Shape Completion with Points in the Shadow (Supplementary Material)

BOWEN ZHANG, Xi'an Jiaotong University, China
XI ZHAO*, Xi'an Jiaotong University, China
HE WANG, University of Leeds, UK
RUIZHEN HU, Shenzhen University, China

CCS Concepts: • **Computing methodologies → Computer vision**.

ACM Reference Format:
Bowen Zhang, Xi Zhao, He Wang, and Ruizhen Hu. 2022. Shape Completion with Points in the Shadow (Supplementary Material). 1, 1 (September 2022), 15 pages. https://doi.org/10.1145/3550469.3555389

1 NETWORK DETAILS

We report the details of the network architecture in Table 1.

O-Predictor. The O-Predictor module has four main components: 1D Deconv, skip transformer(Trans), Encoder1 and MLP. For the 1D Deconv, we list the input channel, the output channel, the convolution kernel size, and the convolution stride. The Encoder1 includes five layers: SA_1, PT_1, SA_2, PT_2 and SA_3, where SA is a single scale grouping version of set abstraction(SA) layers from Pointnet++[Qi et al. 2017] and PT is point transformer(PT) block used in [Zhao et al. 2021]. The parameter S denotes the number of downsampled points, K denotes the number of nearest neighbors for each point and $\#MLP$ denotes the output channels for shared MLPs in the SA layer. For the skip transformer, we list the output channels of MLP_K, MLP_Q and MLP_V. K denotes the number of nearest neighbors for each ray. Note that in this transformer, the key and query are first calculated by MLP_K, MLP_Q, and then they are concatenated to be the input of MLP_V for the prediction of value. The audience is referred to [Xiang et al. 2021] for more details about the skip transformer. For the MLP in the O-Predictor, we list the size of its output channels.

O-Adjustment. There are two main components in O-Adjustment, which are Encoder2 and MLP. For the Encoder2, $\#MLP_1$ and $\#MLP_2$ denote the output channels of the shared MLPs used in the PCN encoder. Dim_g denotes the dimension of the initial global feature v, and Dim_v denotes the dimension of the final global feature g.

Refinement. The offset-constrained refinement contains 2 OCRUs. N_{in} denotes the number of input points, N_{out} denotes the number of output points.

*Corresponding author: Xi Zhao, School of Computer Science and Technology, Xi'an Jiaotong University (xi.zhao@mail.xjtu.edu.cn)

Authors' addresses: Bowen Zhang, Xi'an Jiaotong University, Xi'an, China, zhangbowen.wx@gmail.com; Xi Zhao, Xi'an Jiaotong University, Xi'an, China, zhaoxi.jade@gmail.com; He Wang, University of Leeds, UK, realcrane@gmail.com; Ruizhen Hu, Shenzhen University, Shenzhen, China, ruizhen.hu@gmail.com.

XXXX-XXXX/2022/9-ART $15.00
https://doi.org/10.1145/3550469.3555389

Table 1. The network architecture details.

O-Predictor	1D Deconv	In channel	Out channel	Kernel size	Stride
		512	128	2048	1
	Encoder1	Layer	S	K	#MLP
		SA_1	512	16	[64, 128]
		PT_1	-	16	-
		SA_2	128	16	[128, 256]
		PT_2	-	16	-
		SA_3	-	-	[512, 512]
	Skip Transformer	MLP_K	MLP_Q	MLP_V	K
		[256,128,256]	[256,128,256]	[128,256]	16
	MLP	Out channel [256,128,4]			
O-Adjustment	Encoder2	$\#MLP_1$	$\#MLP_2$	Dim_g	Dim_v
		[128, 256]	[512, 512]	256	512
	MLP	Out channel [256,128,4]			
Refinement		N_{in}	N_{out}		
	$OCRU_1$	2048	2048		
	$OCRU_2$	2048	16384		

2 TRAINING DETAILS

We use Adam optimizer for training the network. Our training process includes three steps. First, we train the initial completion module for 25 epochs with a batachsize of 32. The initial learning rate is 1e-3 and is decayed by 0.2 every 10 epochs. Second, we keep the parameters in the initial completion module unchanged, and train the refinement module for 15 epochs with a batchsize of 32. The initial learning rate setting is the same as the last training step. Third, we train the entire network for 5 epochs with the batchsize of 32. The leaning rate 1e-4 for this step.

3 MORE EXPERIMENTS

3.1 Results for real scan

We test our method on real partial scans from the KITTI dataset [Geiger et al. 2013], which is captured by LiDAR, so the partial scan data are very sparse and unevenly distributed. To apply our method to such data, we normalize the partial scan into a unit cube and then estimate the viewpoint configuration. Finally, we generate the completed point cloud based on the predicted viewpoint and our completion network trained from synthetic data. Some completion results are shown in Fig 1. We find that given input partial scan and viewpoint with very different distributions and features from our training data, our method can still complete the shape reasonably.

3.2 Evaluation and Comparison

The quantitative evaluation results for each category are reported in Table 2, Table 3 and Table 4.

From Table 2 we can see that our method achieves the best performance in terms of average CD and F-Score on the MVP dataset. Compared to SnowflakeNet, 12 out of 16 categories of our results have lower CD values. Although for the other four categories(car, cabinet, sofa and skateboard), our method has slightly higher CD, the visual quality of the results is quite similar. For categories with more complex geometries (such as lamp, chair and bookshelf), our method performs better than all other baseline methods.

Table 2. Evaluation on MVP dataset in term of Chamfer distance and F-Score. The values are [CD/F-score]

Method	PCN	MSN	ME-PCN	VRCNet	PoinTr	PMP-Net++	Snowflake-Net	Ours (with $View^{pred}$)	Ours (with $View^{GT}$)
Airplane	2.825/0.784	1.808/0.845	1.734/0.839	2.027/0.876	1.450/0.867	1.199/0.872	1.040/0.922	1.040/0.908	**0.841**/**0.926**
Bed	9.985/0.434	9.414/0.508	8.776/0.502	8.496/0.583	8.513/0.547	6.487/0.535	4.995/0.665	5.597/0.643	**4.698**/**0.666**
Bench	5.582/0.645	3.902/0.734	4.115/0.714	5.066/0.790	2.968/0.788	2.737/0.751	2.279/0.856	2.157/0.844	**1.826**/**0.864**
Bookshelf	9.144/0.492	7.834/0.577	6.557/0.559	7.659/0.671	6.702/0.657	4.831/0.588	4.041/0.738	4.377/0.729	**3.963**/**0.749**
Bus	3.120/0.754	2.729/0.755	3.212/0.692	2.788/0.782	2.925/0.789	3.047/0.662	2.081/**0.815**	2.175/0.796	**2.025**/0.811
Cabinet	4.562/0.572	4.442/0.607	4.998/0.567	4.106/0.683	4.187/0.650	5.283/0.522	**3.124**/0.705	3.720/0.689	3.387/**0.706**
Car	3.666/0.631	3.430/0.642	3.651/0.616	3.343/0.701	3.136/0.676	3.757/0.563	**2.449**/**0.727**	2.713/0.701	2.568/0.718
Chair	9.044/0.487	6.829/0.599	6.708/0.591	6.234/0.679	4.946/0.653	4.341/0.649	3.628/0.756	3.571/0.736	**3.358**/**0.758**
Guitar	1.391/0.891	0.925/0.912	1.028/0.895	0.998/0.934	0.822/0.929	0.696/0.926	0.620/0.951	0.528/0.957	**0.457**/**0.966**
Lamp	14.41/0.472	10.71/0.626	8.960/0.630	12.31/0.688	7.524/0.646	3.209/0.771	4.496/0.812	2.733/0.817	**2.441**/**0.839**
Motorbike	3.995/0.644	2.926/0.699	3.210/0.676	2.521/0.797	2.293/0.748	2.200/0.698	1.711/0.816	1.673/0.809	**1.627**/**0.818**
Pistol	3.131/0.724	2.421/0.778	2.264/0.782	2.144/0.845	1.775/0.826	1.138/0.867	0.967/0.915	0.888/0.913	**0.831**/**0.922**
Skateboard	1.169/0.875	1.175/0.862	1.459/0.850	0.911/0.914	0.983/0.902	1.048/0.868	**0.687**/0.937	0.827/0.920	0.706/**0.939**
Sofa	5.851/0.535	5.203/0.593	5.126/0.558	5.024/0.643	4.395/0.633	4.361/0.561	**3.139**/**0.710**	3.214/0.681	3.208/0.702
Table	5.962/0.631	4.685/0.707	5.076/0.677	4.876/0.764	4.318/0.709	4.195/0.684	3.191/0.807	3.165/0.791	**2.915**/**0.819**
Watercraft	4.183/0.653	3.112/0.713	3.456/0.685	2.840/0.787	2.454/0.749	2.327/0.732	1.895/**0.825**	1.879/0.806	**1.704**/0.819
Average	5.907/0.617	4.749/0.682	4.680/0.662	4.780/0.741	3.882/0.715	3.381/0.687	2.696/0.796	2.635/0.781	**2.419**/**0.800**

Table 3. Evaluation on MVP dataset in term of SCD_1 and SCD_2 (lower the better)

Method	PCN	MSN	ME-PCN	VRCNet	PoinTr	PMP-Net++	Snowflake-Net	Ours (with $View^{pred}$)	Ours (with $View^{GT}$)
Airplane	3.372/2.825	1.033/3.970	0.996/3.320	1.315/2.880	0.818/3.017	0.575/2.883	0.489/2.119	0.325/2.388	**0.303**/**1.646**
Bed	9.775/10.76	3.509/13.78	3.156/13.00	3.828/11.05	3.380/11.81	2.147/11.57	1.421/7.952	1.173/8.853	**1.042**/**7.052**
Bench	5.714/6.066	2.027/7.772	2.197/7.725	2.824/7.116	1.449/6.414	1.155/8.472	1.057/4.108	0.780/4.242	**0.660**/**3.179**
Bookshelf	8.711/9.911	3.670/11.67	2.722/9.626	3.303/10.02	2.385/9.376	2.578/9.222	1.327/6.546	1.124/6.894	**0.825**/**5.981**
Bus	2.220/3.668	1.129/3.879	1.335/4.562	1.404/3.851	1.274/3.877	1.541/5.260	**0.663**/3.361	0.748/3.131	0.680/**2.824**
Cabinet	3.839/5.255	1.540/6.253	1.647/7.012	1.504/5.314	1.466/5.421	2.159/8.710	**0.905**/4.516	0.989/5.056	0.972/**4.491**
Car	3.459/3.909	1.509/4.519	1.504/4.867	1.540/4.265	1.459/3.932	1.699/6.155	**0.811**/3.687	0.856/3.803	0.875/**3.463**
Chair	8.941/9.860	2.886/11.89	2.824/11.48	2.824/8.952	1.861/9.932	1.311/10.24	1.123/6.564	0.771/6.661	**0.755**/**5.684**
Guitar	1.579/1.394	0.566/1.680	0.587/1.688	0.771/1.258	0.570/1.181	0.719/1.253	0.337/1.204	0.254/0.942	**0.229**/**0.764**
Lamp	14.33/14.35	6.658/18.38	5.263/14.34	8.693/14.73	4.366/15.57	1.137/9.321	2.295/8.700	0.593/6.393	**0.531**/**5.145**
Motorbike	5.005/3.797	1.588/4.475	1.650/4.861	1.496/3.212	1.317/3.212	1.212/4.011	0.903/2.591	**0.610**/2.777	0.656/**2.473**
Pistol	3.667/3.067	1.336/3.936	1.102/3.632	1.128/2.827	1.010/2.988	0.889/2.081	0.433/1.725	0.323/1.597	**0.304**/**1.394**
Skateboard	1.232/1.488	0.608/2.797	0.741/2.827	0.492/1.695	0.599/1.638	0.835/2.710	0.352/1.467	0.349/1.939	**0.321**/**1.302**
Sofa	5.059/6.373	1.934/7.471	1.764/7.517	2.110/6.629	1.610/5.902	1.636/7.768	0.750/5.170	**0.702**/5.201	0.758/**4.803**
Table	6.652/7.155	1.691/8.764	2.055/8.681	1.994/7.238	1.505/10.01	1.544/9.188	0.984/5.848	0.742/5.613	**0.668**/**4.936**
Watercraft	4.251/4.328	1.523/4.825	1.706/5.355	1.351/4.052	1.147/3.818	1.090/5.054	0.620/3.695	0.468/3.720	**0.452**/**3.071**
Average	5.863/6.322	2.210/7.757	2.086/7.341	2.471/6.350	1.709/6.664	1.391/6.954	0.951/4.683	0.678/4.590	**0.646**/**3.896**

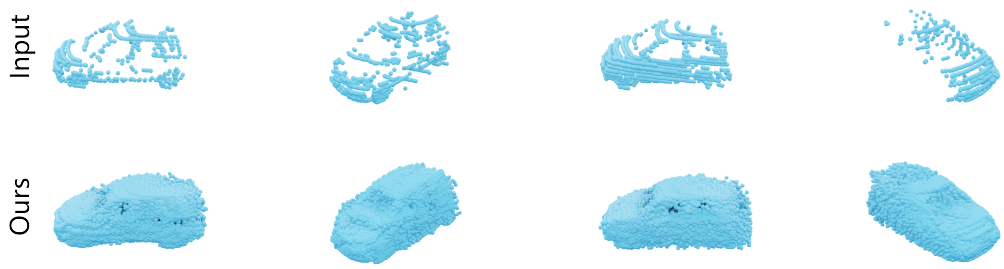

Fig. 1. Completion results on KITTI dataset.

Table 4. Evaluation on MVP dataset in term of *DCD* (lower the better)

Method	PCN	MSN	ME-PCN	VRC-Net	PoinTr	PMP-Net++	Snowflake-Net	Ours (with $View^{pred}$)	Ours (with $View^{GT}$)
Airplane	0.569	0.614	0.641	0.498	0.606	0.666	0.492	0.515	**0.482**
Cabinet	0.644	0.638	0.652	0.534	0.58	0.719	0.537	0.543	**0.526**
Car	0.577	0.614	0.633	**0.523**	0.565	0.706	0.539	0.544	0.527
Chair	0.689	0.683	0.682	0.567	0.654	0.702	0.528	0.551	**0.522**
Lamp	0.730	0.720	0.719	0.620	0.721	0.707	0.573	0.577	**0.548**
Sofa	0.629	0.630	0.649	0.555	0.606	0.713	0.541	0.564	**0.534**
Table	0.629	0.628	0.646	0.507	0.603	0.685	0.481	0.498	**0.466**
Watercraft	0.622	0.649	0.657	0.541	0.628	0.698	0.544	0.560	**0.540**
Bed	0.691	0.685	0.689	0.600	0.661	0.721	0.565	0.586	**0.563**
Bench	0.647	0.651	0.653	0.530	0.605	0.690	0.491	0.506	**0.480**
Bookshelf	0.701	0.682	0.670	0.558	0.620	0.705	0.537	0.544	**0.519**
Bus	0.539	0.591	0.625	0.501	0.535	0.692	0.489	0.497	**0.482**
Guitar	0.522	0.614	0.643	0.486	0.574	0.659	0.478	0.478	**0.450**
Motorbike	0.601	0.643	0.665	**0.534**	0.595	0.703	0.554	0.559	0.542
Pistol	0.629	0.656	0.669	0.538	0.605	0.676	0.521	0.529	**0.512**
Skateboard	0.498	0.580	0.601	0.454	0.538	0.653	0.431	0.451	**0.424**
Average	0.628	0.645	0.658	0.539	0.613	0.696	0.524	0.538	**0.513**

In addition to CD and F-Score, we compute the SCD for all baseline methods on all categories. The results are shown in Table 3. Our method achieves the lowest average SCD_1 and SCD_2 across all categories in the MVP dataset. For some categories such as cabinet, car and sofa, SnowflakeNet has lower SCD_1 values but higher SCD_2 compare to our method. For most categories, our method has the lowest SCD_1 and SCD_2, which indicates our method can improve the evaluation values by producing better shapes for both observed and unobserved parts. We also report the detailed comparison results on DCD in Table 4. Our method has the lowest average DCD value, with the lowest DCD for 14 out of 16 categories. Although our DCD values for the car and motorbike categories are higher than VRC-Net, the visual quality of our results is better.

Table 5. Evaluation on Front2Back dataset in term of $CD - L_1$. (Category names are consistent with [Yao et al. 2020].)

Method	plane	bench	cabinet	car	chair	display	lamp	speaker	rifle	sofa	table	phone	boat	average
Front2Back	0.017	0.021	0.023	0.019	0.021	0.020	0.023	0.027	0.015	0.023	0.019	0.017	0.022	0.0206
Ours	0.009	0.014	0.020	0.012	0.016	0.026	0.011	0.025	0.004	0.014	0.014	0.011	0.008	0.0139

To show how much of the improvement comes from our method leveraging the extra information, we also list the multiple category evaluation values for our approach without a GT viewpoint. From Table 2 and Table 3 we can see that without the GT viewpoint information, the CD, F-Score, SCD_1 and SCD_2 are all getting slightly worse than our full method. However, these values are still better than that of all other baseline methods.

Finally, we compare our method with Front2Back [Yao et al. 2020], which has a similar key insight. The front2Back method uses an occlusion mask from the input and predicts the points on "the other side". It uses depth and normal estimation images as input. As converting our input to their format is not straightforward, we apply our method to their dataset (rather than using it to our data). The quantitative comparison results are shown in Table 5. Our method outperforms Front2Back in all categories. We think the reason is that using just two points along each view ray is insufficient for shapes with a concave area or inside substructures, which is quite common for 3D models. Our method can predict multiple points along each ray and generate better results.

3.3 Ablation Study on O-Predictor

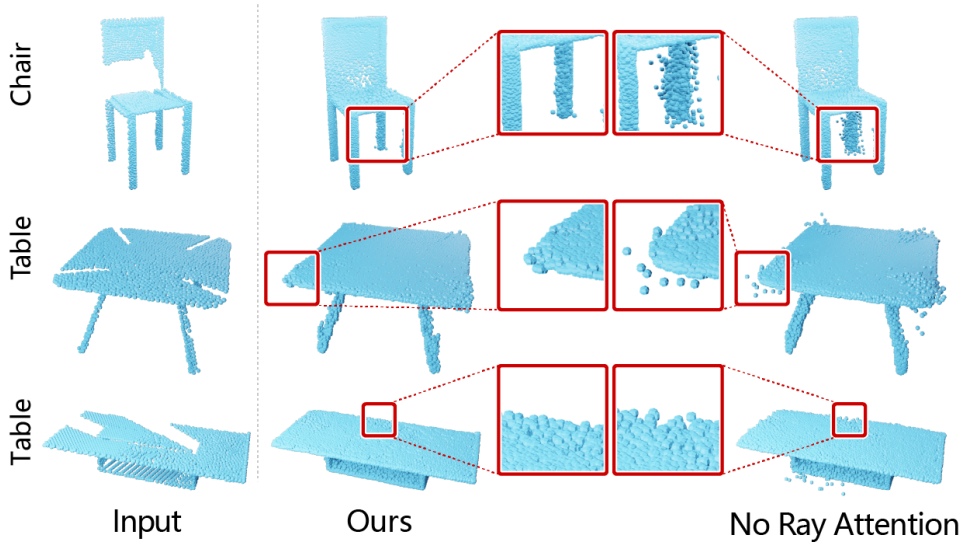

Fig. 2. Ablation study: compare our method with the version that uses point attention instead of ray attention.

In our method, we apply a skip transformer in the offset prediction step that encodes the ray orientation correlations (ray attention). Here we test another version that attention is based on the adjacency between input points corresponding to each ray (point attention), like the point transformer [Zhao et al. 2021]. We compare the result of this version to that of our method.

Table 6. Ablation study on ray attention

Method	*CD*	F-Score	*DCD*	SCD_1	SCD_2
No ray attention	2.421	0.793	0.528	0.696	3.989
Ours	**2.419**	**0.800**	**0.513**	**0.646**	**3.896**

Table 7. Ablation study on the effect of each feature in O-Adjustment Module

Method	*CD*	F-Score	*DCD*	SCD_1	SCD_2
No f_g^2	2.411	0.788	0.532	0.632	4.139
No f_g^1	**2.343**	0.788	0.532	**0.611**	4.025
No $[f_p^1, f_p^2]$	2.381	0.783	0.539	0.628	4.151
No R	2.427	0.787	0.532	0.715	3.909
No P	2.404	0.786	0.535	0.722	3.904
Ours	2.419	**0.800**	**0.513**	0.646	**3.896**

From the quantitative comparison results(Table 6), we can see that when using the point attention instead of ray attention, all evaluation values get slightly worse. The examples in Fig. 2 show that the result of our method with ray attention can produce more precise offsets, so the final results contain less noisy points. Without the ray attention, some points cannot reach the ideal locations and end up floating points around the object. This proves that with the ray attention strategy, the transformer can extract a more useful correlation information and improve the precision of the O-Predictor.

3.4 Ablation Study on O-Adjustment

In offset adjustment module we use feature f_p together with the coarse global feature f_g^2 to predict an adjustment for each offset. f_p is the concatenation of f_g^1, $[f_p^1, f_p^2]$, R and P. To justify the combination of these five features, we test five versions of methods where each of the features is eliminated. The experimental results are shown in Table 7.

We can see that without f_g^2 or $[f_p^1, f_p^2]$, although *CD* and SCD_1 decrease slightly, the SCD_2 increases by a large margin. Without f_g^1, the F-score, *DCD* and SCD_2 values get worse dramatically. When we remove the ray R or the input points P, the performance of methods also decrease.

3.5 Offset Loss

In our experiments, we only use Chamfer Distance as the loss function. However, it is straightforward to think about using an offset loss, which can provide a point-wise movement guidance. To calculate ground truth for offset loss, we find the intersection points between each ray and the original 3D model. The ground truth offset is then the distances from the first intersection point to each following intersection point along the ray. We calculate 1 dimensional Chamfer distance between the predicted offset distance set O and the ground truth offset distance set O^{gt}. The offset loss can be computed by:

$$CD(O, O_{gt}) = \sum_{\hat{d} \in O} \min_{d \in O_{gt}} ||\hat{d} - d||_2^2 + \sum_{d \in O_{gt}} \min_{\hat{d} \in O} ||d - \hat{d}||_2^2 \tag{1}$$

Table 8. The effect of offset loss

Method	CD	F-Score	DCD	SCD_1	SCD_2
With offset loss	2.424	0.797	0.517	0.681	3.905
Ours	**2.419**	**0.800**	**0.513**	**0.646**	**3.896**

For each ray, the number of predicted offsets in O is not necessarily equal to the number of ground truth offset distances in O^{gt}. To compare the predicted offsets to the ground truth offset, which may be of different lengths, we define the offset loss as the summation of 1-dimensional Chamfer distance between O and O^{gt} along with N rays. The loss can be described with the following equation:

$$L_{\text{offset}} = \sum_{i=1}^{N} CD(O_i, O_i^{gt}) \tag{2}$$

We then combine the offset loss with the CD loss for training. The total loss is designed as:

$$L = L_{CD} + \lambda L_{\text{offset}} \tag{3}$$

In experiments, we find it is hard for the training process to converge with a large λ value, and we set λ as 0.001. The quantitative evaluation values for the version with and without the offset loss (ours) are shown in table 8. As the performance is no improvement after using offset loss, we use L_{CD} in our method for simplicity.

3.6 Limitation

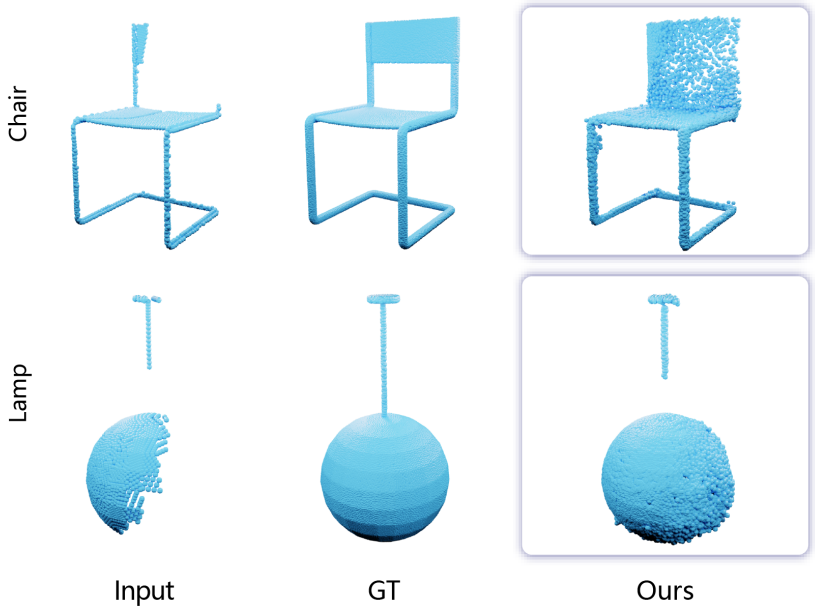

Fig. 3. The limitation of our method

Our experiments demonstrate that our method can retain the details captured in the partial scan, which is helpful for shapes with holes, concave area and thin structures. However, our method cannot generate correct completion results if the detailed structure is not captured. For example, in Figure 3, the chair results cannot recover the hole in the chair back, because the back is blocked by the seat. Therefore, maybe it is not reasonable to produce a unique completion result for a partial scan. How to consider the user's preference and produce multiple possible completion solutions is still an open problem.

In the lamp example, the lower part of the cable is missing because of the insufficient accuracy of the capturing device. It is difficult for our method to complete the missing cable as there are no completion rays of that missing area. The low accuracy of the input causes such confusion. Making the completion methods more robust to low-quality input data is also a problem worth exploring.

4 DETAILS ON VIEWPOINT PREDICTION

In Section 4.4 of our paper, we do an ablation study that assumes the viewpoint GT is not known. Because the viewpoint configuration is key for the completion process, we need to predict it when the GT information is not available. Although predicting viewpoint from the partial scan is not the focus of our paper, we provide a possible way to estimate viewpoint location from the partial point cloud, based on which our method can get satisfactory results. In the future, it would be quite helpful to find a more effective viewpoint prediction method, which can make our completion system rely only on the partial scan data and become more general.

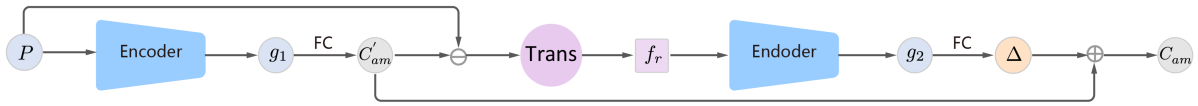

Fig. 4. Viewpoint prediction network

Viewpoint Prediction Network. We use a viewpoint prediction network similar to work in [Chen et al. 2021], which uses PointNet++ for global feature extraction, and feed the global feature to fully connected layers for the prediction of upright direction (a 3D vector). The structure of our viewpoint prediction network is illustrated in Fig. 4. The input is the partial point cloud P, and the predicted viewpoint is the output. We first use global feature g_1 to predict an initial viewpoint Cam', which is further improved by an adjustment to get the final predicted viewpoint position Cam. More specifically, we use PointNet++ to extract the global feature g_1 for the partial point cloud P. g_1 is then passed to fully connected layers to predict the initial viewpoint Cam'. As the results of one-step prediction are inaccurate to some extent, we use the camera rays computed by $P - Cam'$, to further estimate an adjustment. We use the skip transformer(Trans) and the PointNet++ encoder to compute the adjustment value Δ. The final result viewpoint is $Cam' + \Delta$.

Dataset for Training. We prepare partial scan and corresponding viewpoint pairs from the MVP dataset, where the viewpoint is represented by a 3D unit vector on the sphere. The dataset is split into a training set (62400 pairs) and a testing set (41600 pairs) in the same way as the completion dataset.

Training Details. We train the viewpoint prediction network with the Adam Optimizer. The initial learning rate is 1e-6, and is decayed by 0.2 every 25 epochs. The viewpoint prediction network takes 100 epochs to converge, and is then used as a black box for the completion system.

Result. The precision of viewpoint prediction results are listed in Table 9. In the second column of the table, we list the average angle error for each category. In the last two columns, we list the percentage of samples that with the angle error falls into the range $0° \sim 5°$ and $5° \sim 15°$. For each item in the table, we also list the results with and without the adjustment step. From the table, we can see that, after viewpoint adjustment, the average avg falls from 20.72 to 16.94, the overall Acc$[0° \sim 5°]$ rises from 7% to 12%, and the Acc$[5° \sim 15°]$ rise from

Table 9. Evaluation of viewpoint prediction. The former values and the latter values are the values corresponding to viewpoints before and after the adjustment. Avg represents average angle error. Acc[$a^\circ \sim b^\circ$] means the percentage of samples whose angle error falls in this range.

Category	Avg	Acc[$0^\circ \sim 5^\circ$]	Acc[$5^\circ \sim 15^\circ$]
Airplane	20.41/**15.98**	7%/**15%**	37%/**48%**
Cabinet	16.62/**14.84**	7%/**9%**	43%/**50%**
Car	11.81/**9.22**	16%/**27%**	58%/**61%**
Chair	22.64/**18.58**	6%/**9%**	33%/**45%**
Lamp	28.76/**23.96**	5%/**10%**	27%/**37%**
Sofa	16.22/**13.10**	8%/**12%**	44%/**56%**
Table	25.13/**21.16**	4%/**6%**	27%/**39%**
Watercraft	22.74/**18.95**	7%/**11%**	35%/**47%**
Bed	23.72/**20.39**	5%/**8%**	32%/**43%**
Bench	30.35/**25.72**	4%/**5%**	23%/**33%**
Bookshelf	23.32/**20.84**	5%/**7%**	37%/**42%**
Bus	14.94/**12.47**	10%/**17%**	50%/**55%**
Guitar	19.94/**14.39**	7%/**13%**	37%/**50%**
Motorbike	18.27/**11.98**	6%/**19%**	40%/**55%**
Pistol	18.61/**13.18**	7%/**13%**	38%/**53%**
Skateboard	20.89/**15.70**	5%/**14%**	34%/**46%**
All	20.72/**16.94**	7%/**12%**	38%/**48%**

38% to 48%. At least 60% of the prediction corresponds to an error less than 15°. It indicates that the adjustment steps effectively improve the prediction precision. To visually compare the performance of our method with the ablation version that uses the predicted viewpoint, we show some examples results with and without viewpoint ground truth in Fig.5.

5 MORE RESULTS

5.1 Completion Results Visualized with Two Parts

We show some completion results visualized in two parts compared to other methods.

5.2 More Completion Results

We visualize more completion results for each category in Fig 7, Fig 8, Fig 9 and Fig 10.

REFERENCES

Jiayi Chen, Yingda Yin, Tolga Birdal, Baoquan Chen, Leonidas Guibas, and He Wang. 2021. Projective Manifold Gradient Layer for Deep Rotation Regression. *arXiv preprint arXiv:2110.11657* (2021).

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. 2013. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research* 32, 11 (2013), 1231–1237.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html

Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. 2021. SnowflakeNet: Point Cloud Completion by Snowflake Point Deconvolution with Skip-Transformer. *arXiv:2108.04444 [cs]* (Oct. 2021). http://arxiv.org/abs/2108.04444 arXiv: 2108.04444.

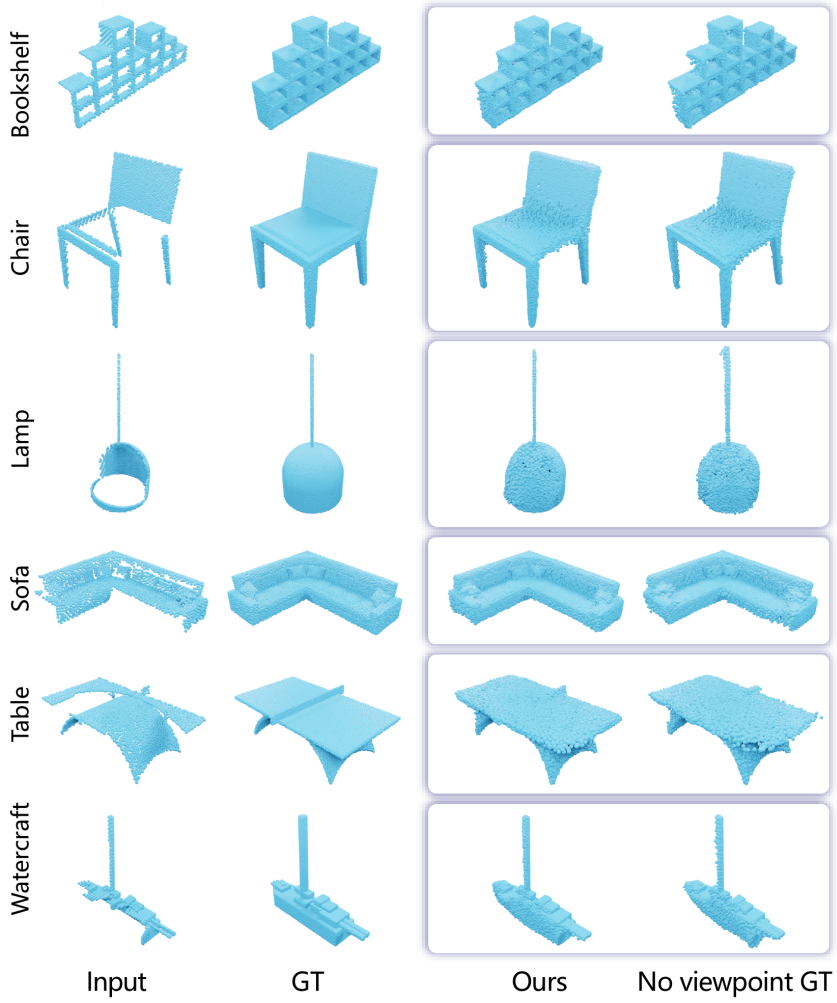

Fig. 5. predicting viewpoint comparison

Yuan Yao, Nico Schertler, Enrique Rosales, Helge Rhodin, Leonid Sigal, and Alla Sheffer. 2020. Front2back: Single view 3d shape reconstruction via front to back prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 531–540.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. 2021. Point Transformer. *arXiv:2012.09164 [cs]* (Sept. 2021). http://arxiv.org/abs/2012.09164 arXiv: 2012.09164.

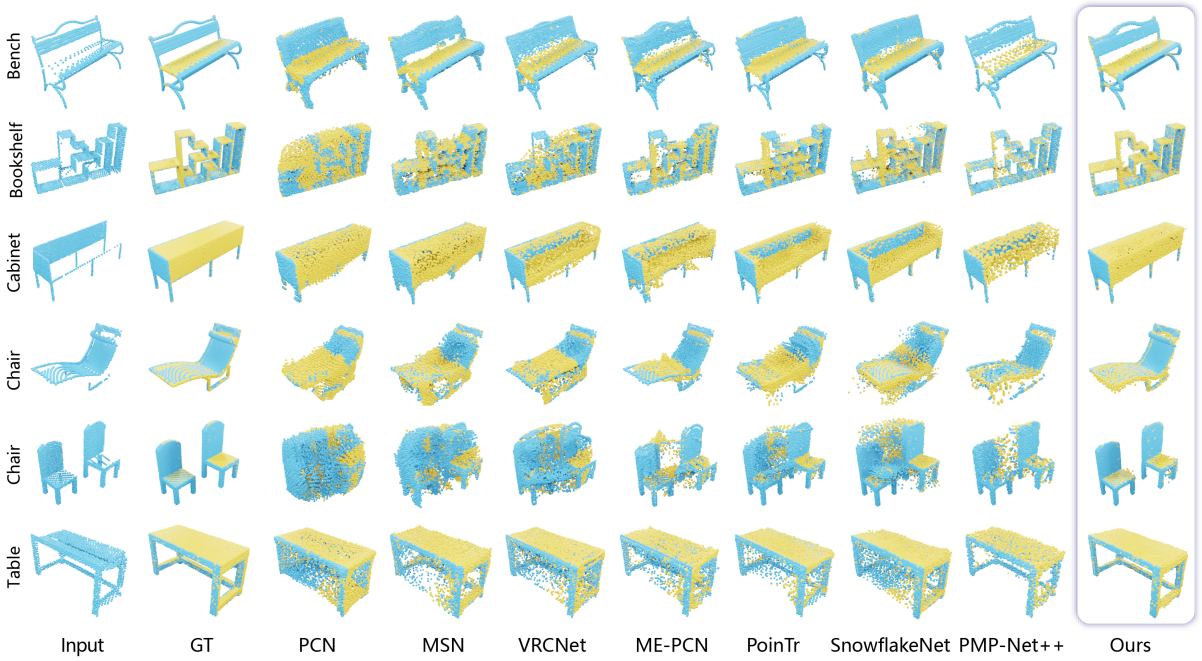

Fig. 6. The completion results are visualized with two parts for comparison. The blue points show the completion for observed part, while yellow points show the completion for unobserved part.

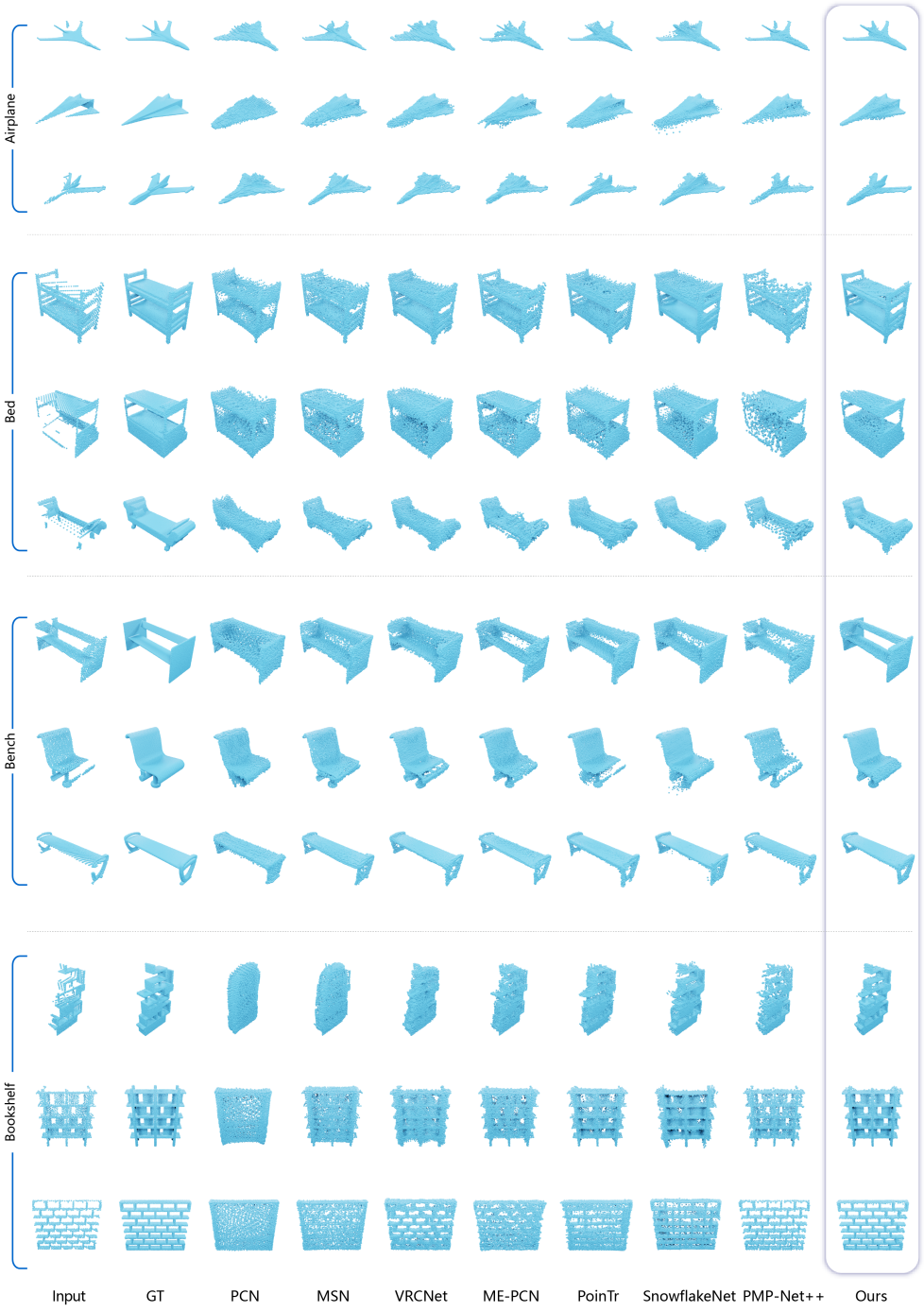

Fig. 7. More results part1.

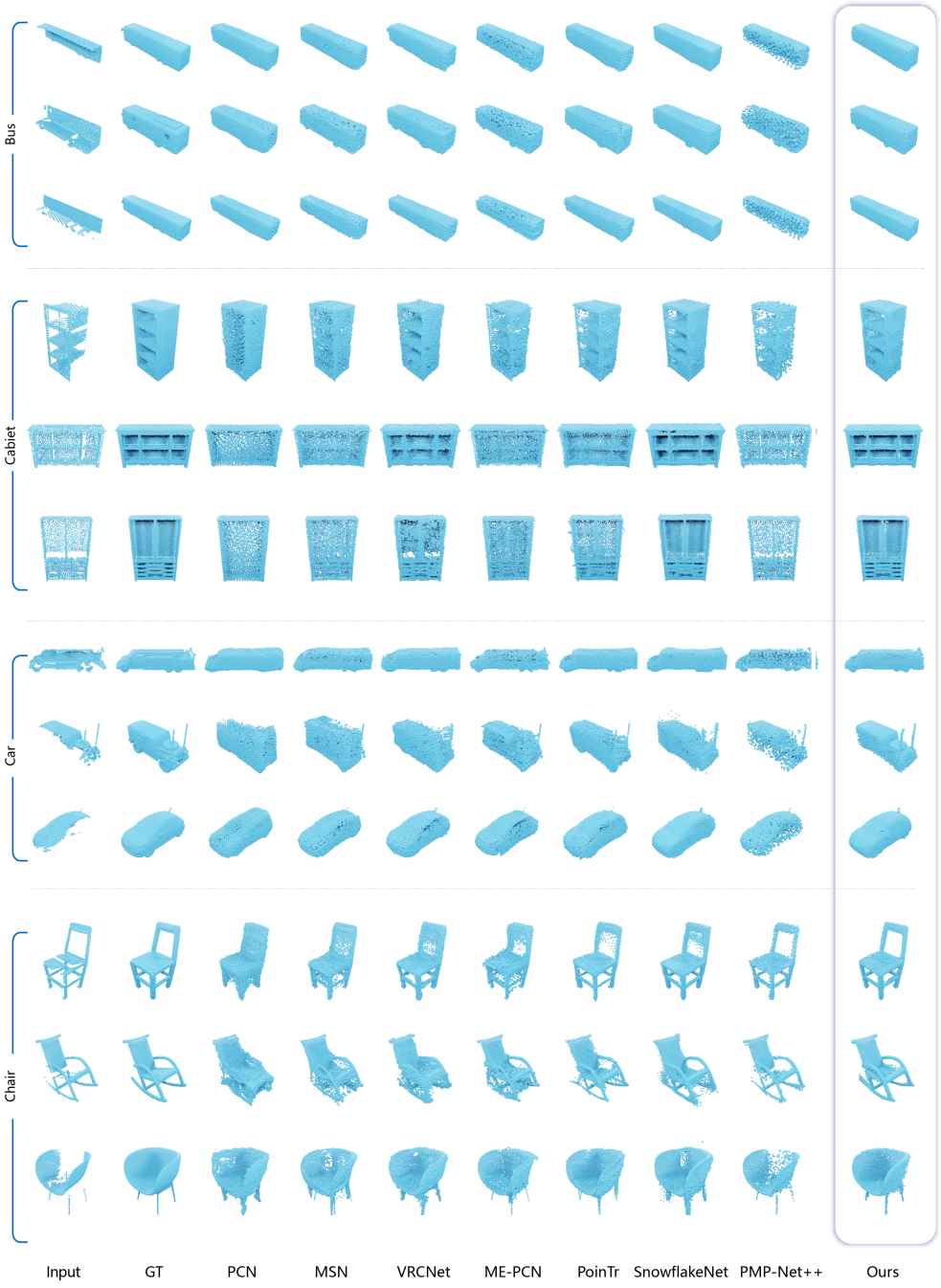

Fig. 8. More results part2.

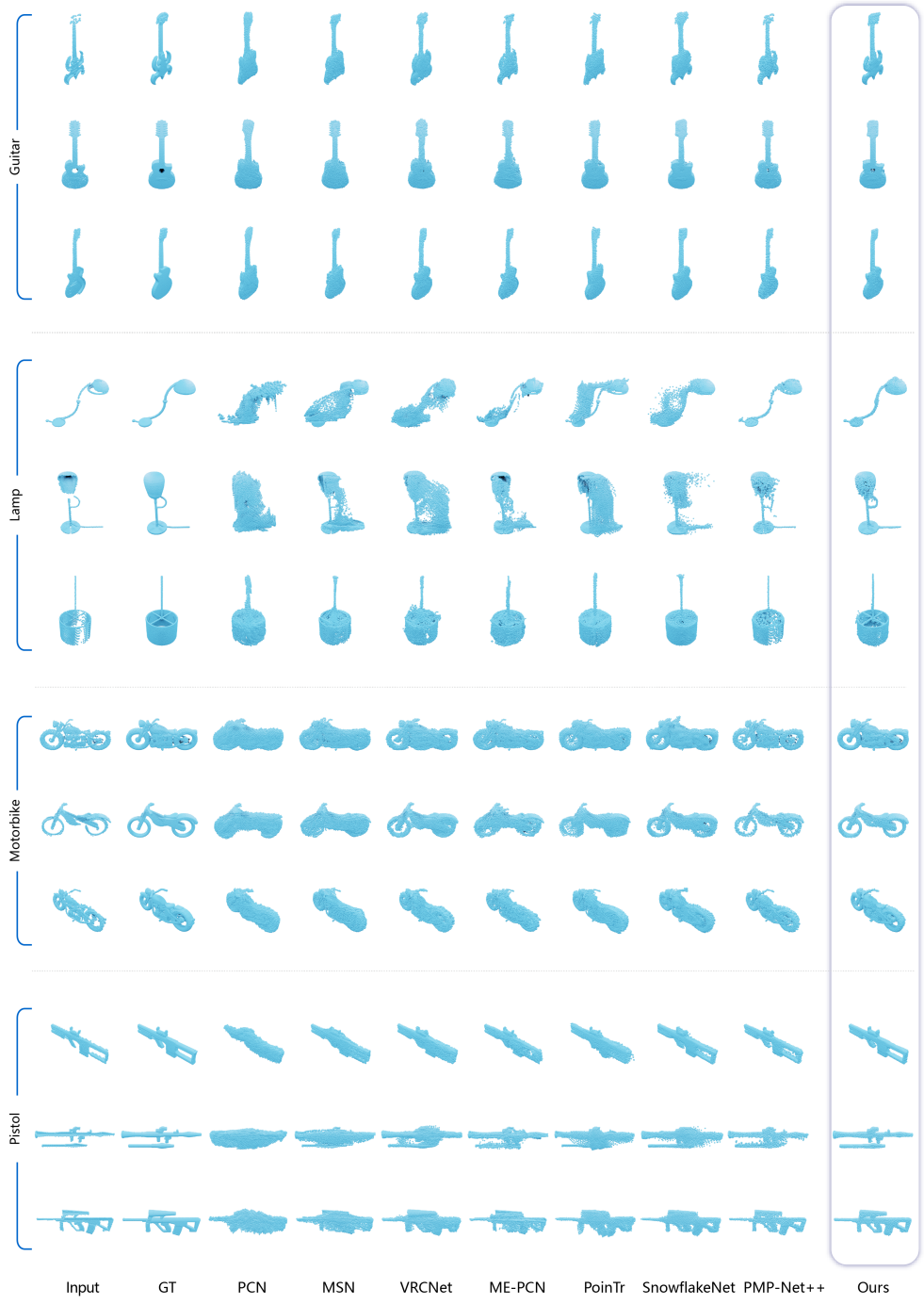

Fig. 9. More results part3.

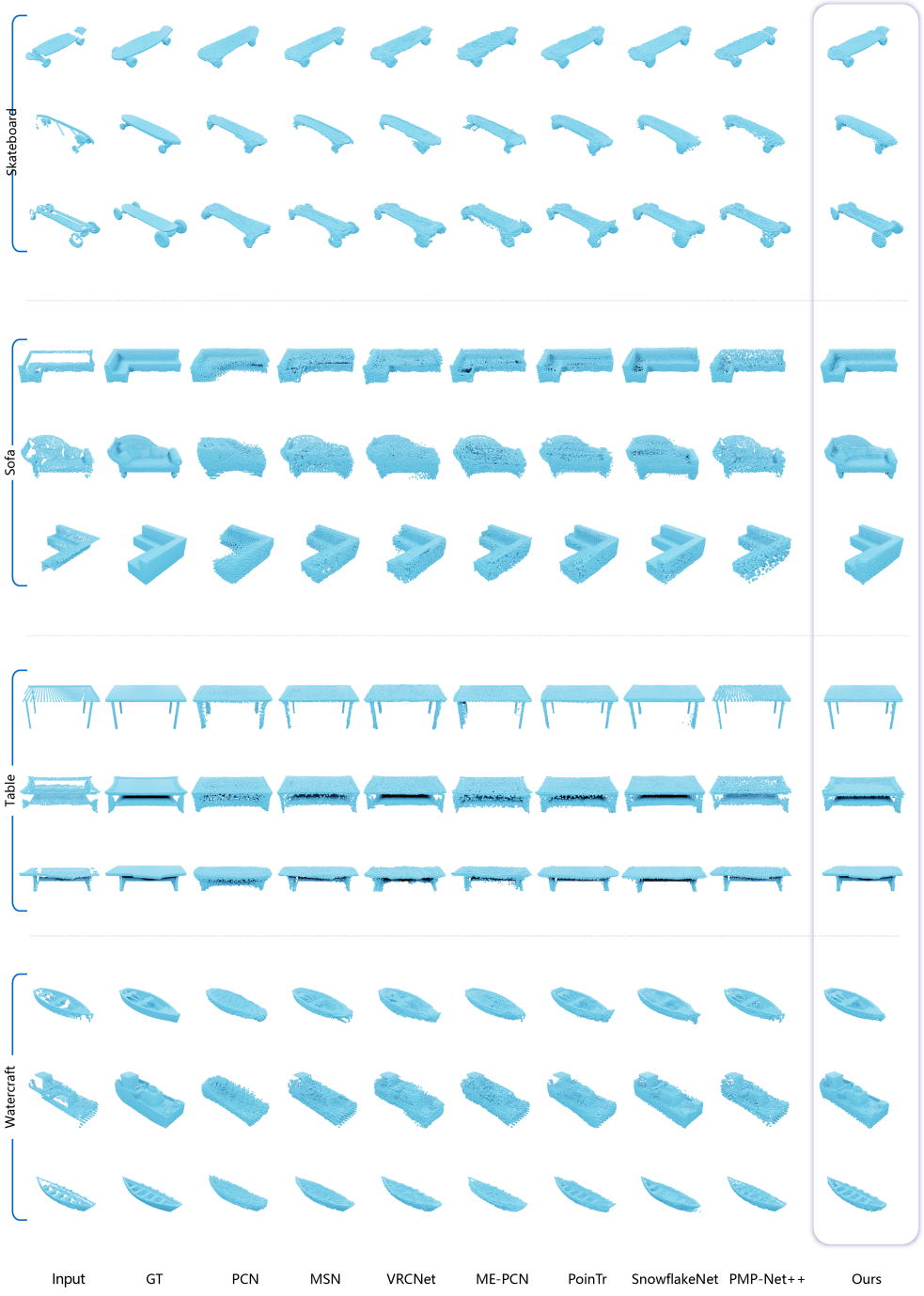

Fig. 10. More results part4.