Supplementary for "SnowflakeNet: Point Cloud Completion by Snowflake Point Deconvolution with Skip-Transformer"

Peng Xiang[1]*, Xin Wen[1,4]*, Yu-Shen Liu[1], Yan-Pei Cao[2], Pengfei Wan[2], Wen Zheng[2], Zhizhong Han[3]
[1]School of Software, BNRist, Tsinghua University, Beijing, China
[2]Y-tech, Kuaishou Technology, Beijing, China [3]Wayne State University [4]JD.com, Beijing, China
xp20@mails.tsinghua.edu.cn wenxin16@jd.com liuyushen@tsinghua.edu.cn
caoyanpei@gmail.com {wanpengfei,zhengwen}@kuaishou.com h312h@wayne.edu

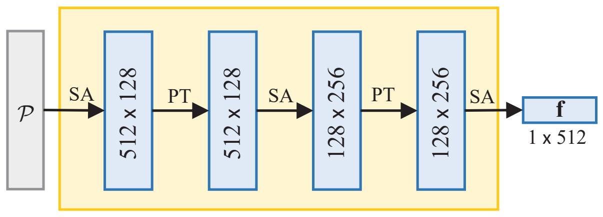

Figure 1. The structure of feature extractor, where "SA" indicates set abstraction and "PT" stands for point transformer.

1. Source Code

We release our source code as part of supplementary material.

2. Detailed Settings

2.1. Network Implementation Details

Feature extractor. As show in Figure 1, the feature extractor consists of three set abstraction (SA) modules (from PointNet++) and two point transformers [3]. The SA modules use k-nearest neighbors (kNN) as grouping operation, where we typically set $k = 16$ throughout the paper, including the kNN strategy in our skip-transformer.

Up-sampling factors. In two datasets (i.e. Completion3D and PCN) that have ground truth shapes with different point densities, the point splitting schemes (up-sampling factors) are different. For the Completion3D dataset, we set $r_1 = 1, r_2 = 2$ and $r_3 = 2$, such that $\mathcal{P}_1, \mathcal{P}_2$ and $\mathcal{P}_3$ have point number $N_1 = 512, N_2 = 1024$ and $N_3 = 2048$, respectively; as for the PCN dataset, we set $r_1 = 1, r_2 = 4$ and $r_3 = 8$, where $N_1 = 512, N_2 = 2048$ and $N_3 = 16384$.

*Equal contribution. This work was supported by National Key R&D Program of China (2020YFF0304100), the National Natural Science Foundation of China (62072268), and in part by Tsinghua-Kuaishou Institute of Future Media Data. The corresponding author is Yu-Shen Liu.

Incomplete shape. Each incomplete shape in Completion3D dataset has 2048 points. In the PCN dataset, the incomplete point clouds have different point numbers, for those point clouds with less than 2048 points, we increase the point number by randomly copying points; for those point clouds with more that 2048 points, we randomly select 2048 points as input.

Multilayer perceptron. The multilayer perceptron in this paper refers to both one-dimensional convolution layer and one-dimensional ResNet block [2].

2.2. Training Loss

Completion loss. The L1 version of Chamfer distance (CD) is defined as

$$\mathcal{L}_{\mathrm{CD}}(\mathcal{X}, \mathcal{Y}) = \sum_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{y}\in\mathcal{Y}} \|\mathbf{x}-\mathbf{y}\| + \sum_{\mathbf{y}\in\mathcal{Y}} \min_{\mathbf{x}\in\mathcal{X}} \|\mathbf{y}-\mathbf{x}\|, \tag{1}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are point sets, $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ are point coordinates, respectively. The L2 version of CD replaces L1-norm in Eq.(1) by L2-norm. Our completion loss is given as

$$\mathcal{L}_{completion} = \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_c, \mathcal{P}_c^{'}) + \sum_{i=1}^{3} \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_i, \mathcal{P}_i^{'}), \tag{2}$$

where $\mathcal{P}_i^{'}$ and $\mathcal{P}_c^{'}$ denote the down-sampled ground truth point clouds that have the same point number as point cloud $\mathcal{P}_i$ and $\mathcal{P}_c$, respectively.

Preservation loss. The *partial matching loss* is defined as

$$\mathcal{L}_{partial}(\mathcal{X}, \mathcal{Y}) = \sum_{\mathbf{x}\in\mathcal{X}} \min_{\mathbf{y}\in\mathcal{Y}} \|\mathbf{x}-\mathbf{y}\|, \tag{3}$$

which is an unidirectional constraint that only requires the output to partially match the input. Therefore, we use it as the preservation loss

$$\mathcal{L}_{preservation} = \mathcal{L}_{partial}(\mathcal{P}, \mathcal{P}_3), \tag{4}$$

where $\mathcal{P}$ is the input point cloud and $\mathcal{P}_3$ is the last output of our SnowflakeNet. And the total loss function is given as

$$\mathcal{L} = \mathcal{L}_{completion} + \lambda\mathcal{L}_{preservation}, \tag{5}$$

where we typically set $\lambda = 1$.

2.3. Training details

For the two datasets (i.e. Completion3D and PCN dataset), we use AdamOptimizer to train SnowflakeNet. It takes 200 steps to *warm-up* the learning rate from 0 to 10^{-3}, and then the learning rate is exponentially decayed by 0.5 for every 50 epochs.

PCN dataset. For each model in the PCN dataset (16384 points in ground truth), there are 8 incomplete shapes that are captured from 8 different views. In each training epoch, we randomly select a single view out of eight for each model. We use 4 NVDIA GTX 2080TI GPUs with a batch size of 80 to train SnowflakeNet, and it takes 300 epochs to converge.

Completion3D. For the Completion3D dataset (2048 points in ground truth), the training process is accomplished using a single NVDIA GTX 2080TI GPU with a batch size of 32. It takes 150 epochs to converge. And we scale all the training shapes of Completion3D by 0.9 to avoid points out of range of *tanh* activation.

3. More Results

3.1. The effect of splitting strategy

Table 1. Effect of splitting strategy.

Splitting strategy	avg.	Couch	Chair	Car	Lamp
one-step	9.53	6.03	11.0	9.94	9.26
two-steps	8.64	6.00	10.3	9.62	8.66
baseline	**8.48**	**5.89**	10.6	**9.32**	**8.12**
three-steps	8.66	6.04	**10.0**	9.53	9.02

The SPD is flexible for increasing the point number. When $r_i = 1$ (r_i is the up-sampling factor of the i-th SPD), it serves to move the point from previous step to a better position; when $r_i > 0$, it splits point by a factor of r_i. In this section, we analyze the influence of different point splitting strategies on the performance of SnowflakeNet, and the other experiment settings are same as ablation study. As shown in Table 1, we additionally test three different splitting strategies. The one-step splitting adopts a single SPD (up-sampling factor is 4) to split $\mathcal{P}_0$ (512 points) to $\mathcal{P}_1$ (2048 points). The two-steps splitting adopts two SPDs (up-sampling factors are both equal to 2) and produces three point clouds $\mathcal{P}_0, \mathcal{P}_1$ and $\mathcal{P}_2$ of the size $512 \times 3, 1024 \times 3$ and 2048×3, respectively. For the three-steps strategy (three SPDs of up-sampling factor 2), we particularly set $N_c = N_0 = 256$ (see Section 3.1) so that it outputs 4 point clouds of the size $256 \times 3, 512 \times 3, 1024 \times 3$ and 2048×3. Note that the baseline is two-steps splitting with an additional SPD (up-sampling factor equals to 1), which serves to rearrange the initial point positions. By comparing one-step splitting with the other strategies, we can find that multiple steps of splitting boosts the performance of SnowflakeNet significantly (in terms of average CD), this should credit to the collaboration between SPDs. By comparing two-steps splitting with three-steps splitting, we can find that the two-steps splitting suffices for generating a sparse point cloud with 2048 points (not necessarily for a dense one with more points), and extra SPDs may not improve the performance but will increase the computational burden. And by comparing the two-steps splitting with the baseline, we can find that the additional SPD which adjusts the initial point positions can facilitate the point splitting process.

3.2. Size of the model

In Table 2, we compare our SnowflakeNet (Ours) with PCN and GRNet in terms of parameter number and FLOPs on PCN dataset. The results demonstrate that SnowflakeNet is significantly smaller than GRNet in terms of parameter number, and it has the lowest computational complexity in terms of FLOPs.

Table 2. Model size of different methods.

Models	PCN	GRNet	Ours
params (M)	**6.86**	76.7	19.2
FLOPs (G)	14.7	25.9	**9.93**

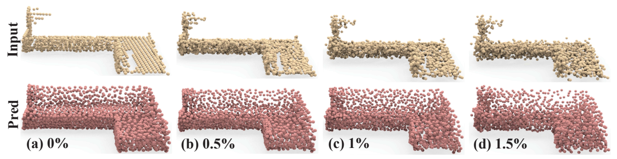

Figure 2. Completion results under increasing noise levels to input.

3.3. Sensitivity to random noise.

We evaluate the performance of our model under increasing Gaussian noise levels, quantitative and qualitative results are shown in Table 3 and Figure 2, respectively. The results indicate that our method performs well under small noise perturbation (0.5%). As noise level increases (1.0%, 1.5%), quantitative results will degrade due to large corruption of partial inputs (e.g. comparing the input in Fig 2(a) with Fig 2(d)), while qualitative results are still visually plausible. Note that we use the pre-trained model from ablation study (without fine tuning) for evaluation, which has not been trained on any noisy inputs.

Table 3. Sensitivity to noise.

Noise levels	avg	Couch	Chair	Car	Lamp
0%	**8.48**	**5.89**	**10.6**	9.32	**8.12**
0.5%	9.01	9.59	11.1	**6.28**	9.04
1.0%	12.1	13.7	11.7	8.13	14.8
1.5%	15.6	17.0	20.2	9.83	15.3

3.4. Effect Justification of Skip-Transformer

In Figure 3, we visualize the point splitting process of the four network variations, as illustrated in Section 4.2 (Effect of skip-transformer). The experiment is conducted on the validation set of Completion3D, so that $\mathcal{P}_1, \mathcal{P}_2$ and $\mathcal{P}_3$ have the point number $N_1 = 512, N_2 = 1024$ and $N_3 = 2048$, respectively. By comparing *Self-att* , *Full* with *No-connect* and *No-att*, we can find that the attention mechanism enables the SPD to capture local structure, thus result in more diverse splitting patterns; by contrast, the splitting patterns of *No-connect* and *No-att* are monotonous, and ignore to comply with local pattern (the columns of the chair back). And by comparing *Full* with *Self-att*, we can find that consecutive point splitting processes in the *Full* model are more consistent and tend to work in an incremental manner. Although $\mathcal{P}_1$ in *Full* also has noise in the back region, after two steps of point splitting, the *Full* model can reduce most of the noise, this should credit to the skip-transformer.

3.5. More visualization results

The ablation study and experiment in supplementary (Section 3.1) are conducted on the four categories (i.e. couch, chair, car and lamp) of validation set in the Completion3D dataset. To prove the generalization ability of our SnowflakeNet, we use the pre-trained model in ablation study to complete chairs in the ScanNet [1] dataset (without fine-tuning). Both the input and prediction shapes have 2048 points, where we up-sample the input point cloud by randomly copying points. As shown in Figure 4 and Figure 5, even for the shapes that are highly noisy and incomplete, our SnowflakeNet is able to predict a complete point cloud with less noise. In Figure 6, we present more snowflake point deconvolution for objects under each category. In Figure 7 and Figure 8, we provide more shape completion results on both the PCN and Completion3D datasets under each category.

References

[1] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 3

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[3] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020. 1

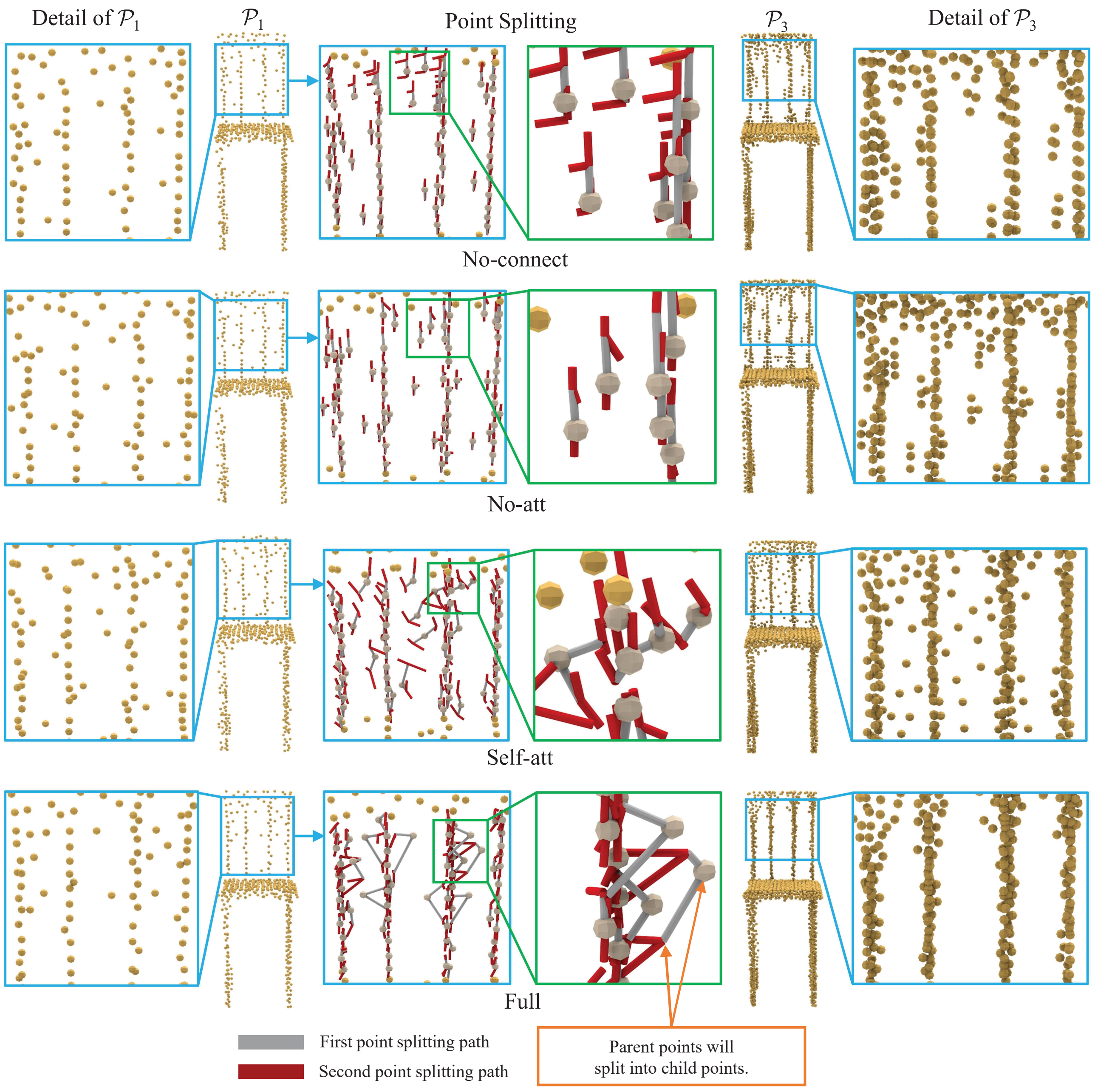
Detail of $\mathcal{P}_1$
$\mathcal{P}_1$
Point Splitting
$\mathcal{P}_3$
Detail of $\mathcal{P}_3$
No-connect
No-att
Self-att
Full
First point splitting path
Second point splitting path
Parent points will split into child points.

Figure 3. Effect justification of skip-transformer. For each model, the second column is $\mathcal{P}_1$ (512 points) and the fifth column is $\mathcal{P}_3$ (2048 points). In the third column, we sample some points from the back of the chair and visualize two steps of point splitting together ($\mathcal{P}_1$ to $\mathcal{P}_2$ to $\mathcal{P}_3$), where the gray lines indicate the first point splitting and the red lines indicate the second. To be more clear, we zoom in the figures in the third column and visualize a smaller region in the fourth column (green boxes). The first and the last column are the visualization of detailed local structure of $\mathcal{P}_1$ and $\mathcal{P}_3$, respectively.

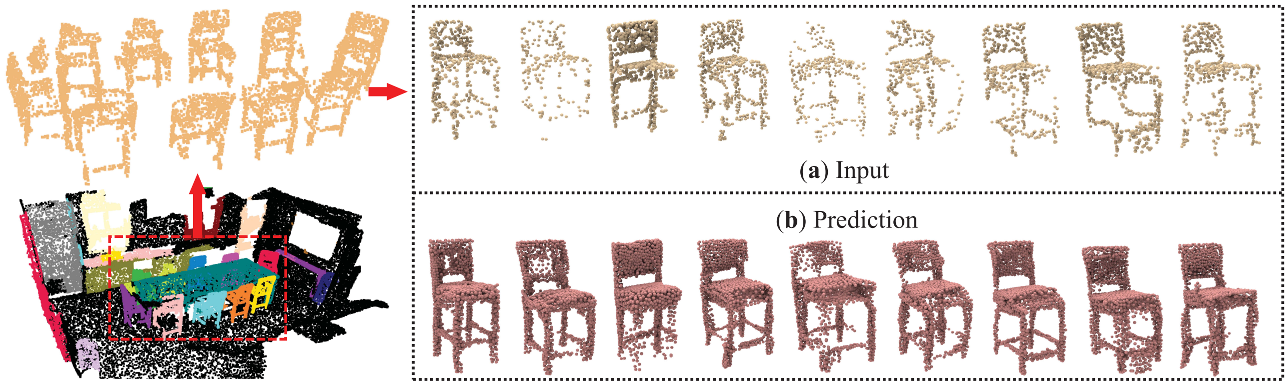

Figure 4. Visualization of completion results on ScanNet chairs. The completion results demonstrate the generalization ability of SnowflakeNet.

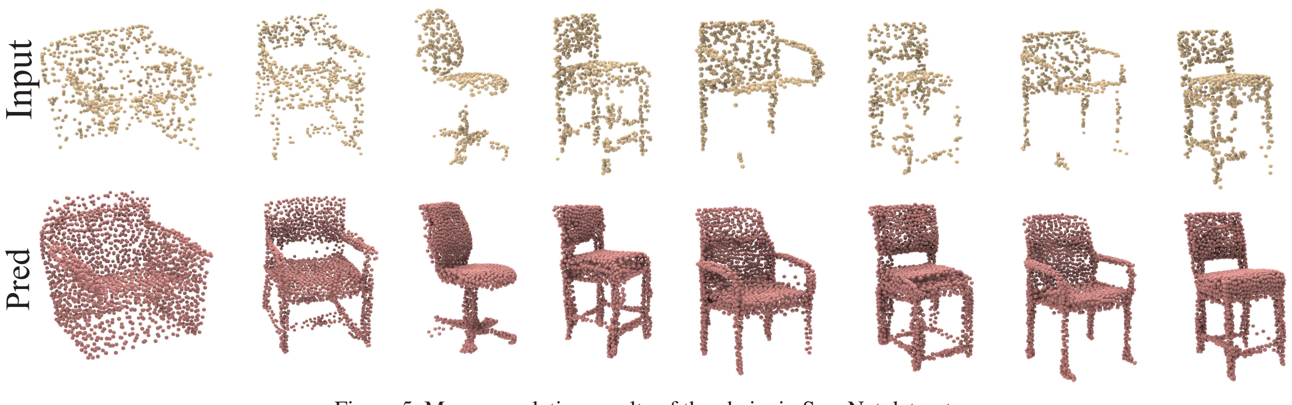

Figure 5. More completion results of the chairs in ScanNet dataset.

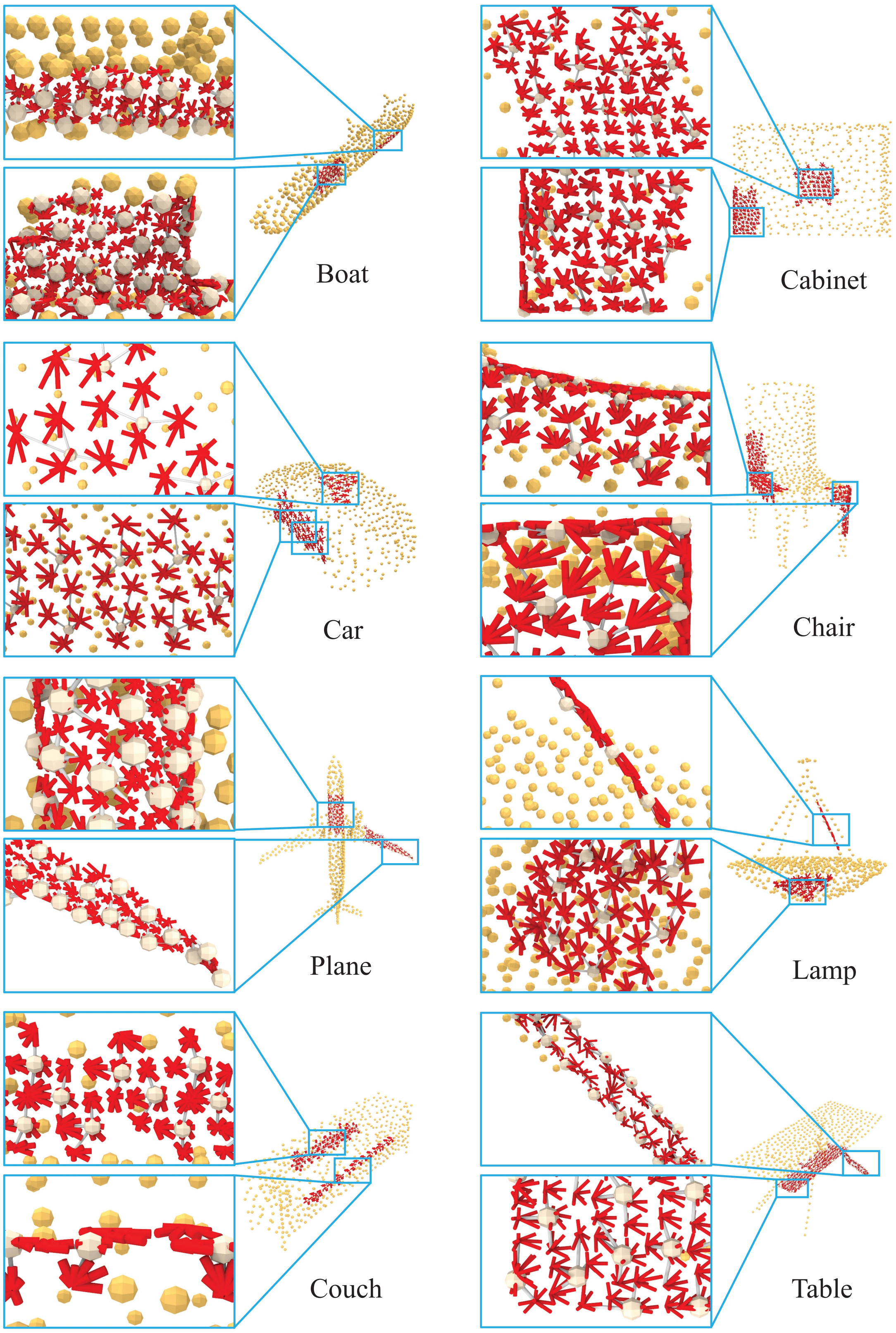

Figure 6. Visualization of snowflake point deconvolution on different objects under each category, the visualization arrangement is the same as Section 4.3 (visualization of point generation process of SPD). For each object, we sample two patches of points and visualize two steps of point splitting together, note that for each point splitting, we only demonstrate the initial parent point and the splitting paths, and the child points are not shown.

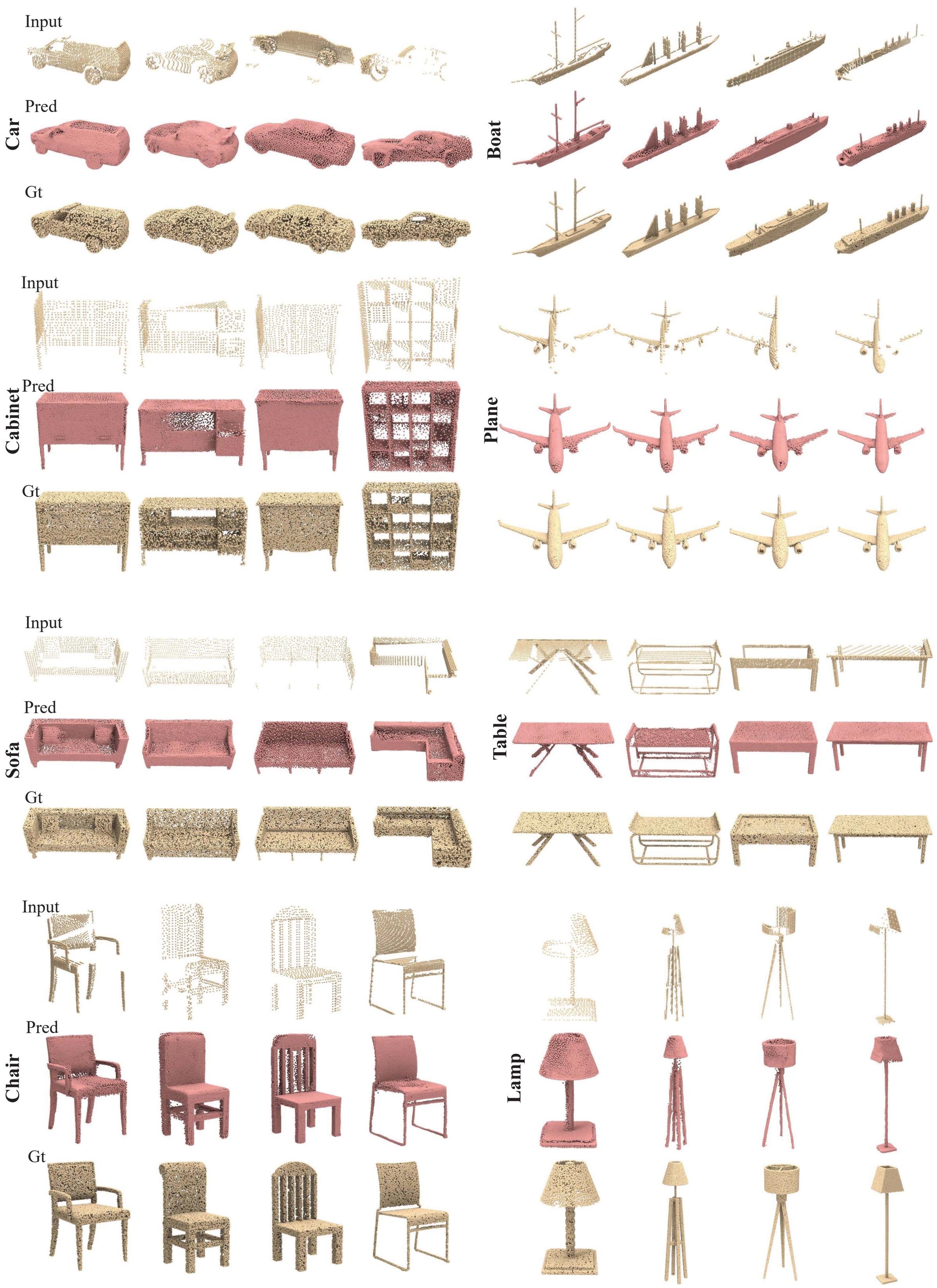

Figure 7. Visualization of completion results on the PCN dataset. For each category, the first row is the input incomplete shape, the second row is the final prediction shape, and the third is the ground truth.

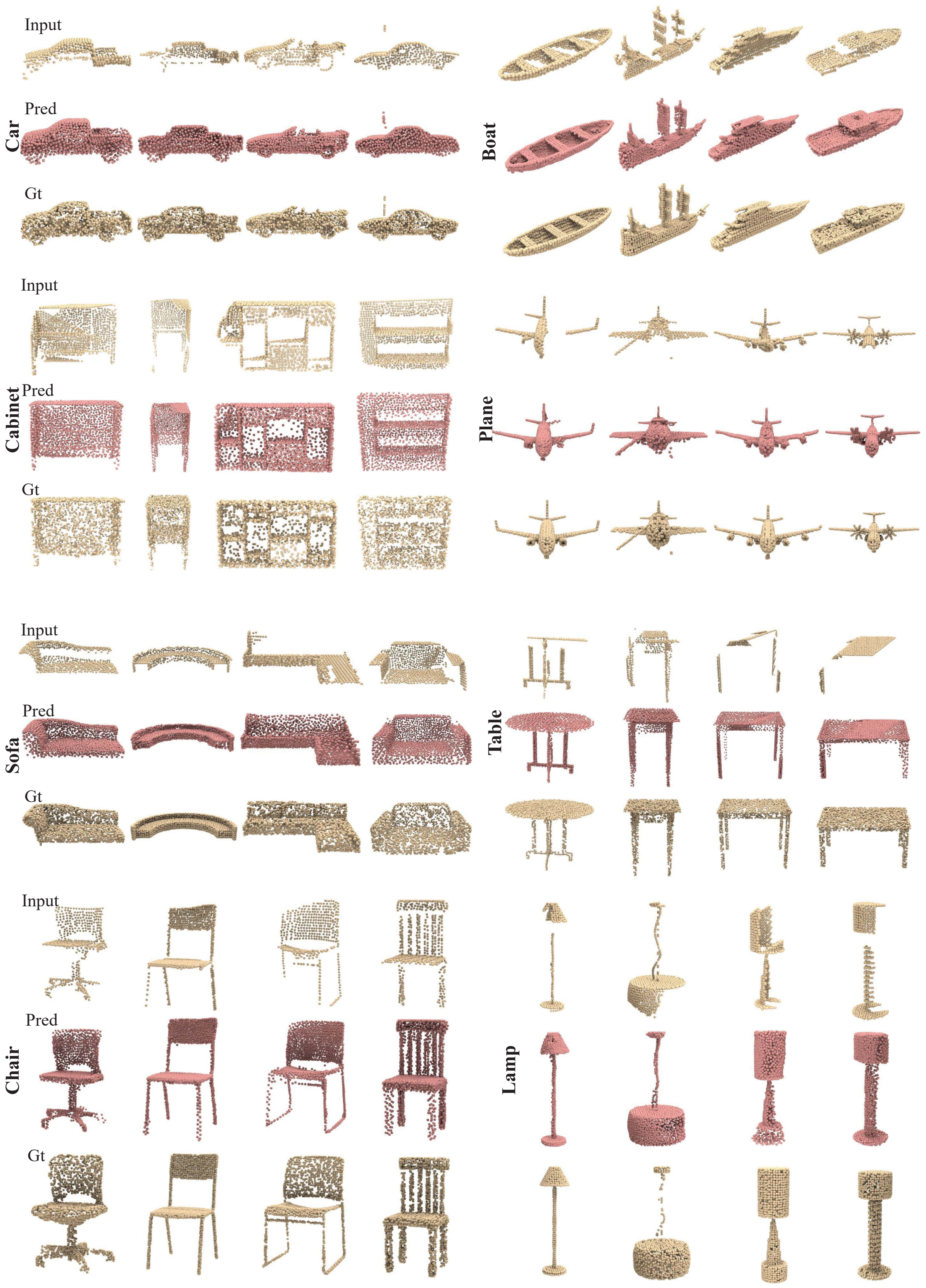

Figure 8. Visualization of completion results on Completion3D. For each category, the first row is the input incomplete shape, the second row is the final prediction shape, and the third is the ground truth.