Denoising Point Clouds in Latent Space via Graph Convolution and Invertible Neural Network

Supplementary Material

1. Outline

The supplementary material includes:

- Evaluation metrics (Sec. 2);
- Network configuration details (Sec. 3);
- Additional comparisons of the generalization ability (Sec. 4);
- Comparison with traditional methods(Sec. 5);
- Additional ablation studies (Sec. 6);
- Runtime and network size for learning based methods (Sec. 7).

2. Evaluation Metrics

CD and P2M. Following Score [9] and IterativePFN [2], we use Chamfer distance (CD) and Point-to-mesh(P2M) distance as our primary evaluation metrics to quantify the performance of point cloud denoising, which have the following standard definitions:

$$\mathbf{CD}(\widehat{X},X)=\frac{1}{|\widehat{X}|}\sum_{\hat{x}\in\hat{X}}\min_{x\in X}\|x-\hat{x}\|+\frac{1}{|X|}\sum_{x\in X}\min_{\hat{x}\in\hat{X}}\|\hat{x}-x\| \tag{1}$$

$$\begin{aligned}\mathbf{P2M}(\widehat{X},M)=&\frac{1}{|\widehat{X}|}\sum_{\hat{x}\in\widehat{X}}\min_{f\in M}d(f,\hat{x})\\&+\frac{1}{|M|}\sum_{f\in M}\min_{\hat{x}\in\widehat{X}}d(\hat{x},f)\end{aligned} \tag{2}$$

where $\widehat{X}$ and X are denoised point cloud and ground-truth, M is the initial mesh corresponding to ground-truth X, f is the triangular face in M, $|X|$ and $|M|$ are the number of points and triangular faces, $\|\cdot\|$ is the L_2 norm.

Uniformity. The uniformity metric was proposed in PU-GAN [7], which uses the FPS and ball query algorithm with radius r_d to sample M seed points and collect a point set at each seed. For each point set $S_j (j = 1 \dots M)$, it defines the expected number of points n_j based on the percentage of the area, and calculates the loss between n_j and the actual number of points $|S_j|$:

$$U_n(S_j)=(|S_j|-n_j)^2/n_j \tag{3}$$

which evaluates the global uniformity. Further, it defines the expected point-to-neighbor distance $\hat{d}$, and calculates the loss between $\hat{d}$ and the actual point-to-neighbor distance $d_{j,k} (k = 1 \dots |S_j|)$ of each point in S_j:

$$U_d(S_j)=\sum_{k=1}^{|S_j|}(d_{j,k}-\hat{d})^2/\hat{d} \tag{4}$$

which evaluates the local uniformity. Combining Eq. (3) and Eq. (4), the definition of uniformity for a point cloud $\widehat{X}$ can be denoted as:

$$U(\widehat{X})=\sum_{j=1}^{M}U_n(S_j)\cdot U_d(S_j) \tag{5}$$

In practice, we collected point set S_j with radius $r_d = \sqrt{p}$ for each $p \in \{0.4\%, 0.6\%, 0.8\%, 1.0\%, 1.2\%\}$. The detailed definitions of n_j and $\hat{d}$ can be found in section 3.5.2 in PU-GAN [7].

Discussion. In the qualitative results of the main paper, we mentioned two issues of existing SOTA deep learning methods including IterativePFN [2] and Score [9]: 1)the denoised points frequently gather together; 2)there are lots of hollow areas. These two issues affects the quality of point clouds, and lead to different biases towards the results of CD, P2M and uniformity metric. To be specific, when calculating the CD in Eq. (1), for ground-truth X located in the hollow area of denoised point cloud $\widehat{X}$, the distance to the nearest point in $\widehat{X}$ will be large. Further, when calculating the local uniformity in Eq. (4), for the denoised points gathered together, the point-to-neighbor distance $d_{j,k}$ will be much less than the expected distance $\hat{d}$. Moreover, the P2M(Eq. (2)) metric with sufficiently large triangular faces is rarely affected by these issues, nevertheless, we typically aim for point clouds with uniform distributions.

3. Network Configuration Details

In this section, we first introduce the design of multiple iteration-modules for our heavy-version denoising framework, which utilizes the iterative denoising technique proposed in IterativePFN [4]. Then, we present the configuration details of our 2-stage multi-level graph convolution module and invertible neural network.

Iterative denoising of our heavy-version framework. Inspired by IterativePFN [4], we construct heavy-version framework by stacking three identical denoising modules to perform iterative denoising internally, as shown in Fig. 1(right), the output of each denoising module is the input of its next module, and we use the reconstruction loss:

Earth Mover's Distance(EMD) to model a training objective aimed at gradually reducing point cloud noise, which can be denoted as:

$$\mathcal{L}^{(i)} = \mathbf{EMD}^{(i)}(\widehat{X}^{(i)}, \widetilde{X}^{(i)}) \tag{6}$$

where $\widehat{X}^{(i)}$ is the output of i-th denoising module, $\widetilde{X}^{(i)} = X + \alpha_i \varepsilon$ is the i-th adaptive ground truth, where $\varepsilon \sim \mathcal{N}(0, I)$ and $\alpha_1 > \alpha_2 > \alpha_3$, we set $\alpha_3 = 0$, which means that $\widetilde{X}^{(3)}$ is exactly the ground-truth X. Finally, we sum adaptive EMD loss across iterations to obtain the final loss:

$$\mathcal{L}_{\mathbf{EMD}} = \sum_{i=1}^{3} \mathbf{EMD}^{(i)}(\widehat{X}^{(i)}, \widetilde{X}^{(i)}) \tag{7}$$

MLGC configuration. The MLGC module of our light-version framework is constructed by one EdgeConv layer for initial feature generation and 12 densely connected EdgeConv layers, as mentioned in the main paper, for l-th EdgeConv layer, the feature of i-th vertex is updated as: $\mathbf{f}_i^{(l)} = \sum_{j:(i,j)\in\mathcal{E}} \text{MLP}(\mathbf{h}_i^{(l-1)} \| \mathbf{h}_i^{(l-1)} - \mathbf{h}_j^{(l-1)})$, and the process of feature dimension adaption is: $C^{(l)} = \text{MLP}(H^{(l)})$, where $H^{(l)} = \{\mathbf{h}_i^{(l)}\}_{i=1}^{N}$ is the set of densely connected features. Thus, we describe the setting format of MLP in each EdgeConv layers and feature dimension adaption layers as: [(input channel, hidden channel, output channel)], and the configuration details of MLGC module in our light-version framework are presented in Tab. 1, we divide the 12 EdgeConv layers into two stages, and except for the last EdgeConv layer of each stage, we concatenate the input and output feature of EdgeConv to provide the densely connected features for next EdgeConv layer and the corresponding feature dimension adaption layer. Moreover, the configuration details of each MLGC module in our heavy-version framework are presented in Tab. 2, which contains 10 densely connected EdgeConv layers. For each vertex, edges are formed with its 32 nearest neighbors

Invertible neural network configuration. Our light-version framework has 12 invertible transformation layers: F_{θ_l}, and each iteration module of our heavy-version framework has 10 invertible transformation layers. Further, one invertible layer consists of 2 blocks, and we use activation normalization(Actnorm) [6] before every invertible block. Thus, with geometric feature $C^{(l)}$ integrated, the transformation process of l-th invertible layer F_{θ_l} can be denoted as:

$$X^{(l)} + C^{(l)} \rightarrow X_c^{(l)}$$
$$X_c^{(l)} \rightarrow [\text{Actnorm} \rightarrow \text{invblock}] \times 2 \rightarrow X^{(l+1)}$$

where actnorm is a form of data dependent initialization, which helps improve the training stability and performance, similar to batch normalization. We present the detailed formulations in Tab. 3, each invertible block is parameterized by a 1-Lipschitz block G, which is recursively constructed by two 1-Lipschitz transformations.

4. Further Comparison of the Generalization Ability

In this section, we provide further comparison of the generalization ability with SOTA methods in two aspects. 1)Denoising results on PUNet dataset with different noise patterns. 2)Quantitative results on Kinect v1 datasets of [14] consisting of 71 real-world scans and more visual results on RueMadame dataset.

4.1. Results on the PUNet dataset with Different Noise Patterns

For the comparison of generalization ability on unseen noise patterns, we further evaluate our method on four noise types with the noise scale set to 1%, 2% and 2.5% of the bounding sphere's radius: 1)Discrete noise, results are shown in Tab. 9 and Fig. 2. Both versions of our framework outperform SOTA deep learning methods on 10K points and are comparable to the SOTA results on 50K points. 2)Laplace noise, results are shown in Tab. 10 and Fig. 3. This noise pattern causes a relatively large disturbance, which is more challenging(see CD and P2M metric results for the noisy point clouds). Nonetheless, both versions of our framework consistently outperform SOTA methods on the Chamfer distance(CD) metric across all resolutions and noise scales, and also exhibit superior results on the Point to Mesh (P2M) distance metric, particularly under sparse point cloud conditions. 3)Non-isotropic Gaussian noise, results are shown in Tab. 11 and Fig. 4. In most cases, our method outperforms other SOTA methods. 4)Uniform distribution of noise, results are shown in Tab. 12 and Fig. 5. Similarly, our denoising results stay ahead of the SOTA results, especially in terms of Chamfer distance(CD) metric. Moreover, the specific implementation details of these noise patterns can be found in supplementary A.3 in [2].

4.2. More Comparisons on real world scans

To further evaluate the denoising performance on real world point clouds, we provide quantitative comparisons Kinect v1 dataset in Tab. 7, our strong denoising performance demonstrates the effectiveness of our method in real-world scenarios. Additionally, we present visual results on two additional scenes of the RueMadame datasets. As shown in Fig. 6, our method can effectively recognize and restore the intrinsic shape of the challenging chaotic noise point regions, such as the door and windowsill. Besides, by effectively eliminating the linear trajectories of point clouds caused by laser scanning, the denoised real world point clouds of our method are relatively uniform and smooth, which is rarely achieved by other methods.

5. Comparison with Traditional Methods

In addition to deep learning based methods, in Tab. 13, we present the denoising results for three traditional methods, including Bilateral filter [3] which depends on normal estimation technique, Jet fitting mechanism [1] which fits n-order polynomial surfaces and projects noisy points onto the surfaces, and WLOP regularization mechanism [5] which performs a resampling process for noisy point clouds. Depending on the manual tuning parameters, these traditional methods can generally perform well in circumstance of 50K resolution, while for point clouds with resolution of 10K, WLOP cannot achieve the positive denoising effect, and other two methods have poor results as well, which indicates that traditional methods have strong denoising bias. Generally, compared with deep learning methods, traditional methods struggle to achieve excellent denoising effects.

stage	input	EdgeConv	feature adaption
-	3	(6, 16)	-
1	16	[(32, 64, 32)]	[(48, 64, 3+D_a)]
	48	[(96, 64, 32)]	[(80, 64, 3+D_a)]
	80	[(160, 64, 32)]	[(112, 64, 3+D_a)]
	112	[(224, 64, 32)]	[(144, 64, 3+D_a)]
	144	[(288, 64, 32)]	[(176, 64, 3+D_a)]
	176	[(352, 64, 32)]	[(208, 64, 3+D_a)]
	208	[(416, 64, 32)]	[(240, 64, 3+D_a)]
	240	[(480, 96, 96)]	[(96, 64, 3+D_a)]
2	96	[(192, 64, 24)]	[(120, 64, 3+D_a)]
	120	[(240, 64, 24)]	[(144, 64, 3+D_a)]
	144	[(288, 64, 24)]	[(168, 64, 3+D_a)]
	168	[(336, 96, 96)]	[(96, 64, 3+D_a)]

Table 1. Network configuration of 2-stage multi-level graph convolution module in light-version framework. ”input” indicates the dimension of input features for EdgeConv layer, $D_a = 48$ is the augmented dimension for invertible neural network.

6. Further Ablation Studies

Ablation study on noise channels in the latent space. As mentioned in the main paper, our invertible denoising framework can bijectively map a high-dimensional prior latent space with $D(\widetilde{z}) = 3 + D_a$, where noise components and intrinsic clean points are separated into different channels: $\widetilde{z} = [z_c, z_n]$. To obtain a noise-free latent code, we set $z_n = 0$. Thus, to investigate the optimal noise channel division for disentanglement, we set the noise channel z_n to 1, 4, 12, 24, and 48, respectively, as shown in Tab. 4. The division of various noise channels z_n for our light-version framework can achieve similar denoising results, while the division of $z_n = 24$ has the best generalization ability for unseen noise scale of 2.5%. Therefore, we set $z_n = 24$ as the default value, approximately half of the latent space dimension 51 (augmented dimension $D_a = 48$). In general, when there are sufficient augmented latent dimensions, dividing noise channels can effectively achieve noise disentanglement.

Ablation study on the MLGC module. In the main paper, we respectively remove the whole MLGC module and stage-2 of it to demonstrate the importance of its local struc-

stage	input	EdgeConv	feature adaption
-	3	(6, 16)	-
1	16	[(32, 64, 32)]	[(48, 64, 3+D_a)]
	48	[(96, 64, 32)]	[(80, 64, 3+D_a)]
	80	[(160, 64, 32)]	[(112, 64, 3+D_a)]
	112	[(224, 64, 32)]	[(144, 64, 3+D_a)]
	144	[(288, 64, 32)]	[(176, 64, 3+D_a)]
	176	[(352, 64, 32)]	[(208, 64, 3+D_a)]
2	96	[(192, 64, 24)]	[(120, 64, 3+D_a)]
	120	[(240, 64, 24)]	[(144, 64, 3+D_a)]
	144	[(288, 64, 24)]	[(168, 64, 3+D_a)]
	168	[(336, 96, 96)]	[(96, 64, 3+D_a)]

Table 2. Network configuration of 2-stage multi-level graph convolution module in each iteration modules of heavy-version framework. where $D_a = 32$.

	Forward	Inverse
$C^{(l)}$ integration	$X_c^{(l)} = X^{(l)} + C^{(l)}$	$X^{(l)} = X_c^{(l)} - C^{(l)}$
Actnorm	$y = (x - u)/exp(\alpha)$	$x = y * exp(\alpha) + u$
invertible block	$y = \left(\frac{\mathrm{Id}+G}{2}\right)^{-1}(x) - x$	$x = \left(\frac{\mathrm{Id}-G}{2}\right)^{-1}(y) - y$
1-Lip G	$y = f_1(f_0(x)), f_i(x) = \varphi(\bar{W}_i x + b_i)$	

Table 3. Summarization of the components in invertible neural network. Where u and α are trainable parameters, φ denotes the LipSwish function, and $\bar{W}_i x + b_i$ is a linear mapping which satisfies the spectral norm $\|\bar{W}_i\|_2 < 1$.

z_n	**10K points**					
	1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M
1	18.29	4.97	25.59	8.27	28.65	10.50
4	18.28	4.98	25.72	8.30	28.77	10.61
12	18.25	4.95	25.60	8.19	28.61	10.38
24	18.25	4.96	25.67	8.24	**28.49**	**10.34**
48	**18.21**	**4.93**	**25.58**	**8.19**	28.81	10.54

Table 4. Ablation study for different noise channels: z_n in latent space. CD and P2M distances are multiplied by 10^5.

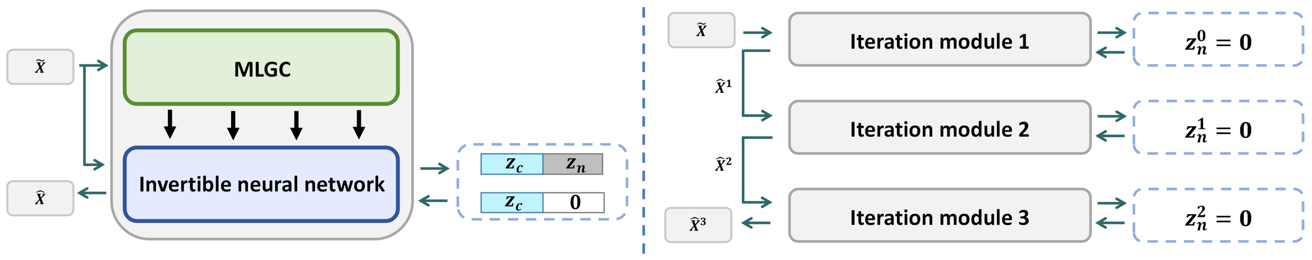

Figure 1. **Left:** architecture of our light-version framework with one denoising iteration. **Right:** architecture of our heavy-version framework with three denoising iterations.

Ablation	10K points					
	1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M
w/o DC	18.61	5.06	26.45	8.48	29.62	10.86
DG	18.72	5.01	26.76	8.41	30.17	10.86
ours(light)	**18.25**	**4.96**	**25.67**	**8.24**	**28.49**	**10.34**

Table 5. Ablation study for dense connection(DC) and dynamic graph(DG) construction. CD and P2M distances are multiplied by 10^5.

Ablation	10K points					
	1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M
PDflow	21.26	6.74	32.46	13.24	36.27	17.02
INN	25.93	10.36	38.85	16.45	50.11	24.56
PDflow+MLGC	22.30	6.91	33.05	12.84	38.86	17.32
INN+MLGC	**18.25**	**4.96**	**25.67**	**8.24**	**28.49**	**10.34**

Table 6. Ablation study for invertible mapping choice(10K points).

Method	Kinect v1	
	CD	P2M
Noisy	14.49	9.32
PCN [12]	13.73	8.75
GPDNet [11]	14.83	8.69
DMRDenoise [8]	22.78	12.89
PDFlow [10]	13.34	8.69
ScoreDenoise [9]	13.22	8.18
Pointfilter	13.77	**7.91**
IterativePFN	13.2	8.43
Ours	**13.07**	8.57

Table 7. Results on the Kinect v1 dataset.

Methods	# params	Time
PCN [12]	27.9M	106.66s
DMRDenoise [8]	225K	**2.99s**
PDflow [10]	470K	12.48s
Score [9]	**178K**	3.03s
Pointfilter [16]	1361K	60.43s
IterativePFN [4]	3195K	19.05s
Ours(heavy)	1443K	11.56s
Ours(light)	679K	7.75s

Table 8. Comparison of parameter count and runtime across various learning-based methods for denoising a noisy point cloud containing 50K points with 2% Gaussian noise.

ture features. To further investigate the rationality of MLGC module design, we additionally conduct an ablation study in two cases in Tab. 5: 1)Removing the dense connections between each EdgeConv layers. We only maintain shortcut connections to aggregate the outputs of all EdgeConv layers at the end of each stage to extract multi-level features, which is similar to the configuration of DGCNN [15] and IterativePFN [2], and the results in Tab. 5 demonstrates the positive denoising effect of contextual semantic information brought by repeated dense connections. 2)Dynamically constructing graph in the feature space of each EdgeConv layer. Compared with our default light-version framework with static graph constructed on raw point cloud, the results also deteriorated, which indicates that for our MLGC module with dense connections, it is not necessary to capture semantic k-nearest neighboring structure of the feature space, which increases the burden of MLGC module capturing local structure and affects the inference speed.

Ablation study on invertible mapping choice. To clarify the motivation of the choice of our invertible neural network(INN) and demonstrate its contribution, we provide an explicit comparison between the choice of PDflow and our invertible neural network in Tab. 6, where we trained PDflow+MLGC using our filtering alternative. Our INN, without MLGC, underperformed compared to PDflow, mainly because INN lacks the capability to aggregate neighboring features, resulting in a poor understanding of local areas. However, the integration of MLGC significantly boosts our INN's performance, surpassing PDflow

Method	10K points						50K points					
	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Noisy	12.14	6.01	30.63	13.22	37.83	18.05	6.86	4.55	14.20	10.71	19.03	15.20
Pointfilter [16]	12.45	6.78	22.02	8.30	24.26	9.04	6.99	4.92	8.63	5.98	9.50	6.63
PD-flow [10]	9.06	4.71	18.99	6.39	22.01	7.41	4.37	3.05	5.97	3.93	7.00	4.66
Score [9]	12.78	5.40	21.88	7.10	24.51	8.23	4.51	2.92	6.22	3.92	7.11	4.54
IterativePFN [2]	6.77	3.70	16.57	4.72	18.87	5.32	3.50	2.54	**4.30**	**2.82**	**4.68**	**3.03**
Ours(light)	**6.41**	3.71	15.95	4.78	18.40	5.37	**3.35**	2.58	4.45	2.94	4.92	3.18
Ours(heavy)	6.45	**3.67**	**15.70**	**4.48**	**18.03**	**4.98**	3.37	**2.56**	4.31	2.84	4.72	3.07

Table 9. Denoising results on the PUNet dataset with discrete noise. CD and P2M distances are multiplied by 10^5.

Method	10K points						50K points					
	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Noisy	51.44	27.20	122.67	88.38	168.75	131.00	29.68	23.13	90.43	80.44	133.44	121.98
Pointfilter [16]	30.30	10.52	64.94	35.37	97.18	63.12	11.95	7.62	39.33	31.62	69.26	59.59
PD-flow [10]	25.70	8.96	45.32	23.87	68.51	43.06	8.39	5.65	30.63	25.53	62.49	55.25
Score [9]	29.54	10.00	47.72	22.78	63.79	36.15	8.35	4.97	17.82	12.57	29.79	22.87
IterativePFN [2]	24.29	6.18	34.81	11.60	45.71	19.63	6.63	**3.42**	10.34	**6.05**	19.12	**13.02**
Ours(light)	21.54	6.23	31.05	12.35	38.54	18.05	5.74	3.60	10.21	6.82	**18.02**	13.35
Ours(heavy)	**20.88**	**5.78**	**28.93**	**11.33**	**36.39**	**17.31**	**5.33**	3.43	**9.69**	6.78	18.80	14.58

Table 10. Denoising results on the PUNet dataset with Laplace noise. CD and P2M distances are multiplied by 10^5.

Method	10K points						50K points					
	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Noisy	37.05	16.50	81.35	49.92	107.59	72.80	19.18	13.35	52.47	43.35	75.67	65.18
Pointfilter [16]	25.86	8.82	42.44	17.14	57.16	28.06	10.26	6.35	19.38	13.42	31.84	23.96
PD-flow [10]	21.21	6.90	33.48	14.11	42.41	21.39	6.64	4.26	15.79	11.98	27.47	22.32
Score [9]	24.94	7.58	37.63	14.50	46.29	21.01	7.20	4.05	13.88	9.17	22.06	15.91
IterativePFN [2]	20.23	5.07	30.52	8.67	35.51	12.11	6.10	3.09	8.47	**4.73**	12.72	**7.83**
Ours(light)	18.08	5.04	26.11	8.67	30.06	11.59	4.96	3.07	7.65	4.95	**11.85**	8.01
Ours(heavy)	**17.60**	**4.72**	**24.89**	**8.01**	**29.50**	**11.57**	**4.66**	**2.94**	**7.14**	4.77	11.98	8.40

Table 11. Denoising results on the PUNet dataset with non-isotropic Gaussian noise. CD and P2M distances are multiplied by 10^5.

Method	10K points						50K points					
	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Noisy	12.49	6.39	34.94	14.81	45.70	20.47	8.15	4.81	17.45	11.43	23.22	16.21
Pointfilter [16]	12.83	6.82	26.05	8.44	29.64	9.24	7.71	4.87	9.83	5.95	10.73	6.56
PD-flow [10]	9.09	4.70	20.56	6.36	24.08	7.50	4.59	3.05	6.29	3.97	7.51	4.89
Score [9]	13.13	5.40	24.85	7.06	28.23	8.17	5.10	2.90	7.00	3.86	8.04	4.58
IterativePFN [2]	6.79	3.72	20.40	4.82	24.48	5.58	4.47	**2.54**	6.04	3.02	6.57	3.32
Ours(light)	**6.39**	3.70	17.76	4.75	20.88	5.42	3.73	2.56	4.87	2.99	5.32	3.30
Ours(heavy)	6.45	**3.65**	**17.35**	**4.45**	**20.20**	**4.98**	**3.69**	2.55	**4.63**	**2.88**	**4.97**	**3.13**

Table 12. Denoising results on the PUNet dataset with uniform noise. CD and P2M distances are multiplied by 10^5.

Method	10K points						50K points					
	1% noise		2% noise		2.5% noise		1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Noisy	36.9	16.03	79.39	47.72	105.02	70.03	18.69	12.82	50.48	41.36	72.49	62.03
Bilateral [3]	31.47	14.95	51.94	26.75	73.83	44.31	17.09	13.9	17.98	13.98	25.34	19.55
Jet [1]	31.91	13.42	55.25	31.14	61.51	36.56	7.95	4.39	13.67	8.77	16.68	11.18
WLOP [5]	60.41	30.85	88.59	52.95	109.68	74.29	9.61	6.27	19.94	14.53	28.44	21.66
Ours(light)	<u>18.25</u>	<u>4.96</u>	<u>25.67</u>	<u>8.24</u>	<u>28.49</u>	<u>10.34</u>	<u>4.96</u>	<u>3.06</u>	<u>7.06</u>	<u>4.47</u>	<u>9.18</u>	<u>5.92</u>
Ours(heavy)	**17.81**	**4.65**	**24.41**	**7.58**	**26.99**	**9.67**	**4.70**	**2.95**	**6.46**	**4.25**	**8.63**	**5.81**

Table 13. Denoising results of traditional methods on the PUNet dataset with Gaussian noise. CD and P2M distances are multiplied by 10^5.

with MLGC. This indicates that our INN, with its unconstrained and expressive architecture, can naturally combine with MLGC and effectively enhance MLGC's feature representation capabilities. In contrast, PDflow's affine coupling layer, constrained by dimension partitioning, is incompatible with MLGC

7. Runtime and Number of Parameters for Learning-based Methods

In this section, we conduct a comparison of inference speed and the number of parameters among various existing learning-based methods. As described in Sec. 3, our denoising framework employs a deep neural network architecture, but thanks to the architecture of the invertible neural network, each monotone block can share weights in both the forward and inverse transformation processes, leading to a reduction in the number of parameters in our model. As shown in Tab. 8, ScoreDenoise [9] and DMRDenoise [8] exhibit the smallest number of parameters and the fastest inference speeds, respectively. However, in general, both the light and the heavy versions of our networks also perform competitively in these two metrics. Additionally, it is worth noting that when compared to IterativePFN [2], which is recognized for its exceptional denoising capabilities among existing deep learning methods, both versions of our framework not only consistently deliver better results for the vast majority of testing models, but also have faster inference speed and a smaller number of parameters.

References

[1] Frédéric Cazals and Marc Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121–146, 2005. 3, 6

[2] Dasith de Silva Edirimuni, Xuequan Lu, Zhiwen Shao, Gang Li, Antonio Robles-Kelly, and Ying He. Iterativepfn: True iterative point cloud filtering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13530–13539, 2023. 1, 2, 4, 5, 6

[3] Julie Digne and Carlo De Franchis. The bilateral filter for point clouds. *Image Processing On Line*, 7:278–287, 2017. 3, 6

[4] Xian-Feng Han, Jesse S Jin, Ming-Jie Wang, and Wei Jiang. Iterative guidance normal filter for point cloud. *Multimedia Tools and Applications*, 77:16887–16902, 2018. 1, 4

[5] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM transactions on graphics (TOG)*, 28(5):1–7, 2009. 3, 6

[6] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018. 2

[7] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7203–7212, 2019. 1

[8] Shitong Luo and Wei Hu. Differentiable manifold reconstruction for point cloud denoising. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1330–1338, 2020. 4, 6

[9] Shitong Luo and Wei Hu. Score-based point cloud denoising. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4583–4592, 2021. 1, 4, 5, 6

[10] Aihua Mao, Zihui Du, Yu-Hui Wen, Jun Xuan, and Yong-Jin Liu. Pd-flow: A point cloud denoising framework with normalizing flows. In *European Conference on Computer Vision*, pages 398–415. Springer, 2022. 4, 5

[11] Francesca Pistilli, Giulia Fracastoro, Diego Valsesia, and Enrico Magli. Learning graph-convolutional representations for point cloud denoising. In *European conference on computer vision*, pages 103–118. Springer, 2020. 4

[12] 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. In *Computer graphics forum*, pages 185–203. Wiley Online Library, 2020. 4

[13] Andrés Serna, Beatriz Marcotegui, François Goulette, and Jean-Emmanuel Deschaud. Paris-rue-madame database: a 3d mobile laser scanner dataset for benchmarking urban detection, segmentation and classification methods. In *4th international conference on pattern recognition, applications and methods ICPRAM 2014*, 2014. 10

[14] Peng-Shuai Wang, Yang Liu, and Xin Tong. Mesh denoising via cascaded normal regression. *ACM Trans. Graph.*, 35(6): 232–1, 2016. 2

[15] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 4

[16] 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, 2020. 4, 5

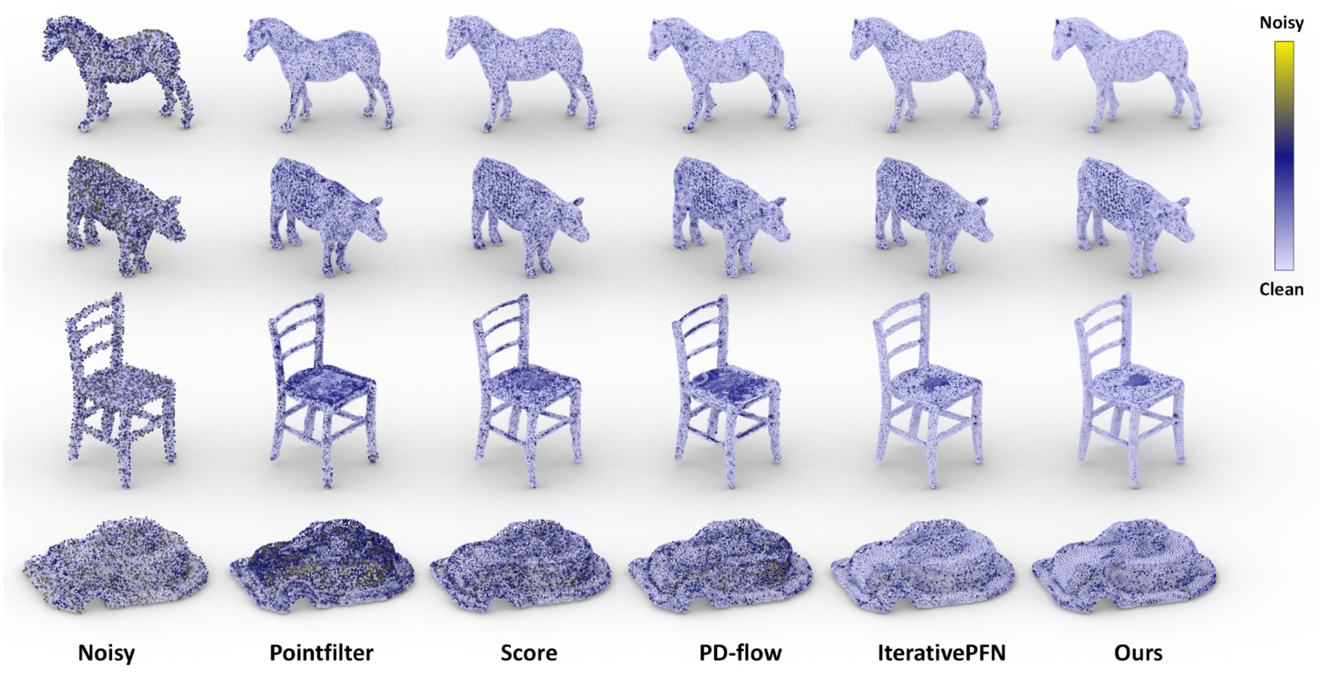

Figure 2. Visual results of point-wise P2M distance for 10K resolution shapes with discrete noise and a scale parameter of 2% of the bounding sphere radius.

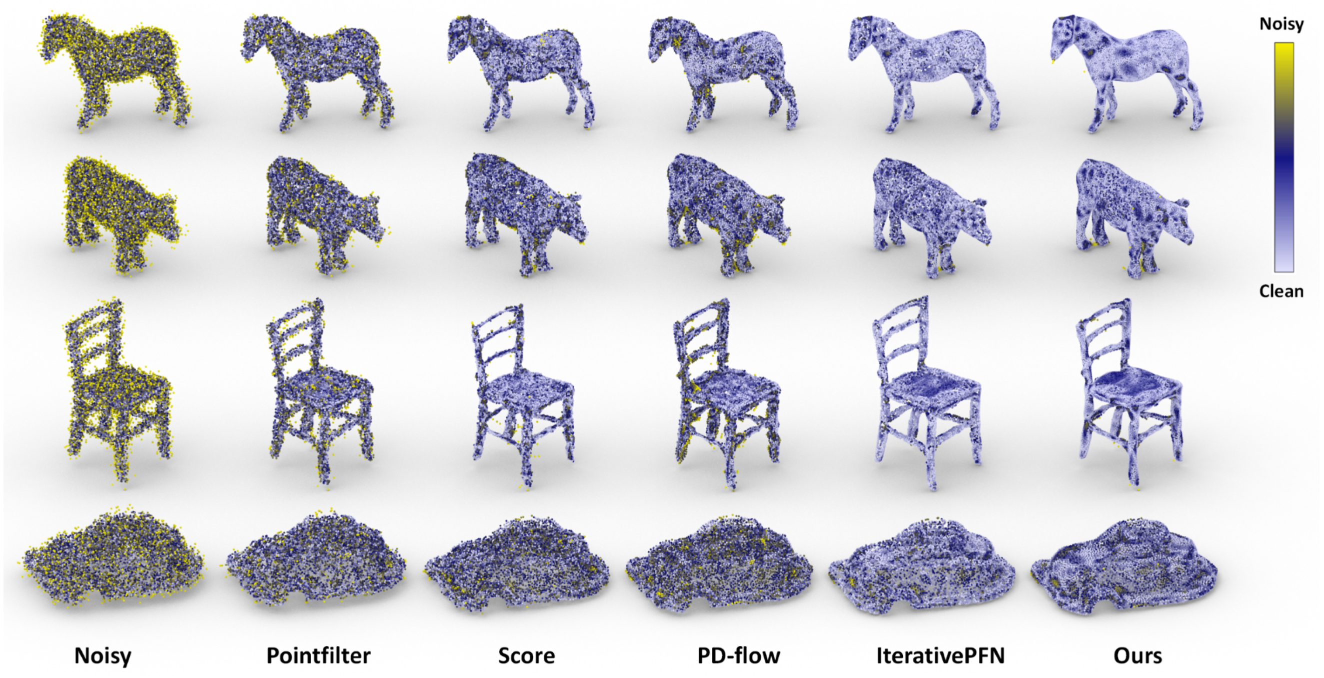

Figure 3. Visual results of point-wise P2M distance for 10K resolution shapes with Laplace noise and a scale of 2% of the bounding sphere radius.

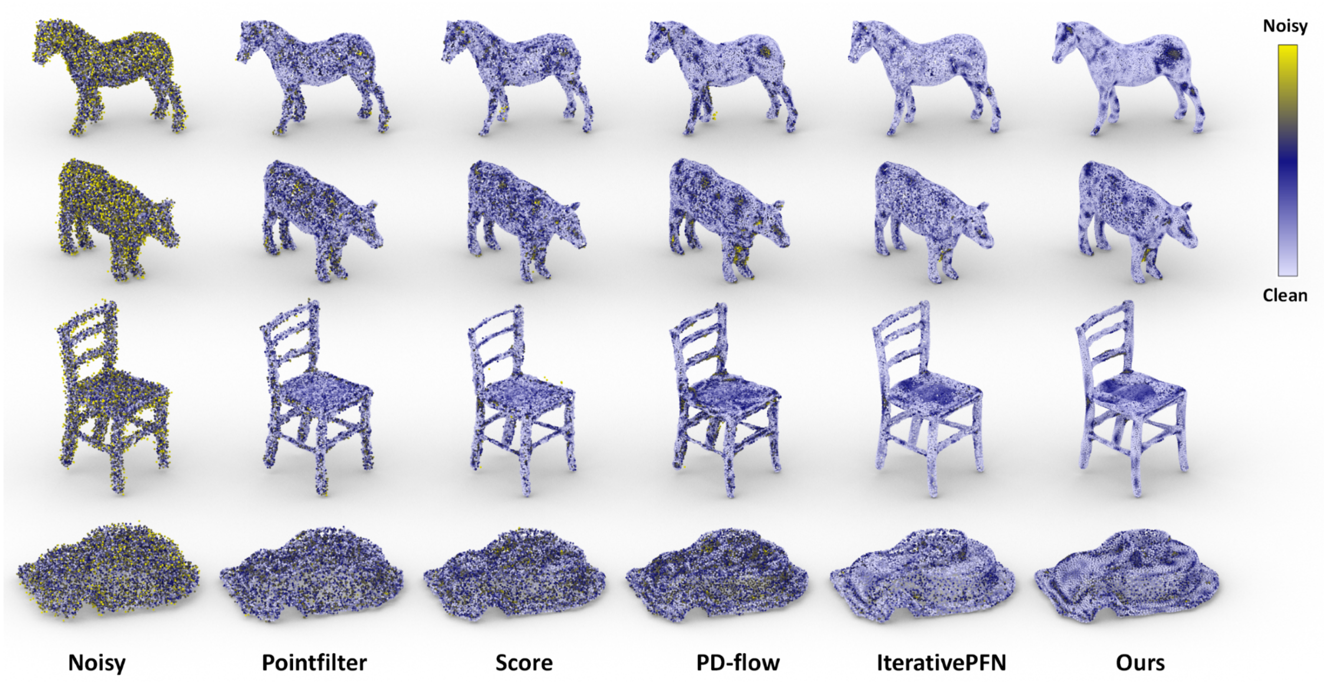

Figure 4. Visual results of point-wise P2M distance for 10K resolution shapes with non-isotropic Gaussian noise and a scale of 2% of the bounding sphere radius.

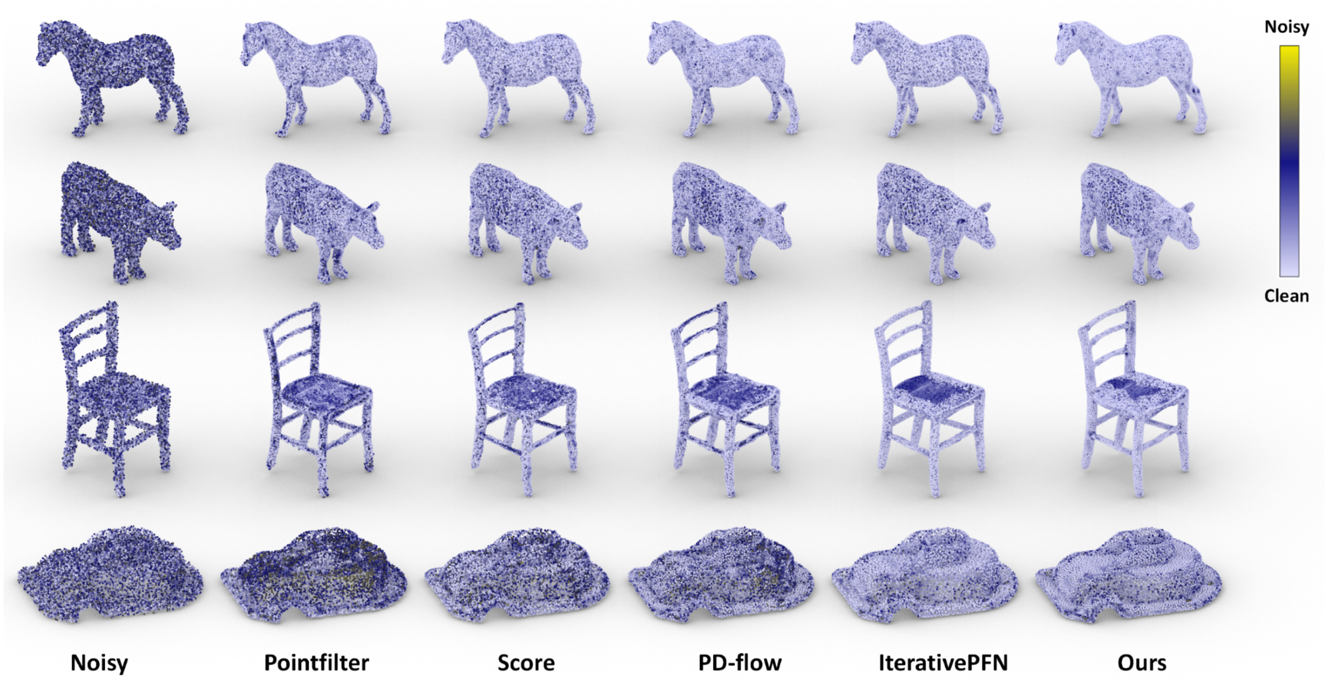

Figure 5. Visual results of point-wise P2M distance for 10K resolution shapes with noise uniformly distributed within a 3D sphere of radius s. Here, s corresponds to the noise scale and is equal to 2% of the bounding sphere radius.

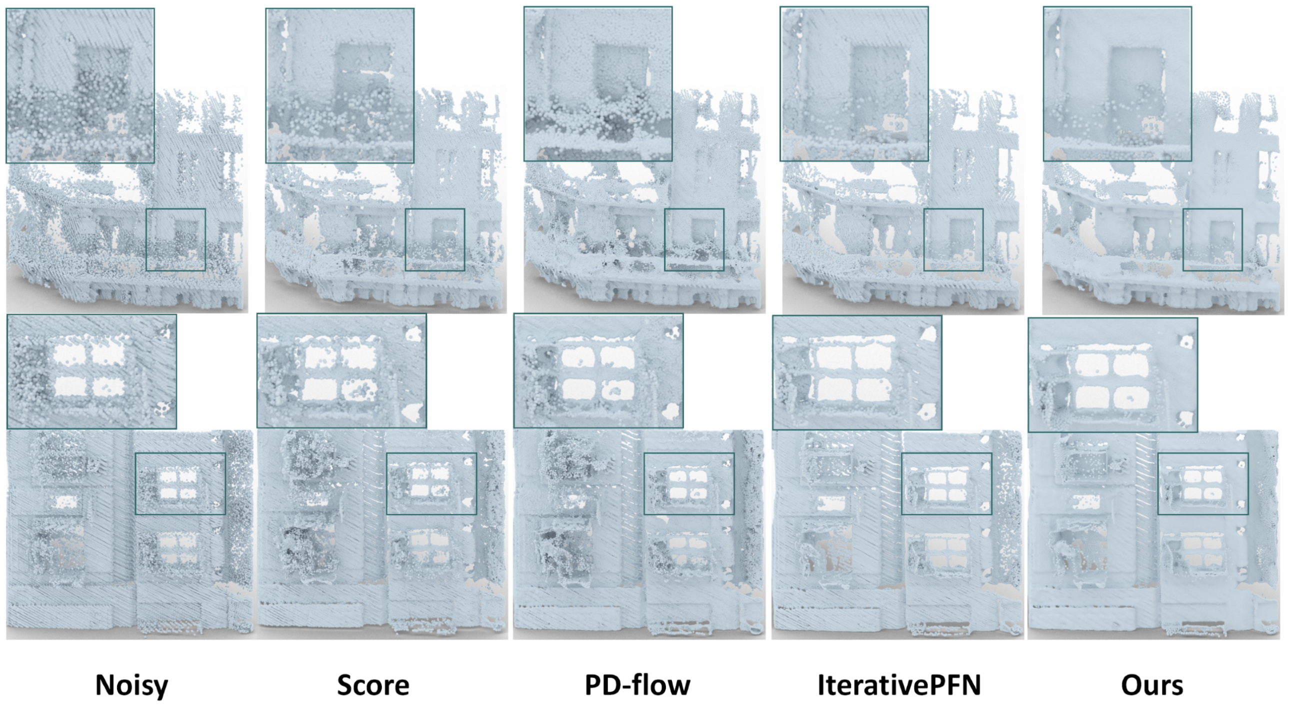

Figure 6. Visual results on two additional scenes of the RueMadame dataset [13].