Appendix

A. Network Architectures

We first elaborate on the network architectures of our SpareNet.

A.1. Encoder

The input for our point encoder E is a partial and low-res point cloud X with M points in 3D. As shown by Table 7, the encoder E consists of four sequential Channel-Attentive EdgeConv (CAE) blocks with layer output sizes 256, 256, 512, 1024. The k of k-NN is set as 8. The slope of all LeakyReLU layers is set to 0.2. We concatenate the outputs of the four layers, and feed them into a final shared MLP-BN-ReLU layer with dimension 2048. The output shape code $\mathbf{g}$ with dimension 4096 is the concatenation of two global poolings of the above result: a maximum pooling and an average pooling.

Layer	C_{in}	$\mathbf{F}_1$	$\mathbf{F}_2$	$\mathbf{F}_3$
1	3	[256]	[16, 256]	/
2	256	[256]	[16, 256]	[256]
3	256	[512]	[32, 512]	[512]
4	512	[1024]	[64, 1024]	[1024]

Table 7: Channels of the CAE blocks in point encoder E.

A.2. Generator

Our style-based point generator G employs K (K=32) surface elements to form a complex shape. For each surface element, the generator maps a $n \times 2$ unit square ($n = N/K$) into a $n \times 3$ surface through three sequential style-based folding layers and one linear layer. The output sizes of the four layers are 4096, 2048, 1024 and 3.

We use two linear layers with $\{4096, 3059\}$ neurons to transform the shape code $\mathbf{g}$ into modulation parameters $\gamma_{\mathbf{g}}$ and $\beta_{\mathbf{g}}$ for the three style-based folding layers, with 3059 being the total size of modulation parameters in the three style-based folding layers. We partition the modulation parameters into parts according to the size of activation $\bar{\mathbf{h}}_{in}$ in each style-based folding layer, and assign them to each layer respectively. The learned $\gamma_{\mathbf{g}}$ and $\beta_{\mathbf{g}}$ are shared among all the K surface elements.

A.3. Renderer

The size of a rendered depth map $H \times W$ is set to 256×256 in experiments. The eight viewpoints for the multi-view rendering are set as the eight corners of a cube: $(\pm 1, \pm 1, \pm 1)$. We adopt the radius $\rho = 3$ in point rendering.

A.4. Refiner

In each refiner R, instead of directly concatenating the previous output points (containing N points) and the partial input points (denoted by X, containing M points) into one point cloud, we attach a flag to each point before concatenation: a 0 is attached if the point comes from X while a 1 label is attached otherwise. This results in a point cloud with $N + M$ 4-dimensional points, which is fed into a minimum density sampling [21] that samples N points out of the $N + M$ points. A residual network then learns point-wise residuals for the re-sampled N points to adjust their positions. The residual network consists of 7 CAE blocks as depicted in Table 8. We concatenate the outputs of the first and the third CAE blocks, which is fed into the fourth CAE block and outputs residuals for the refined point coordinates.

A.5. Discriminator

The real samples for training the discriminator D is the concatenation of the depth maps rendered from X and Y_r^1, while the fake samples come from concatenating the depth maps of X and Y_{gt}, as illustrated in Figure 10. The discriminator consists of four Conv-LeakyReLu-Dropouts and one Linear, with spectral normalization [24] applied on the Conv and Linear weights. The number of channels are 16, 32, 64, 128 for the four convolutional layers. Each Conv has a kernel size of 3, a stride of 2 and a padding size of 1. The slope of LeakyReLU is 0.2. The drop rate of all Dropouts is 0.75. A last Linear outputs a scalar for discrimination.

Layer	C_{in}	$\mathbf{F}_1$	$\mathbf{F}_2$
1	4	[64]	[4, 64]
2	64	[128]	[8, 128]
3	128	[1024]	[64, 1024]
4	1088	[512]	[32, 512]
5	512	[256]	[16, 256]
6	256	[128]	[8, 128]
7	128	[3]	/

Table 8: Channels of the CAE blocks in the residual network of refiner.

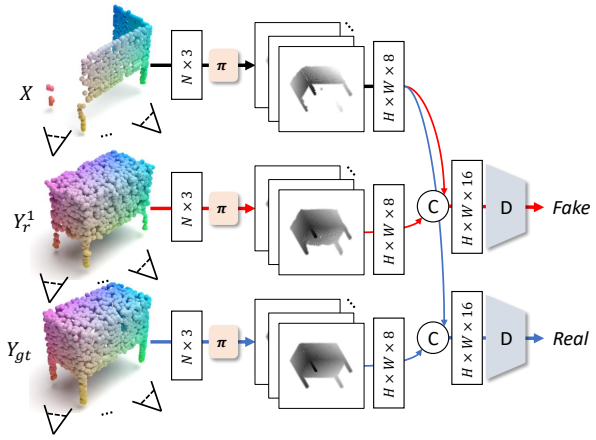

Figure 10: Discriminator D is trained using depth maps generated from the multi-view point renderer π.

B. Importance of the Image Domain Supervisions

We remove all the image-domain losses but instead implement the discriminator with a PointNet. We denote such setting as the *Point-based*, and report its comparisons with our proposed setting in Tables 9, 10, 11 and Figure 15. These results all verify the effectiveness of our proposed image-domain losses in point cloud completion.

Method	plane	cabinet	car	chair	lamp	sofa	table	vessel	avg
Proposed	**1.131**	**2.014**	**1.783**	**2.050**	**2.063**	**2.333**	**1.729**	**1.790**	**1.862**
Point-based	1.563	2.355	2.144	2.379	2.508	2.886	2.133	2.170	2.267

Table 9: Completion comparison on ShapeNet in terms of EMD $\times 10^3$ (lower is better).

Method	plane	cabinet	car	chair	lamp	sofa	table	vessel	avg
Proposed	**0.307**	**0.691**	**0.142**	**1.113**	**0.774**	**0.945**	**0.668**	**0.523**	**0.645**
Point-based	1.961	0.826	1.592	4.275	1.406	5.153	0.673	1.074	2.120

Table 10: Completion comparison on ShapeNet in terms of FPD $\times 0.1$ (lower is better).

Method	plane	cabinet	car	chair	lamp	sofa	table	vessel	avg
Proposed	**0.176**	**0.664**	**0.362**	**0.616**	**0.631**	**0.789**	**0.498**	**0.384**	**0.515**
Point-based	0.331	0.809	0.471	0.812	0.834	1.187	0.649	0.557	0.706

Table 11: Completion comparison on ShapeNet in terms of CD $\times 10^3$ (lower is better).

C. More Qualitative Results

We show more qualitative completion results in Figures 11, 12, 13. We also demonstrate more qualitative comparisons in Figure 14 with respect to the rendered depth maps. Moreover, in Figure 17, we illustrate the car completion results based on the real-world LiDAR scans from KITTI.

D. Rendering with Different Point Cloud Resolutions

We also illustrate the multi-view depth maps of the same shape that are rendered with different point numbers in Figure 16. It demonstrates that a denser point cloud can alleviate the point scattering artifacts in rendered depth maps.

E. Robustness Study

We finally visualize point completion results of the same 3D shape, but from partial points that are sampled from different view angles (Figure 18) or with different sampling densities (Figure 19). These results verify that our method is robust to various acquisition conditions, such as different viewpoint and point density.

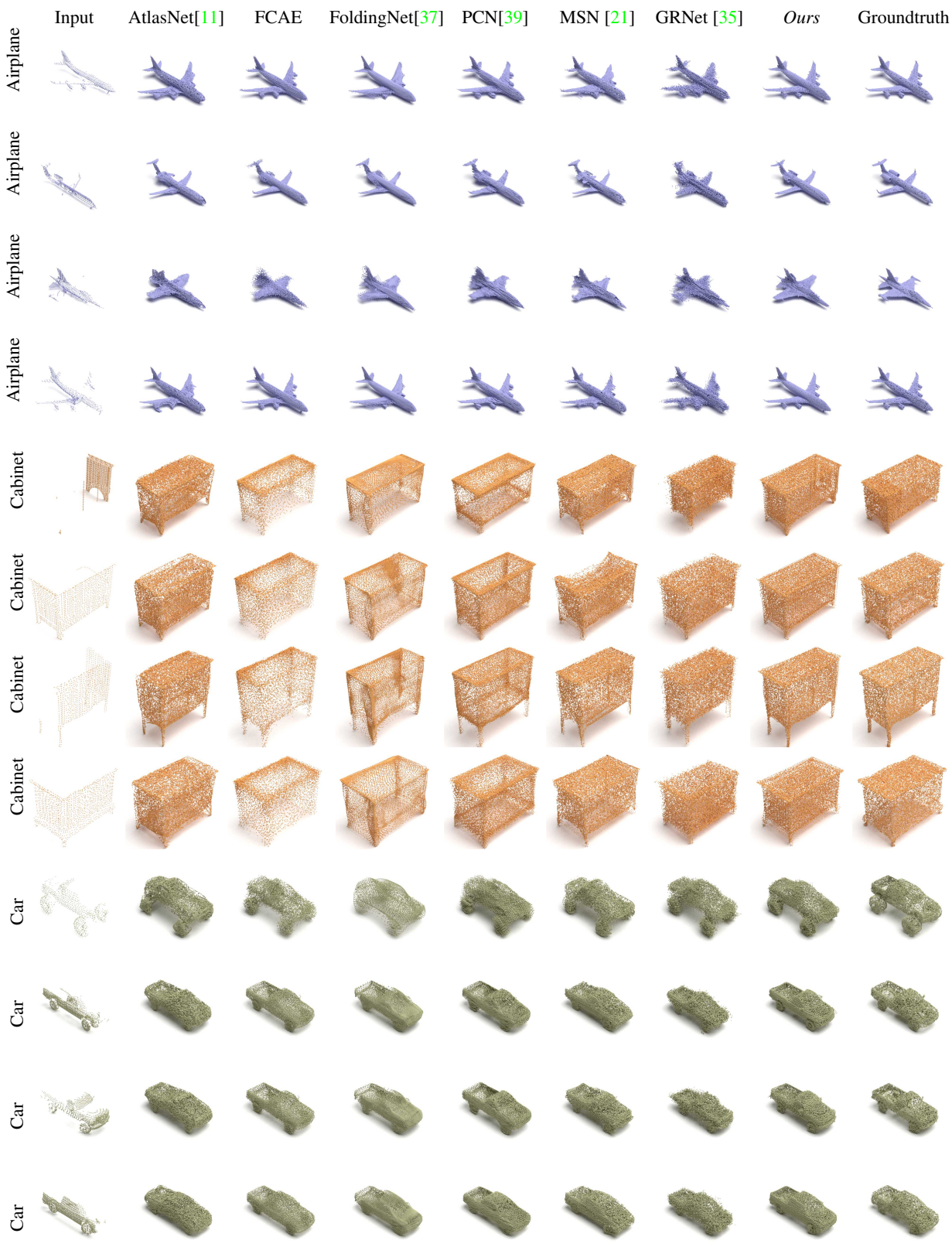

Figure 11: Visualized completion comparison on ShapeNet.

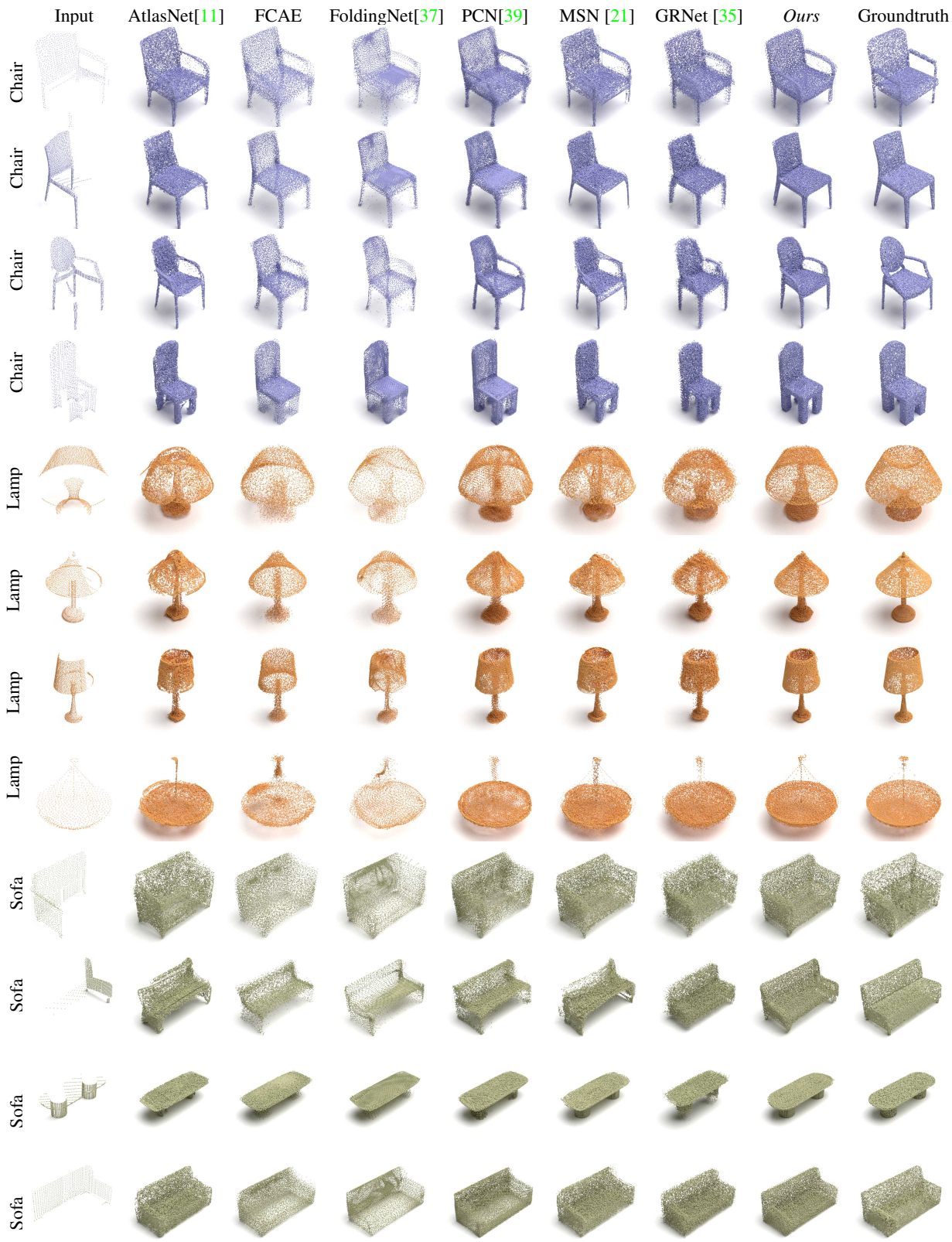

Figure 12: Visualized completion comparison on ShapeNet.

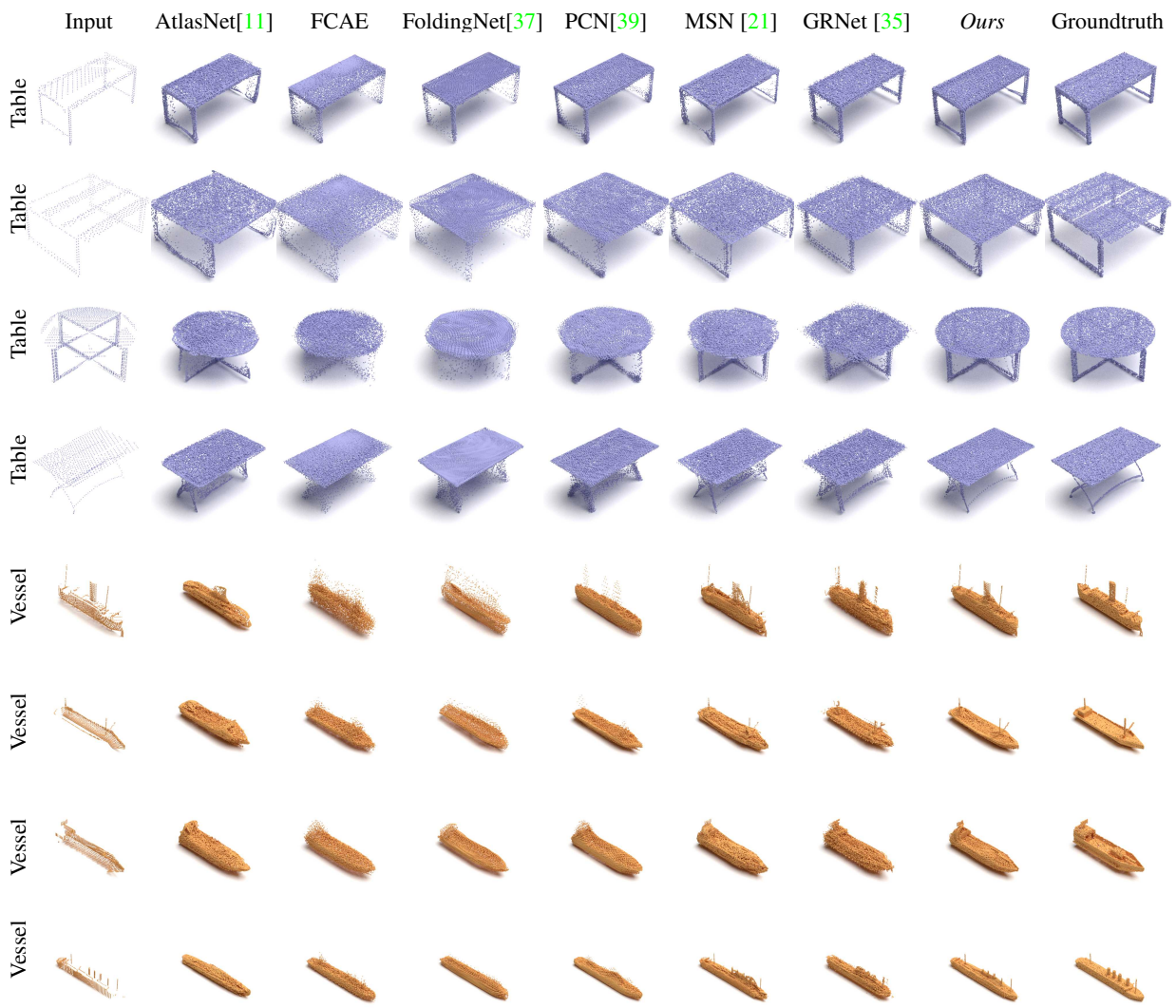

Figure 13: Visualized completion comparison on ShapeNet.

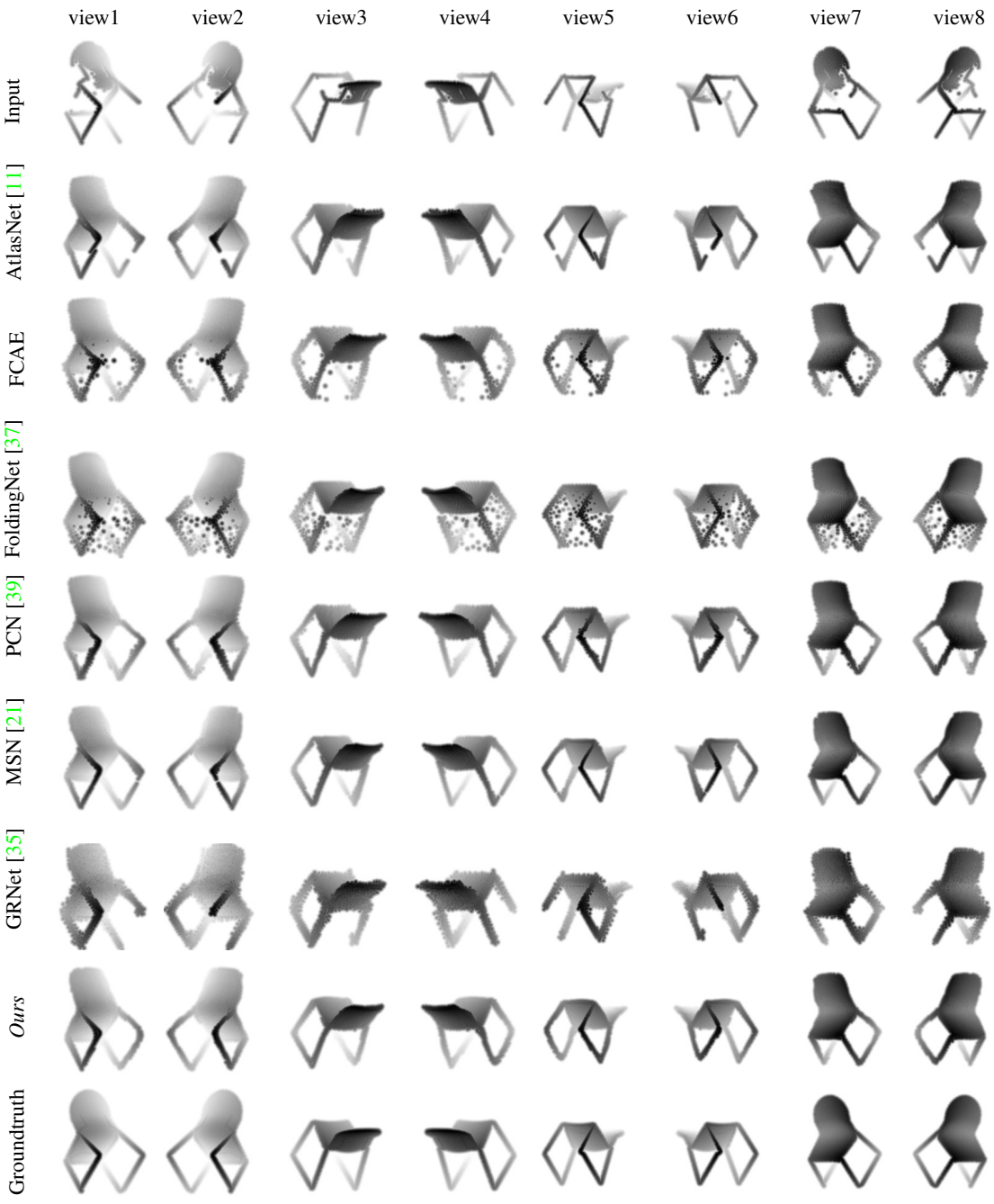

Figure 14: Completion comparison visualized in rendered depth maps.

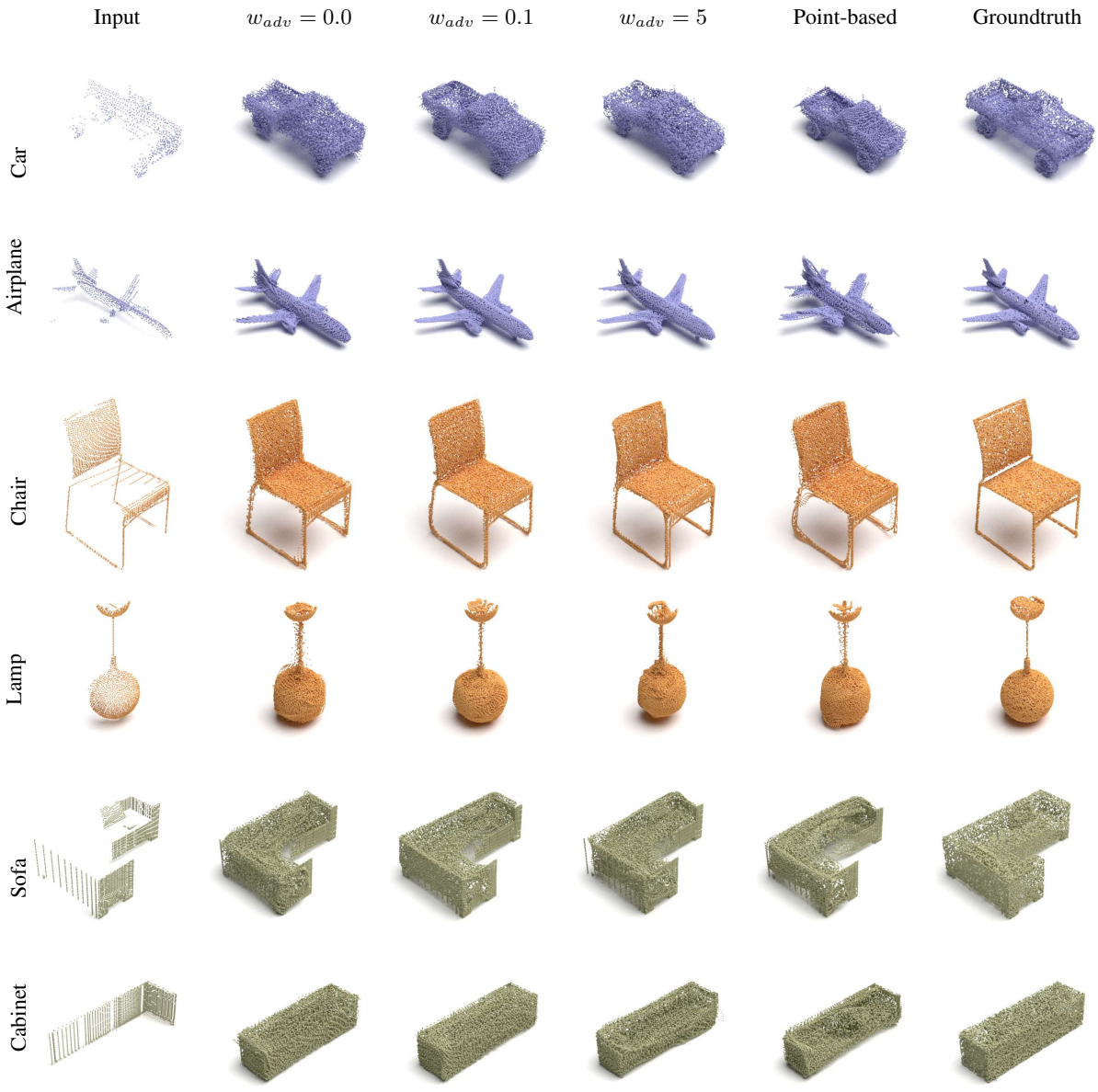

Figure 15: Visualized comparison of models with or without rendering supervisions. In comparison, the proposed rendered discriminator is more capable to examine the local details than the point-based discriminator.

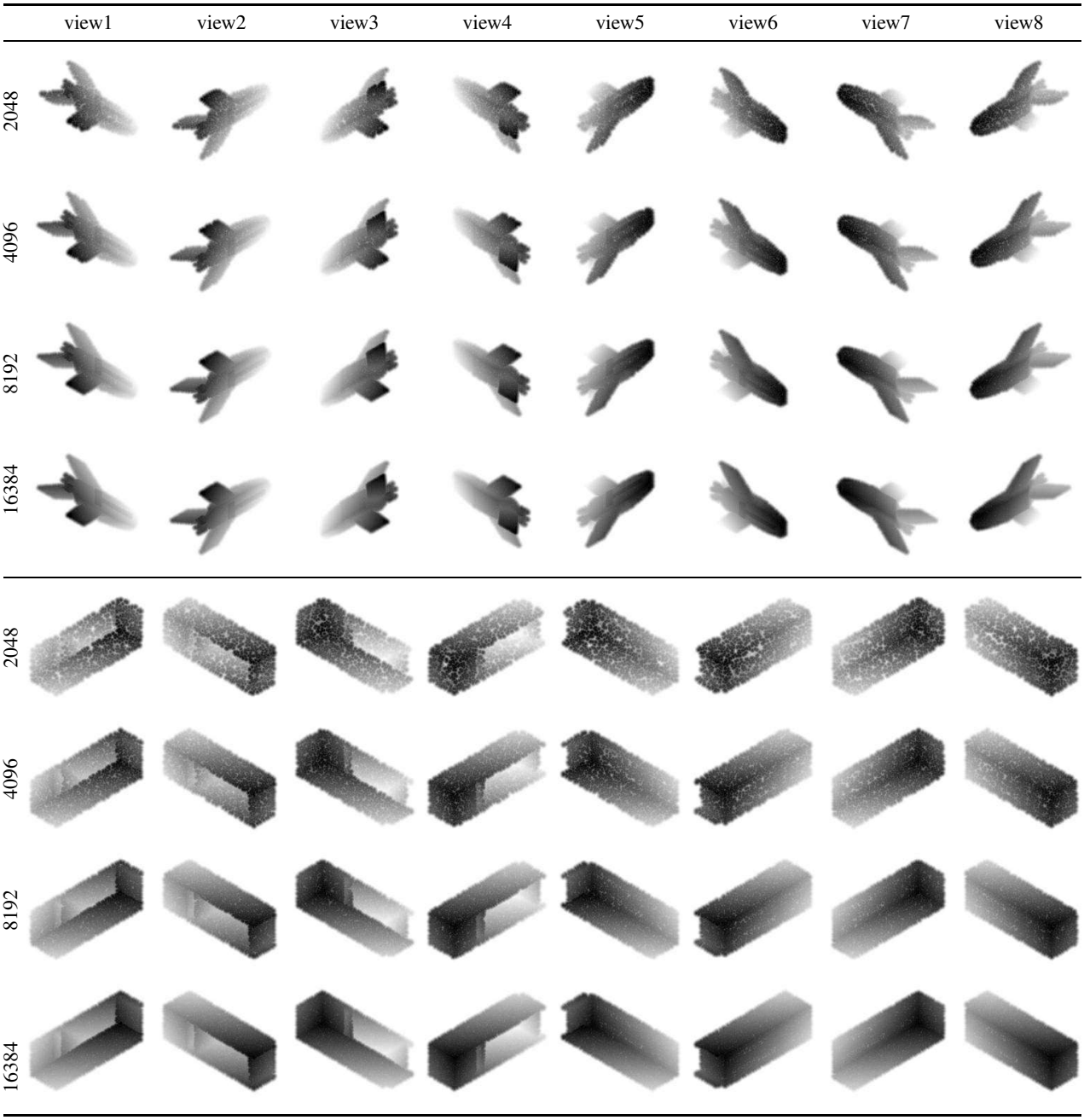

Figure 16: Multi-view depth maps rendered with different point numbers. A denser point cloud can alleviate the point scattering artifacts in its rendered depth maps.

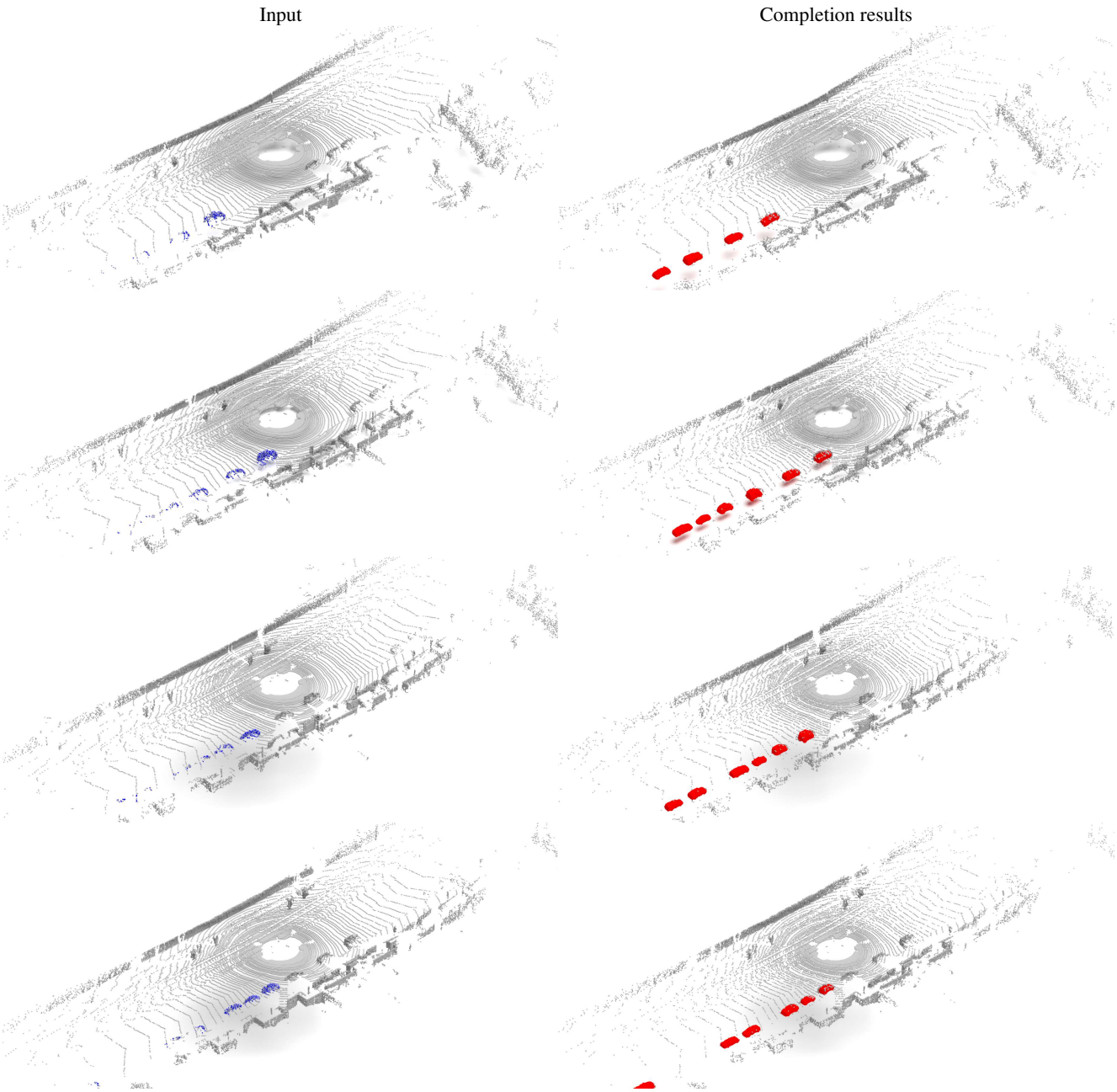

Figure 17: Visualized car completion results based on real-world LiDAR scans from the KITTI dataset [9]. The left frames show the input partial points in blue, the right frames show the completed point clouds of cars in red.

Figure 18: Completing the same 3D shape from partial points that are sampled from four different view angles.

Figure 19: Completing the same 3D shape from partial points of various densities (from 1500 points to 3000 points).