Supplementary Material for
"Score-Based Point Cloud Denoising"

Shitong Luo, Wei Hu
Wangxuan Institute of Computer Technology
Peking University
{luost, forhuwei}@pku.edu.cn

Overview

This supplementary material is organized as follows.

- In Section A, we present more experiments using different noise models such as non-isotropic Gaussian noise, uni-directional noise, Laplace noise, uniform noise, and discrete noise.
- In Section B, we present the full results and analysis of ablation studies.
- In Section C, we adapt the unsupervised training objective of TotalDn [4] to train an unsupervised adaptation of our model, and compare the adaptation to TotalDn.
- In Section D, we present hyper-parameters and other implementation details for reproducing our model.
- In Section E, we show more visual results on both synthetic point clouds and real-world point clouds.

A. Additional Quantitative Results

In this section, we present more quantitative results under different noise types. Note that, though evaluated under different noise types, the model is *trained only using Gaussian noise*. The purpose of these additional experiments is to show our model's generalizability to different noise types which are unseen during training.

We only present the results of stronger baselines (MRPCA[6], GLR[9], PCNet[7], and DMR[5]) and leave out the weaker ones because their performance is clearly inferior.

A.1. Non-isotropic Gaussian Noise

We set the covariance matrix of the 3D Gaussian distribution to the following positive definite matrix:

$$\mathbf{\Sigma} = s^2 \times \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{4} \\ -\frac{1}{2} & 1 & -\frac{1}{4} \\ -\frac{1}{4} & -\frac{1}{4} & 1 \end{bmatrix}, \tag{1}$$

where s is the scale parameter controlling the magnitude of noise. We set s to 1%, 2% and 3% of the shape's bounding sphere radius to generate point clouds at different noise levels.

The table below is the result.

Test-set: PU	10K 1%		10K 2%		10K 3%		50K 1%		50K 2%		50K 3%	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
MRPCA [6]	2.676	0.689	**3.605**	**1.007**	5.108	2.081	**0.669**	**0.102**	2.058	1.088	5.789	4.138
GLR [9]	2.910	1.048	3.779	1.332	4.975	2.195	0.694	0.163	1.605	0.850	3.880	2.758
PCNet [7]	3.432	1.129	7.393	3.940	12.952	8.654	1.040	0.344	1.458	0.627	2.414	1.402
DMR [5]	4.487	1.718	5.046	2.148	5.916	2.866	1.156	0.463	1.554	0.791	2.458	1.557
Ours	**2.470**	**0.456**	**3.682**	**1.084**	**4.776**	**2.000**	**0.712**	**0.149**	**1.317**	**0.591**	**2.085**	**1.176**

Table 1. Comparison among competitive denoising algorithms under the non-isotropic Gaussian noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

Our method outperforms other methods under the non-isotropic Gaussian noise and, generalizes much better to this novel noise type than other deep-learning-based methods.

A.2. Uni-directional Noise

We only perturb the x-component of point clouds using Gaussian noise. The standard deviation is set to 1%, 2% and 3% of the shape's bounding sphere radius to generate point clouds at different noise levels. Below is the denoising result:

Test-set: PU	10K 1%		10K 2%		10K 3%		50K 1%		50K 2%		50K 3%	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
MRPCA [6]	1.712	0.646	2.564	0.767	**3.237**	**1.063**	**0.446**	**0.056**	0.841	0.271	1.879	1.070
GLR [9]	2.033	1.026	2.837	1.139	3.472	1.434	0.454	0.089	**0.763**	0.262	1.507	0.812
PCNet [7]	1.530	0.432	3.466	1.360	5.638	2.914	0.690	0.201	1.005	0.370	1.546	0.830
DMR [5]	4.350	1.674	4.581	1.830	5.015	2.169	1.032	0.372	1.181	0.488	1.585	0.864
Ours	**1.442**	**0.279**	**2.412**	**0.543**	**3.391**	**1.108**	**0.486**	**0.063**	**0.780**	**0.235**	**1.486**	**0.799**

Table 2. Comparison among competitive denoising algorithms under the uni-directional noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

Our method outperforms other methods under uni-directional noise and, generalizes much better to this novel noise type than other deep-learning-based methods.

A.3. Laplace Noise

We perturb point clouds using Laplace noise. The scale of Laplace noise is set to 1%, 2% and 3% of the shape's bounding sphere radius to generate point clouds at different noise levels. Below is the denoising result:

Test-set: PU	10K 1%		10K 2%		10K 3%		50K 1%		50K 2%		50K 3%	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
MRPCA [6]	2.950	0.724	**4.216**	**1.428**	7.951	4.441	**0.816**	0.203	4.047	2.827	11.629	9.535
GLR [9]	3.223	1.121	4.751	2.090	7.977	4.773	0.962	0.374	3.269	2.325	8.675	7.162
PCNet [7]	4.616	1.940	11.082	7.218	20.981	15.922	1.190	0.458	2.854	1.868	7.555	6.020
DMR [5]	4.600	1.811	5.441	2.469	6.918	3.714	1.243	0.537	1.881	1.077	3.609	2.634
Ours	**2.915**	**0.674**	**4.601**	**1.799**	**6.332**	**3.271**	**0.823**	**0.231**	**1.658**	**0.869**	**2.728**	**1.681**

Table 3. Comparison among competitive denoising algorithms under Laplace noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

Our method outperforms other methods under Laplace noise and, generalizes much better to this novel noise type than other deep-learning-based methods.

A.4. Uniform Noise

We use the uniform distribution on a 3D ball with radius s to generate noise:

$$p(\boldsymbol{x}; s) = \begin{cases} \frac{3}{4\pi s^3} & \|\boldsymbol{x}\|_2 \leq s \\ 0 & \text{Otherwise} \end{cases}. \tag{2}$$

We set the radius s to 1%, 2% and 3% of the shape's bounding sphere radius to generate point clouds at different noise levels. Below is the denoising result:

Test-set: PU	10K 1%		10K 2%		10K 3%		50K 1%		50K 2%		50K 3%	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
MRPCA [6]	1.555	0.633	2.754	0.684	3.229	0.765	0.502	**0.044**	0.660	**0.092**	1.016	0.307
GLR [9]	1.850	1.015	2.948	1.052	3.400	1.109	**0.485**	0.071	**0.656**	0.132	**0.903**	0.293
PCNet [7]	**1.205**	0.337	3.378	1.018	5.044	1.995	0.806	0.228	1.064	0.358	1.218	0.451
DMR [5]	4.307	1.640	4.445	1.693	4.685	1.857	1.064	0.391	1.159	0.464	1.287	0.572
Ours	1.277	**0.248**	**2.467**	**0.418**	**3.079**	**0.654**	**0.506**	**0.047**	**0.690**	**0.129**	**0.917**	**0.282**

Table 4. Comparison among competitive denoising algorithms under the uniform noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

In this setting, our method outperforms other deep-learning-based methods and is on par with the two optimization-based methods.

It worth noting that in Analysis section, we assume the noise distribution is uni-modal and its mode is 0, but the uniform noise does not satisfy the assumptions. Thus, this experiment demonstrates that our model is effective to a broader family of noise beyond the assumptions.

A.5. Discrete Noise

We perturb point clouds using the following noise model:

$$p(\boldsymbol{x}; s) = \begin{cases} 0.1 & \boldsymbol{x} = (\pm s, 0, 0) \text{ or } (0, \pm s, 0) \text{ or } (0, 0, \pm s) \\ 0.4 & \boldsymbol{x} = (0, 0, 0) \\ 0 & \text{Otherwise} \end{cases}, \tag{3}$$

where s controls the scale of noise. We set the scale parameter s to 1%, 2% and 3% of the shape's bounding sphere radius to generate point clouds at different noise levels. Below is the denoising result:

Test-set: PU	10K 1%		10K 2%		10K 3%		50K 1%		50K 2%		50K 3%	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
MRPCA [6]	1.522	0.629	2.353	0.674	**2.607**	0.743	**0.404**	**0.044**	**0.488**	**0.074**	**0.681**	**0.207**
GLR [9]	1.838	1.014	2.665	1.047	2.952	1.116	0.413	0.072	0.550	0.138	0.786	0.307
PCNet [7]	**1.177**	0.307	2.870	0.871	4.028	1.674	0.669	0.204	0.857	0.310	0.986	0.385
DMR [5]	4.288	1.625	4.388	1.668	4.566	1.785	1.039	0.379	1.151	0.461	1.246	0.540
Ours	1.249	**0.251**	**2.177**	**0.416**	2.653	**0.653**	**0.449**	**0.048**	**0.615**	**0.136**	**0.812**	**0.277**

Table 5. Comparison among competitive denoising algorithms under the uni-directional noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

Our method outperforms other deep-learning-based methods but does not outperform optimization-based methods when the point cloud is denser. However, the performance gap between our method and the optimization-based methods is generally not large.

In Analysis section, we assume the noise distribution is continuous. This experiment using discrete noise models indicates that our model is effective to a broader family of noise beyond the assumptions.

B. Additional Results and Analysis of Ablation Studies

In Experiment section, we present ablation studies to evaluate the contribution of following main designs of our method: (1) score-based denoising algorithm, (2) neighborhood-covering training objective, and (3) ensemble score function.

Following are the complete version of Table 3 in the paper and visual results:

Dataset: PU	10K, 1%		10K, 2%		10K, 3%		50K, 1%		50K, 2%		50K, 3%	
Ablation	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
(1)	3.237	0.994	5.241	2.258	7.471	4.049	1.487	0.749	2.793	1.882	4.578	3.425
(1) + iter.	3.237*	0.994*	5.241*	2.258*	6.073	2.953	1.487*	0.738	2.270	1.378	2.884	1.930
(2)	4.726	2.188	5.740	2.748	5.976	3.036	1.347	0.598	1.826	0.985	2.319	1.395
(3)	2.522	0.471	4.021	1.280	6.872	3.497	0.768	0.178	1.585	0.803	4.188	2.971
Full	**2.521**	**0.463**	**3.686**	**1.074**	**4.708**	**1.942**	**0.716**	**0.150**	**1.288**	**0.566**	**1.928**	**1.041**

Table 6. Comparison of the unsupervised adaptation of our method, TotalDn[4], and optimization-based methods under simulated LiDAR noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4. (*) The best performance is achieved after running for only 1 iteration.

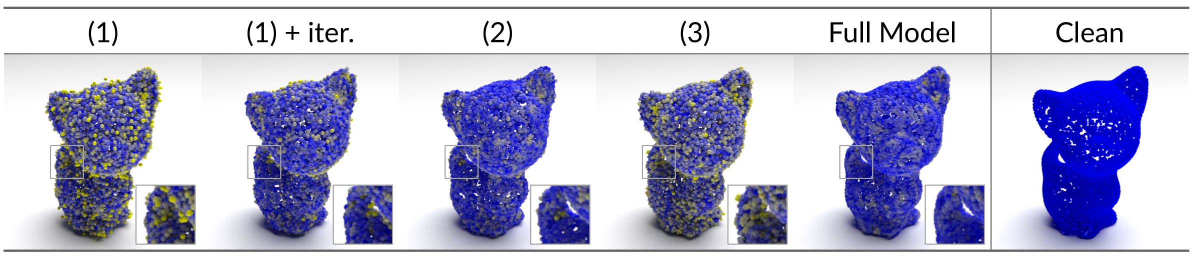

Figure 1. Visual results of ablation studies. The resolution and noise level are 50K and 3%.

As shown above, all the main component contributes positively to the performance.

Specifically, the score-based denoising algorithm (1) is shown superior to displacement-based methods widely employed in previous works [7, 4]. It leads to less outliers as shown in the figure. This is because for displacement-based methods, both the direction and magnitude of displacements should be accurate enough in order to achieve good quality. If the magnitude of displacement is over-estimated, the point will overreach the clean surface. Otherwise if the magnitude of displacement is under-estimated, the point will still be far away from the clean surface. In contrast, score-based methods only require the direction of score to be accurate because it iteratively moves points towards the surface at decaying step sizes. When the score is under-estimated, more than one step of updates will eventually move the point closer to the surface. When the score is over-estimated, the decaying step size will prevent over-denoising.

The neighborhood-covering training objective (2) not only train the network to predict scores on the input noisy points but also the neighborhood of points. This is crucial to score-based denoising because during gradient-ascent, points move around and their positions change. This relies on scores defined on positions other than their original positions. It can be seen in the figure that with such training objective, the denoised point cloud is smoother with details preserved better.

The ensemble score function (3) improves the robustness of the denoiser especially in high noise level cases. In high noise level cases, an individual local score function might be less accurate because it is far away from the clean surface. By considering more than 1 local score functions, the estimated score will be more reliable.

C. Unsupervised Learning

We adapt the unsupervised training objective in [4] to train an unsupervised version of our model:

$$\mathcal{L}_{\text{unsup}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\boldsymbol{x}_j \sim \mathrm{NN}(\boldsymbol{x}_i)} \left[\left\| \mathbb{E}_{\boldsymbol{x}_j \sim \mathrm{NN}(\boldsymbol{x}_i, \boldsymbol{X})} \left[\boldsymbol{x}_j - \boldsymbol{x}_i \right] - \mathcal{S}_i(\boldsymbol{x}_j) \right\|_2^2 \right]. \tag{4}$$

Except for the objective function, the training setting is identical to the supervised learning. We compare this unsupervised adaptation to the unsupervised-learning-based denoising model TotalDn [4] using both Gaussian noise and simulated LiDAR noise. We also compare them to optimization-based denoising methods because optimization-based methods do not rely on data. The results are as follows:

Test-set: PU	10K 1%		10K 2%		10K 3%		50K 1%		50K 2%		50K 3%	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
Bilateral [2]	3.646	1.342	5.007	2.018	6.998	3.557	0.877	0.234	2.376	1.389	6.304	4.730
Jet [1]	2.712	0.613	4.155	1.347	6.262	2.921	0.851	0.207	2.432	1.403	5.788	4.267
MRPCA [6]	2.972	0.922	3.728	1.117	5.009	1.963	0.669	0.099	2.008	1.033	5.775	4.081
GLR [9]	2.959	1.052	3.773	1.306	4.909	2.114	0.696	0.161	1.587	0.830	3.839	2.707
TotalDn [4]	3.390	**0.826**	7.251	3.485	13.385	8.740	1.024	0.314	2.722	1.567	7.474	5.729
Ours (Unsup.)	**3.107**	0.888	**4.675**	**1.829**	**7.225**	**3.762**	**0.918**	**0.265**	**2.439**	**1.411**	**5.303**	**3.841**

Table 7. Comparison of the unsupervised adaptation of our method, TotalDn[4], and optimization-based methods under Gaussian noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

	Bilateral [3]	Jet [1]	MRPCA [6]	GLR [9]	TotalDn [4]	**Ours(Unsup.)**
CD	3.279	3.385	2.886	2.663	4.090	**3.420**
P2M	2.215	2.319	1.933	1.920	2.869	**2.331**

Table 8. Comparison of the unsupervised adaptation of our method, TotalDn[4], and optimization-based methods under simulated LiDAR noise. CD is multiplied by 10^4 and P2M is multiplied by 10^4.

The unsupervised adaptation of our method outperforms TotalDn[4]. However, both unsupervised-learning-based methods do not perform better than optimization-based methods.

D. Hyper-parameters and Implementation Details

Please find the code and other details at: `https://github.com/luost26/score-denoise`.

E. Additional Visual Results

E.1. Real-World Datasets

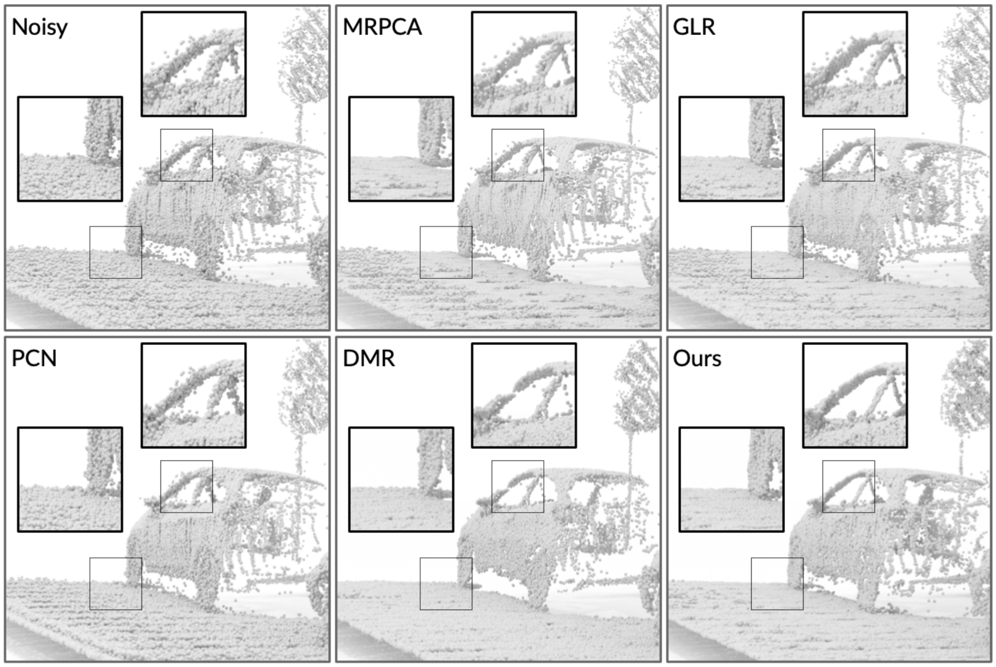

Figure 2. *Paris-rue-Madame* [8]

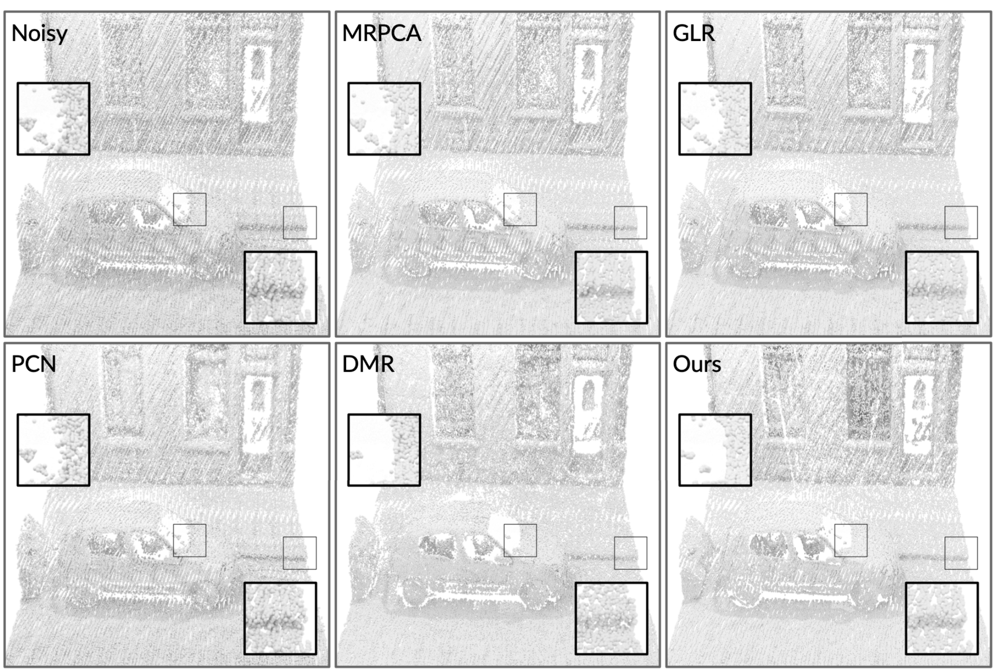

Figure 3. *Paris-rue-Madame* [8]

E.2. Synthetic Datasets

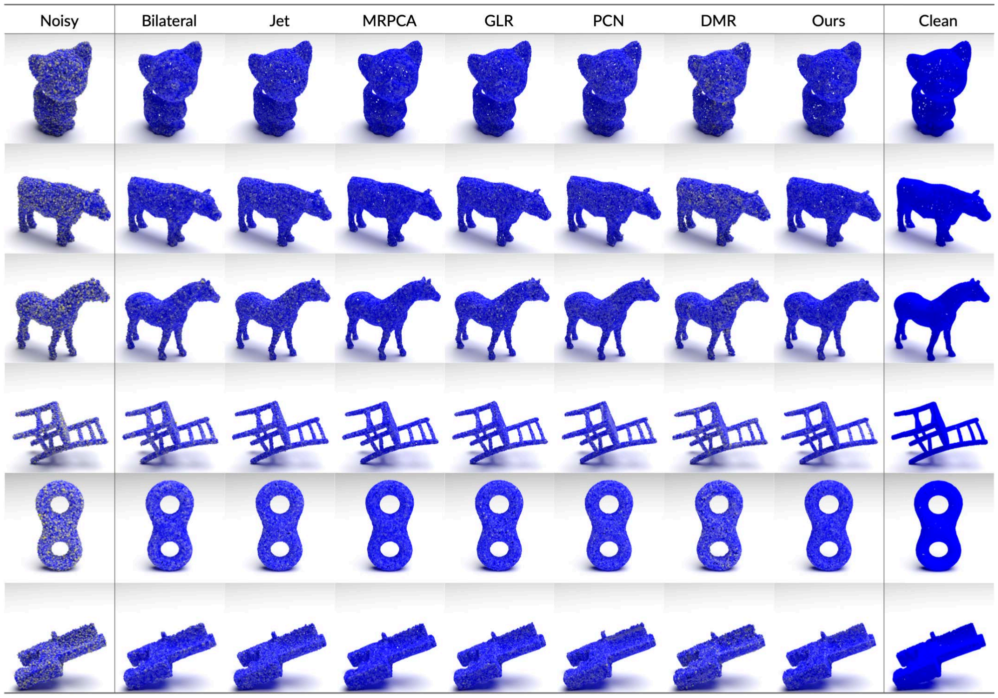

Figure 4. Gaussian 1%.

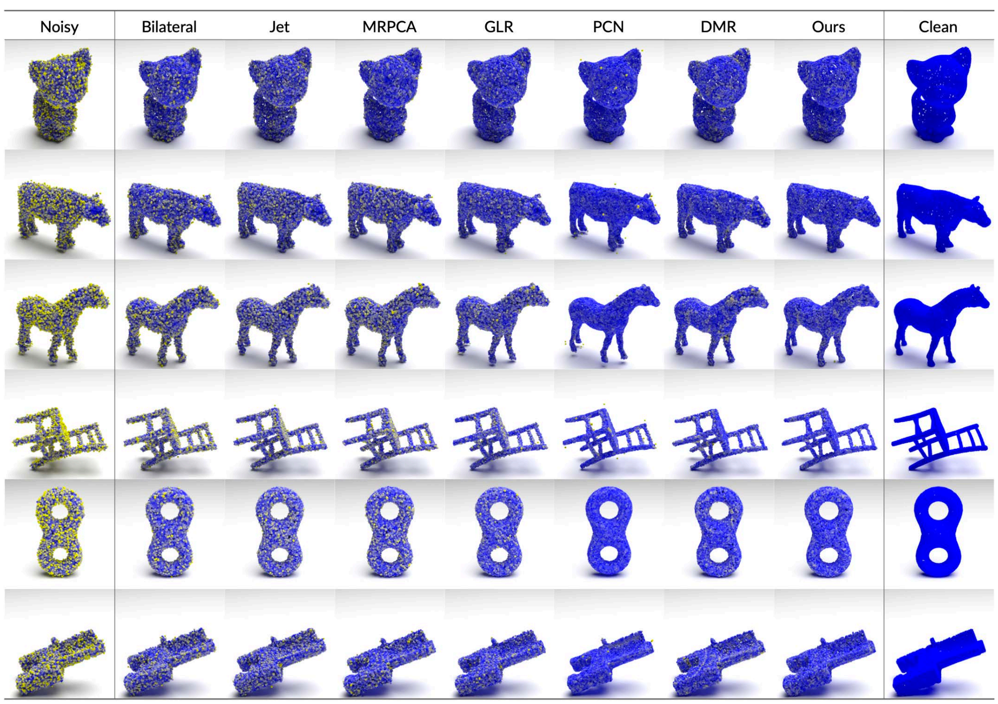

Figure 5. Gaussian 2%.

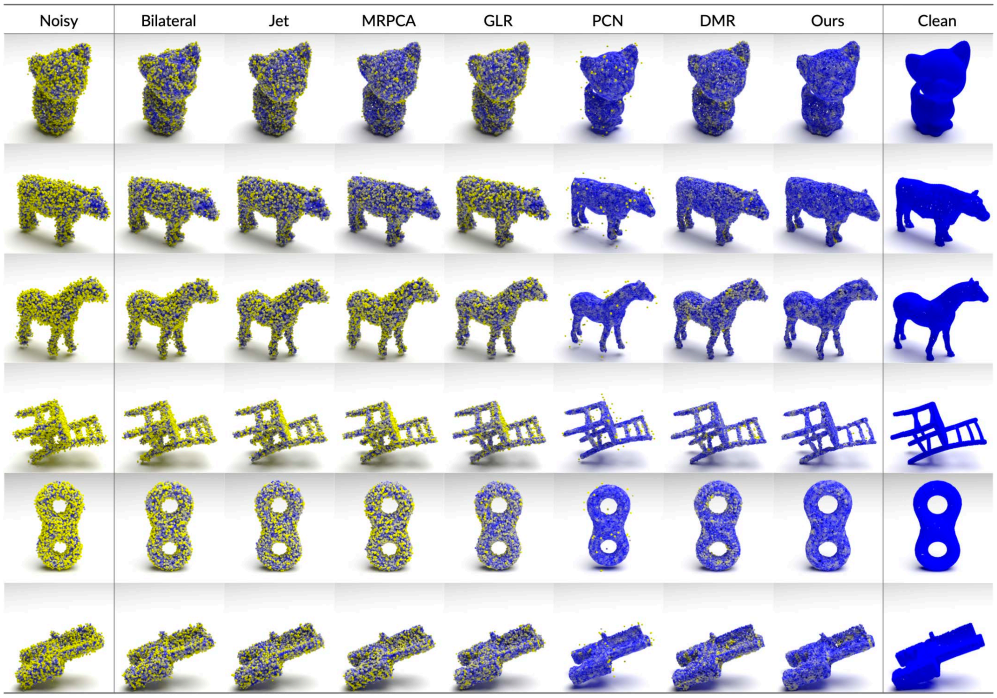

Figure 6. Gaussian 3%.

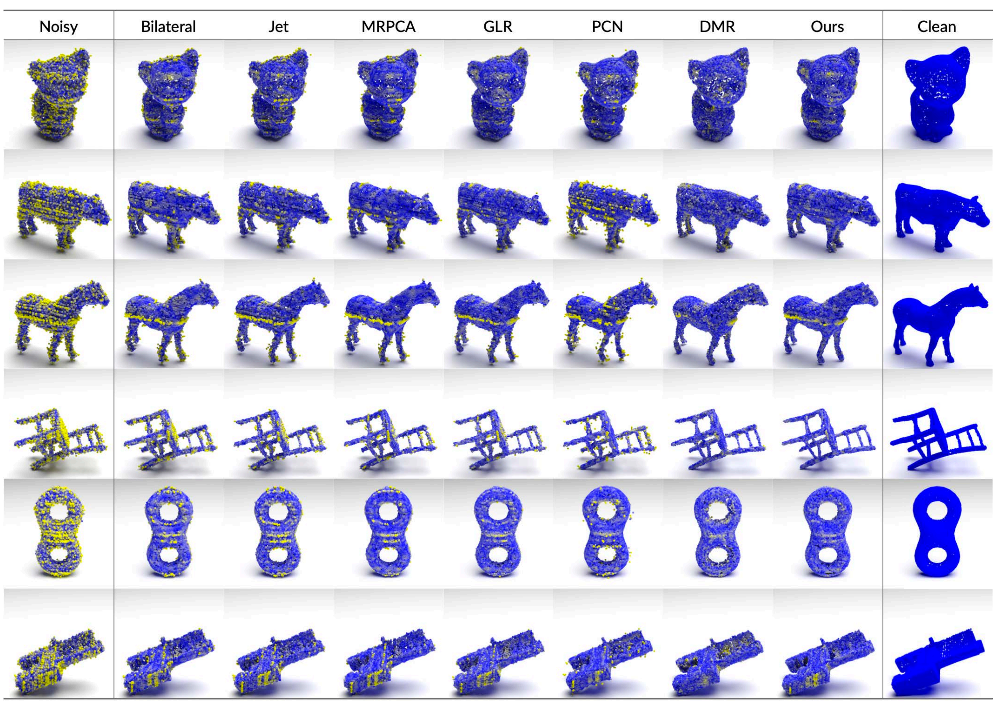

Figure 7. Simulated LiDAR noise.

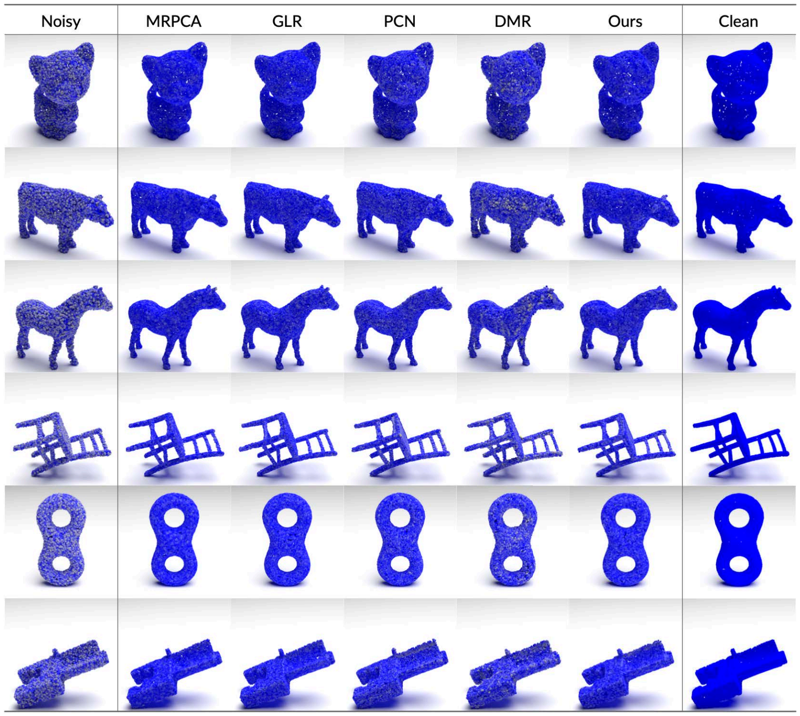

Figure 8. Uniform noise 2%.

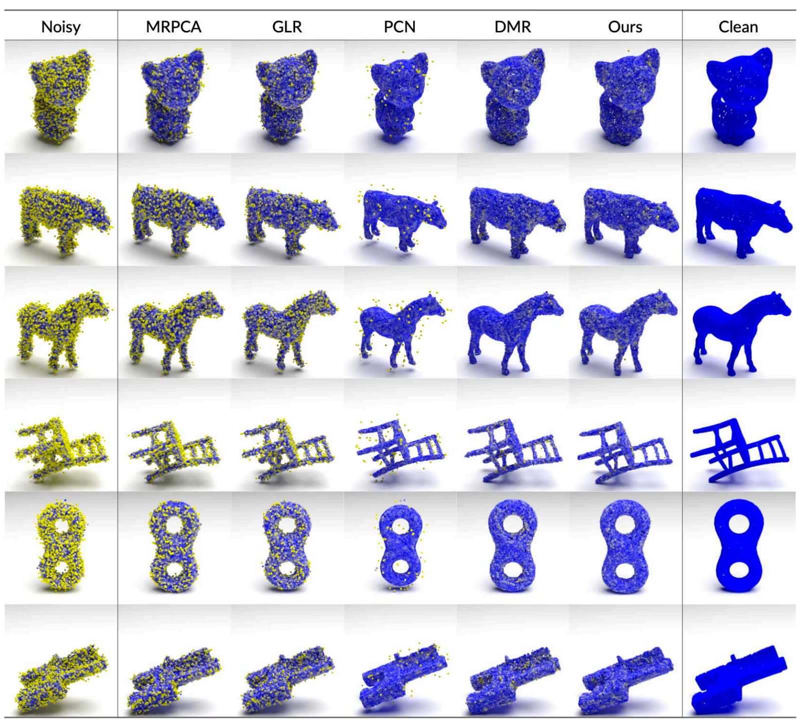

Figure 9. Laplace noise 2%.

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. 5

[2] Julie Digne and Carlo De Franchis. The bilateral filter for point clouds. *Image Processing On Line*, 7:278–287, 2017. 5

[3] Shachar Fleishman, Iddo Drori, and Daniel Cohen-Or. Bilateral mesh denoising. In *ACM SIGGRAPH 2003 Papers*, pages 950–953. 2003. 5

[4] Pedro Hermosilla, Tobias Ritschel, and Timo Ropinski. Total denoising: Unsupervised learning of 3d point cloud cleaning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 52–60, 2019. 1, 4, 5

[5] 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. 1, 2, 3

[6] Enrico Mattei and Alexey Castrodad. Point cloud denoising via moving rpca. In *Computer Graphics Forum*, volume 36, pages 123–137. Wiley Online Library, 2017. 1, 2, 3, 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. In *Computer Graphics Forum*, volume 39, pages 185–203. Wiley Online Library, 2020. 1, 2, 3, 4

[8] 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. 2014. 6

[9] Jin Zeng, Gene Cheung, Michael Ng, Jiahao Pang, and Yang Cheng. 3D point cloud denoising using graph Laplacian regularization of a low dimensional manifold model. *IEEE Transactions on Image Processing*, 29:3474–3489, December 2019. 1, 2, 3, 5