Supplementary Material of IterativePFN: True Iterative Point Cloud Filtering

Dasith de Silva Edirimuni[1], Xuequan Lu[1*], Zhiwen Shao[2†], Gang Li[1], Antonio Robles-Kelly[1,4], Ying He[3]
[1]School of Information Technology, Deakin University
[2]School of Computer Science and Technology, China University of Mining and Technology
[3]School of Computer Science and Engineering, Nanyang Technological University
[4]Defense Science and Technology Group, Australia
{dtdesilva, xuequan.lu, gang.li, antonio.robles-kelly}@deakin.edu.au,
zhiwen_shao@cumt.edu.cn, yhe@ntu.edu.sg

In this supplementary document, to the main paper, we provide the following:

A. Further comparisons on synthetic and scanned data

A.1. Additional visual comparisons on the Rue Madame and Kinect datasets

A.2. Comparison of conventional methods on the PUNet dataset with Gaussian noise and Kinect dataset

A.3. Quantitative comparisons on PUNet dataset with different noise patterns

B. Further ablations on iteration number

C. Ablation study on patch stitching

D. Filtering results for PointNet++ based encoders

E. Runtimes for learning based methods

A. Further Comparisons on Synthetic and Scanned Data

In this section, we provide additional visual and quantitative results on synthetic and scanned data.

A.1. Additional visual comparisons on the Rue Madame and Kinect datasets

Table 2 and Fig. 7, of the main paper, present quantitative results on the Kinect dataset and visual results on the RueMadame dataset, respectively. To further demonstrate our method's robustness, in the presence of real world noise, we provide visual results on two additional scenes of the RueMadame dataset, given in Fig. 1. Moreover, we provide visual comparisons of 4 scans from the Kinect v2 dataset in Fig. 2, which could not be included in the main paper due to constraints of space.

As seen in Fig. 1, our method performs considerably better compared to other methods when filtering noise for the RueMadame scenes. In scene 3, the roof and body of the vehicle is best filtered by our method while the filtering results of other methods leave behind significant noise artifacts. Furthermore, as illustrated in Fig. 2, we are able to better filter complex Kinect v2 scans, such as the model of the Boy, as opposed to state-of-the-art methods. This is supported by the results presented in Table 2 of the main paper, where we achieve the best results on the CD metric. It indicates that our method produces filtered point clouds closer in point distribution, and proximity, to the clean point clouds of the Kinect v2 dataset, than other state-of-the-art methods.

A.2. Comparison of conventional methods on the PUNet dataset with Gaussian noise and Kinect dataset

In the main paper we were not able to present a comparison of the filtering performance of conventional methods, due to limited space. In Table 1 and Table 2, we present quantitative results for the Bilateral filter [2], Jet fitting mechanism [1] and WLOP regularization mechanism [4]. The Bilater filter relies on initial point normals which were calculated using Principal Component Analysis (PCA) [3] as it is a widely used normal estimation technique. WLOP regularizes and downsamples noisy point clouds. These must then be upsampled and we use the Edge Aware Resampling (EAR) mechanism of Huang et al. [5] to accomplish this.

In general, conventional methods perform poorly compared to learning based methods. In addition, they require parameter tuning to obtain best results which can be tedious and time-consuming. Methods such as the Bilateral filter further rely on initial point normals to filter effec-

*Corresponding author: X. Lu, supported by PJ03906.PG00507.F002.
†Z. Shao is supported by the NSFC (No. 62106268).

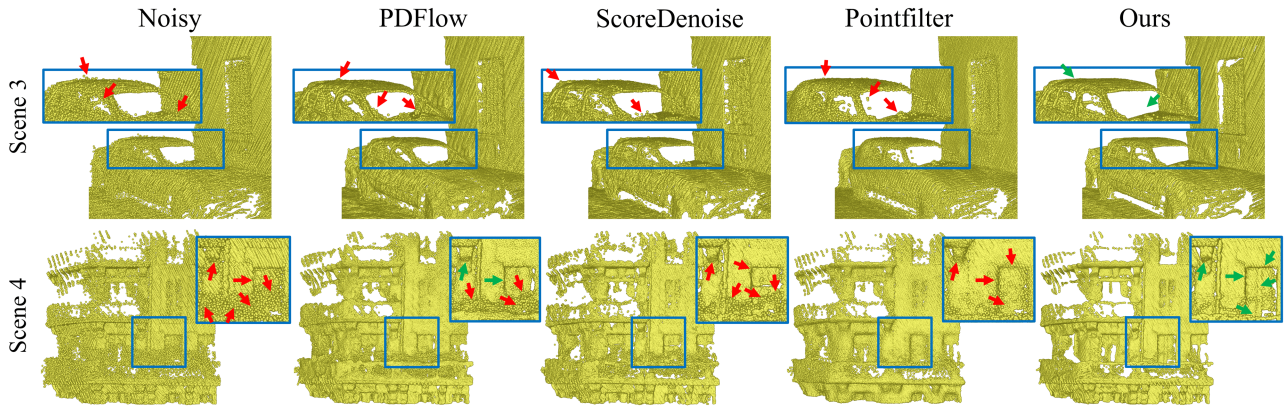

Figure 1. Visual results on two additional scenes of the RueMadame dataset. Green and red arrows are used to indicate accurately and inaccurately filtered regions, respectively.

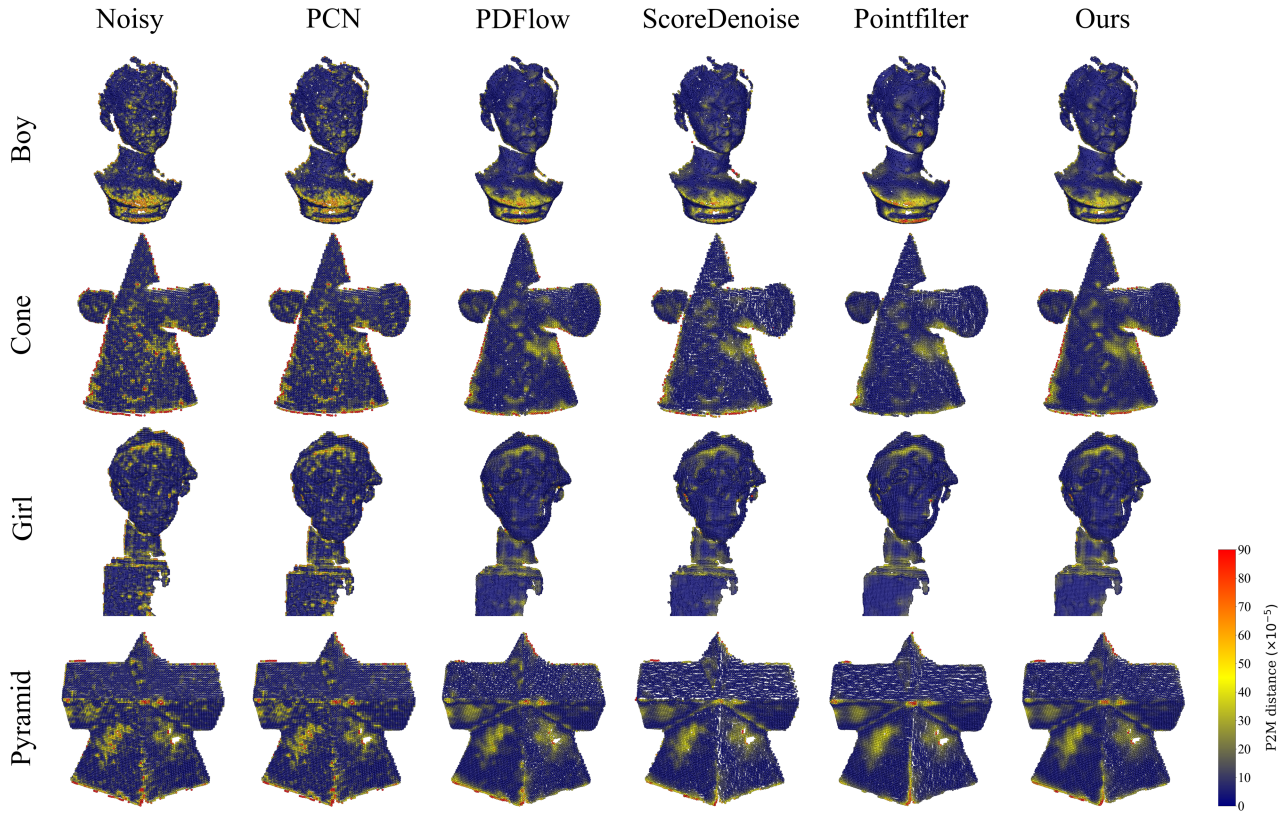

Figure 2. Visual results of point-wise P2M distances for 4 scans of the Kinect v2 dataset.

tively and the accuracy of these normals have an effect on the overall filtered output. WLOP, by contrast, recovers good P2M results on the Kinect v1 dataset. This is possibly due to the resampling procedure where the downsampled output from WLOP is upsampled by the EAR mechanism. Although this is reasonably successful for the Kinect v1 scanned data with relatively simple topology, for more complex topologies as those associated with the shapes in the PUNet dataset, the resampling procedure is less successful. As shown in Table 1, the resampling procedure yields high CD and P2M values as the WLOP mechanism is not able to optimally identify clean surfaces and omits geometric details. Moreover, WLOP performs poorly on the Kinect v2 dataset which illustrates its inability to generalize well across different filtering scenarios.

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
Bilateral	31.47	14.95	51.94	26.75	73.83	44.31	17.09	13.9	17.98	13.38	25.34	19.55
Jet	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	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	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**	**6.05**	**3.02**	**8.03**	**4.36**	**10.15**	**5.88**

Table 1. Filtering results of conventional methods on the PUNet dataset with Gaussian noise. CD and P2M distances are multiplied by 10^5.

Method	Kinect v1		Kinect v2	
	CD	P2M	CD	P2M
Bilateral	13.65	9.27	20.14	12.1
Jet	13.41	8.78	19.82	11.82
WLOP	13.89	**7.65**	33.0	14.79
Ours	**13.2**	8.43	**18.69**	**10.92**

Table 2. Filtering results on the Kinect v1 and Kinect v2 datasets. CD and P2M distances are multiplied by 10^5.

A.3. Quantitative comparisons on PUNet dataset with different noise patterns

In addition to Gaussian noise, we are interested in comparing filtering results for different noise patterns. In this section, we follow the work of ScoreDenoise [7] and investigate the filtering performance of methods on the following noise types:

Non-isotropic Gaussian noise where we set the covariance matrix of the Gaussian noise distribution to:

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

The noise scale parameter s is set to 1%, 2% and 2.5% of the bounding sphere's radius. Results are presented in Table 3 and Fig. 4. Much like the case of isotropic Guassian noise, Table 1 of the main paper, our method outperforms others and generalizes well to this noise pattern.

Discrete noise with the following distribution:

$$p(\boldsymbol{x}; s) = \begin{cases} 0.1, & \boldsymbol{x} = (\pm s, 0, 0) \text{ or } \boldsymbol{x} = (0, \pm s, 0) \\ & \text{or } \boldsymbol{x} = (0, 0, \pm s), \\ 0.4, & \boldsymbol{x} = (0, 0, 0) \\ 0, & \text{Otherwise} \end{cases} \tag{2}$$

where s is set to 1%, 2% and 2.5% of the bounding sphere's radius. Results are presented in Table 4 and Fig. 5. Our method outperforms others across resolutions and noise scales.

Laplace noise with the noise scale set to 1%, 2% and 2.5% of the bounding sphere's radius. Results are presented in Table 5 and Fig. 6. This noise pattern has a relatively high noise intensity as illustrated by the CD and P2M metric results for the noisy point clouds. Nevertheless, our method consistently outperforms other methods and recovers filtered point clouds with low CD and P2M distances from the clean counterparts.

Uniform distribution of noise within a 3D sphere of radius s, given by,

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

where s is set to 1%, 2% and 2.5% of the bounding sphere's radius. Results are presented in Table 6 and Fig. 7. This noise pattern corresponds to noise sampled uniformly within a sphere of radius s, and as pointed out by [7], is an example of a noise distribution that is not uni-modal and is unlike Gaussian or Laplace noise. Our method again outperforms other state-of-the-art methods.

B. Further Ablations on Iteration Number

In Table 7 we consider the impact of iteration number under higher noise settings. The maximum training noise for all models is 2%, much lower than the unseen, high noise settings of Table 7. We observe that the optimum number of filtering iterations for our network is 4. Variants containing higher numbers of ItMs appear to over-specialize on the training noise scales and perform sub-optimally at higher, unseen noise scales. Moreover, the DPFN network, with 1 ItM comprising 6 Dynamic EdgeConv layers as opposed to 4 ItMs with 4 Dynamic EdgeConv layers each, performs poorly with increasing noise. This result demonstrates the importance of multiple ItMs and *true* iterative filtering, as opposed to a deep, 1 iteration, network with the same number of parameters.

C. Ablation Study on Patch Stitching

During inference, given a noisy point cloud, we construct patches of 1000 nearest neighbors from a set of reference points $\{\boldsymbol{x}_r\}_{r=1}^{R}$ determined using farthest point sam-

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	36.17	15.9	78.88	47.75	104.2	69.69	18.55	12.8	50.37	41.38	72.32	62.0
Bilateral	30.92	14.98	52.56	27.77	74.19	45.08	17.08	13.99	18.99	14.52	27.72	22.07
Jet	31.57	13.54	55.39	31.59	60.89	35.89	7.87	4.38	13.84	9.0	17.21	11.74
WLOP	60.02	29.87	92.14	56.69	108.78	70.55	9.67	6.34	20.53	15.13	29.59	22.68
PCN	36.12	15.86	78.74	47.6	104.04	69.53	10.82	6.37	20.49	14.42	34.17	26.98
GPDNet	22.71	7.17	43.34	19.24	59.25	31.68	10.54	6.46	33.09	25.38	51.0	41.67
DMRDenoise	47.8	22.76	50.96	25.26	53.47	27.62	12.05	7.6	14.63	9.9	17.57	12.48
PDFlow	21.03	6.78	32.93	13.76	37.12	17.81	6.53	4.19	13.12	9.6	20.56	15.97
ScoreDenoise	24.79	7.54	37.01	14.09	42.41	19.1	7.11	4.0	13.19	8.62	14.92	9.96
Pointfilter	23.99	7.2	35.29	11.84	41.57	15.71	7.57	4.36	9.64	5.6	12.4	7.51
Ours	**19.94**	**4.94**	**30.3**	**8.6**	**34.39**	**11.23**	**6.02**	**3.05**	**8.3**	**4.6**	**11.52**	**6.99**

Table 3. Filtering 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	11.6	5.87	30.08	12.73	36.99	17.35	6.71	4.44	13.76	10.31	18.36	14.56
Bilateral	19.6	15.33	27.84	15.08	28.78	14.56	15.61	14.38	15.85	14.09	15.74	13.74
Jet	17.82	10.46	38.94	22.82	41.06	23.74	4.57	3.22	8.13	5.67	8.94	6.22
WLOP	59.72	29.85	66.12	35.0	68.2	36.57	8.28	5.31	10.11	6.77	11.32	7.73
PCN	11.64	5.89	30.06	12.71	36.95	17.31	6.21	4.14	8.86	5.95	10.26	7.05
GPDNet	7.88	4.33	19.19	6.34	22.32	7.65	4.6	3.14	7.75	5.29	10.38	7.45
DMRDenoise	48.4	24.25	48.42	23.74	48.43	23.86	11.1	6.9	12.5	7.95	12.68	8.13
PDFlow	8.71	4.64	18.86	6.33	23.37	8.29	4.33	3.04	6.72	4.53	7.38	5.0
ScoreDenoise	12.44	5.32	21.82	7.07	28.81	11.72	4.48	2.91	6.16	3.87	7.32	4.62
Pointfilter	11.14	5.72	20.56	6.92	22.49	7.57	5.43	3.92	6.12	4.16	6.43	4.34
Ours	**6.41**	**3.67**	**16.49**	**4.66**	**18.72**	**5.21**	**3.47**	**2.53**	**4.26**	**2.8**	**4.62**	**2.99**

Table 4. Filtering 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	49.83	25.95	117.73	83.79	162.23	124.83	28.69	22.23	86.64	76.74	127.21	115.91
Bilateral	36.87	17.95	80.15	51.36	112.47	78.99	18.2	14.66	41.25	35.58	71.13	63.96
Jet	35.87	15.64	63.13	38.0	70.8	44.24	9.4	5.43	18.63	13.04	28.33	21.79
WLOP	60.94	31.19	103.49	69.38	126.41	88.96	11.73	7.98	32.5	25.96	52.69	44.16
PCN	49.75	25.87	117.6	83.64	162.1	124.66	13.71	8.89	47.62	40.94	86.11	78.13
GPDNet	28.61	10.13	63.79	36.67	96.58	65.43	15.33	10.53	58.14	49.37	93.35	82.92
DMRDenoise	48.45	23.48	55.12	28.84	59.61	32.72	12.6	8.12	17.62	12.62	23.45	17.98
PDFlow	25.38	8.76	43.19	22.05	51.87	30.12	8.21	5.52	22.76	17.89	45.08	38.08
ScoreDenoise	29.03	9.64	46.02	21.36	51.84	27.3	8.25	4.89	16.77	11.67	18.51	12.79
Pointfilter	27.73	8.61	42.5	16.73	54.66	25.55	8.27	4.85	12.44	7.59	17.54	11.69
Ours	**23.93**	**6.02**	**33.96**	**11.0**	**42.93**	**17.55**	**6.53**	**3.36**	**9.99**	**5.8**	**16.6**	**10.96**

Table 5. Filtering 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	11.89	6.21	35.01	14.17	44.54	19.62	7.96	4.7	16.93	10.99	22.45	15.55
Bilateral	19.92	15.42	31.4	14.78	33.86	14.46	16.49	14.41	17.25	14.14	17.01	13.6
Jet	18.15	10.55	42.27	22.72	46.06	24.38	5.6	3.28	9.56	5.8	10.42	6.4
WLOP	59.88	29.79	71.06	37.63	67.11	35.71	8.12	5.17	10.63	7.2	12.05	8.22
PCN	11.93	6.22	34.96	14.14	44.48	19.56	7.24	4.25	10.62	6.09	11.81	6.92
GPDNet	7.84	4.31	22.31	6.59	27.25	8.22	5.45	3.24	9.97	5.87	13.66	8.6
DMRDenoise	49.14	24.01	47.69	22.61	48.06	22.67	11.18	6.9	12.23	7.71	12.19	7.72
PDFlow	8.74	4.64	20.26	6.24	24.65	8.38	4.56	3.03	6.82	4.54	7.55	5.09
ScoreDenoise	12.74	5.35	24.67	7.01	31.4	11.68	5.05	2.88	6.91	3.78	8.02	4.74
Pointfilter	11.39	5.7	24.47	6.92	27.76	7.68	6.31	3.96	7.42	4.2	7.63	4.23
Ours	**6.45**	**3.68**	**20.09**	**4.74**	**24.16**	**5.45**	**4.43**	**2.54**	**5.99**	**2.98**	**6.42**	**3.24**

Table 6. Filtering results on the PUNet dataset with noise uniformly distributed within a 3D sphere of radius s. Here, s corresponds to the noise scale. CD and P2M distances are multiplied by 10^5.

Ablation: 10K points	2.75% noise		3% noise		3.25% noise	
	CD	P2M	CD	P2M	CD	P2M
$\mathcal{L}_a$ & 1 it.	40.76	15.46	46.33	19.62	53.55	25.36
$\mathcal{L}_a$ & 2 it.	37.53	13.2	43.91	17.82	52.8	24.72
$\mathcal{L}_a$ & 4 it.	**36.31**	**12.3**	**41.87**	**16.44**	**51.2**	**23.6**
$\mathcal{L}_a$ & 8 it.	37.44	13.15	45.39	19.1	56.65	27.97
$\mathcal{L}_a$ & 12 it.	37.95	13.59	45.14	18.86	55.17	26.79
$\mathcal{L}_a$ & DPFN	39.49	14.48	47.06	20.11	57.35	28.25
$\mathcal{L}_b$ & 4 it.	37.62	13.2	44.22	18.13	54.36	25.93

Table 7. Ablation results for different iteration numbers and loss functions at higher noise. CD and P2M distances are multiplied by 10^5.

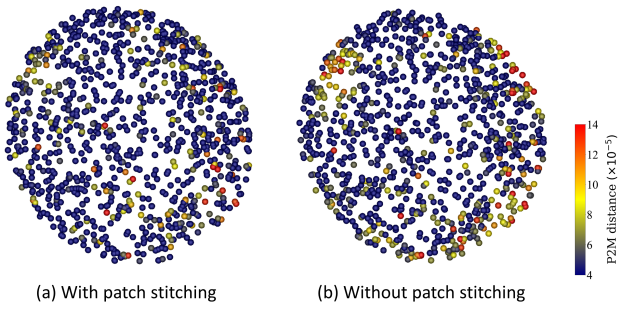

Figure 3. Patch (a) illustrates a noisy patch filtered with patch stitching and patch (b) corresponds to the same noisy patch filtered without patch stitching. The point-wise P2M distance is given for each point. As shown by patch (b), we observe that boundary points of patches tend to have higher P2M distances, i.e., filtered results away from the center are less accurate, when filtering without patch stitching. However, results closer to the center have higher accuracy. Therefore, patch stitching attempts to recover the best filtered points from overlapping patches and reduce the filtering error at boundary points. This can be seen in patch (a), where stitching is incorporated. We observe more accurate filtering results near the boundary.

Ablation	10K points					
	1% noise		2% noise		2.5% noise	
	CD	P2M	CD	P2M	CD	P2M
without PS	21.19	5.45	32.38	10.2	38.67	14.98
with PS	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**

Table 8. Ablation results with and without patch stitching (PS). CD and P2M distances are multiplied by 10^5.

pling. Typically, for a noisy point cloud of 50K points, we construct 300 such patches and filter each patch with all patch points being simultaneously filtered. As these 300 patches have overlapping regions, some points may be repeated amongst multiple patches which leads to some noisy points having multiple filtering results associated to them. Therefore, it is necessary to select the best filtered points that correspond to the best filtering result for each point amongst all patches. To achieve this, inspired by Zhou et al. [9], we design a generalized patch stitching mechanism to utilize Gaussian weights when selecting best filtered points. These weights are designed such that noisy points closer to a patch's central reference point, $\boldsymbol{x}_r$, are weighted higher than points further away from it. This is motivated by the observation that points at the patch boundary are filtered less favorably compared to those close to the central point. During inference, we recover the best filtered point, corresponding to a given initial noisy point, by selecting the filtered result within the patch where the noisy point was weighted highest.

The impact of incorporating patch stitching can be seen in Fig. 3. Here, patch (b) is filtered without patch stitching. We see that many boundary points have high point-wise P2M values, indicating these filtered results lie further away from the clean surface. By contrast, patch (a), filtered

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
Ours (PN++)	22.9	5.93	33.07	10.28	36.92	13.06	6.18	3.09	9.23	5.22	12.1	7.47
Ours (PN++ w/ GRU)	22.81	5.81	33.15	10.31	37.08	13.17	6.15	3.07	9.19	5.2	12.04	7.44
Ours	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**	**6.05**	**3.02**	**8.03**	**4.36**	**10.15**	**5.88**

Table 9. Filtering results on the PUNet dataset for IterativePFNs with Pointnet++ based encoders as compared to the original graph convolution based implementation. CD and P2M distances are multiplied by 10^5.

using patch stitching, has fewer boundary points with high P2M values. Table 8 shows that patch stitching leads to better overall filtering, especially at high noise scales.

D. Filtering Results for PointNet++ based Encoders

We are motivated that filtering is the reverse Markov process that iteratively removes noise. To obtain the filtered point $\boldsymbol{x}_i^\tau$, we only need the filtered point from the previous iteration, i.e., $\boldsymbol{x}_i^{\tau-1}$. Hence, a simple Encoder-Decoder module (ItM) suffices to obtain filtered displacements. To test our hypothesis, and demonstrate the generalization of our method to other 3D point set convolution architectures, we create variant IterativePFNs with 4 ItMs composed of Pointnet++ based encoders:

- **PN++**: a vanilla Pointnet++ based encoder where the graph convolution of point cloud patches, of our original implementation, is replaced by direct point set convolution of these patches using a Pointnet++ achitecture.
- **PN++ w/ GRU**: similar to [8], we create a Pointnet++ based variant which incorporates GRU layers within the encoder to maintain an additional *memory* of feature information from the previous iteration.

Our original network, which uses a DGCNN-based encoder, performs better as shown in Table 9. Using an additional GRU layer within the encoder provides no meaningful performance gain as the results of the vanilla Pointnet++ implementation performs comparably to the PN++ w/ GRU network. This reinforces our hypothesis that filtering a point, $\boldsymbol{x}_i^\tau$, at the subsequent iteration, is a reverse Markov process which requires only the filtered point, $\boldsymbol{x}_i^{\tau-1}$, from the preceding iteration.

E. Runtimes for Learning based Methods

Finally, we compare runtimes across different learning based methods. We do not compare conventional methods as they usually require parameter tuning and, therefore, user interaction, to obtain best results. Furthermore, the bilateral filtering mechanism, and the upsampling step for WLOP, both require point normals to be computed separately. This process can be quite time-consuming. Our method is quite competitive, ranking third in runtime while also achieving best results on synthetic and scanned data. By contrast, DMRDenoise [6], which has the shortest runtime, generally performs poorly on the filtering task. ScoreDenoise [7], while performing relatively faster, still produces sub-optimal filtering results. The ScoreDenoise runtime mentioned in Table 10 corresponds to filtered results after the default number of 30 gradient ascent steps, i.e., 30 test time iterations. However, as filtered results retain significant amounts of noise, additional iterations will be required to further decrease this noise which inevitably contributes to higher runtimes. GPDNet, Pointfilter and PCN perform worst in terms of efficiency. In the case of Pointfilter and PCN, the one patch per central point filtering strategy contributes to these high runtimes since each input patch is used to filter a single central point. By contrast, our method filters all patch points simultaneously while performing iterative filtering internally and, coupled with the generalized patch stitching mechanism we introduce, yields the best filtering results with short runtimes.

Method	Time (s)
PCN	167.75
GPDNet	79.87
DMRDenoise	**11.08**
PDFlow	32.74
ScoreDenoise	18.7
Pointfilter	84.32
Ours	22.91

Table 10. Runtimes of different learning based methods for filtering a noisy point cloud of 50K points with 2% Gaussian noise.

References

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

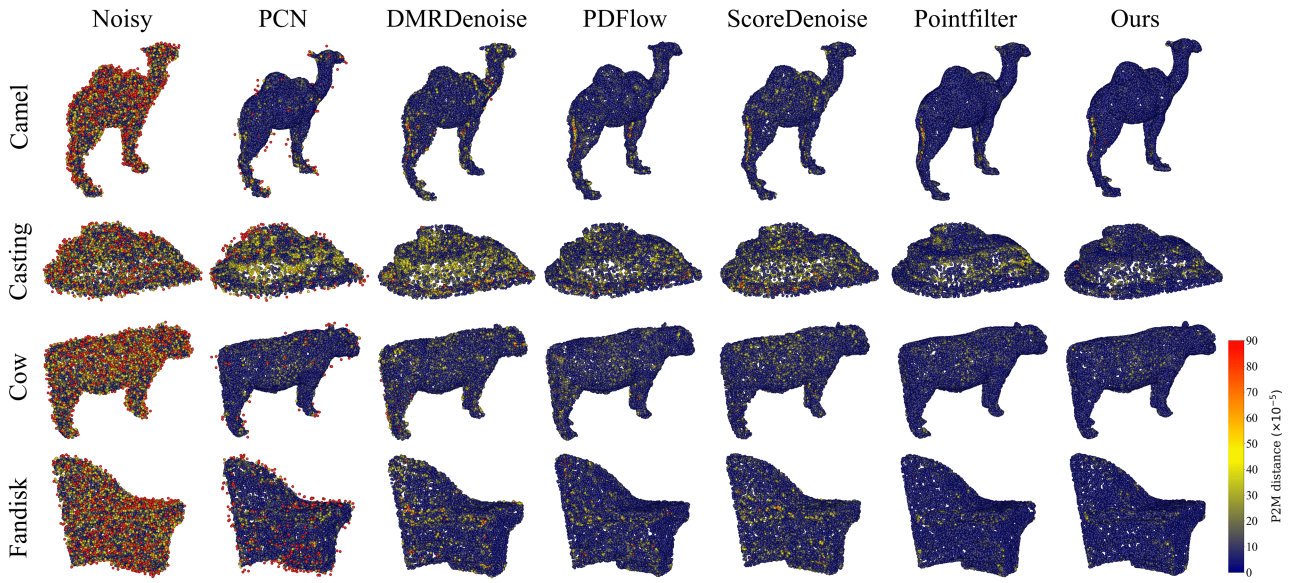

Figure 4. Visual results of point-wise P2M distance for 50K resolution shapes with non-isotropic Gaussian noise and a scale parameter of 2% of the bounding sphere radius.

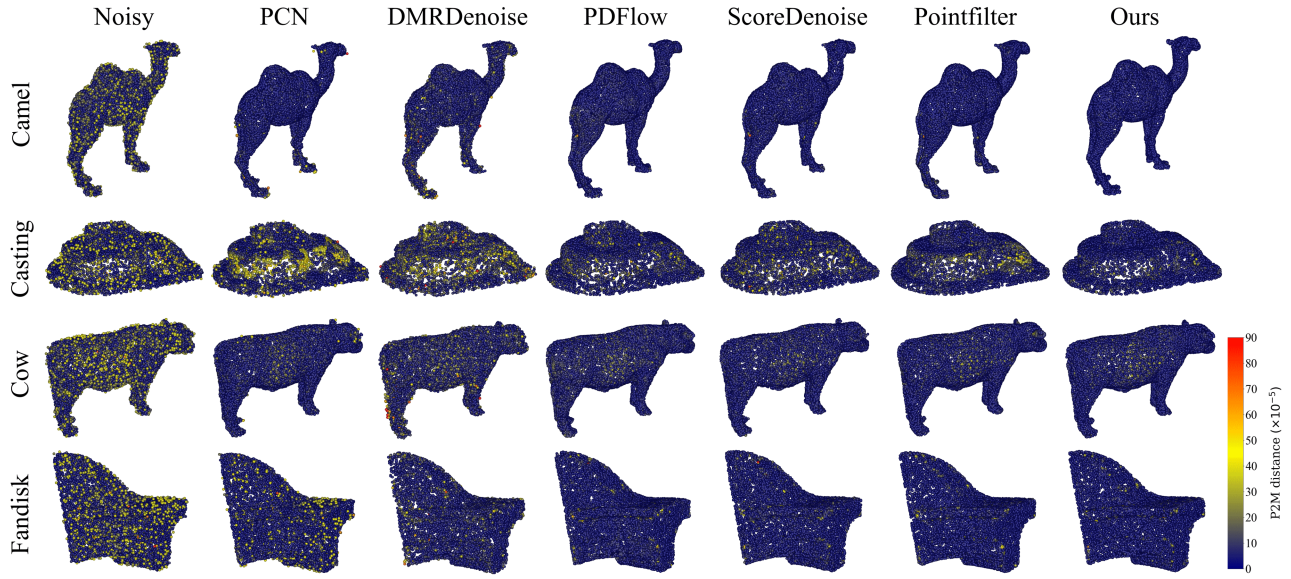

Figure 5. Visual results of point-wise P2M distance for 50K resolution shapes with discrete noise and a scale parameter of 2% of the bounding sphere radius.

[2] Julie Digne and C. D. Franchis. The bilateral filter for point clouds. *Image Process. Line*, 7:278–287, 2017. 1

[3] Hugues Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, 1992. 1

[4] Hui Huang, Dan Li, Hongxing Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM SIGGRAPH Asia 2009 papers*, 2009. 1

[5] Hui Huang, Shihao Wu, Minglun Gong, D. Cohen-Or, U. Ascher, and Hongxing Zhang. Edge-aware point set resampling. *ACM Transactions on Graphics (TOG)*, 32:1 – 12, 2013. 1

[6] Shitong Luo and Wei Hu. Differentiable manifold reconstruction for point cloud denoising. In *Proceedings of the 28th ACM International Conference on Multimedia*, page 1330–1338. Association for Computing Machinery, 2020. 6

[7] Shitong Luo and Wei Hu. Score-based point cloud denoising.

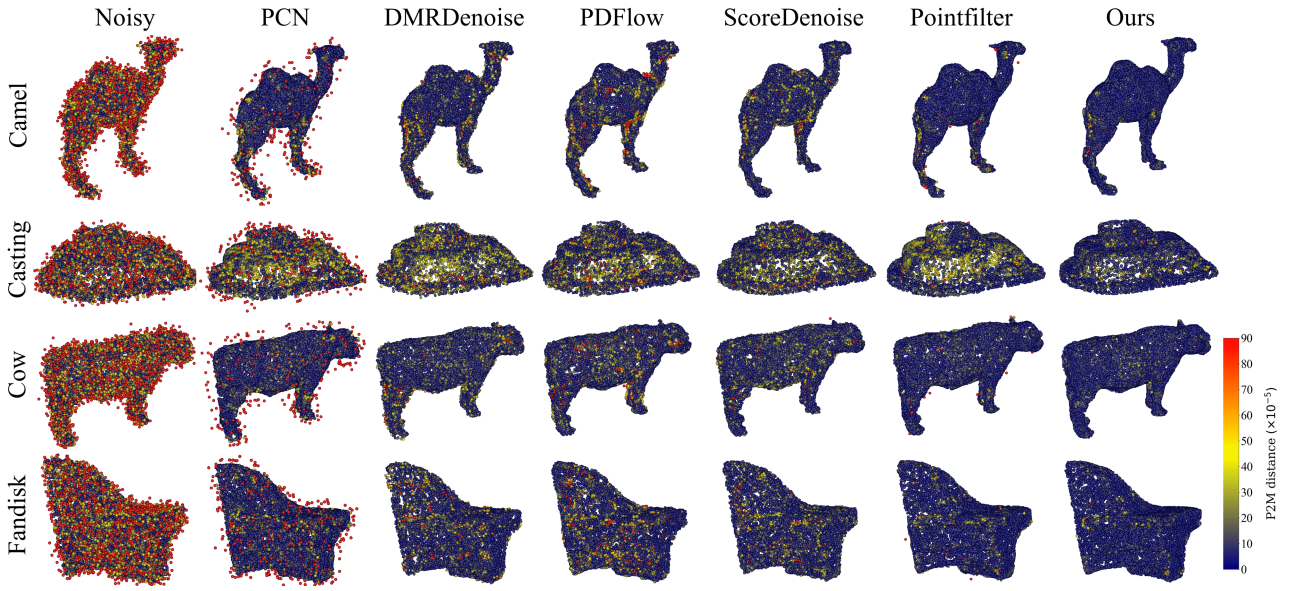

Figure 6. Visual results of point-wise P2M distance for 50K resolution shapes with Laplace noise and scale of 2% of the bounding sphere radius.

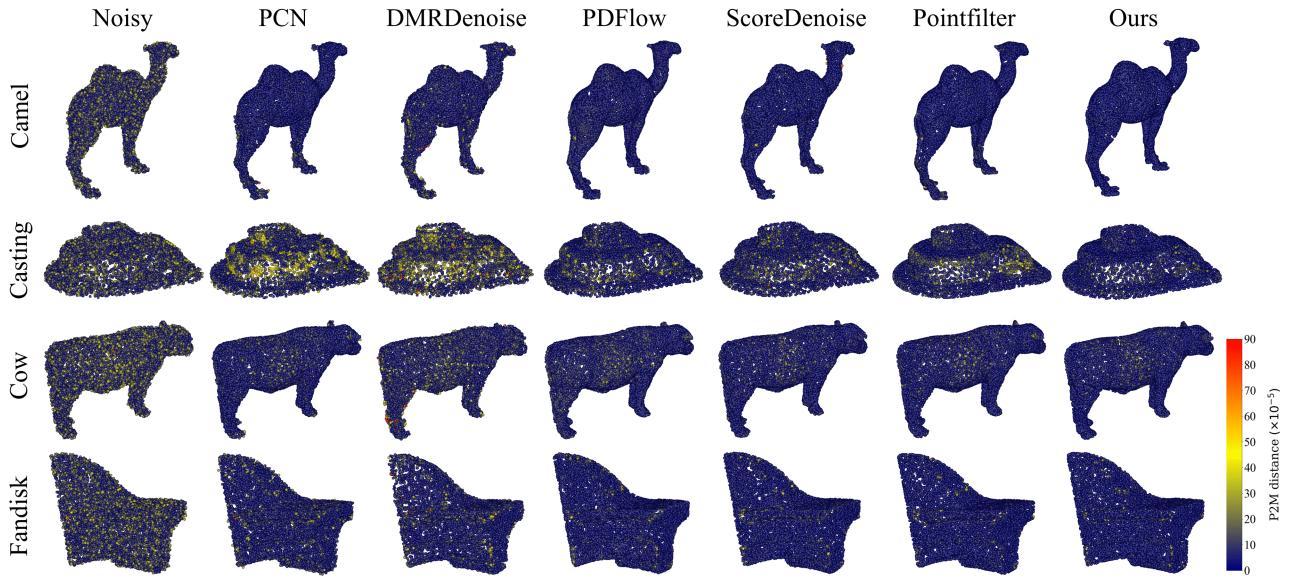

Figure 7. Visual results of point-wise P2M distance for 50K resolution shapes with noise uniformly distributed within a 3D sphere of radius s. Here, s corresponds to the noise scale and is equal to 2% of the bounding sphere radius.

In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4583–4592, October 2021. 3, 6

[8] X Wen, P Xiang, Z Han, Y Cao, P Wan, W Zheng, and Y Liu. Pmp-net: Point cloud completion by learning multi-step point moving paths. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7439–7448, 2021. 6

[9] Jun Zhou, Wei Jin, Mingjie Wang, Xiuping Liu, Zhiyang Li, and Zhaobin Liu. Fast and accurate normal estimation for point clouds via patch stitching. *Computer-Aided Design*, 142, 2022. 5