ACL-SPC: Adaptive Closed-Loop system *for* Self-Supervised Point Cloud Completion

Sangmin Hong[1*] Mohsen Yavartanoo[2*] Reyhaneh Neshatavar[2] Kyoung Mu Lee[1,2]
[1]IPAI, [2]Dept. of ECE & ASRI, Seoul National University, Seoul, Korea
{mchiash2,myavartanoo,reyhanehneshat,kyoungmu}@snu.ac.kr

Abstract

Point cloud completion addresses filling in the missing parts of a partial point cloud obtained from depth sensors and generating a complete point cloud. Although there has been steep progress in the supervised methods on the synthetic point cloud completion task, it is hardly applicable in real-world scenarios due to the domain gap between the synthetic and real-world datasets or the requirement of prior information. To overcome these limitations, we propose a novel self-supervised framework ACL-SPC for point cloud completion to train and test on the same data. ACL-SPC takes a single partial input and attempts to output the complete point cloud using an adaptive closed-loop (ACL) system that enforces the output same for the variation of an input. We evaluate our ACL-SPC on various datasets to prove that it can successfully learn to complete a partial point cloud as the first self-supervised scheme. Results show that our method is comparable with unsupervised methods and achieves superior performance on the real-world dataset compared to the supervised methods trained on the synthetic dataset. Extensive experiments justify the necessity of self-supervised learning and the effectiveness of our proposed method for the real-world point cloud completion task. The code is publicly available from this link.

1. Introduction

Along with the development of autonomous driving cars and robotics, the usage of depth sensors such as LiDARs has increased. These sensors can collect numerous points in the 3D space, and the combination of these points forms a 3D representation called a point cloud. Point cloud representation has been widely used in many applications as it is highly convertible to other 3D data representations, *e.g.*, voxel and mesh, and accessible for obtaining information from the real world. However, point clouds obtained from a real-world sensor, *e.g.*, a LiDAR, are often incomplete and sparse due to occlusion, limitations of sensor resolution, and viewing angle [49] leading to loss of some geometric information and difficulty in proceeding with further applications *e.g.*, object detection [26] and object segmentation [7]. We define such point clouds as partial point clouds. Therefore, point cloud completion is a crucial task that infers completing geometric 3D shapes by using such partial point cloud observations.

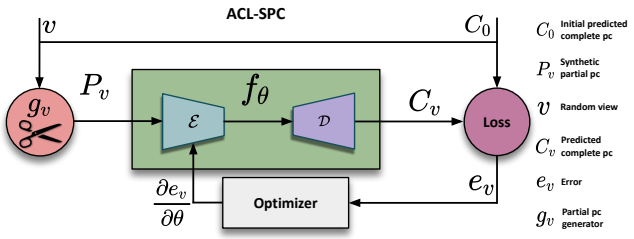


Figure 1. **Overview of our proposed pipeline.** We first generate C_0 using the initial partial point cloud. Then, multiple synthetic point clouds P_v are generated from the random views of C_0. We input the generated P_v to the network and make predicted complete point clouds. We take the loss between C_0 and C_v to optimize the parameters of the network f_θ.

With the advent of deep learning, previous data-driven works [40, 43, 49] have been able to solve this task using complete point cloud ground-truths. Even though such methods have achieved decent performance, they are not applicable in real-world scenarios where the ground-truth point clouds are not easy to obtain. For these reasons, researchers have recently attempted to overcome the lack of high-quality and large-scale paired training data using multiple views of the point cloud in unsupervised and weakly-supervised manners. Especially, recent methods [15, 21] leverage multi-view consistency of the desired object, which shows effectiveness in supervising 3D shape prediction. PointPnCNet [21] claims that its method is based on self-supervised learning. However, combining multi-view consistency enables reconstructing a complete 3D point cloud and can be weak supervision. Moreover, collecting multiple partial views of an object in real-world

*equal contribution

scenarios is difficult as gathering ground-truth point clouds. Therefore, the necessity for multi-view consistency prevents this method from being fully self-supervised. Meanwhile, other methods [6, 13, 41, 50] exploit unpaired partial and complete point clouds [6, 41] or pre-trained models [13, 50] on synthetic data to overcome the difficulty of collecting ground-truth. However, the need for unpaired data limits the methods' applicability to a few categories.

To overcome the challenges mentioned above, we propose a novel and the first self-supervised method called ACL-SPC for point cloud completion using only a single partial point cloud. We develop an adaptive closed-loop (ACL) [2] system as shown in Figure 1 to design our self-supervised point cloud completion framework ACL-SPC. In ACL-SPC, an encoder adaptively reacts to the variance in the input by adjusting its parameters to generate the same output. Using our developed ACL, our method tries to generate a complete point cloud from a single partial input captured from an unknown viewpoint without any prior information or multi-view consistency and also simulates several synthetic partial point clouds from the reconstructed point cloud. Under our defined novel loss function, our ACL-SPC can learn to generate the same complete point cloud from all such synthetic point clouds and the initial partial point cloud without any supervision. In the experiments, we demonstrate the ability of our method to restore a complete point cloud and the effect of our designed loss functions on saving fine details and improving quantitative performance. We also evaluate our method with various datasets, including real-world scenarios, and verify that our method can be applied in practice. Evaluation results show that our method is comparable to other unsupervised methods and performs better than the supervised method trained on a synthetic dataset.

Our main contributions can be summarized as follows:

- We propose ACL-SPC by developing an adaptive control-loop ACL framework to solve the point cloud completion problem in a self-supervised manner.
- We also design an effective self-supervised loss function to train our method without requiring any other information and using only a single partial point cloud taken from an unknown viewpoint.
- Our method achieves superior performance in real-world scenarios compared to methods trained on synthetic datasets and comparative performance among other unsupervised methods.

2. Related Works

2.1. Supervised point cloud completion

Point cloud completion is the task of reconstructing a complete geometry of a shape from partial point clouds. Before the advancement of the deep neural network, some traditional geometric-based methods [9, 22, 32] have been attempted to complete shapes using the geometric priors from a partial input without any external data. Other methods [20, 29, 30, 36, 38] have been proposed to handle the point cloud completion task by utilizing the symmetry property of the object to complete the incomplete parts.

With the development of deep learning, some learning-based methods [9, 23, 34] have shown up to solve complete the partial point cloud using a large amount of data. However, these methods have converted a partial point cloud into voxels to apply convolutional neural networks (CNNs), which leads to computational complexity and losing some geometrical information of point clouds. PCN [49] as the first data-driven approach learned a completion network directly from point clouds rather than converting to other representations. Further, various works have proposed developed architectures using a novel rooted tree structure [37], 3D grids as the intermediate representation [43], feedback refinement module [46], and transformers [48, 52] to improve the performance. However, these methods deviate from real-world scenarios, as gathering ground truth point clouds is cost-inefficient and not practical.

2.2. Unsupervised point cloud completion

Due to the aforementioned limitations of supervised methods for point cloud completion, existing unsupervised approaches [15, 16, 21] have been proposed to handle point cloud completion tasks where ground truth data are unavailable. Meanwhile, weakly supervised methods [15, 16] have attempted to predict the complete point cloud using multiple partial views, which are not always available in real-world scenarios. Later, PointPnCNet [21] introduced an inpainting framework with geometric consistency to overcome the above issue and claim that the method is the first self-supervised work for this task. Nevertheless, this method has exploited geometric consistency between multi-views of an object and showed that without this supervision, they could not complete the partial point cloud. Moreover, there have been attempts to utilize unpaired complete point clouds from synthetic datasets to solve the difficulty of accessing ground-truths [6, 13, 41, 50]. As a domain gap exists between unpaired complete point clouds and partial point clouds, these methods design architectures that can transform from one domain to the other and eventually solve the issue. However, these methods can only be suitable for categories available in synthetic datasets.

2.3. Self-supervised Learning

Self-supervised learning has attracted increasing attention in computer vision due to its practicality and ability to avoid the need for expensive annotated datasets. Following the advances in CNNs, recent self-supervised learning

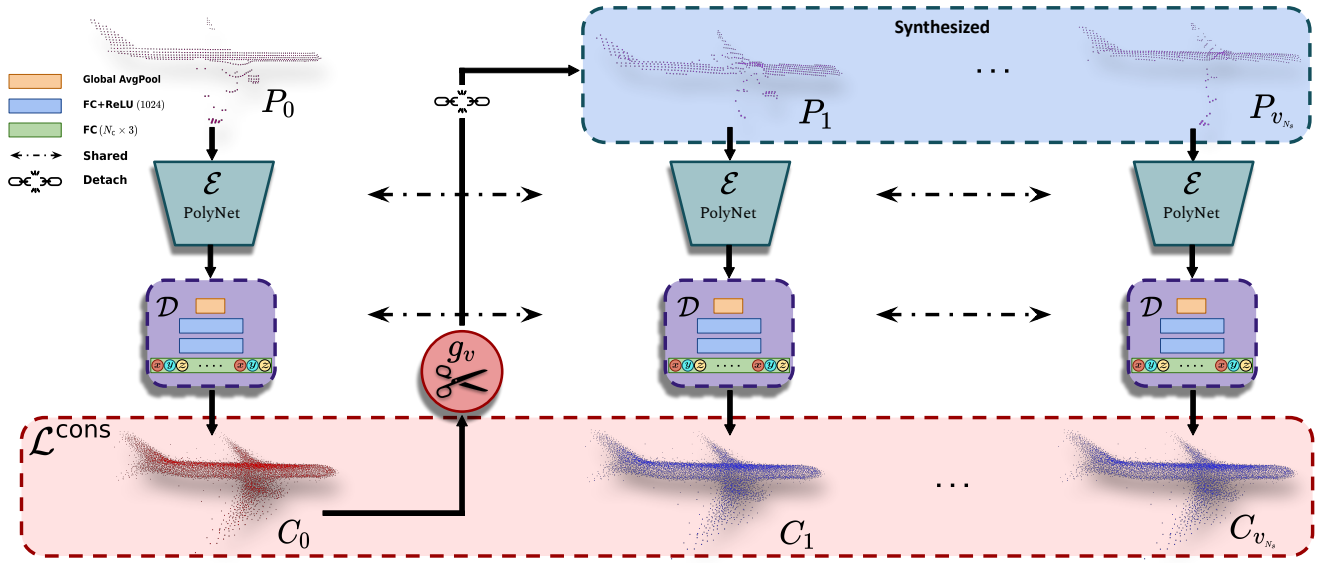


Figure 2. **The framework of ACL-SPC.** Our framework consists of an encoder-decoder style network where the parameters are shared between the objects. The network adopts the PolyNet [47] as the encoder and three fully connected (FC) layers as the decoder. Our network first takes the input partial point cloud and generates an estimated complete point cloud. Using this point cloud, we make multiple synthesized partial point clouds as new inputs. Again, the network outputs estimated complete point clouds from the synthesized partial point clouds. We apply consistency loss between multiple estimated complete point clouds and optimize the parameters of the network.

methods have incorporated generative [14, 17, 18, 31, 39, 54] and contrastive approaches [1, 10, 44], to learn the features from unlabeled data where an input itself provides supervision. Furthermore, researchers [11, 19, 27, 33, 35, 45, 51, 53] have started to apply self-supervised methods on point clouds to overcome the cumbersome task of annotating. These works have successfully shown great performance on feature learning to handle tasks such as classification [11, 19, 27, 33, 35, 45, 53], segmentation [11, 19, 27, 33, 35, 45, 53] or upsampling [51]. In this way, we propose the first self-supervised method for point cloud completion using only a partial point cloud as input without any prior information.

3. Method

In control theory [24, 25], closed-loop systems have many applications in various areas such as aerospace, electronics, and biomedical. Especially adaptive closed-loop (ACL) is a system where a controller automatically gives a compensated signal for the variation in the system so that the overall result remains the same [2, 24, 25]. In an ACL system, a controller outputs an appropriate signal after receiving feedback from the error between the desired output and the generated one. Meanwhile, obtaining a complete point cloud generator invariant to the view of captured partial point clouds is essential for point cloud completion tasks. We believe that the aforementioned attribute of the ACL system can be utilized for this task because it is appropriate for constructing the same complete point cloud, whatever a partial point cloud of an object comes in as an input. Therefore, we develop the concept of ACL for point cloud completion and introduce a novel self-supervised partial point-cloud completion framework (ACL-SPC).

3.1. ACL-SPC

Using a conventional ACL as a point cloud completion system requires the target complete the point cloud and several partial point cloud observations to optimize the system. However, accessing the target complete point cloud and several partial observations in real-world scenarios is not always possible. Therefore, we develop the ACL system such that it generates the complete point cloud using only a single partial observation and without requiring the target complete point cloud for optimizing the system, as shown in Figure 2. To achieve this goal, we employ a learnable model f_θ on an input partial point cloud observation $P_0 \in \mathbb{R}^{N_p \times 3}$ as follows:

$$C_0 = f_\theta(P_0), \tag{1}$$

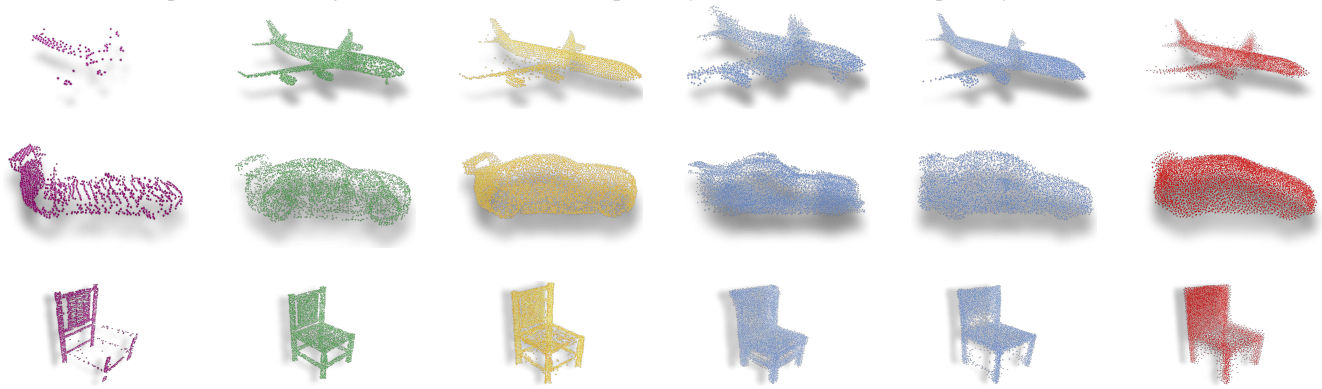
where $C_0 \in \mathbb{R}^{N_c \times 3}$ is the generated complete point cloud and N_p and N_c refers to the number of points in the input and output point cloud, respectively. Then we apply a partial point cloud generator g_v to generate a set of partial point clouds P_v from the generated point cloud C_0 as follows:

$$\forall v \in \{v_i\}_{i=1}^{N_s},\ P_v = g_v(C_0), \tag{2}$$

where v_i is a random parameter to generate N_s number of different partial point clouds. We again employ the same

Supervision	Method	Airplane			Car			Chair			Average		
		P↓	C↓	CD↓	P↓	C↓	CD↓	P↓	C↓	CD↓	P↓	C↓	CD↓
Unsupervised	DPC [16]	-	-	3.91	-	-	3.47	-	-	4.30	-	-	3.89
	Gu *et al*. [15]	0.91	1.05	**1.95**	1.27	1.41	**2.68**	1.69	1.64	**3.33**	1.29	1.36	**2.65**
	PointPnCNet [21]	1.58	1.74	3.32	1.98	2.98	4.96	2.72	2.68	5.40	1.75	2.46	4.56
Self-supervised	**Ours**	1.20	**0.80**	2.01	1.65	**1.28**	2.93	2.25	**1.46**	3.71	1.70	**1.18**	2.88

Table 1. **Quantitative results on three categories airplane, car and chair.** We also calculate average values among the categories. P, C, and CD refers to precision, coverage, and Chamfer distance, respectively. All the values are multiplied by 100.



(a) Input (b) GT (c) Multi-View (d) GRNet [43] (e) Gu *et al*. [15] (f) **Ours**

Figure 3. **Qualitative comparison on the ShapeNet dataset.** We visualize a) the partial input, b) completed ground truth point cloud, c) multi-view point cloud, results on d) GRNet, e) Gu *et al*., and f) ours. The multi-view point cloud is the concatenation of five random partial views of an object. Our result show that our method can recover most of the missing parts from the partial input.

model f_θ on the generated synthetic partial point clouds P_v to generate the same point cloud C_0 as follows:

$$\forall v \in \{v_i\}_{i=1}^{N_s},\ C_v = f_\theta(P_v) = f_\theta(g_v(C_0)), \qquad (3)$$

where C_v is the predicted complete point cloud for the generated point cloud C_0. Then, we optimize the system by a loss function between the predicted complete point clouds C_v and the generated initial complete point cloud C_0. Accordingly, our ACL-SPC learns to generate the same complete point cloud for different partial point cloud observations P_v synthesized from the generated C_0. Since the learnable model f_θ is optimized to map any partial point cloud P_v to its corresponding target complete point cloud C_0, the generated point cloud C_0 as the output of f_θ on the input partial point cloud P_0 must predict the target complete point cloud.

3.2. Loss functions

To train our network f_θ, we use two self-supervised loss functions. First, to optimize the ACL-SPC and guarantee to generate the same predicted complete point clouds, we design the consistency loss function $\mathcal{L}^{\text{cons}}$ between the predicted complete point clouds C_v and C_0 as follows:

$$\mathcal{L}^{\text{cons}} = \frac{1}{N_c \times N_s} \sum_{v \in \{v_i\}_{i=1}^{N_s}} ||C_v - C_0||_2^2, \qquad (4)$$

where $||.||_2$ represents the L_2 norm. We further utilize the weighted Chamfer distance [21] loss $\mathcal{L}^{\text{wcd}}$ between the predicted complete point cloud C_0 and the input partial point cloud P_0. The weighted Chamfer distance is invariant to the permutation of the order of points which is composed of two parts with the corresponding weights as follows:

$$\mathcal{L}^{\text{wcd}} = \frac{\alpha}{N_c} \sum_{p \in C_0} \min_{q \in P_0} ||p-q||_2 + \frac{\beta}{N_p} \sum_{q \in P_0} \min_{p \in C_0} ||q-p||_2. \qquad (5)$$

The first term measures the mean distance for each point in the source point cloud C_0 to the closest point in the target point cloud P_0, while the second term measures the mean distance from each point in the target point cloud P_0 to its nearest point in the source point cloud C_0. Therefore, the second term leads the predicted point cloud C_0 to cover the points in the target point cloud P_0 while the first term performs as a regularizer. We set $\alpha = 0.1$ and $\beta = 0.9$ to enforce the points to cover the non-missing parts of the point cloud and let the remaining points be flexible to fill in the missing parts. The total loss $\mathcal{L}^{\text{total}}$ is the weighted summation of two aforementioned loss functions as follows:

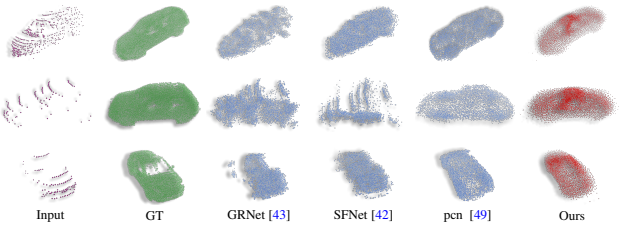
$$\mathcal{L}^{\text{total}} = \lambda_{\text{cons}} \mathcal{L}^{\text{cons}} + \mathcal{L}^{\text{wcd}}. \qquad (6)$$

3.3. Training details

Our model f_θ includes an encoder $\mathcal{E}$ which learns the local and global features from the partial input point clouds

Supervision	Method	P	C	CD
Supervised	GRNet [43]	**4.63**	6.90	**11.53**
	SFNet [42]	14.12	12.64	26.76
	pcn [49]	9.83	17.96	27.79
Unsupervised	Gu [15]	8.70	10.70	19.40
	PointPnCNet [21]	9.00	10.00	19.00
Self-supervised	**Ours**	11.67	**5.63**	17.30

(a) Quantitative results.



(b) Qualitative results.

Figure 4. **Evaluation on the SemanticKITTI [3] dataset.** We compare our results with various supervised and unsupervised methods by a) evaluating our quantitative results in terms of precision (P), coverage (C), and Chamfer distance (CD) and b) visualizing their outputs. For supervised methods, we use their pretrained models on the synthetic PCN [49] dataset to evaluate on the SemanticKITTI.

and a decoder $\mathcal{D}$ to generate the points of the complete point clouds as shown in Figure 2. We use PolyNet [47], a powerful spatial graph CNN, as the encoder $\mathcal{E}$ which consists of four squeezed PolyConv layers with the sizes of 64, 128, 256, and 512, respectively. We apply a random downsampling followed by max-pooling after the first three PolyConv layers to reduce the point size to 512, 128, and 32, respectively. We employ a global average pooling after the last PolyConv layer to eliminate the spatial dependency and obtain 512 invariant features to various partial observations and point permutations. We use three fully connected (FC) layers as the decoder $\mathcal{D}$ with the sizes of 1024, 1024, and $N_c \times 3$, respectively, where the ReLU non-linear activation function is applied on the outputs of the first and the second FC layers. In fact, g_v is a non-learnable function that generates the synthetic partial point clouds by projecting the generated complete point cloud to a depth map at a random view v from azimuth $[0°, 360°]$ and elevation $[-20°, 40°]$. We then back-project the depth map into 3D. To avoid double backpropagation and optimize f_θ only once, we use the detach operator [28] as shown in Figure 2. We exploit the Adam optimizer and optimize the model for every 32 different input partial point clouds and their N_s synthetic partial point clouds and set $N_s = 8$ and $\lambda_{\text{cons}} = 10$ in our baseline. For the inference, we feed the input partial point cloud to the trained model f_θ to directly generate the corresponding complete point cloud, which takes $12ms$ on average for each sample with a NVIDIA RTX 2080Ti.

4. Experiments

4.1. Datasets and metrics

In this section, we discuss the training and evaluation datasets and the metrics used to compare our proposed method ACL-SPC with the related methods.

Synthetic Datasets: ShapeNet [5] is a large-scaled dataset including curated 3D shapes represented by CAD models, which consists of 55 categories. Among them, we focus on three categories, airplanes, cars, and chairs, to maleaaintain the same setup as the previous works [15, 16, 21]. Followed by these works, we capture RGB-D data from five random views for each object, transfer them to 3D, and resample 3096 points of them to generate a set of partial point clouds. The ground truth complete point clouds with the fixed 8192 number of points are used for evaluation.

Real-World Datasets: Similar to previous works [6,13,50], we evaluate our method using three sources of real scans: ScanNet [8] (chairs and tables), MatterPort3D [4] (chairs and tables), and KITTI [12] (car). The ScanNet and MatterPort3D datasets are richly annotated 3D reconstructions of indoor environments, whereas the KITTI dataset is of outdoor scenes. We resample the points to 2048 points to match the previous work's settings [6, 13, 50]. SemanticKITTI dataset [3] is derived from the KITTI dataset [12], including only the car objects which are captured in multi-views from sequence 00 to 10 when parked. We take out the sequence 08 for testing and exploit the other sequences for training. Note that the input points are resampled to 1024 for convenience. As the SemanticKITTI dataset has no ground truth complete point cloud, we follow the same steps as in the previous work [15] to generate them by aggregating partial point clouds.

Metrics. We utilize the Chamfer distance (CD) between the reconstructed point cloud and the ground truth to evaluate the performance of our ACL-SPC method. Chamfer distance is the average distance between each point in a point cloud and the nearest point in the other as follows:

$$\begin{aligned} \mathcal{CD}(C_0, \text{GT}) = &\frac{1}{N_c} \sum_{p \in C_0} \min_{q \in GT} ||p - q||_2 \\ &+ \frac{1}{N_g} \sum_{q \in GT} \min_{p \in C_0} ||q - p||_2, \end{aligned} \tag{7}$$

where GT is defined as the ground truth complete point cloud with N_g points. The first and second term refers to precision and coverage, respectively. Precision infers how much the generated points are distributed well compared to the ground-truth data, while coverage refers to how much the missing parts of the partial point cloud are filled in. Accordingly, the coverage is an important metric

Supervision	Method	ScanNet				MatterPort3D				KITTI	
		Chair		Table		Chair		Table		Car	
		UCD↓	UHD↓	UCD↓	UHD↓	UCD↓	UHD↓	UCD↓	UHD↓	UCD↓	UHD↓
Unsupervised	pcl2pcl [6]	17.3	10.1	9.1	11.8	15.9	10.5	6.0	11.8	9.2	14.1
	ShapeInversion [50]	3.2	10.1	3.3	11.9	3.6	10.0	3.1	11.8	2.9	13.8
	+UHD [50]	4.0	9.3	6.6	11.0	4.5	9.5	5.7	10.7	5.3	12.5
	Cycle4Comp. [41]	5.1	6.4	3.6	5.9	8.0	8.4	4.2	6.8	3.3	5.8
	DE [13]	2.8	5.4	2.5	5.2	3.8	6.1	2.5	5.4	1.8	3.5
	OptDE [13]	2.6	5.5	1.9	**4.6**	3.0	5.5	**1.9**	5.3	**1.6**	**3.5**
Self-supervised	**Ours**	**1.4**	**4.7**	**1.8**	5.1	**1.8**	**4.8**	2.1	**4.9**	2.0	4.9

Table 2. **Quantitative results on the real-world datasets [4, 8, 12] in the categories of chair, table, and car.** We evaluate the method in terms of UCD and UHD where the values are multiplied by 10^2 and 10^4 respectively.

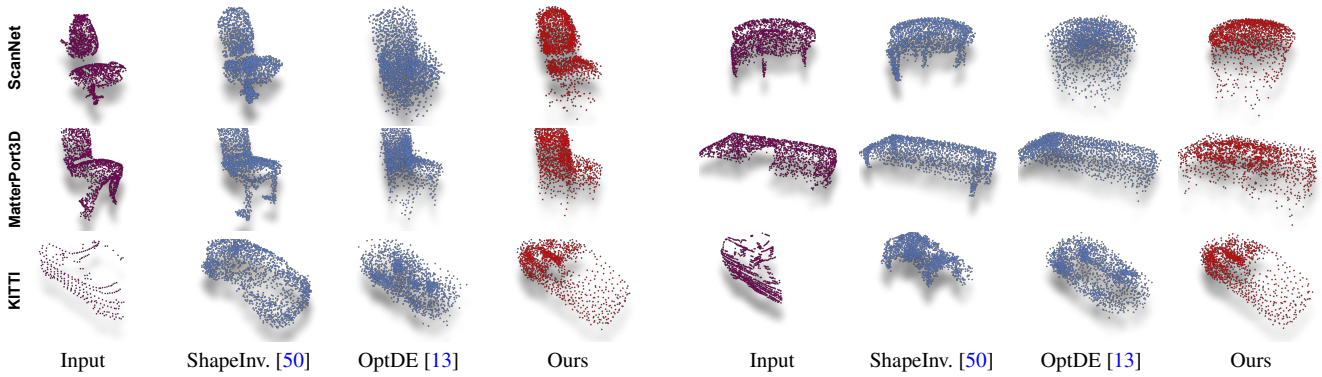


Figure 5. **Qualitative results on the real-world datasets [4, 8, 12] in the categories of chair, table, and car.**

for point cloud completion tasks, reflecting the effectiveness of the methods to fill the missing parts. Additionally, we utilize two metrics called Unidirectional Chamfer Distance (UCD) and Unidirectional Hasudorff Distance (UHD) for the real-world datasets [4, 8, 12] in the same way as previous works [6, 13, 50]. To calculate UCD, we obtain the first term of $\mathcal{CD}(P_0, C_0)$ between the partial input point cloud P_0 and the predicted complete point cloud C_0 using equation 7. Similarly, we measure the UHD with the single side of Hausdorff distance as follows:

$$\mathcal{UHD}(P_0, C_0) = \max_{p \in P_0} \min_{q \in C_0} ||p - q||_2 \tag{8}$$

Although the two metrics do not reflect the completeness of shape, they can give a fair comparison where ground-truth data is unavailable.

4.2. Evaluation on synthetic dataset

In this section, we qualitatively and quantitatively evaluate our ACL-SPC method on the ShapeNet dataset and compare the results with the related methods [15, 16, 21]. We train our network for each category separately with 1000 epochs by a learning rate of 0.001, which is decayed by 0.5 for every 200 epochs to generate $N_c = 8192$ points. We visualize and compare the results of our method with the supervised [43] and unsupervised [15] methods in Figure 3.

We note that GRNet [43] and Gu *et al*. [15] utilize the GT and the multi-view information as their supervision, respectively. Using the multi-view information leads to achieving a high-quality appearance for Gu *et al*. [15] because the concatenated point cloud from five random partial point clouds is almost as the GT as shown in Figure 3c. Even without this information, our ACL-SPC method shows comparable results in completing the missing parts of the input in a fully self-supervised manner. Moreover, our quantitative results in Table 1 show that our method can outperform the unsupervised methods DPC [16] and PointPnCNet [21] with a large gap 1.01 and 1.68 with the CD on average while performing only 0.23 lower performance compared to Gu *et al*. method. Therefore, our method can learn even better without any prior information compared to some of the unsupervised methods that have leveraged multiple partial views. Moreover, our method outperforms all the unsupervised methods by the coverage metric, which shows its superiority in covering the missing parts.

4.3. Evaluation on real-world dataset

We evaluate our self-supervised method ACL-SPC on SemanticKITTI [3] dataset and compare the results with both supervised and unsupervised methods. We train our network for 500 epochs with a learning rate of 0.001, and it is decayed by 0.5 for every 200 epochs to output $N_c = 8192$

number of points. We exploit pretrained models of the supervised methods [42,43] on the synthetic PCN [49] dataset to test on the real-world SemanticKITTI dataset. As shown in Figure 4a, our method outperforms the unsupervised methods Gu *et al*. [15] and PointPnCNet [21] in terms of coverage and CD. It also achieves better coverage compared to the supervised method GRNet [43], while it shows superior performance in all metrics compared to the supervised method SFNet [42]. Moreover, we can see through Figure 4b that the supervised methods perform poorly on the real-world dataset compared to our method due to the domain gap with the synthetic dataset, which emphasizes the generalizability of our self-supervised method in real-world scenarios. Additionally, we can validate that the coverage is more important than other metrics as our method shows better qualitative results than GRNet [43] even though our method shows larger precision and CD values.

Furthermore, we quantitatively and qualitatively evaluate our method on ScanNet [8], MatterPort3d [4], and KITTI [12] dataset as shown in Table 2 and Figure 5, respectively. In contrast to the unsupervised methods [6, 13, 41, 50] that require synthetic datasets in addition to the real-world datasets for training because they either need unpaired ground truth [6, 41] or pretrained model [13, 41], our method is trained only on real-world datasets. Except for some metrics in the table categories, our method generally performs better than the state-of-the-art [13] in ScanNet and MatterPort3D datasets as shown in Table 2. However, in the KITTI dataset, our method is slightly behind the state-of-the-art by 0.4 and 1.4 on UCD and UHD, respectively. We also qualitatively compare the results with unsupervised methods [13, 50] as shown in Figure 5. The ShapeInversion [50] generally fails to generate the missing parts in some cases, such as the chair class of MatterPort3D datasets. Meanwhile, OptDE [13] generates lots of noise in most samples, especially in the categories of the ScanNet [8] dataset. In contrast, our method generates plausible points in the missing parts of the input in all samples. One drawback of our results is that the output point cloud is not uniformly distributed, being sparser in the input regions where points were absent. Thus, even without requiring any synthetic datasets, we illustrate that our method is competitive compared to other unsupervised methods.

4.4. Ablation study

In this section, we further analyze our novel ACL-SPC method by extensive ablations studies on test-time adaptation, the effect of defined loss functions, the number of synthesized data, training on the multi-class dataset, and training on a dataset including only one view of objects.

Supervision	**P↓**	**C↓**	**CD↓**
Supervised	17.29	8.57	25.86
Self-supervised	11.67	**5.63**	17.30
Test-time adapt.	**9.62**	7.09	**16.71**

Table 3. **Evaluation on Test-time adaptation.** We train the network in three modes: supervised, self-supervised, and test-time adaptation. The values are multiplied by 100.

4.4.1 Test-time adaptation

Similar to previous works [6, 13, 41, 50], we show that our method can also be effective for test-time adaptation. We train and test our network in three different schemes, as shown in Table 3. First, we train the network in the supervised setting on a synthetic dataset and then evaluate it on a real-world dataset. Second, we train our network in a self-supervised manner, without any pretraining, and train and test it on the real-world dataset. Finally, on the test-adaptation setting, it goes through the pretraining stage and then moves on to the adaptation stage with our ACL-SPC framework. Table 3 displays the precision, coverage, and CD of each experiment setting. The results show the suitability of our ACL-SPC method not only for self-supervised learning but also for test-time adaption.

4.4.2 Effect of each loss

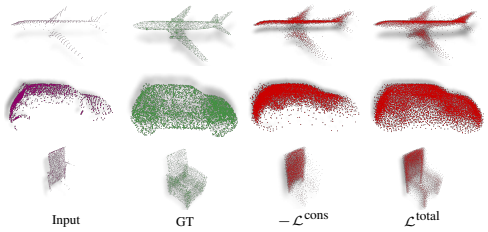
We evaluate the effects of each loss by taking out each loss at a time for each experiment. We report the quantitative results for the experiments without $\mathcal{L}^{\text{wcd}}$, without $\mathcal{L}^{\text{cons}}$, and with the total loss $\mathcal{L}^{\text{total}}$ in Figure 6a. Taking out $\mathcal{L}^{wcd}$ affects critically in the results as there is no guarantee to cover points in the input. On the other hand, excluding $\mathcal{L}^{\text{cons}}$ results in a worse coverage value which proves the importance of our proposed ACL-SPC to fill the missing parts of partial input point clouds. To visualize this effect, we qualitatively compare our results with and without $\mathcal{L}^{\text{cons}}$ in Figure 6b. Without $\mathcal{L}^{\text{cons}}$, only the input is covered while the missing parts are still uncovered. We note that since without $\mathcal{L}^{\text{wcd}}$ there is no constraint to generate points in the location of the input, the method produce all points in the same position.

4.4.3 Number of synthesized data

We analyze how the number of synthesized data can influence point cloud completion results. Table 4 presents the precision, coverage, and CD values when N_s is set to 1, 4, and 8. According to the results, we achieve the best performance by $N_s = 8$ on average among other setups, while there is no dramatic difference as it shows only 0.03, 0.01, and 0.03 difference compared to having one number of synthesized data. Consequently, having more synthesized data slightly enhances the performance.

Loss	Airplane			Car			Chair		
	P↓	C↓	CD↓	P↓	C↓	CD↓	P↓	C↓	CD↓
$-\mathcal{L}^{wcd}$	1.15	18.76	19.91	4.66	26.51	31.17	3.91	26.41	30.32
$-\mathcal{L}^{cons}$	**1.10**	1.22	2.32	**1.45**	2.11	3.56	**1.99**	1.74	3.73
$\mathcal{L}^{total}$	1.20	**0.81**	**2.01**	1.65	**1.28**	**2.93**	2.25	**1.46**	**3.71**

(a) Quantitative results.



(b) Qualitative results.

Figure 6. **Evaluation on the effects of loss functions.** We show a) precision (P) and coverage (C) values multiplied by 100 for each experiment with different losses, and b) qualitative results.

Ablation	Setup	Airplane			Car			Chair			Average		
		P↓	C↓	CD↓	P↓	C↓	CD↓	P↓	C↓	CD↓	P↓	C↓	CD↓
Num Syns	1	**1.19**	0.85	2.04	1.63	**1.27**	**2.90**	2.37	1.45	3.82	1.73	1.19	2.92
	4	1.23	0.82	2.05	**1.62**	1.30	2.92	2.55	**1.43**	3.97	1.80	**1.18**	2.98
	8	1.20	**0.81**	**2.01**	1.65	1.28	2.93	**2.25**	1.46	**3.71**	**1.70**	**1.18**	**2.89**
Class	Single	**1.20**	0.81	**2.01**	**1.65**	1.28	2.93	**2.25**	1.46	**3.71**	**1.70**	1.18	**2.89**
	Multi	1.40	**0.79**	2.19	1.66	**1.25**	**2.91**	2.35	**1.42**	3.76	1.80	**1.15**	2.96
Views	1	1.23	0.89	2.12	**1.63**	**1.27**	**2.90**	**2.15**	1.54	**3.69**	**1.67**	1.23	2.90
	5	**1.20**	**0.81**	**2.01**	1.65	1.28	2.93	2.25	**1.46**	3.71	1.70	**1.18**	**2.89**

Table 4. **Quantitative effects of the number of synthetic partial views, single-/multi-class training, and single-/multi-view training.** We present the values of precision (P), coverage (C), and Chamfer distance (CD) multiplied by 100.

4.4.4 Training on multi-class

In our main experiment, we train our model on a specific class and present the results in section 4.2. However, in real-world scenarios where the classes of objects are not identified, it is necessary to train the network on multi-class objects. Table 4 shows the quantitative results of the ShapeNet dataset when trained with multi-class objects. The results demonstrate not much difference in the performance as the precision and CD values were 0.10 and 0.07 worse, while the coverage was 0.03 better in the multi-class training. Thus, we believe that our network can learn to understand the appropriate features of a particular object even when there are various classes in the training set.

4.4.5 Single-view training

As mentioned in section 2, recent works [15, 16, 21] leverage multi-partial views for supervision. Even though our method does not take the advantage of multi-view supervision, we validate the power of our method to be trained on only a single view per object. As the training set includes multiple partial views for the same object, we take out these views and leave only one partial view to prove that it does not significantly affect our method's performance. According to Table 4, our method shows only 0.01 CD difference with the model trained with only the single partial view included in the training set. Through the results, we confirm that our method can still perform as expected even without multi-views of an object in the training set.

5. Conclusion

In this paper, we propose ACL-SPC, the first self-supervised point cloud completion method from only a single input partial point cloud. Our method learns to complete partial point clouds by adaptively controlling the output in a closed-loop system. We also introduce a consistency loss to generate the same complete point cloud and learn the geometric features of the object. Our extensive experiments demonstrate that our method can be more useful in real-world scenarios without performance degradation than other methods. In most cases, our method shows better performance in the coverage than precision showing the excellent performance of filling in the missing parts.

Limitations and future works. One remaining limitation of our method is that there is no constraint to not generate redundant points, which results in high precision values. To improve the precision and reduce noise, we will apply more constraints for future works. We will further find applications of our self-supervised framework in other point cloud restoration tasks such as denoising and upsampling.

Acknowledgement. This work was supported in part by the IITP grants [No.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University), No.2022-0-00156, No. 2021-0-02068, and No.2022-0-00156], and the NRF grant [No. 2021M3A9E4080782] funded by the Korea government (MSIT).

References

[1] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In *CVPR2022*, 2022. 3

[2] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013. 2, 3

[3] 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. 5, 6

[4] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *3DV*, 2017. 5, 6, 7

[5] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*. 5

[6] Xuelin Chen, Baoquan Chen, and Niloy J Mitra. Unpaired point cloud completion on real scans using adversarial training. In *ICLR*, 2020. 2, 5, 6, 7

[7] Xieyuanli Chen, Shijie Li, Benedikt Mersch, Louis Wiesmann, Jürgen Gall, Jens Behley, and Cyrill Stachniss. Moving object segmentation in 3d lidar data: A learning-based approach exploiting sequential data. *RA-L*, 2021. 1

[8] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 5, 6, 7

[9] Angela Dai, Charles Ruizhongtai Qi, and Matthias Niessner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *CVPR*, 2017. 2

[10] Bi'an Du, Xiang Gao, Wei Hu, and Xin Li. Self-contrastive learning with hard negative sampling for self-supervised point cloud learning. In *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 3

[11] Benjamin Eckart, Wentao Yuan, Chao Liu, and Jan Kautz. Self-supervised learning on 3d point clouds by learning discrete generative models. In *CVPR*. 3

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 5, 6, 7

[13] Jingyu Gong, Fengqi Liu, Jiachen Xu, Min Wang, Xin Tan, Zhizhong Zhang, Ran Yi, Haichuan Song, Yuan Xie, and Lizhuang Ma. Optimization over disentangled encoding: Unsupervised cross-domain point cloud completion via occlusion factor manipulation. In *ECCV*, 2022. 2, 5, 6, 7

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014. 3

[15] 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. In *ECCV*, 2020. 1, 2, 4, 5, 6, 7, 8

[16] Eldar Insafutdinov and Alexey Dosovitskiy. Unsupervised learning of shape and pose with differentiable point clouds. In *NeurIPS*, 2018. 2, 4, 5, 6, 8

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 3

[18] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *ICML*, 2017. 3

[19] Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. In *ECCV*, 2022. 3

[20] Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3d geometry. ACM, 2006. 2

[21] Himangi Mittal, Brian Okorn, Arpit Jangid, and David Held. Self-supervised point cloud completion via inpainting. In *BMVC*, 2021. 1, 2, 4, 5, 6, 7, 8

[22] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. ACM. 2

[23] Duc Thanh Nguyen, Binh-Son Hua, Khoi Tran, Quang-Hieu Pham, and Sai-Kit Yeung. A field model for repairing 3d shapes. In *CVPR*, 2016. 2

[24] Norman S. Nise. *Control Systems Engineering*. John Wiley & Sons, Inc., USA, 3rd edition, 2000. 3

[25] Katsuhiko Ogata et al. *Modern control engineering*, volume 5. Prentice hall Upper Saddle River, NJ, 2010. 3

[26] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *CVPR*, 2019. 1

[27] Yatian Pang, Wenxiao Wang, Francis E. H. Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, 2022. 3

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 5

[29] Mark Pauly, Niloy J. Mitra, Johannes Wallner, Helmut Pottmann, and Leonidas J. Guibas. Discovering structural regularity in 3d geometry. 2008. 2

[30] Joshua Podolak, Philip Shilane, Aleksey Golovinskiy, Szymon Rusinkiewicz, and Thomas Funkhouser. A planar-reflective symmetry transform for 3d shapes. 2006. 2

[31] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *ICML*, 2016. 3

[32] Kripasindhu Sarkar, Kiran Varanasi, and Didier Stricker. Learning quadrangulated patches for 3d shape parameterization and completion. *CoRR*, 2017. 2

[33] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. In *NeurIPS*. 3

[34] Abhishek Sharma, Oliver Grau, and Mario Fritz. Vconv-dae: Deep volumetric shape learning without object labels. In *ECCV*, 2016. 2

[35] Charu Sharma and Manohar Kaul. Self-supervised few-shot learning on point clouds. In *NeurIPS*. 3

[36] Minhyuk Sung, Vladimir G. Kim, Roland Angst, and Leonidas Guibas. Data-driven structural priors for shape completion. 2015. 2

[37] Lyne P. Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *CVPR*, 2019. 2

[38] S. Thrun and B. Wegbreit. Shape from symmetry. In *ICCV*, 2005. 2

[39] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *ICML*, 2016. 3

[40] Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *CVPR*, 2020. 1

[41] Xin Wen, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. Cycle4completion: Unpaired point cloud completion using cycle transformation with missing region coding. In *CVPR*, 2021. 2, 6, 7

[42] Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In *ICCV*, 2021. 5, 7

[43] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *ECCV*, 2020. 1, 2, 4, 5, 6, 7

[44] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*, 2020. 3

[45] Siming Yan, Zhenpei Yang, Haoxiang Li, Li Guan, Hao Kang, Gang Hua, and Qixing Huang. Implicit autoencoder for point cloud self-supervised representation learning, 2022. 3

[46] Xuejun Yan, Hongyu Yan, Jingjing Wang, Hang Du, Zhihong Wu, Di Xie, Shiliang Pu, and Li Lu. Fbnet: Feedback network for point cloud completion. In *ECCV*, 2022. 2

[47] Mohsen Yavartanoo, Shih-Hsuan Hung, Reyhaneh Neshatavar, Yue Zhang, and Kyoung Mu Lee. Polynet: Polynomial neural network for 3d shape recognition with polyshape representation. In *3DV*, 2021. 3, 5

[48] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, 2021. 2

[49] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *3DV*, 2018. 1, 2, 5, 7

[50] Junzhe Zhang, Xinyi Chen, Zhongang Cai, Liang Pan, Haiyu Zhao, Shuai Yi, Chai Kiat Yeo, Bo Dai, and Chen Change Loy. Unsupervised 3d shape completion through gan inversion. In *CVPR*, 2021. 2, 5, 6, 7

[51] Wenbo Zhao, Xianming Liu, Zhiwei Zhong, Junjun Jiang, Wei Gao, Ge Li, and Xiangyang Ji. Self-supervised arbitrary-scale point clouds upsampling via implicit neural representation. 3

[52] Haoran Zhou, Yun Cao, Wenqing Chu, Junwei Zhu, Tong Lu, Ying Tai, and Chengjie Wang. Seedformer: Patch seeds based point cloud completion with upsample transformer. In *ECCV*, 2022. 2

[53] Junsheng Zhou, Xin Wen, Yu-Shen Liu, Yi Fang, and Zhizhong Han. Self-supervised point cloud representation learning with occlusion auto-encoder. 3

[54] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 3