

# Supplementary: Self-Supervised Point Cloud Completion via Inpainting

Himangi Mittal  
hmittal@andrew.cmu.edu

Carnegie Mellon University  
Pittsburgh, PA, USA

Brian Okorn  
bokorn@andrew.cmu.edu

Arpit Jangid  
ajangid@andrew.cmu.edu

David Held  
dheld@andrew.cmu.edu

## 1 Appendix

### 1.1 Architecture Details

For all results and ablations, we keep the output size of our network as 8192 points, where the global decoder  $D_g$  generates 4096 points and the local decoder  $D_\ell$  generates 512 points for each region, to make an overall size of 4096 points across all local regions. Similarly, the input size is kept consistent for all the ablations; that is, the input size is 3096 and 387 for global encoder  $E_g$  and local encoder  $E_\ell$  respectively. Each region in  $X$  is dropped with a probability of removal of 20% and the resulting synthetically occluded point cloud  $\hat{X}$  is passed to the global encoder  $E_g$ . In parallel, the input partial point cloud is subdivided into 8 regions along the axial planes of the canonical frame. Each region not artificially removed or marked as missing is then independently encoded using the local encoder,  $E_\ell$ . When encoding each region of the input cloud, regions that are marked as missing based on the threshold number of points are replaced with zeros equal to the threshold. In our method, we set this threshold as 4. We allow a small overlap of 0.02 cm between neighboring regions for the ShapeNet dataset and 0.02m for the KITTI dataset. The architecture of local encoder  $E_g$  and global decoder  $D_g$  are similar to the PCN [1]. For local encoder  $E_\ell$  and local decoder  $D_\ell$ , we use the architecture of PCN [1], but reduce the number of hidden units to 1/8th of the original number. We use Adam optimizer with a learning rate of  $1 \times 10^{-4}$  and train our network for 400K iterations.

### 1.2 Data preparation

**Shapenet:** We obtain a point cloud from the RGB-D data by backprojecting 2.5D depth images to 3D similar to Gu *et al.* [2]. In contrast to DPC [2], we do not use the color in-

formation. The centers of the oriented clouds are then shifted to the origin before passing it to our shape completion network. Specifically, we use the 3D partial shape classification branch of IT-net pre-trained on ModelNet40 to generate the pose transformations, as it does not require the ground-truth pose annotations for training. Since our method does not require perfect pose alignment, using IT-Net pretrained on ModelNet40 instead of ShapeNet is sufficient for our purpose, as it represents an off-the-shelf canonical frame estimator for our model classes. We refer the reader to IT-Net [8] for details on this pose canonicalization method.

Originally, the ShapeNet [9] dataset has 5 views. When training on  $N$  views, we only consider a fixed set of  $N$  random views, which is chosen at the beginning of training; the network is only trained on these  $N$  views and the other views of an object are discarded.

**Semantic KITTI:** At training time, we subdivide the observations of a single instance into groups of 20 sequential observations and randomly sample a set of four views for multi-view training. When evaluating accuracy on this dataset, all 20 frames are combined using ground truth odometry to form the ground truth shape of each instance. This merged cloud is only used for evaluation and is not present during training. At inference time, only a single view is used.

## 1.3 Ablation Studies

In this section, we present a more exhaustive ablation study focusing on the number of views, architecture changes, number of input points used for training and mention the details of the ablation of densification of input point clouds for the KITTI dataset.

### 1.3.1 Number of views

We evaluate the sensitivity of our method to the number of views available at training time in Supplementary Figure 1. We show the results both with and without inpainting in green and red lines respectively. It can be observed that our model is able to outperform the baseline with 2 views and 3 views, even though the baseline Gu *et al.* [8] is trained with 4 views. This demonstrates that our method is able to take advantage of a reduced number of views, due to our use of inpainting. We also show the qualitative results with varying numbers of training views in the Supplementary Figure 2; as can be seen, the results of 2 views and 3 views are qualitatively very similar to the results with 4 views.

### 1.3.2 Architecture Changes

**Global and Local Encoders and Decoders** We analyze whether to use both global and local encoders and decoders in our network. The results can be found in Supplementary Table 1. It can be observed that a combination of global and local encoders and decoders gives the best performance among all the possible combinations.

**Number of levels** In addition to the two levels in our parallel model (global and local), we experiment with adding another branch where the partial point cloud is partitioned into  $3 \times 3 \times 3$  regions. For this branch, we use an independent local encoder and decoder. The input size of a region to the encoder is taken as 115 points (to maintain a total input size of 3096) and the size of the predicted point cloud is 152 points for each region (to maintain a total output size of 4096 for the local decoder). For computing the loss, we divide the

original input (before dropping points) into regions and subsample the points to have at most 304 points in each region. The results are in Supplementary Table 3. We notice that further partitioning of the partial point cloud and the additional branch do not give a significant improvement in the performance.

### 1.3.3 Number of input points

We evaluate the effect of the number of points in the point cloud on the performance of our method. To test this, we create new versions of the test set with varying numbers of points; for each object, we resample the point cloud (without replacement) from the input point cloud with a varying number of sampled points. We evaluate the Chamfer Distance metric as a function of the number of points in the input point cloud on the ShapeNet and KITTI [10] dataset during testing. We evaluate our method on the number of points ranging from 100 to 4000 and present the results in Figure 3. As expected, performance degrades as we reduce the number of available points.

### 1.3.4 Densification of KITTI point clouds

To evaluate the quantitative effects of simply densifying the input point cloud without completing occluded regions, we design a simple densification method. For each point in the input partial point cloud, we find its 10 nearest neighbors and estimate the eigenvalues of this local neighborhood. An ellipsoid is formed using these values and points are uniformly sampled within this volume. This approximates the local surface. From Table 2 of the main paper, the improvement of our method over the results of this densification method demonstrates that our model is completing the partial point clouds rather than simply densifying the partial input cloud.

### 1.3.5 Performance Analysis with respect to Occlusions

We conduct an experiment to assess the impact of occlusions in the input partial point clouds on the ability of the model to complete the given shape. To do so, we introduce artificial occlusions by removing a certain number of regions from the input during testing (we have divided the input into 8 total regions). Given that the original input is already naturally occluded, we artificially remove at most three regions because beyond that, the input is barely visible. The results are shown in Table 2; we can observe that as the number of artificial occlusions in the input increases, there is a slight drop in performance for all categories. However, the model is considerably robust to the additional occlusions.

## 1.4 Metrics

In this section, we report different metrics for further analysis of our method.

### 1.4.1 Precision and Coverage of observed and unobserved regions

For a detailed analysis, we compute the precision and coverage of the observed and unobserved regions of the input point cloud. To categorize points as observed or unobserved, we compute the distance between each point in the predicted point cloud and its nearest neighbor in the input point cloud. We compute the mean and standard deviation of these distances for each point cloud and use 1 standard deviation over this mean as a threshold. Points with the

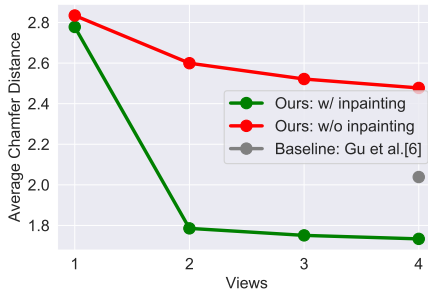


Figure 1: Quantitative Results on the number of views (1, 2, and 3) (with **green** and without inpainting **red**) used during network training. Our original method trains on 4 views. All the values reported are average Chamfer Distance metric over the ShapeNet (Airplane, Car, Chair) and KITTI dataset. We are able to outperform the baseline using a limited number of views due to our use of inpainting.

nearest neighbor distance greater than this threshold are considered as unobserved, while all other points are considered observed. The precision and coverage are computed separately for each of these types of points and we report the results in the Supplementary Table 4. As expected, we find that the precision and coverage of the observed regions are slightly better than that of unobserved regions in the input partial point cloud; however, the results are relatively similar for the observed and unobserved regions, which provides further evidence that we are completing (and not just densifying) the input (see also Section 1.3.4).

### 1.4.2 F1-Score

Following Xie et al [5], we evaluate the F1-score@1%, which is the harmonic mean between precision and recall, on the ShapeNet dataset. In this context, “precision” is the percentage of the points in the predicted point cloud which are within a specified distance threshold with the ground truth. “Recall” is the percentage of the points in the ground truth point cloud that are within a distance threshold with the predicted point cloud. Precision helps to measure the accuracy of the prediction and recall measures the coverage of the prediction. In this metric, we use  $d = 1\%$  of the side length of the predicted point cloud. It can be observed from the Supplementary Table 5 that our method is able to outperform the baseline DPC [9] when evaluated on this metric. We do not report the results on Gu *et al.* [9] since their code is not open-source.

### 1.4.3 Uniformity Metric

We also evaluate the uniformity metric following Xie et al [5] on the ShapeNet and KITTI datasets. In the Supplementary Table 6, we compare our method with the baseline DPC on the ShapeNet dataset. Our method gives a similar performance with the baseline with respect to this metric, revealing that both methods have similar uniformity of predicted points.

For the KITTI dataset, we compare our method with the ablation of our method without inpainting, as DPC does not train and evaluate on KITTI and Gu *et al.* [9] do not have open-source code. We report the results on KITTI in Supplementary Table 7 and show the

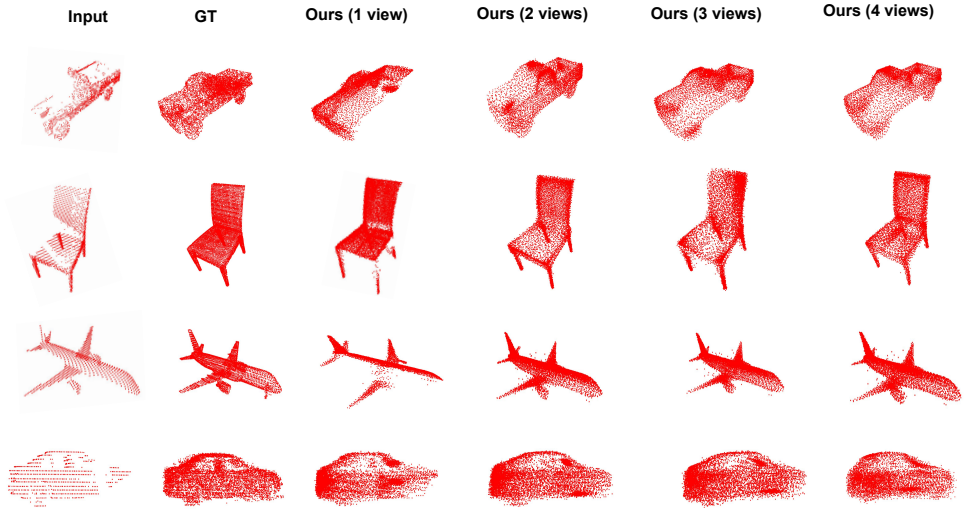


Figure 2: Qualitative results on varying the number of views given as input to the PointP-nCNet. The first, second, third, and fourth row shows the results on the ShapeNet test set of car, chair, plane, and Semantic KITTI [1] dataset respectively. As can be seen, the results of 2 views and 3 views are qualitatively very similar to the results with 4 views. This demonstrates that our method is able to take advantage of a reduced number of views, due to our use of inpainting.

improvement in the performance of our model when using inpainting.

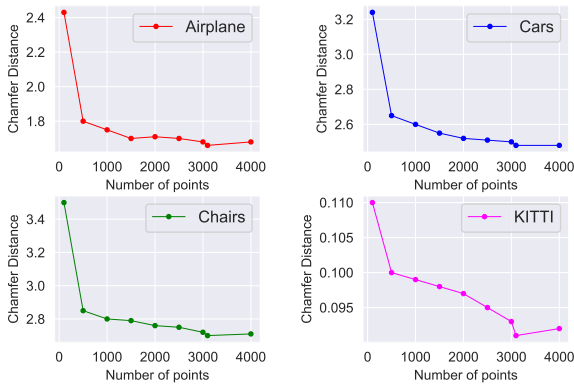


Figure 3: Quantitative Results of the Chamfer Distance metric with respect to the number of points in the input point cloud during testing.

## 1.5 Qualitative Results

We present additional visualizations of the complete predicted point cloud generated by our network, PointPnCNet.

**Cars:** As can be observed from the Supplementary Figure 4, our model is able to complete the finer details of a car such as the headlight of a car and generates a more defined outer boundary in comparison to DPC [9]. We also show that our network has the ability to not only complete the shapes of general cars, but also the shape of a truck as shown in the third row of Supplementary Figure 4. We show a few failure cases as well on the car category in the Supplementary Figure 5. Our method is unable to create detailed shapes of various sports cars. Further, for the truck in the second row, our method fails to create a gap between the front and back of a truck.

**Chairs:** We present in Supplementary Figure 6 that our method is able to generate finer completion results on different types of chairs such as a sofa and desk chair than DPC [9]. It is able to complete the front, back, and arms of the chair. There are also a few failure cases where the network generates noisy results especially near the legs of a chair as seen in Supplementary Figure 7.

**Airplanes:** From Supplementary Figure 8, we observe that the network is able to complete the front, back, and wings of the planes. Supplementary Figure 9 shows some failure cases in which it also generates some noisy points near the wings of the planes.

**KITTI:** We show the visualizations where our network is able to complete the partial point cloud cars from the LiDAR scans of the Semantic KITTI dataset in the first and second row of Supplementary Figure 10. Additionally, there are a few failure cases where the network is unable to generate the details in a fine manner such as the tire of a car as seen in the third and fourth row of Supplementary Figure 10. We also show the completion results of the partial point clouds in a scene in the Supplementary Figure 11.

**ShapeNet Categories:** We present the qualitative results on the 5 other categories of the ShapeNet dataset - Cabinet, Lamp, Sofa, Table, and Vessel in the Figure 12. We compare the results of our method with our ablation of without inpainting. It can be observed that our method is able to complete the shape of the incomplete point clouds whereas our method without inpainting outputs noisy points.

## 1.6 Comparison with supervised method

To analyze the performance gap between self-supervised method and supervised method, we compare the performance of our method with a fully supervised method, PCN [7] on 8 categories of the ShapeNet dataset and present the results in Table 8. Since our method builds on the architecture of PCN, we compare our method to fully-supervised PCN; the choice of architecture is somewhat orthogonal to our proposed method of inpainting. We observe that the fully supervised PCN outperforms our self-supervised method, as expected. However, our results indicate that our method has reduced the gap between self-supervised and fully supervised approaches. In Table 8, we also compare our method to the ablation of "no inpainting" across 8 object categories of ShapeNet and show consistent improvement in performance.

$E_g$	$E_\ell$	$D_g$	$D_\ell$	Airplane	Car	Chair	KITTI
				CD	CD	CD	CD
✓		✓		1.830	2.710	3.260	0.336
✓			✓	1.930	2.560	3.480	1.042
✓		✓	✓	1.820	2.580	3.320	0.357
	✓	✓		1.950	2.790	3.520	0.329
	✓		✓	2.010	2.610	3.730	0.392
	✓	✓	✓	1.930	2.650	3.610	0.362
✓	✓	✓		1.860	2.840	3.110	0.388
✓	✓		✓	1.850	2.530	3.250	1.131
✓	✓	✓	✓	<b>1.660</b>	<b>2.480</b>	<b>2.700</b>	<b>0.095</b>

Table 1: We study the performance of the architecture styles through combinations of local and global encoders and decoders on the Airplane, Car, Chair of the Shapenet dataset and KITTI dataset via Chamfer Distance metric. It can be observed that a combination of global and local encoders and decoders gives the best performance among all the possible combinations.

Number of regions removed	Airplane	Car	Chair	KITTI
0	<b>1.66</b>	<b>2.48</b>	<b>2.70</b>	<b>0.095</b>
1	1.67	2.50	2.73	0.097
2	1.76	2.60	2.79	0.100
3	1.89	2.67	2.95	0.102

Table 2: Chamfer Distance onShapenet and KITTI with varying numberof removed regions

Ablation	Airplane	Car	Chair	KITTI
	CD	CD	CD	CD
Adding third level	2.070	2.550	3.030	0.123
<b>Our method (2 levels)</b>	<b>1.660</b>	<b>2.480</b>	<b>2.700</b>	<b>0.095</b>

Table 3: Quantitative Results on the architecture changes in our method. All the Chamfer Distance metric values reported for Shapenet are multiplied with 100. It can be observed that adding a third level does not give a significant improvement in the performance.

Region	Airplane	Car	Chair	KITTI
<b>Observed Precision</b>	0.771	1.113	1.767	0.625
<b>Unobserved Precision</b>	0.824	1.127	1.844	0.640
<b>Observed Coverage</b>	0.848	1.490	1.344	0.531
<b>Unobserved Coverage</b>	0.857	1.496	1.355	0.648

Table 4: Precision and Coverage for the observed and unobserved regions on the Shapenet and KITTI dataset.


Method	Airplane	Car	Chair
DPC[ 	0.423	0.364	0.315
<b>Ours</b>	<b>0.626</b>	<b>0.450</b>	<b>0.409</b>

Table 5: F1-Score@1% on the Shapenet dataset.

	Airplane		Cars		Chairs	
p	DPC [1]	Ours	DPC	Ours	DPC	Ours
<b>0.4%</b>	0.775	0.775	0.758	0.757	0.785	0.787
<b>0.6%</b>	0.674	0.673	0.646	0.647	0.664	0.664
<b>0.8%</b>	0.490	0.490	0.489	0.489	0.498	0.497
<b>1.0%</b>	0.395	0.394	0.388	0.389	0.385	0.385
<b>1.2%</b>	0.256	0.257	0.246	0.245	0.265	0.264

Table 6: Uniformity Metric on the Shapenet dataset compared with the baseline DPC [1].

p	w/o inpainting	Ours
<b>0.4%</b>	0.815	0.750
<b>0.6%</b>	0.773	0.658
<b>0.8%</b>	0.697	0.527
<b>1.0%</b>	0.588	0.496
<b>1.2%</b>	0.517	0.384

Table 7: Uniformity Metric on the KITTI dataset compared with our baseline of our model without inpainting.

	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Vessel
<b>Ours (w/o inpainting)</b>	0.026	0.045	0.031	0.039	0.041	0.039	0.040	0.033
<b>Ours (Self-Supervised)</b>	0.016	0.027	0.024	0.027	0.030	0.029	0.025	0.026
<b>PCN (Fully Supervised)[2]</b>	0.005	0.010	0.008	0.010	0.011	0.011	0.008	0.009

Table 8: Quantitative results of comparison of our self-supervised method (Ours) with fully supervised method, PCN [2], and ablation of our method without inpainting.

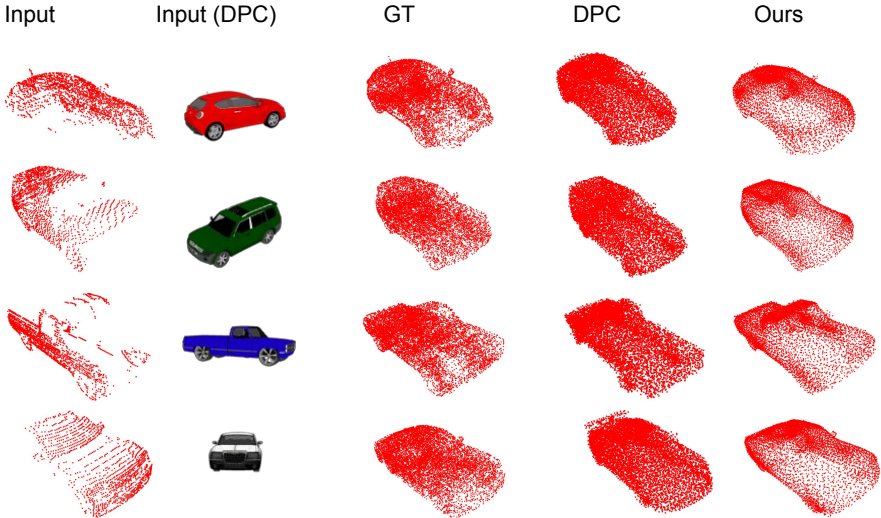


Figure 4: Success cases on the Car category of the ShapeNet dataset. It can be observed that our model is able to complete the finer details of a car such as the headlight of a car and generates a detailed outer boundary in comparison to DPC [1] in all the rows. It is also able to generate the shape of a truck as can be seen in the third row.

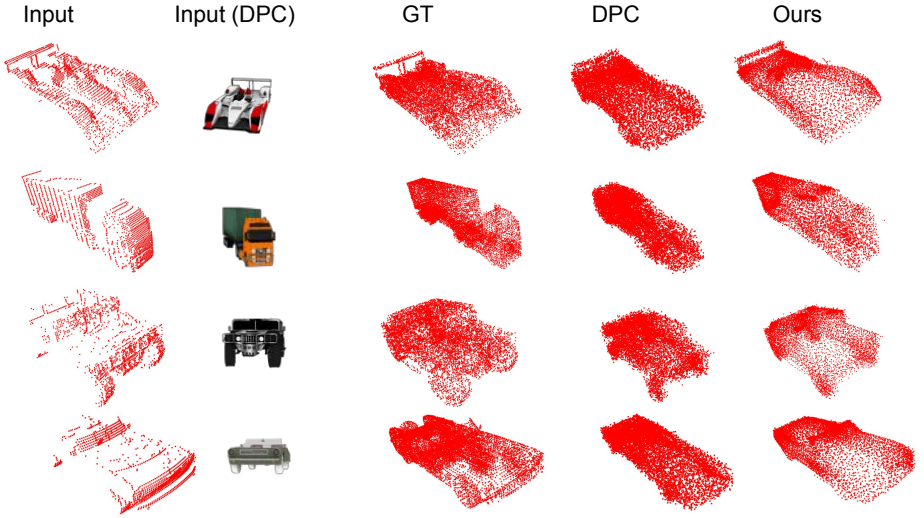


Figure 5: Failure cases on the Car category of the ShapeNet dataset. We compare our method with the results of DPC [4]. Our method fails to create the detailed shapes of various sports cars. Further, for the truck in the second row, our method fails to create a gap between the front and back of a truck.

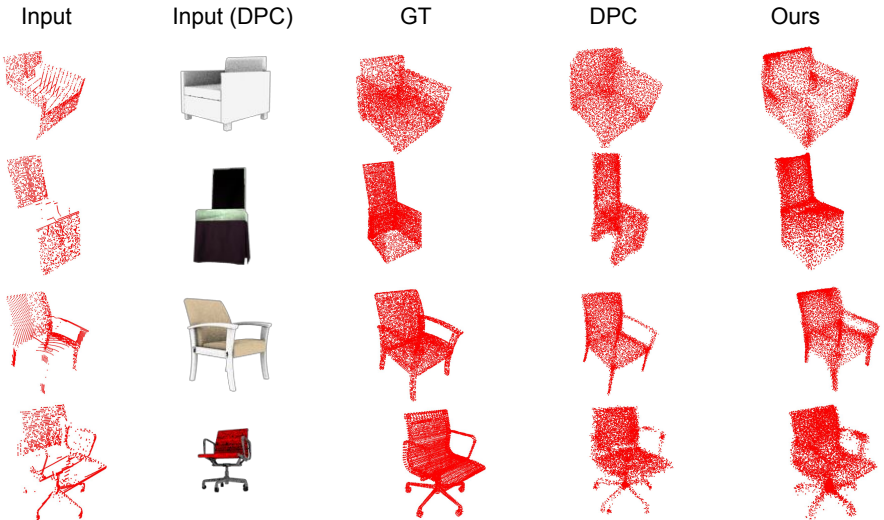


Figure 6: Success cases on the Chair category of the ShapeNet dataset. We compare our method with the results of DPC [4]. Our method is able to show finer completion results on different types of chairs such as a sofa and desk chair than DPC [4]. It is able to complete the front, back, and arms of the chair.

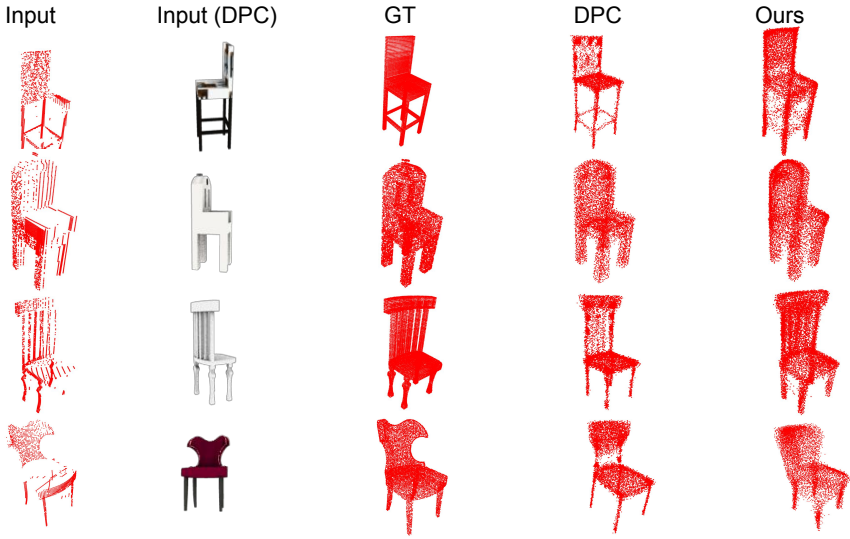


Figure 7: Failure cases on the Chair category of the ShapeNet dataset. We compare our method with the results of DPC [4]. It can be observed in these cases that the network generates noisy results especially near the legs of a chair.

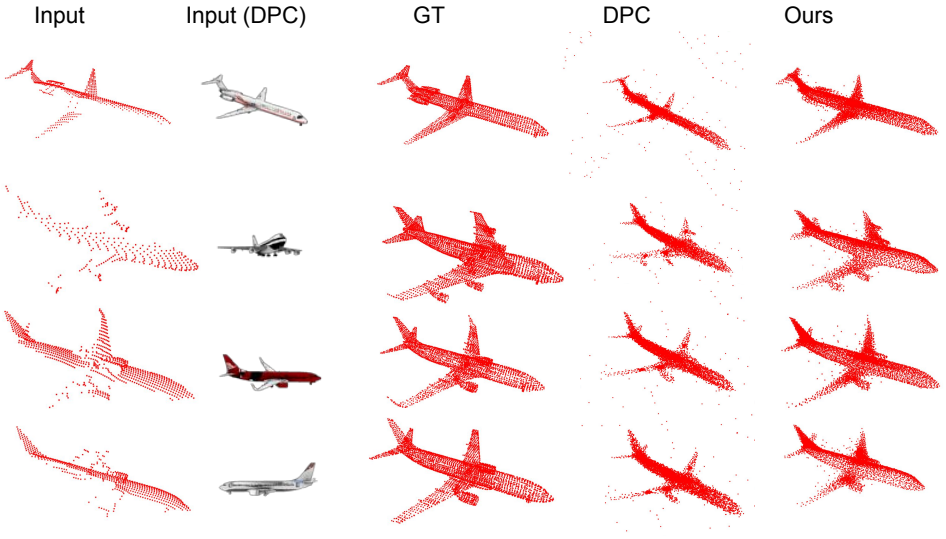


Figure 8: Success cases on the Plane category of the ShapeNet dataset. We compare our method with the results of DPC [4]. It can be observed that the network is able to complete the front, back and wings of the planes.

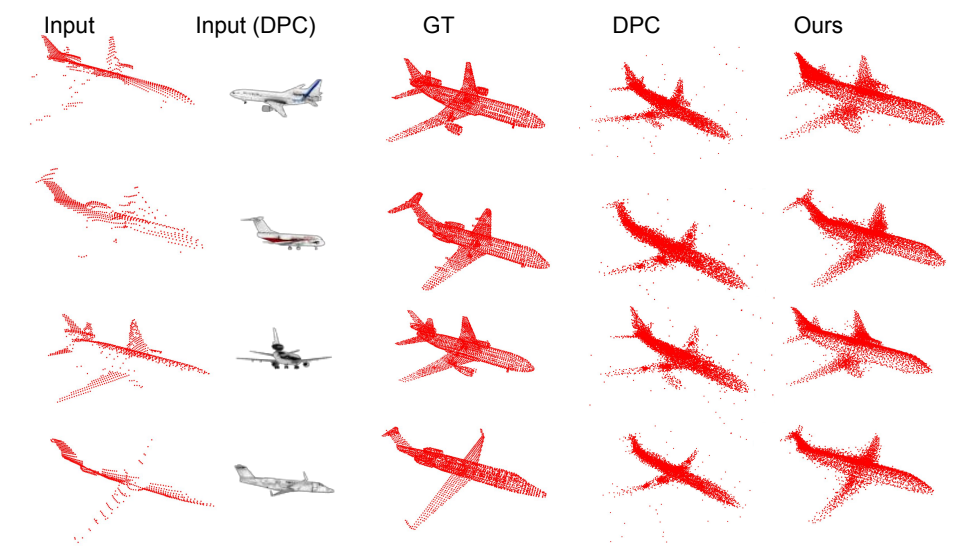


Figure 9: Failure cases on the Plane category of the ShapeNet dataset. We compare our method with the results of DPC [14]. The network generates some noisy points near the wings of the planes.

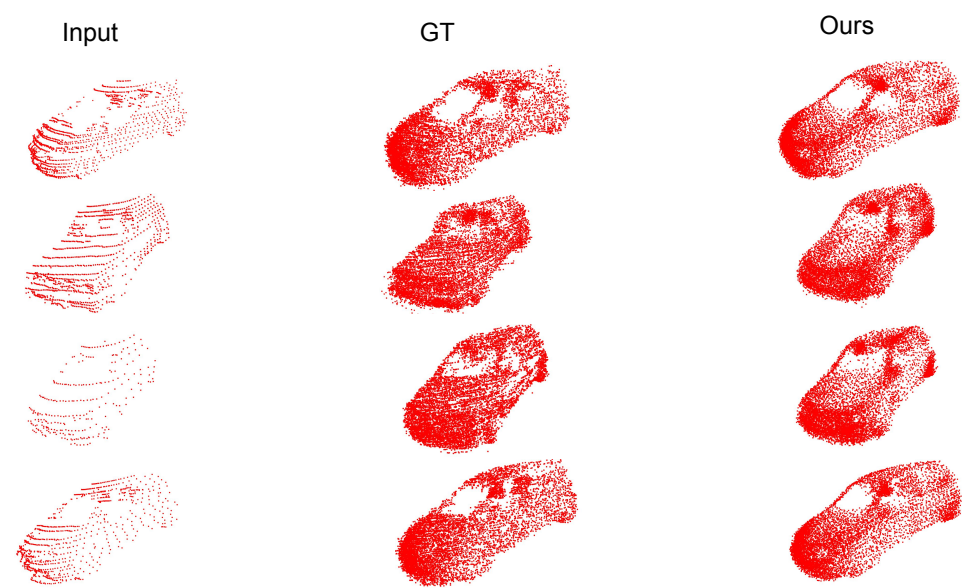


Figure 10: Completion of partial point cloud cars from the LiDAR scans of the Semantic KITTI dataset (first and second row). The third and fourth row show some failure cases of the network where the network is unable to generate the smaller details such as a tire of a car.

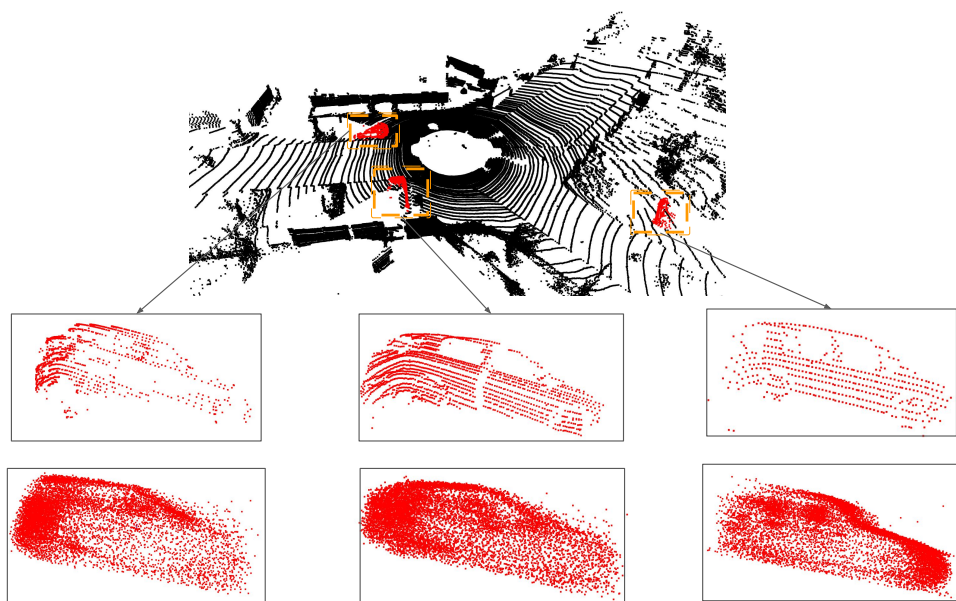


Figure 11: Completion of partial point cloud of cars in a LiDAR scan of the Semantic KITTI dataset.

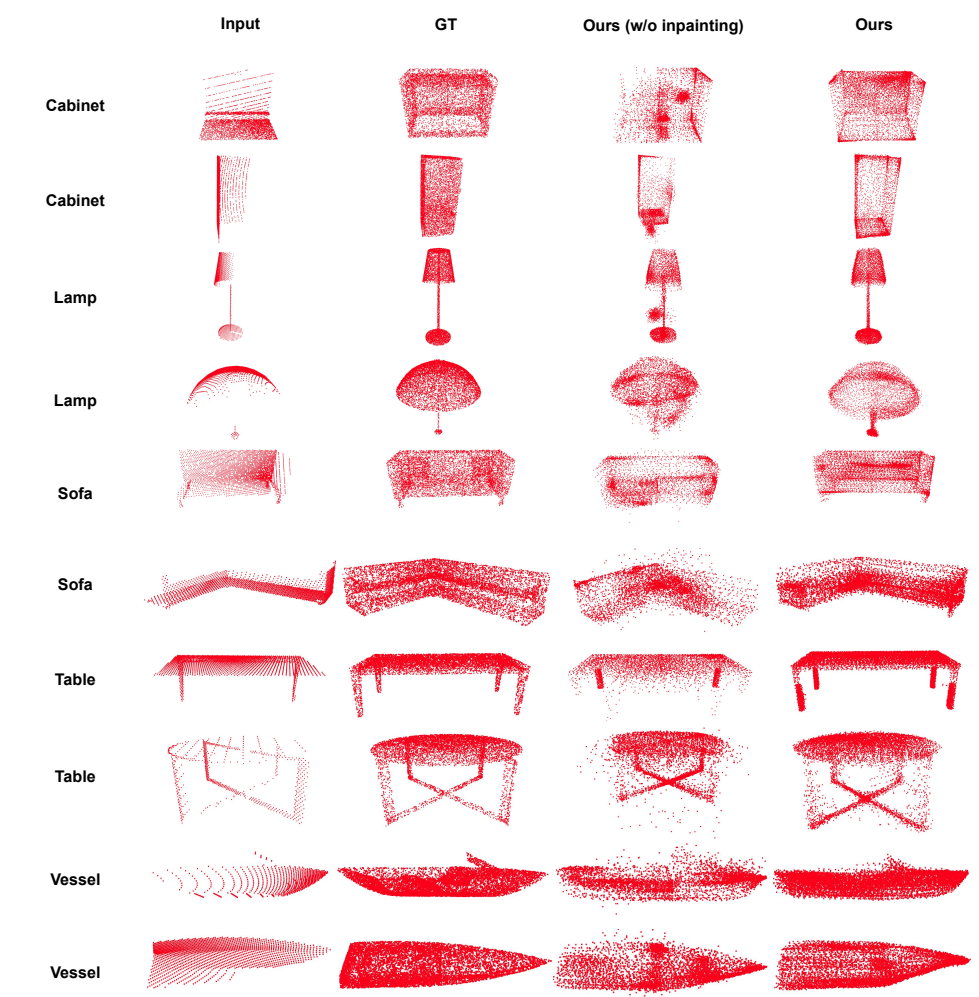


Figure 12: Qualitative Results on five categories of Shapenet compared to our ablation of without inpainting.

## References

- [1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, 2019.
- [2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.
- [3] Jiayuan Gu, Wei-Chiu Ma, Sivabalan Manivasagam, Wenyuan Zeng, Zihao Wang, Yuwen Xiong, Hao Su, and Raquel Urtasun. Weakly-supervised 3d shape completion in the wild. pages 283–299, 2020.
- [4] Eldar Insafutdinov and Alexey Dosovitskiy. Unsupervised learning of shape and pose with differentiable point clouds. In *NeurIPS*, pages 2802–2812, 2018.
- [5] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. pages 365–381, 2020.
- [6] Wentao Yuan, David Held, Christoph Mertz, and Martial Hebert. Iterative transformer network for 3d point cloud. *arXiv preprint arXiv:1811.11209*, 2018.
- [7] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *3DV*, pages 728–737, 2018.