-point buffers are updated after each training batch by Eq. 5. The epoch-dependent momentum is given by:

$$m_e = 0.9999 - (0.9999 - 0.996)\frac{\cos(\pi e/799) + 1}{2}. \tag{24}$$

Non-floating teacher buffers are copied from the student.

D.4 Clean and Decoder Codebooks

Training-only clean codebook. The clean codebook has 128 vectors in $\mathbb{R}^{64}$ and receives no gradient. At epoch 20, immediately before its loss becomes active, its state is reset. The first active clean-teacher batch initializes all centers from ℓ_2-normalized teacher features. The first center is the feature least aligned with the batch mean direction. Each subsequent center minimizes its maximum cosine similarity to both that mean direction and the centers already selected. This greedy rule is continued until 128 centers are obtained.

For later batches, a teacher feature is assigned to the center of maximum cosine similarity. If a_i is its hard assignment, the running count and sum for center k are updated as follows:

$$\begin{aligned} n_k &\leftarrow 0.99 n_k + 0.01 \sum_i \mathbb{1}[a_i = k], \\ \mathbf{s}_k &\leftarrow 0.99 \mathbf{s}_k + 0.01 \sum_i \mathbb{1}[a_i = k] \bar{\mathbf{z}}_i^{\mathrm{t}}, \\ \mathbf{c}_k &\leftarrow \frac{\mathbf{s}_k / n_k}{\|\mathbf{s}_k / n_k\|_2}. \end{aligned} \tag{25}$$

Assignment probabilities use temperatures 0.04 for the teacher and 0.10 for the student. The utilization u is the fraction of centers receiving at least one hard assignment in the current batch, and the multiplier in Eq. 7 is $h_{\mathcal{C}} = \mathrm{clip}((u - 0.50)/0.25, 0, 1)$. A center with no hard assignment for 100 consecutive active batches is considered dead. At most 16 dead centers are revived per batch, longest-inactive first, using clean-teacher features that have the lowest maximum cosine similarity to the live centers. The revived count and sum are initialized to one and to the replacement vector, respectively.

Inference-retained decoder codebooks. The four deep-to-shallow decoder codebooks contain $(512, 384, 256, 192)$ vectors with dimensions $(108, 72, 48, 32)$. They are initialized from normalized Gaussian vectors. For noisy skip feature $\mathbf{s}_i$, assignment uses normalized features and centers:

$$a_{ik} = \mathrm{softmax}_k\left(\cos(\mathbf{s}_i, \mathbf{e}_k)/0.1\right), \qquad \mathbf{q}_i = \sum_k a_{ik} \mathbf{e}_k. \tag{26}$$

Counts and feature sums are updated from these soft weights with EMA momentum 0.99 on every training forward pass. A center is then set to $\mathbf{s}_k/(n_k + 10^{-5})$; cosine assignment normalizes a copy of the stored center. Every 1,000 updates, a center whose last positive assignment was more than 5,000 updates earlier is replaced by the corresponding most-used center plus Gaussian noise of standard deviation 0.1, followed by normalization. The commitment coefficient is zero. The forward value is the soft structural prior with a straight-through gradient to the noisy skip feature. Unlike the clean codebook, these four codebooks and NSRM remain in the inference graph. In evaluation mode, codebook EMA updates and dead-center replacement are disabled; the stored centers are used without modifying their counts or vectors.

D.5 Hierarchical Targets and Exact Training Objective

At encoder levels 1 and 2, the feature dimensions are 48 and 72 and the normalized level weights are 4/7 and 3/7. Projected student and teacher features are 64-dimensional. Curvature targets are defined as:

$$t_\kappa = \frac{\log(1 + 100\kappa)}{\log 101}. \tag{27}$$

The curvature prediction passes through a sigmoid before a mean smooth ℓ_1 loss. At each level, density is standardized using the mean and population standard deviation over all clean-teacher features at that level; the standard deviation is clamped below at 10^{-4}. Density also uses mean smooth ℓ_1. Normals are ℓ_2-normalized and use the sign-invariant mean loss $1 - |\widehat{\mathbf{n}}^\mathsf{T}\mathbf{n}|$. The three geometry terms are averaged, and the codebook, feature, and geometry losses have weights 0.30, 0.40, and 0.30.

The reconstruction loss is distinct from the reported CD. For each batch item, let $d_i^{P\to C}$ and $d_j^{C\to P}$ be Euclidean nearest- neighbor distances after clamping their squared values below at 10^{-9}. Define the auxiliary quantities:

$$S^{P\to C} = \sum_i \left[\exp(-0.5 d_i^{P\to C}) + 10^{-7}\right], \quad \phi(d, S) = -\log \frac{\exp(-0.5d)}{S^{10^{-7}}}. \tag{28}$$

For batch size B, the reconstruction loss is defined as:

$$\mathcal{L}_{\text{rec}} = \frac{1}{2B} \sum_{b=1}^{B} \left[\sum_i \phi(d_{b,i}^{P\to C}, S_b^{P\to C}) + \sum_j \phi(d_{b,j}^{C\to P}, S_b^{C\to P}) \right]. \tag{29}$$

Eqs. 28– 29 reproduce the submitted implementation, including both occurrences of 10^{-7}.

For HGCPD, let $\mathbf{g}_r$ and $\mathbf{g}_d$ denote gradients of $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{HGCPD}}$ with respect to the shared GMEM and noisy-student encoder parameters. For surface guidance, $\mathbf{h}_r$ and $\mathbf{h}_s$ denote gradients of $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{surf}}$ with respect to the predicted coordinates. Each auxiliary gradient is projected with Eq. 17. The distillation schedule used in Eq. 18 is defined as:

$$s_e = \begin{cases} 0, & e < 20, \\ \min((e-19)/20, 1), & 20 \le e < 80, \\ 1 - 0.75\min((e-80)/40, 1), & e \ge 80. \end{cases} \tag{30}$$

Thus $s_{20} = 0.05$, it reaches one at epoch 39, remains one through epoch 80, and reaches 0.25 at epoch 120. The loss-value weights are detached. To realize the projected gradients exactly, the implementation adds a zero-valued term whose derivative is $-\lambda\alpha\mathbf{g}_r$, where $\alpha = \min(\mathbf{g}_a^\mathsf{T}\mathbf{g}_r/\|\mathbf{g}_r\|_2^2, 0)$, separately for HGCPD and surface guidance.

The per-batch execution order is: (1) compute reconstruction, gate, surface, and HGCPD losses in one forward pass; (2) project the surface gradient against the reconstruction gradient in coordinate space and compute λ_s; (3) project the HGCPD gradient against the reconstruction gradient on shared encoder parameters and compute λ_d; (4) assemble Eq. 19 with detached weights and the two zero-valued gradient corrections; and (5) run one ordinary backward pass and one Adam step. The gate loss is weighted by 0.01 and is not projected. Decoder codebook commitment is also not projected and has coefficient zero. The teacher EMA is updated after the optimizer step.

D.6 Training Hyperparameters and Checkpoint Rule

The plateau scheduler was driven by one-pass CD on the 20 official PUNet evaluation geometries at 10K resolution, fixed $\sigma = 0.015$, and deterministic validation seed 12345. Validation used 50 FPS patches per 10K cloud. These same 20 geometries are used in the reported PUNet tables, so those tables do not constitute evaluation on geometry that was untouched by training control. The PC-Net geometries were not used by either the optimizer or scheduler.

The checkpoint callback saved every five-epoch snapshot. All reported results for the complete HGCPD+NASR model use the epoch-299 file, whose validation loss is 2.2323×10^{-4}. The reporting rule selected the snapshot after 300 completed epochs. We applied this rule unchanged across point counts, noise levels, corruption distributions, datasets, and metrics. The run resumed from its epoch-124 checkpoint with model, optimizer, scheduler, and global-step states restored. The lowest archived validation loss through epoch 309 was 2.2313×10^{-4} at epoch 274, so the reported snapshot was not selected by minimum validation loss. Training used one seeded run without independent retraining replicates.

The HGCPD-only row in Table 4 uses the separately trained pgd-epoch309-val_loss0.00022087.ckpt checkpoint with seed 2025. Clean-codebook and hierarchical distillation are enabled, whereas NSRM and UGRH are disabled. This checkpoint is used unchanged across the six reported PUNet conditions with weighted patch fusion, two restoration passes at 1% and 2% noise, and three passes at 2.5% noise. Each patch executes one student-network forward per restoration pass. There is no additional internal sampling loop.

The NSRM-without-UGRH row uses the separately trained pgd-epoch229-val_loss0.00022466.ckpt checkpoint with seed 2025. GMEM, the decoder codebooks, and NSRM are enabled, while UGRH and all HGCPD branches and losses are absent. The checkpoint is evaluated under the same six PUNet conditions and restoration-pass schedule as the other component variants.

The UGRH-without-NSRM row uses a separate epoch-239 checkpoint trained with seed 2025. GMEM, decoder codebooks, and UGRH are enabled; NSRM and HGCPD branches and losses are absent. Both single-component NASR variants use nearest-patch fusion, two restoration passes at 1% and 2% noise, and three at 2.5% noise on the 20 PUNet test shapes.

Item	Reported-model setting
Training data	40 PUNet meshes; 10K and 50K samples
Training noise	Isotropic Gaussian; $\sigma \sim \mathcal{U}(0.005, 0.020)$
Rotation	Independent $x/y/z$ angles from $[-180^\circ, 180^\circ]$
Patch sampling	1,000 noisy KNN points; paired clean indices; noisy-seed centering
Epoch size / batch	80,000 patches; training and validation batch sizes 20
Loader	8 workers; shuffled; pinned memory
Optimizer	Adam; lr 5×10^{-4}; $\beta = (0.9, 0.999)$; $\epsilon = 10^{-8}$; weight decay 0; AMSGrad off
LR scheduler	ReduceLROnPlateau, mode min; factor 0.5; patience 5 validation events; relative threshold 10^{-4}; cooldown 0; scheduler epsilon 10^{-8}; minimum lr 10^{-9}
Validation / saving	Every 5 epochs; all snapshots retained (save_top_k $= -1$)
Configured duration	800 epochs; no step limit
Reported snapshot	Epoch 299; 300 completed epochs; 1,200,000 optimizer steps; lr 3.125×10^{-5}
Precision / hardware	16-bit mixed precision; one NVIDIA GeForce RTX 3090
Randomness	Python, NumPy, and PyTorch seed 2025; validation seed 12345; deterministic trainer disabled
Parameters	1.402M full feature network; 1.044M active inference graph

Table 26: Data, optimization, and runtime settings for the reported complete model.

Item	Reported-model setting
Encoder point counts	1,000 / 800 / 600 / 400 / 200
Encoder feature widths	32 / 48 / 72 / 108 / 162
Decoder widths	108 / 72 / 48 / 32
Local neighborhoods	GMEM 16; feature aggregation 16; cross-attention 16; interpolation 8
Feature mask	Ratio 0.30; nominal group size 16; 19 centers and 300 points for $n = 1{,}000$
Clean codebook	128×64; EMA 0.99; teacher/student temperatures 0.04/0.10
Codebook health	utilization floor 0.50; full-strength target 0.75; confidence exponent 2
Dead clean centers	100 active batches; at most 16 revivals per batch
Hierarchical targets	levels 1 and 2; projection width 64; level weights $4/7, 3/7$
HGCPD mixture	codebook / feature / geometry = 0.30 / 0.40 / 0.30
Teacher EMA	0.996 to 0.9999 cosine schedule over configured 800 epochs
Distillation schedule	warm-up 20; ramp 20; decay starts 80 for 40 epochs; terminal factor 0.25
HGCPD balancing	target gradient ratio 0.005 to 0.015; maximum $\lambda_d = 0.40$; conflict projection on
Surface guidance	Charbonnier $\epsilon_s = 10^{-4}$; target ratio 0.01; maximum $\lambda_s = 80$; conflict projection on
UGRH gate loss	mean smooth ℓ_1; weight 0.01
Decoder codebooks	sizes 512 / 384 / 256 / 192; temperature 0.1; EMA 0.99; commitment 0

Table 27: Architecture and auxiliary-training hyperparameters.

D.7 Patch-Based Inference

Each noisy test cloud is first normalized using its own coordinate-wise bounding-box midpoint and maximum radius. FPS with random_start=False chooses $6N/1{,}000$ seeds. The 1,000 nearest input points form each patch and are centered at its seed. For patch m, the squared seed distances returned by KNN are divided by the farthest squared distance in that patch, giving $\widetilde{d}_{mi}$. All overlapping predictions are fused with Eq. 20; uncovered points, if any, retain their input coordinates. Processing chunks contain $\lfloor N/(10 \times 1{,}000) \rfloor$ patches and affect memory only. The fused cloud is denormalized to the input coordinate system and written with eight digits after the decimal point.

For the matched Gaussian protocol, the same network and patch procedure are applied twice at 1% and 2% noise and three times at 2.5%. Non-Gaussian inputs are processed twice. PC-Net uses the same two-pass/three-pass rule at 1%/2.5%. Each patch invokes the student network once per external pass. The legacy sampling_timesteps command-line field is not consumed by the model forward and adds no internal updates. For the local PC-Net comparisons, ASDN uses the released ASDN_0.ckpt, and IterativePFN uses denoisenet-ep-99.ckpt. Both use 1,000-point patches, sixfold patch coverage, overlap stitching, seed 2025, the same corrupted inputs and metric implementation, and the same two-pass/three-pass rule, without fine-tuning. No teacher, masked branch, clean codebook, clean feature, or clean geometry is available during inference.

D.8 Exact Evaluation Metrics

For CD, the clean reference point set $\mathbf{Y}$ supplies $\mathbf{c}_\mathbf{Y}$ and $r_\mathbf{Y}$ from Eq. 21; the same transform is applied to the prediction $\mathbf{P}$. The reported value is given by:

$$\mathrm{CD}(\mathbf{P}, \mathbf{Y}) = 10^4 \left[\frac{1}{|\mathbf{P}|} \sum_{\mathbf{p} \in \mathbf{P}} \min_{\mathbf{y} \in \mathbf{Y}} \|\widetilde{\mathbf{p}} - \widetilde{\mathbf{y}}\|_2^2 + \frac{1}{|\mathbf{Y}|} \sum_{\mathbf{y} \in \mathbf{Y}} \min_{\mathbf{p} \in \mathbf{P}} \|\widetilde{\mathbf{y}} - \widetilde{\mathbf{p}}\|_2^2 \right]. \tag{31}$$

This is PyTorch3D Chamfer distance with squared Euclidean norm, mean point reduction, both directions, and their sum.

For P2M, mesh vertices rather than the sampled clean point set supply the bounding-box midpoint and maximum-radius scale. The prediction is transformed with the same mesh-derived values. PyTorch3D's bidirectional point_mesh_face_distance returns:

$$\mathrm{P2M}(\mathbf{P}, \mathcal{T}) = 10^4 \left[\frac{1}{|\mathbf{P}|} \sum_{\mathbf{p} \in \mathbf{P}} d^2(\mathbf{p}, \mathcal{T}) + \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} d^2(f, \mathbf{P}) \right]. \tag{32}$$

Here, $\mathcal{F}$ is the mesh face set and both terms use the library's exact point–triangle kernels with the default minimum triangle area 0.005; no surface points are sampled for this calculation. Each table entry is the arithmetic mean of the per-shape metric over 20 PUNet or ten PC-Net shapes.

Evaluation used Python 3.10.19, PyTorch 2.3.1 with CUDA 11.8, PyTorch3D 0.7.8, PyTorch Lightning 2.6.1, torch-cluster 1.6.3, point-cloud-utils 0.31.0, NumPy 2.2.5, SciPy 1.15.3, and pandas 2.3.3. All shape-level statistical tables use this same metric implementation.

E Expanded Qualitative Results

Figures 6–9 extend the point-wise P2M visualization in Figure 3 to sixteen additional PUNet shapes at 50K points and 2.5% isotropic Gaussian noise. The method order, color mapping, viewing conditions, and borderless transparent layout are unchanged. We include every unused shape for which all seven displayed renderings are available; Pig is omitted because its P2P-Bridge rendering is unavailable. The grouping follows the official shape order and was fixed without reference to the relative performance of our method.

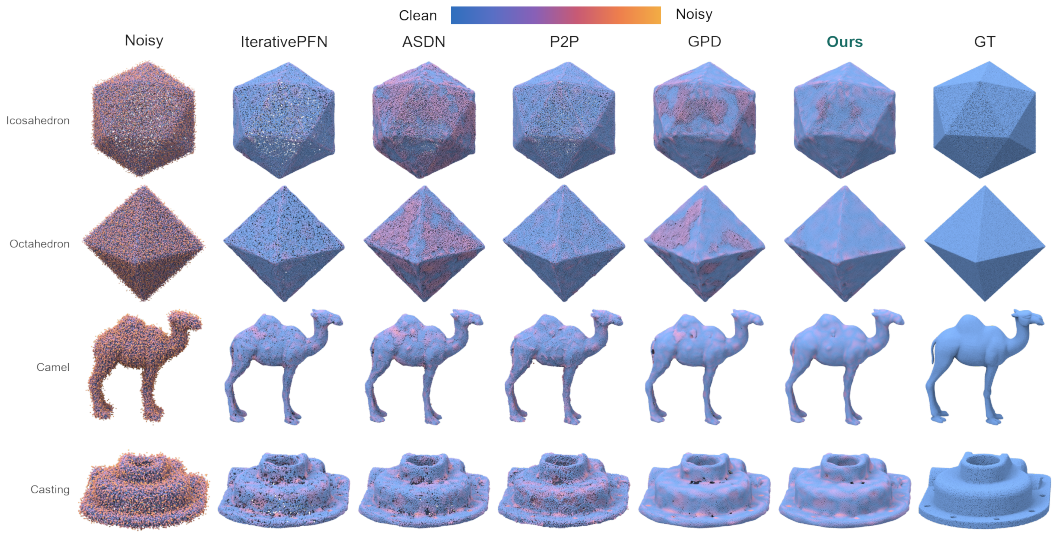

Figure 6: Expanded point-wise P2M distance visualizations (group 1): Icosahedron, Octahedron, Camel, and Casting. Columns and colors follow Figure 3.

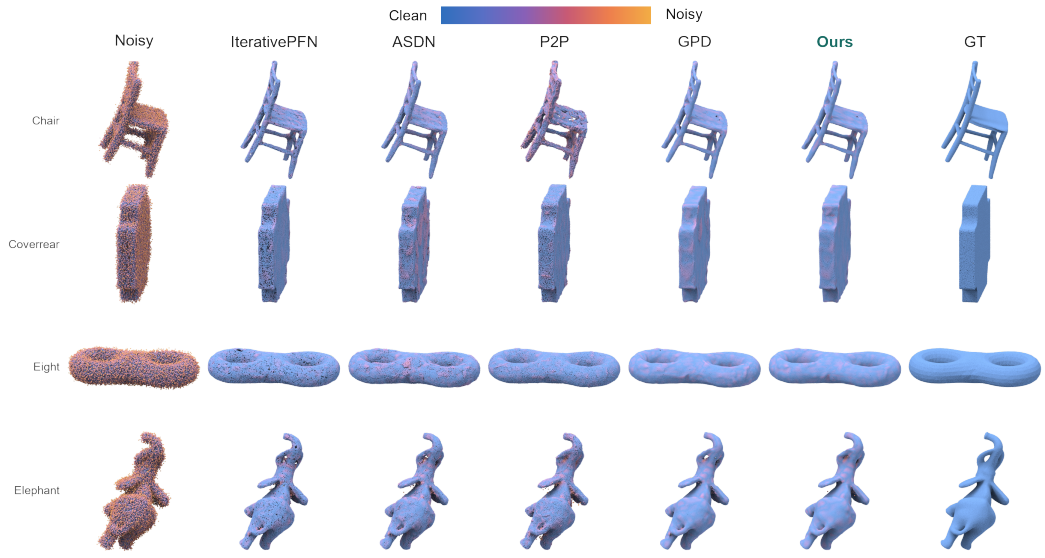

Figure 7: Expanded point-wise P2M distance visualizations (group 2): Chair, Coverrear, Eight, and Elephant. Columns and colors follow Figure 3.

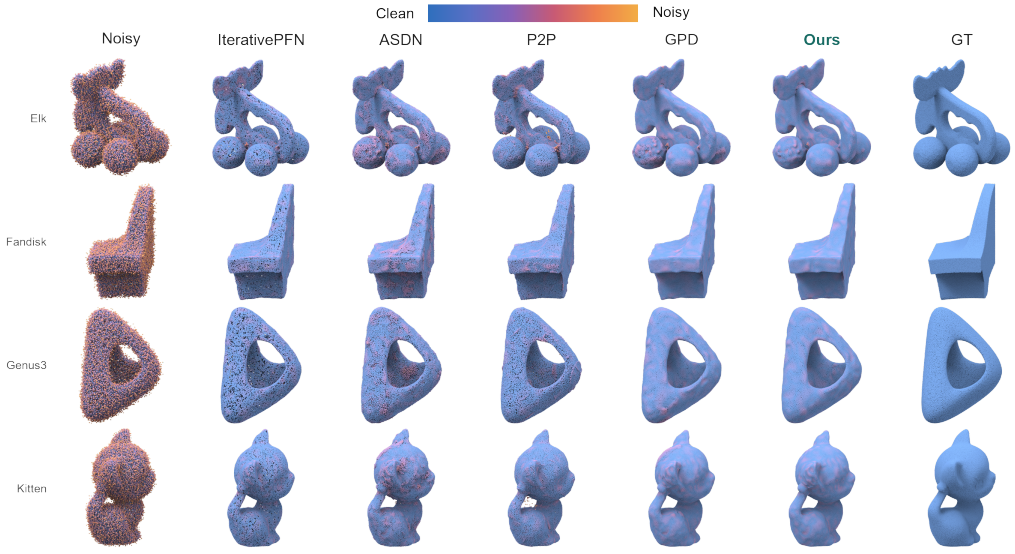

Figure 8: Expanded point-wise P2M distance visualizations (group 3): Elk, Fandisk, Genus3, and Kitten. Columns and colors follow Figure 3.

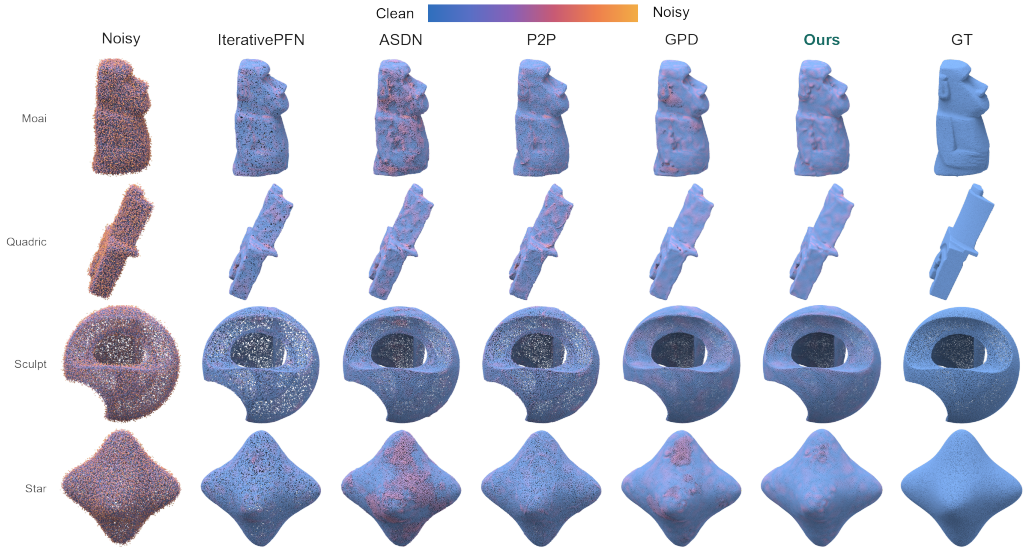

Figure 9: Expanded point-wise P2M distance visualizations (group 4): Moai, Quadric, Sculpt, and Star. Columns and colors follow Figure 3.