Supplementary Material

1 Overview

This document provides additional technical details and more visualization results. Specifically, we first describe the details of the network architecture for the experiments in Sec. 2. Then we discuss the details of the proposed GAN-based point cloud generation model in Sec. 3. Furthermore, we show more visual and quantitative results of our method in Sec. 4. Finally, in Sec. 5, we present more visualization results compared with PointFlow [7].

2 Network Architecture

For the generator, the structure is illustrated in Fig. 2. We generate four resolutions (256, 512, 1024, and 2048) of point clouds from a 128-dimensional latent vector created by the normal distribution $\mathcal{N}(0, 0.2)$. Specifically, we stack 4 deconvolution networks (DECONV Networks) to generate multi-resolution feature maps. Each deconvolution network has two branches: one for capturing global information and one for bilateral interpolation. For bilateral interpolation, the detailed structure is shown in Fig. 1.The output size of 4 deconvolution networks are 256×32, 512×64, 1024×128, and 2018×256, respectively. It is important to note that our four deconvolution networks are not shared for four resolutions. To generate the coordinates of 3D point clouds, after each deconvolution network, we use Multi-Layer Perceptrons (MLPs) with neuron sizes of 512, 256, 64, and 3, respectively. After MLPs, we use $Tanh$ as the activation function to generate the final 3D coordinates.

For the discriminator, in Fig. 3, we adopt four PointNet-like [5] networks as our discriminators. Specifically, we modify the network to accommodate different resolutions. In the experiments, we found that too many convolution layers are harmful for low-resolution point clouds. Besides, for different resolutions, the discriminators are not shared.

3 Training of the Proposed PDGN

We employ the same training strategy in Goodfellow *et al.* [2] to train the generator and discriminator. We alternate between one step for optimizing the discriminator and one step for optimizing the generator to train our method. We did not use any more tricks in our training. We adopt Adam [3] with the learning rate 10^{-4} for both generator and discriminator. In the experiments, we observe that the proposed model is easy to train and stable during training. This may be due to the generation strategy, which can progressively generate point clouds from the low resolution to the high resolution. In our progressive generator,

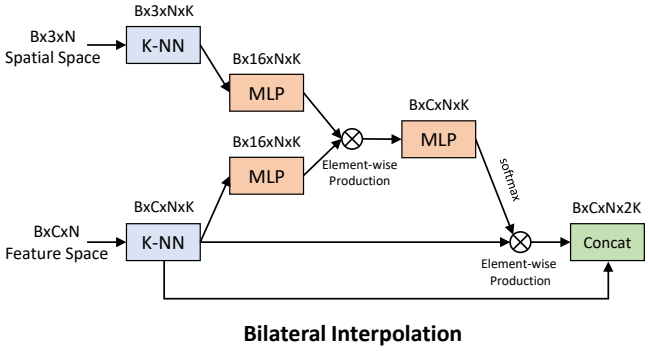

Fig. 1. Learning-based bilateral interpolation in our experiments.

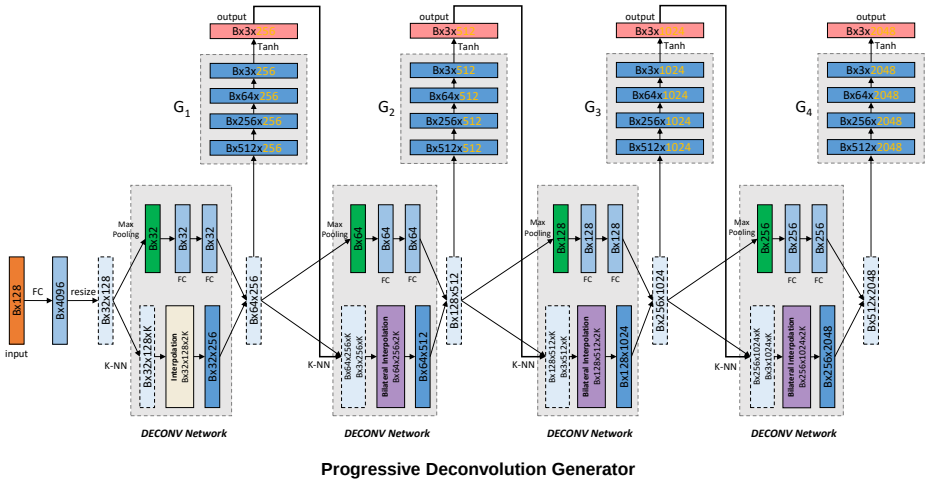

Fig. 2. The architecture of our progressive deconvolution generator. G_1, G_2, G_3, and G_4 are four MLPs for 256, 512, 1024, and 2048 resolutions, respectively. FC is the fully connected layer. k-NN represents the k nearest neighboring points.

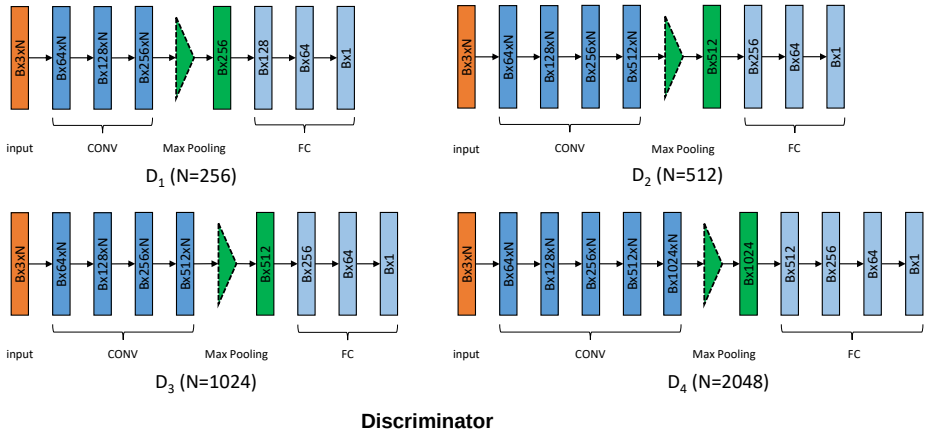

Fig. 3. The architecture of our discriminators. D_1, D_2, D_3, and D_4 are discriminators with resolutions of 256, 512, 1024, and 2048, respectively. FC is the fully connected layer, while CONV represents 1×1 convolution operating on each point.

the low-resolution network is easier to train due to the simple shape with fewer points. The stability of the low-resolution network contributes to the training of the high-resolution network. Furthermore, we also found that our method convergences quickly during training.

4 More Visualization and Quantitative Results

4.1 Generated multi-resolution point clouds

We visualize more generated point clouds of four resolutions 256, 512, 1024, and 2048, respectively. As shown in Figs. 6, 7, 8, 9, 10, 11, and 12, they contain seven categories including "Airplane", "Chair", "Car", "Table", "Lamp", "Pistol", and "Guitar". From the figure, it can be clearly seen that the generated multi-resolution point clouds are consistent.

4.2 Features in progressive deconvolution network

We analyze the outputs of different resolutions of the deconvolution network and visualize them in the feature space. As shown in Fig. 4, we visualize the generated point clouds on the "Airplane" and "Chair" categories with the outputs of four deconvolution networks by using k-means clustering in the feature space.

4.3 Three views of the generated point clouds

As shown in Fig. 13, we visualize seven categories including "Airplane", "Chair", "Car", "Table", "Lamp", "Pistol", and "Guitar". It can be seen that our generated point clouds are realistic.

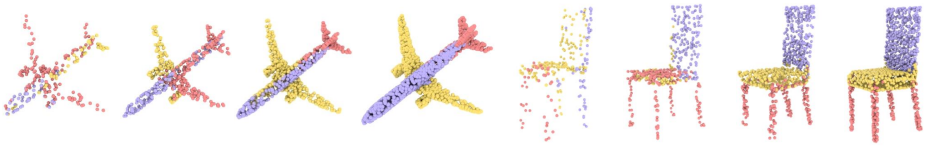

Fig. 4. Visualization results for features in four deconvolution networks by k-means clustering, colored onto the corresponding point clouds. For the "Airplane" and "Chair" categories, each category has resolutions 256, 512, 1024, and 2048, respectively. Different parts of the chair are distinguished more clearly as the resolution of the deconvolution network increases.

4.4 Ablation study on the parameter k

To study the effect of parameter k in the bilateral interpolation on the final generation result, we perform the ablation studies on parameter k. Specifically, k represents the number of the nearest neighboring points in the bilateral interpolation. We select $k \in [2, 4, \cdots, 36]$ with interval 2. The metric results are shown in Fig. 5. It can be seen that setting k values around 20 can obtain better performance than other choices. Actually, if k is too small, the small neighborhood cannot produce the discriminative geometric features of the points, leading to the poor generation results. If k is too large, the large neighborhood results in the high computational cost of the deconvolution operation. Therefore, for a good trade-off between the quality of generated point clouds and computational cost, we set k to 20 in the experiments.

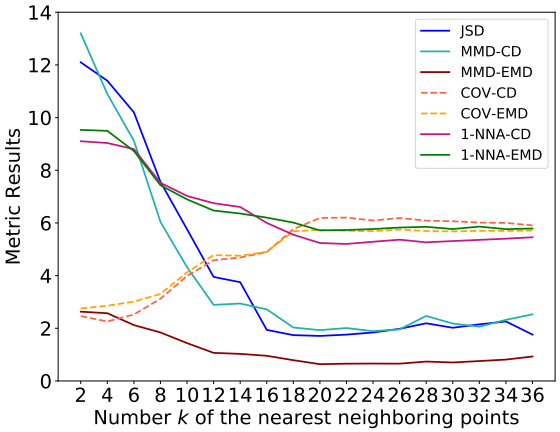

Fig. 5. Metric results in the cases of different k on the "Chair" category. Note that we magnify the results of COV-$*$ and 1-NNA-$*$ by a factor of 10.

4.5 Quantitative results of the generated point clouds

As shown in Table. 1, we provide the metric results on the "Car" category and the mean results of all 16 categories. On the "Car" category, metric results of our PDGN are comparable to the results of PointFlow [7]. This may be because the "Car" category does not have many thin structures. On all 16 categories, our PDGN is better than r-GAN [1], tree-GAN [6], and PointFlow [7]. Note that we reported the mean results of PointFlow for all 16 categories by running the official code on 16 categories.

Table 1. The comparison results of different methods. The All (16) presents the mean results of all 16 categories. The best results are highlighted in **bold**. Note that JSD scores and MMD-EMD scores are multiplied by 10^2. MMD-CD scores are multiplied by 10^3. Lower JSD, MMD-CD, MMD-EMD, 1-NNA-CD, and 1-NNA-EMD show better performance, while higher COV-CD and COV-EMD indicate better performance.

Category	Model	JSD (↓)	MMD (↓)		COV (%, ↑)		1-NNA (%, ↓)	
			CD	EMD	CD	EMD	CD	EMD
Car	r-GAN [1]	12.8	1.27	8.74	15.06	9.38	97.87	99.86
	l-GAN (CD) [1]	4.43	1.55	6.25	38.64	18.47	63.07	88.07
	l-GAN (EMD) [1]	2.21	1.48	5.43	39.20	39.77	69.74	68.32
	PC-GAN [4]	5.85	1.12	5.83	23.56	30.29	92.19	90.87
	PointFlow [7]	0.87	**0.91**	**5.22**	**44.03**	**46.59**	60.65	62.36
	PDGN (ours)	**0.75**	1.07	5.27	41.17	42.86	**57.89**	**61.53**
All (16)	r-GAN [1]	17.1	2.10	15.5	58.00	29.00	-	-
	tree-GAN [6]	10.5	1.80	10.7	**66.00**	39.00	-	-
	PointFlow [7]	8.42	2.34	7.82	45.85	52.32	58.01	60.22
	PDGN (ours)	**6.45**	**1.68**	**6.21**	56.58	**53.65**	**56.85**	**59.31**

5 Visualization Comparison with PointFlow

As shown in Figs. 14 and 15, we compare with the advanced method PointFlow [7]. Specifically, we use the trained model provided by PointFlow on GitHub to generate point clouds. As mentioned in PointFlow [7], it fails in the cases with many thin structures (like chairs). However, due to the progressive generator from the low resolution to the high resolution, our method performs well on point clouds with thin structures (like chairs).

References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: ICML (2018)

2. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NeurIPS (2014)
3. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
4. Li, C.L., Zaheer, M., Zhang, Y., Póczos, B., Salakhutdinov, R.: Point cloud gan. arXiv preprint arXiv:1810.05795 (2018)
5. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
6. Shu, D.W., Park, S.W., Kwon, J.: 3d point cloud generative adversarial network based on tree structured graph convolutions. In: ICCV (2019)
7. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: ICCV (2019)

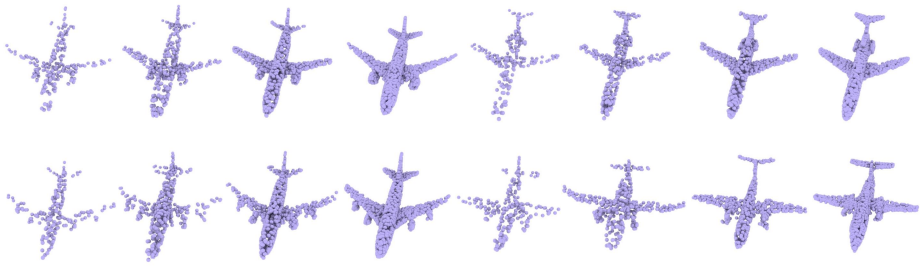

Fig. 6. Visualization results on the "Airplane" category. The resolutions are 256, 512, 1024, and 2048, respectively.

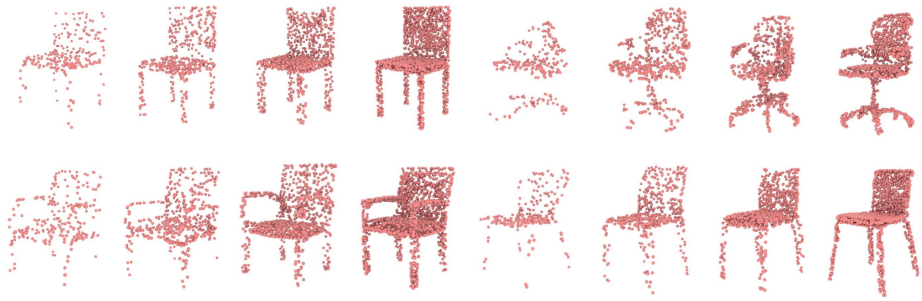

Fig. 7. Visualization results on the "Chair" category. The resolutions are 256, 512, 1024, and 2048, respectively.

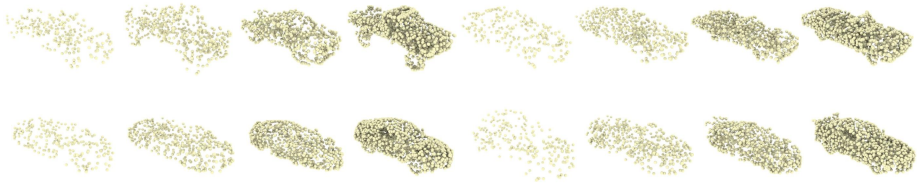

Fig. 8. Visualization results on the "Car" category. The resolutions are 256, 512, 1024, and 2048, respectively.

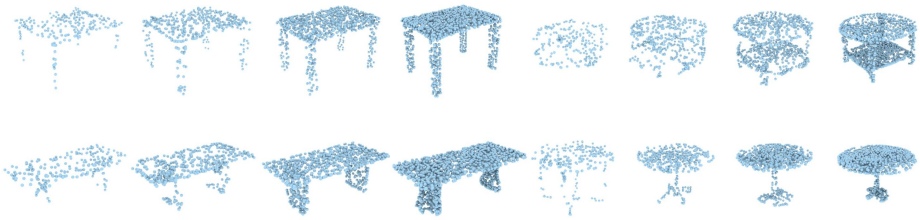

Fig. 9. Visualization results on the "Table" category. The resolutions are 256, 512, 1024, and 2048, respectively.

Fig. 10. Visualization results on the "Lamp" category. The resolutions are 256, 512, 1024, and 2048, respectively.

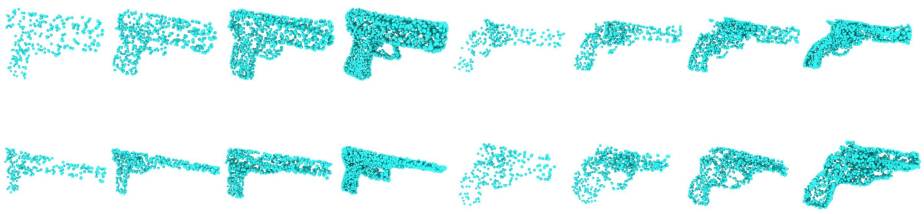

Fig. 11. Visualization results on the "Pistol" category. The resolutions are 256, 512, 1024, and 2048, respectively.

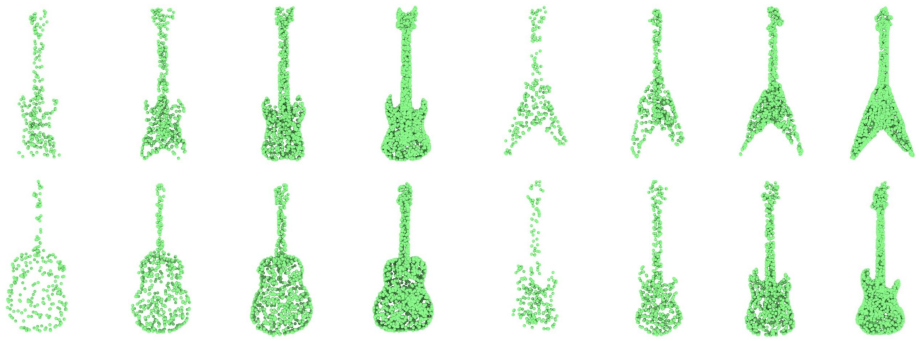

Fig. 12. Visualization results on the "Guitar" category. The resolutions are 256, 512, 1024, and 2048, respectively.

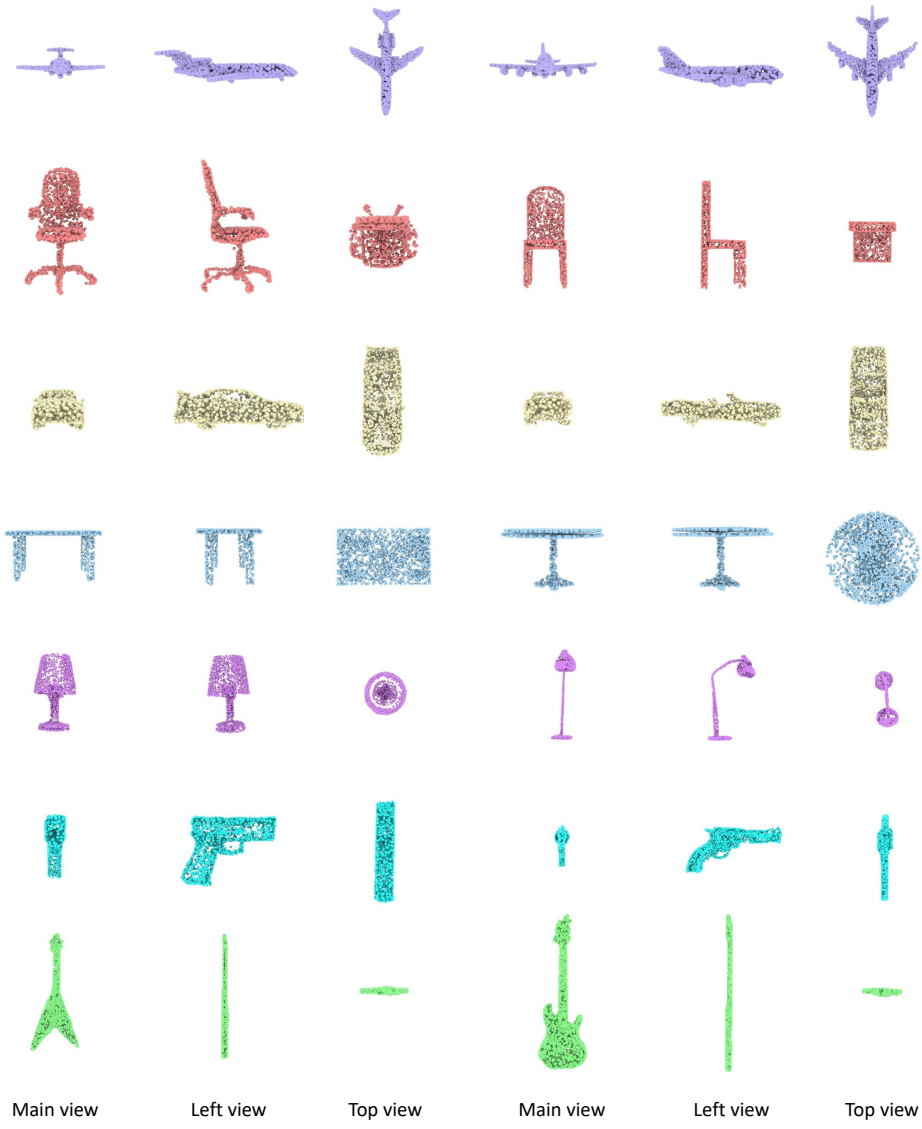

Fig. 13. Three views of generated point clouds ("Airplane", "Chair", "Car" "Table", "Lamp", "Pistol", and "Guitar"). The resolution of each generated point clouds is 2048.

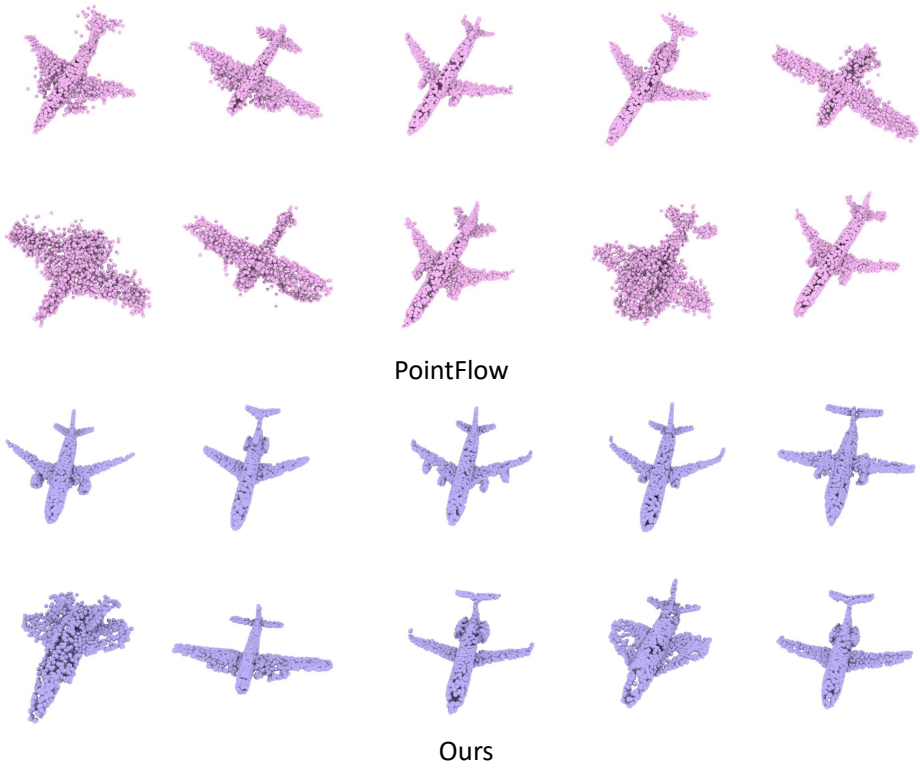

Fig. 14. Visualization results of our method and PointFlow on the "Airplane" category. The resolution of each generated point clouds is 2048.

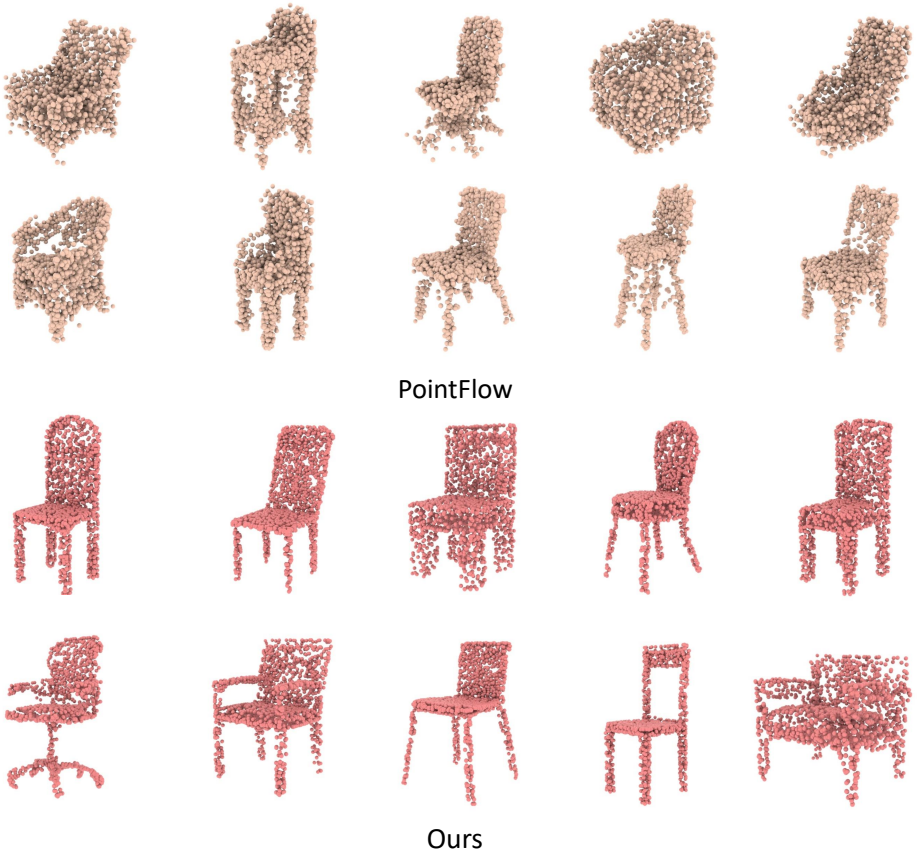

Fig. 15. Visualization results of our method and PointFlow on the "Chair" category. The resolution of each generated point clouds is 2048.