Learning Local Displacements for Point Cloud Completion

Yida Wang[1], David Joseph Tan[2], Nassir Navab[1], Federico Tombari[1,2]
[1]Technische Universität München [2]Google Inc.

Abstract

We propose a novel approach aimed at object and semantic scene completion from a partial scan represented as a 3D point cloud. Our architecture relies on three novel layers that are used successively within an encoder-decoder structure and specifically developed for the task at hand. The first one carries out feature extraction by matching the point features to a set of pre-trained local descriptors. Then, to avoid losing individual descriptors as part of standard operations such as max-pooling, we propose an alternative neighbor-pooling operation that relies on adopting the feature vectors with the highest activations. Finally, upsampling in the decoder modifies our feature extraction in order to increase the output dimension. While this model is already able to achieve competitive results with the state of the art, we further propose a way to increase the versatility of our approach to process point clouds. To this aim, we introduce a second model that assembles our layers within a transformer architecture. We evaluate both architectures on object and indoor scene completion tasks, achieving state-of-the-art performance.

1. Introduction

Understanding the entire 3D space is essential for both humans and machines to understand how to safely navigate an environment or how to interact with the objects around them. However, when we capture the 3D structure of an object or scene from a certain viewpoint, a large portion of the whole geometry is typically missing due to self-occlusion and/or occlusion from its surrounding. To solve this problem, geometric completion of scenes [2, 27, 32] and objects [16, 20, 39, 44, 45] has emerged as a task that takes on a 2.5D/3D observation and fills out the occluded regions, as illustrated in Fig. 1.

There are multiple ways to represent 3D shapes. Point cloud [3, 6], volumetric grid [8, 27], mesh [11] and implicit surfaces [18, 21, 40] are among the most common data formats. These representations are used for most 3D-related computer vision tasks such as segmentation, classification and completion. For what concerns geometric completion, most works are focused on either point cloud or volumetric data. Among them, the characteristic of having an explicitly defined local neighbourhood makes volumetric data easier to process with 3D convolutions [7, 41, 42]. One drawback introduced by the predefined local neighborhood is the inaccuracy due to the constant resolution of the voxels, meaning that one voxel can represent several small structures.

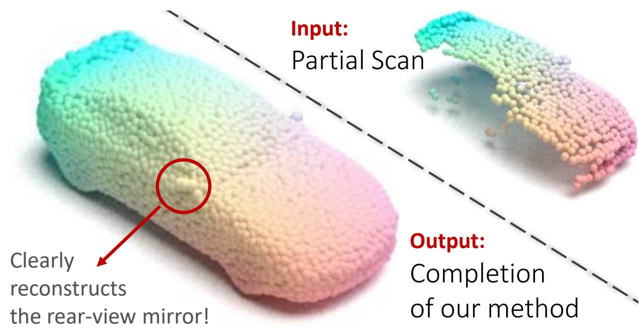

Figure 1. From the input partial scan to our object completion, we visualize the amount of detail in our reconstruction.

On the other hand, point clouds have the advantage of not limiting the local resolution, although they come with their own sets of drawbacks. Mainly, there are two problems in processing point clouds: the undefined local neighborhood and unorganized feature map. Aiming at solving these issues, PointNet++ [23], PMP-Net [35], PointConv [37] and PointCNN [13] employ k-nearest neighbor search to define a local neighborhood, while PointNet [22] and SoftPoolNet [33] adopt the pooling operation to achieve permutation invariant features. Notably, point cloud segmentation and classification were further improved by involving k-nearest neighbor search to form local features in PointNet++ [23] compared to global features in PointNet [22]. Several variations of PointNet [22] also succeeded in improving point cloud completion as demonstrated in FoldingNet [43], PCN [45], MSN [16]. Other methods such as SoftPoolNet [33] and GRNet [39] explicitly present local neighbourhood in sorted feature map and voxel space, respectively.

This paper investigates grouping local features to improve the point cloud completion of objects and scenes. We apply these operation in encoder-decoder architectures which iteratively uses a feature extraction operation with the help of a set of displacement vectors as part of our parametric model. In addition, we also introduce a new pooling mechanism called neighbor-pooling, aimed at down-sampling the data in the encoder while, at the same time, preserving individual feature descriptors. Finally, we propose a new loss function that gradually reconstructs the target from the observable to the occluded regions. The proposed approach is evaluated on both object completion dataset with ShapeNet [3], and semantic scene completion on NYU [25] and CompleteScanNet [36], attaining significant improvements producing high resolutions reconstruction with fine-grained details.

2. Related works

This section focuses on the three most related fields – point cloud completion, point cloud features and semantic scene completion.

Point cloud completion. Given the partial scan of an object similar to Fig. 1, 3D completion aims at estimating the missing shape. In most cases, the missing region is due to self-occlusion since the partial scan is captured from a single view of the object. Particularly for point cloud, FoldingNet [43] and AtlasNet [11] are among the first works to propose an object completion based on PointNet [22] features by deforming one or more 2D grids into the desired shape. Then, PCN [45] extended their work by deforming a collection of much smaller 2D grids in order to reconstruct finer structures.

Through encoder-decoder architectures, ASFM-Net [38] and VRCNet [20] match the encoded latent feature with a completion shape prior, which produce good coarse completion results. To preserve the observed geometry from the partial scan for the fine reconstruction, MSN [16] and VRCNet [20] bypass the observed geometries by using either the minimum density sampling (MDS) or the farthest point sampling (FPS) from the observed surface and building skip connections. By embedding a volumetric sub-architecture, GRNet [39] preserves the discretized input geometries with the volumetric U-connection without sampling in the point cloud space. In more recent works, PMP-Net [35] gradually reconstructs the entire object from the observed to the nearest occluded regions. Also focusing on only predicting the occluded geometries, PoinTr [44] is among the first few transformer methods targeted on point cloud completion by translating the partial scan proxies into a set of occluded proxies to further refine the reconstruction.

Point cloud features. Notably, a large amount of work in object completion [11,16,33,35,39,43,45] rely on PointNet features [22]. The main advantage of [22] is its capacity to be permutation invariant through max-pooling. This is a crucial characteristic for the input point cloud because its data is unstructured.

However, the max-pooling operation disassembles the point-wise features and ignores the local neighborhood in 3D space. This motivated SoftPoolNet [33] to solve this problem by sorting the feature vectors based on the activation instead of taking the maximum values for each element. In effect, they were able to concatenate the features to form a 2D matrix so that a traditional 2D convolution from CNN can be applied.

Apart from building feature representation through pooling operations, PointNet++ [23] samples the local subset of points with the farthest point sampling (FPS) then feeds it into PointNet [22]. Based on this feature, SA-Net [34] then groups the features in different resolutions with KNN for further processing, while PMP-Net [35] uses PointNet++ features to identify the direction to which the object should be reconstructed. PoinTr [44] also solves the permutational invariant problem without pooling by adding the positional coding of the input points into a transformer.

Semantic scene completion. All the point cloud completion are designed to reconstruct a single object. Extending these methods from objects to scenes is difficult because of the difference in size and content. When we tried to train these methods for objects, we noticed that the level of noise is significantly increased such that most objects in the scene are unrecognizable. Evidently, for semantic scene completion, the objective is not only to build the full reconstruction of the scene but also to semantically label each component.

On the other hand, there have been a number of methods for semantic scene completion based on voxel grids that was initiated by SSCNet [27]. Using a similar volumetric data with 3D convolutions [7, 41, 42], VVNet [12] convolves on the 3D volumes which are back-projected from the depth images, revealing the camera view instead of a TSDF volume. Later works such as 3D-RecGAN [42] and ForkNet [32] use discriminators to optimize the convolutional encoder and decoder during training. Since 3D convolutions are heavy in terms of memory consumption especially when the input is presented in high resolution, SketchSSC [4] learns the 3D boundary of all objects in the scene to quickly estimate the resolution of the invariant features.

Although there are quite many methods targeting on volumetric semantic scene completion, there are still no related works proposed explicitly for point cloud semantic scene completion which we achieved in this paper.

3. Operators

Whether reconstructing objects or scenes from a single depth image, the objective is to process the given point cloud of the partial scan $\mathcal{P}_{\text{in}}$ to reconstruct the complete structure $\mathcal{P}_{\text{out}}$. Most deep learning solutions [16, 20, 33, 43, 45] solve this problem by building an encoder-decoder architecture. The encoder takes the input point cloud to iteratively *down*-sample it into its latent feature. Then, the decoder iteratively *up*-sample the latent feature to reconstruct the object or scene. In this section, we illustrate our novel down-sampling and up-sampling operations that cater to point cloud completion. Thereafter, in the following sections, we use our operators as building blocks to assemble two different encoder-decoder architectures that perform object completion and semantic scene completion. We also discuss the associated loss functions.

3.1. Down-sampling operation

To formalize the down-sampling operation, we denote the input as the set of feature vectors $\mathcal{F}_{\text{in}} = \{\mathbf{f}_i\}_{i=1}^{|\mathcal{F}_{\text{in}}|}$ where $\mathbf{f}_i$ is a feature vector and $|\cdot|$ is the number of elements in the set. Note that, in the first layer of the encoder, $\mathcal{F}_{\text{in}}$ is then set to the coordinates of the input point cloud. We introduce a novel down-sampling operation inspired from the Iterative Closest Point (ICP) algorithm [1, 5]. Taking an arbitrary anchor $\mathbf{f}$ from $\mathcal{F}_{\text{in}}$, we start by defining a vector $\delta \in \mathbb{R}^{D_{\text{in}}}$. From the trainable variable δ, we find the feature closest to $\mathbf{f} + \delta$ and compute the distance. This is formally formulated as a function

$$d(\mathbf{f}, \delta) = \min_{\forall \tilde{\mathbf{f}} \in \mathcal{F}_{\text{in}}} \|(\mathbf{f} + \delta) - \tilde{\mathbf{f}}\| \tag{1}$$

where δ represents a displacement vector from $\mathbf{f}$. Multiple displacement vectors are used to describe the local geometry, each with a weight $\sigma \in \mathbb{R}$. We then assign the set as $\{(\delta_i, \sigma_i)\}_{i=1}^{s}$ and aggregate them with the weighted function

$$g(\mathbf{f}) = \sum_{i=0}^{s} \sigma_i \tanh \frac{\alpha}{d(\mathbf{f}, \delta_i) + \beta} \tag{2}$$

where the constants α and β are added for numerical stability. Here, the hyperbolic tangent in $g(\mathbf{f})$ produces values closer to 1 when the distance $d(\cdot)$ is small and closer to 0 when the distance is large. In practice, we can speed-up (1) with the k-nearest neighbor search for each anchor. A simple example of this operation is depicted in Fig. 2. This illustrates the operation in the first layer where we process the point cloud so that we can geometrically plot a feature in $\mathcal{F}_{\text{in}}$ with respect to $\{(\delta_i, \sigma_i)\}_{i=1}^{s}$.

Furthermore, to enforce the influence of the anchor in this operation, we also introduce the function

$$h(\mathbf{f}) = \rho \cdot \mathbf{f} \tag{3}$$

that projects $\mathbf{f}$ on $\rho \in \mathbb{R}^{D_{\text{in}}}$, which is a trainable parameter. Note that both functions $g(\cdot)$ and $h(\cdot)$ produce a scalar value.

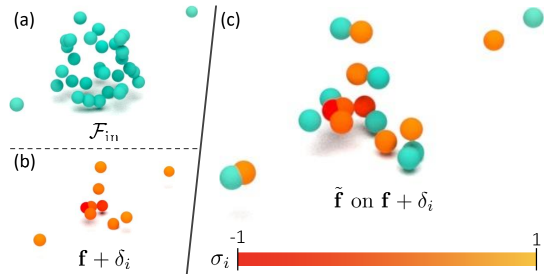

Figure 2. (a) k-nearest neighbor in reference to an anchor $\mathbf{f}$; (b) displacement vectors around the anchor $\mathbf{f} + \delta_i$ and the corresponding weight σ_i; and, (c) closest features $\tilde{\mathbf{f}}$ to $\mathbf{f} + \delta_i$ for all i.

Thus, if we aim at building a set of output feature vectors, each with a dimension of D_{out}, we construct the set as

$$\mathcal{F}_{\text{out}} = \left\{ [g_b(\mathbf{f}_a) + h(\mathbf{f}_a)]_{b=1}^{D_{\text{out}}} \right\}_{a=1}^{|\mathcal{F}_{\text{in}}|} \tag{4}$$

where different sets of trainable parameters $\{(\delta_i, \sigma_i)\}_{i=1}^{s}$ are assigned to each element, while different ρ for each output vector. Moreover, the variables s in (2) and D_{out} in (4) are the hyper-parameters. We label this operation as the *feature extraction*.

It is noteworthy to mention that the proposed down-sampling operation is different from 3D-GCN [15], which only takes the cosine similarity. While still being scale-invariant, hence suitable for object classification and segmentation, they ignore the metric structure of the local 3D geometry; consequently, making completion difficult because the original scale of the local geometry is missing.

Neighbor pooling. The final step in our down-sampling operation is to reduce the size of $\mathcal{F}_{\text{out}}$ with pooling. However, unlike Graph Max-Pooling (GMP) [15], that takes the element-wise maximum value of the feature across all the vectors, we select the subset of feature vectors with the highest activations. Therefore, while GMP disassembles their features as part of their pooling operation, we preserve the feature descriptors from $\mathcal{F}_{\text{out}}$. From the definition of $\mathcal{F}_{\text{out}}$ in (4), we base our activation for each vector $\mathbf{f}_a$

$$\mathcal{A}_a = \sum_{b=1}^{D_{\text{out}}} \tanh |g_b(\mathbf{f}_a)| \tag{5}$$

on the results of $g(\cdot)$ from (2). Thereafter, we only take the $\frac{1}{\tau}$ of the number of feature vectors with the highest activations.

3.2. Up-sampling operation

The up-sampling and pooling operations in the encoder reduce the point cloud to a latent vector. In this case, if we directly use the operation in (4), the first layer in the decoder ends up with one vector since $|\mathcal{F}_{\text{in}}|$ is one. Subsequently, all

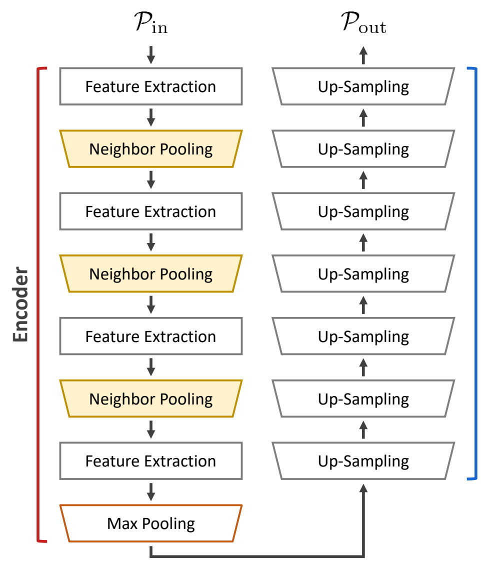

Figure 3. This architecture is composed of the proposed operators to build its encoder and decoder.

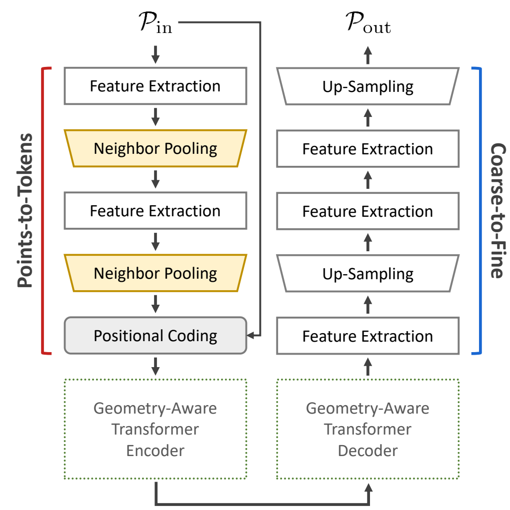

Figure 4. This architecture is derived from the transformers backbone, where we use the proposed operators to convert the input 3D points to tokens and to perform the coarse-to-fine strategy.

the other layers in the decoder result in a single vector. To solve this issue, our up-sampling iteratively runs (4) so that, denoting $\mathcal{F}_{\text{in}}$ as the input to the layer, we build the set of output feature vectors as

$$\begin{aligned}\mathcal{F}_{\text{up}} &= \{\mathcal{F}^{u}_{\text{out}}\}_{u=1}^{N_{\text{up}}} \\ &= \left\{[g^u_b(\mathbf{f}_a) + h^u_b(\mathbf{f}_a)]_{b=1}^{D_{\text{out}}}\right\}_{a=1,u=1}^{a=|\mathcal{F}_{\text{in}}|,u=N_{\text{up}}}\end{aligned} \tag{6}$$

which increases the number of vectors by N_{up}. As a result, $\mathcal{F}_{\text{up}}$ is a set of $N_u \cdot |\mathcal{F}_{\text{in}}|$ feature vectors. In addition to the list of hyper-parameters in Sec. 3.1, our up-sampling operation also takes N_{up} as a hyper-parameter.

4. Encoder-decoder architectures

In order to uncover the strengths of our operators in Sec. 3 (*i.e.* feature extraction, neighbor pooling and up-sampling), we used them as building blocks to construct two different architectures. The first directly implements our operators to build an encoder-decoder while the second takes advantage of our operators to improve the transformers derived from PoinTr [44]. We refer the readers to the Supplementary Materials for the detailed parameters of the architectures.

4.1. Direct application

The objective of the first architecture is to establish that building it solely from the proposed operators (with the additional max-pooling) can already be competitive in point cloud completion. We then propose an encoder-decoder architecture based on our operators alone as shown in Fig. 3.

The encoder is composed of four alternating layers of feature extraction and neighbor pooling. As the number of points from the input is reduced by 128 times, we use a max-pooling operator to extract a vector as our latent feature. Taking the latent feature from the encoder, the decoder is then constructed from a series of up-sampling operators, resulting in a fine completion of 16,384 points.

4.2. Transformers

The second architecture aims at showing the diversity of the operators to improve the state-of-the-art from PoinTr [44] that uses transformers. We therefore propose a transformer-based architecture that is derived from [44] and our operators as summarized in Fig. 4.

Before computing the attention mechanisms in the transformer, the partial scan are subsampled due to the memory constraint of the GPU. PoinTr [44] implements the Farthest Point Sampling (FPS) to reduce the number of points and MLP to convert the points to features. Conversely, our architecture applies the proposed operators. Similar to Sec. 4.1, this involves alternating the features extraction and neighbor pooling. Since the Fourier feature [28] and SIRENs [26] have proven that the sinusoidal activation is helpful in presenting complex signals and their derivatives in layer-by-layer structures, a positional coding based on the 3D coordinates is then added to the features. In Fig. 4, we refer this block as *points-to-token*. Thereafter, we use the geometry-aware transformers from [44] which produces a coarse point cloud.

From the coarse point cloud, we then replace their coarse-to-fine strategy with our operators. This includes a series of alternating feature extraction and up-sampling operators as shown in Fig. 4.

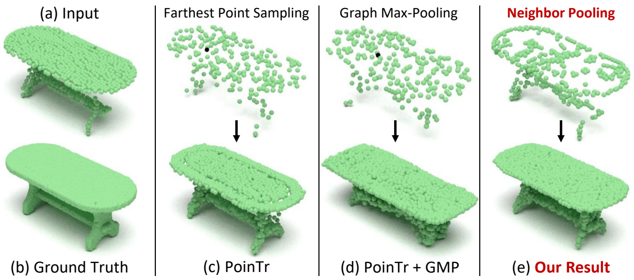

Figure 5. The first row compares the point tokens chosen by Farthest Point Sampling (FPS) in PointTr [44], Graph Max-Pooling (GMP) [15] in PointTr [44] and our proposed neighbor pooling in our transformer architecture. These tokens are then fed to the transformer and the coarse-to-fine strategy to produce the reconstruction shown in the second row.

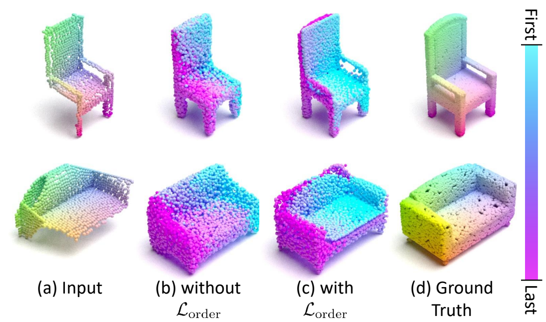

Figure 6. Compares the order of the point clouds reconstructed in the object completion with and without $\mathcal{L}_{\text{order}}$

It it noteworthy to emphasize the difference between our architecture from PoinTr [44] and to understand the implication of the changes. The contributions of points-to-tokens and coarse-to-fine to the overall architecture is illustrated in Fig. 5. We can observe from this figure that the FPS from PoinTr [44] only finds the distant points while the results of our neighbor pooling sketches the contours of the input point cloud to capture the meaningful structures of the object. Notably, by looking at our sketch, we can already identify the that the object is a table. This is contrary to the random points from PoinTr [44]. Moreover, our coarse-to-fine strategy uniformly reconstructs the planar region on the table as well as its base. Later, in Sec. 7, we numerically evaluate these advantages in order to show that the individual components has their own merits.

Since we previously discussed in Sec. 3.1 the difference of our down-sampling operation against 3D-GMP [15], we became curious to see the reconstruction in Fig. 5 if we replace the FPS in PoinTr [44] with the cosine similarity and GMP of [15]. Similar to PoinTr, the new combination selects distant points as its tokens while the table in their final reconstruction increased in size. In contrast, our tokens are more meaningful and the final results are more accurate.

5. Loss functions

Given the input point cloud $\mathcal{P}_{\text{in}}$ (*e.g.* from a depth image), the objective of completion is to build the set of points $\mathcal{P}_{\text{out}}$ that fills up the missing regions in our input data. Since we train our architecture in a supervised manner, we denote $\mathcal{P}_{\text{gt}}$ as the ground truth.

Completion. To evaluate the predicted point cloud, we impose the Earth-moving distance [9]. Comparing the output points to the ground truth and vice-versa, we end up with

$$\mathcal{L}_{\text{out}\to\text{gt}} = \sum_{p\in\mathcal{P}_{\text{out}}} \|p - \phi_{\text{gt}}(p)\|_2 \tag{7}$$

$$\mathcal{L}_{\text{gt}\to\text{out}} = \sum_{p\in\mathcal{P}_{\text{gt}}} \|p - \phi_{\text{out}}(p)\|_2 \tag{8}$$

where $\phi_i(p)$ is a bijective function that finds the closest point in the point cloud $\mathcal{P}_i$ to p.

Order of points in $\mathcal{P}_{\text{out}}$. After training with (7) and (8), we noticed that the points in the output reconstruction are ordered from left to right as shown in Fig. 6(b). We want to take advantage of this organization and investigate this behavior further. Assuming the idea that, among the points in $\mathcal{P}_{\text{out}}$, we are confident that the input point cloud must be part of it, we introduce a loss function that enforces that the first subset in $\mathcal{P}_{\text{out}}$ is similar to $\mathcal{P}_{\text{in}}$. We formally write this loss function as

$$\mathcal{L}_{\text{order}} = \sum_{p\in\mathcal{P}_{\text{in}}} \mathcal{S}(\theta_{\text{out}}(p)) \cdot \|p - \phi_{\text{out}}(p)\|_2 \tag{9}$$

where $\theta_{\text{out}}(p)$ is the index of the closest point in $\mathcal{P}_{\text{out}}$ based on $\phi_{\text{out}}(p)$ while

$$\mathcal{S}(\theta) = \begin{cases} 1, & \text{if } \theta \leq |\mathcal{P}_{\text{in}}| \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

is a step function that returns one if the index is within the first $|\mathcal{P}_{\text{in}}|$ points.

When we plot the results with $\mathcal{L}_{\text{order}}$ in Fig. 6(c), we noticed that the order in $\mathcal{P}_{\text{out}}$ moves from the observed to the occluded. In addition, fine-grained geometrical details such as the armrest of the chair are visible when training with $\mathcal{L}_{\text{order}}$; thus, improving the overall reconstruction.

Semantic scene completion. In addition to the architecture in Sec. 4 and the loss functions in (7), (8) and (9) for completion, a semantic label is added to each point in the predicted cloud $\mathcal{P}_{\text{out}}$. Given N_c categories, we denote the

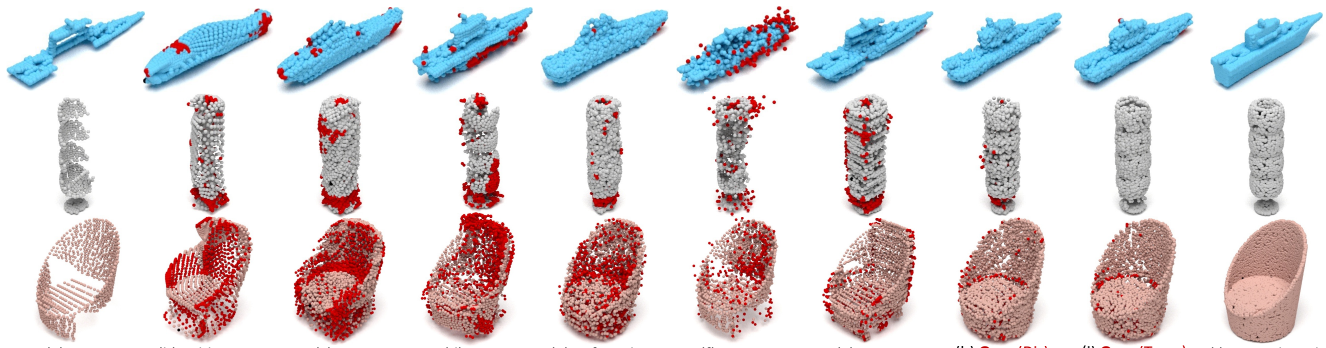

Figure 7. Object completion results where we highlight the errors in red points.

label for each point as a one-hot code $\mathbf{l}_i = [l_{i,c}]_{c=1}^{n_c}$ for the i-th point in $\mathcal{P}_{\text{out}}$ and the c-th category. Since training is supervised, the ground truth point clouds are also labeled with the semantic category.

After establishing the correspondence between the predicted point cloud to the ground truth in (7) in training, we also extract the ground truth semantic label $\hat{\mathbf{l}}_i$. It then follows that the binary cross-entropy of the i-th point is computed

$$\epsilon_i = -\frac{1}{N_c}\sum_{c=i}^{N_s} \hat{l}_{i,c}\log l_{i,c} + (1-\hat{l}_{i,c})(1-\log l_{i,c}) \quad (11)$$

and formulate the semantic loss function as

$$\mathcal{L}_{\text{semantic}} = \frac{\gamma}{|\mathcal{P}_{\text{in}}|}\sum_{i=i}^{|\mathcal{P}_{\text{in}}|}\epsilon_i \quad (12)$$

where the weight

$$\gamma = \frac{0.01}{\mathcal{L}_{\text{out}\to\text{gt}} + \mathcal{L}_{\text{gt}\to\text{out}}} \quad (13)$$

triggers to increase the influence of the $\mathcal{L}_{\text{semantic}}$ in training as the completion starts to converge. Note that γ is an important factor, since the output point cloud is erratic in the initial iterations, which means that it can abruptly change from one iteration to the next before the completion starts converging.

6. Experiments

To highlight the strengths of the proposed method, this section focuses on two experiments – object completion and semantic scene completion.

6.1. Object completion

We evaluate the geometric completion of a single object on the ShapeNet [3] database where they have the point clouds of the partial scans as input and their corresponding ground truth completed shape. The input scans are composed of 2,048 points while the database provides a low resolution output of 2,048 points and a high resolution of 16,384 points. We follow the standard evaluation on 8 categories where all objects are roughly normalized into the same scale with point coordinates ranging between -1 to 1.

Numerical results. We conduct our experiments based on three evaluation strategies from Completion3D [29], PCN [45] and MVP [20]. Evaluating on 8 objects (*plane*, *cabinet*, *car*, *chair*, *lamp*, *sofa*, *table*, *vessel*), they measure the predicted reconstruction through the L2-Chamfer distance, L1-Chamfer distance and the F-Score@1%, respectively. Note that, in this paper, we also follow the standard protocol where the value presented for the Chamfer distance is multiplied by 10^3. Although Table 1 only shows the average results across all categories, we refer the readers to the supplementary materials for the more detailed comparison.

One of the key observations in this table is the capacity of our direct architecture to surpass most of the other methods' results. Among 11 approaches, our Chamfer distance is only worse than 3 methods while our F-Score@1% is better than all of them. This therefore establishes the strength of our operators since our first architecture is solely composed of it. Moreover, our second architecture, which combines our operators with the transformer, reduces the error by 3-5% on the Chamfer distance and increases the accuracy by 4.5% on the F-Score@1%.

The table also examines the effects of $\mathcal{L}_{\text{order}}$ to our reconstruction. Training with $\mathcal{L}_{\text{order}}$ improves our results by 0.12-0.13 in Chamfer distance and 0.013-0.021 in F-Score@1%, validating our observations in Fig. 6.

Qualitative results. We compare our object completion results in Fig. 7 with the recently proposed methods: FoldingNet [43], PCN [45], MSN [16], SoftPoolNet [33], VRCNet [20] and PoinTr [44]. The red points in the figure highlight the errors in the reconstruction. All the approaches reconstructs a point cloud with 16,384 points with the excep-

Method	Completion3D *L2-Chamfer*	PCN *L1-Chamfer*	MVP *F-Score@1%*
FoldingNet [43]	19.07	14.31	–
SoftPoolNet [33]	11.07	9.20	0.666
TopNet [29]	14.25	12.15	0.576
PCN [45]	18.22	9.64	0.614
MSN [16]	–	9.97	0.690
GRNet [39]	10.64	8.83	0.677
ECG [19]	–	–	0.736
NSFA [47]	–	–	0.770
CRN [30]	9.21	8.51	0.724
SCRN [31]	9.13	8.29	–
VRCNet [20]	8.12	–	0.781
PoinTr [44]	9.22	8.38	0.741
ASFM-Net [38]	6.68	–	–
Ours (Direct)	8.35	8.46	0.801
–without $\mathcal{L}_{\text{order}}$	8.47	8.59	0.788
–input $\mathcal{P}_{\text{gt}}$	5.11	5.37	0.923
Ours (Transformer)	**6.64**	**7.96**	**0.816**
–without $\mathcal{L}_{\text{order}}$	6.74	8.09	0.795
–input $\mathcal{P}_{\text{gt}}$	4.46	4.95	0.962

Table 1. Evaluation on Completion3D [29], PCN [45] and MVP [20] datasets with their corresponding metrics for the object completion task.

tion for FoldingNet with 2,048 points and MSN with 8,192.

Since FoldingNet and PCN take advantage of their mathematical assumption where they rely on deforming one or more planar grids, they tend to over-smooth their reconstruction where finer details such as the boat is flattened. In contrast, our method can perform better on the smooth regions as well as the finer structures. Nevertheless, the more recent approaches like [16,20,33,44] can also produce more descriptive reconstruction on the boat. However, they produce more errors which is highlighted in the unconventional lamp or chair. Overall, our reconstructions are closer to the ground truth.

Failure cases. In addition to the qualitative results, we also examine the failure cases in Fig. 8. Most of them are objects with unusual structures like the car without the wheels. Another issue is when there is an insufficient amount of input point cloud to describe the object such as the chair. Notably, compared to the state-of-the-art, our reconstructions are still better in these situations.

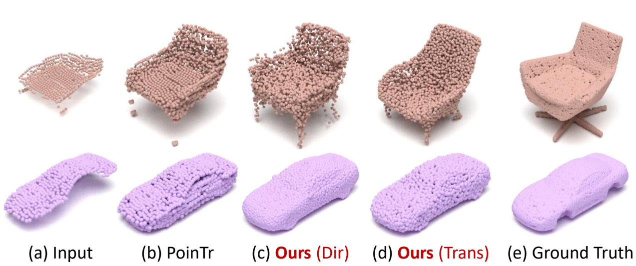

Figure 8. Examples of the failure cases in object completion.

Method	Resolution	*Average IoU*
Lin *et al*. [14]	60	12.0
Geiger and Wang [10]	60	19.6
SSCNet [27]	60	30.5
VVNet [12]	60	32.9
SaTNet [17]	60	34.4
ForkNet [32]	80	37.1
CCPNet [46]	240	38.5
SketchSSC [4]	60	41.1
SISNet [2]	60	**52.4**
Ours (Direct)	60	40.0
–with $\gamma = 1$ *in* $\mathcal{L}_{\text{semantic}}$	60	37.2
Ours (Transformer)	60	42.4
–with $\gamma = 1$ *in* $\mathcal{L}_{\text{semantic}}$	60	38.9

Table 2. Semantic scene completion on NYU [25] dataset. The value in resolution (x) is the output volumetric resolution which is $x \times 0.6x \times x$.

6.2. Semantic scene completion

This evaluation aims at reconstructing the scene from a single depth image through a point cloud or an SDF volume where each point or voxel is categorized with a semantic class. Originally introduced for 2.5D semantic segmentation, NYU [25] and ScanNet [6], which were later annotated for semantic completion by [27, 36], are among the most relevant benchmark datasets in this field. These datasets include pairs of depth image and the corresponding semantically labeled 3D reconstruction.

Semantic scene completion with voxels. NYU are provided with real scans for indoor scenes which are acquired with a Kinect depth sensor. Following SSCNet [27], the semantic categories include 12 classes of varying shapes and sizes: *empty space*, *ceiling*, *floor*, *wall*, *window*, *chair*, *bed*, *sofa*, *table*, *tvs*, *furniture* and *other objects*.

Since the other point cloud completion do not handle semantic segmentation, we start our evaluation by comparing with the voxel-based approaches which perform the both the completion and the semantic segmentation such as [2, 4, 10, 12, 14, 17, 27, 32, 46]. Considering that the volumetric data evaluates through the IoU, we need to convert our point clouds to voxel grids to make the comparison.

One of the significant advantage of point clouds over voxels is that we are not constrained to a specific resolution. Since most method evaluate on $60 \times 36 \times 60$, we converted our point cloud to this resolution. Our approach achieves competitive average IoU of 42.4% which is better than all the other methods except for SISNet [2]. However, it is noteworthy to mention that our method faces additional errors associated to the conversion from point cloud to vox-

Method	CompleteScanNet	NYU
FoldingNet [43]	11.25	14.66
AtlasNet [11]	8.92	10.12
PCN [45]	8.19	9.98
MSN [16]	7.28	8.65
SoftPoolNet [33]	8.27	9.29
GRNet [39]	4.56	5.80
VRCNet [20]	4.29	5.45
PoinTr [44]	5.08	5.92
Ours (Direct)	3.17	4.72
Ours (Transformer)	**3.04**	**4.38**

Table 3. Evaluation on CompleteScanNet [36] and NYU [25] dataset for scene completion, measuring the average Chamfer distance trained with L2 distance (multiplied by 10^3) with the output resolution of 16,384.

els. In addition, the ground truth voxels for the furnitures in the NYU dataset is a solid volume which is not a plausible format for point cloud approaches which focuses more on the surface reconstruction. This in effect decreases the IoU of our method.

Moreover, Table 2 includes a small ablation study to verify the contribution of γ from (13) in $\mathcal{L}_{\text{semantic}}$. If we discard (13) by setting γ to one, the IoU for our models decrease by 7.5-9%; thus, proving the advantage in adaptively weighing the semantic loss function.

Point cloud scene completion. Another relevant dataset is from ScanNet [6] which was supplemented with the ground truth semantic completion by CompleteScanNet [36]. This include a total of 45,451 paired partial scan and semantic completion for training. Our evaluation in Table 3 takes 2,048 points as input and reconstructs the scene with 16,384 points. Since there is no previous work that focused on point cloud scene completion, we compare against methods that were designed for a single object completion such as PCN [45], MSN [16], SoftPoolNet [33] and GRNet [39]. Based on our evaluation in Table 3, both versions of our architectures attain the best results. Notably, we also compared these methods on the NYU dataset in Table 3. Similarly, the proposed architectures also achieve the state-of-the-art in point cloud scene completion.

7. Ablation study

This section focuses on the strengths of our operator in our transformer architecture. Although we adapt the transformer from PoinTr [44], we argue that every component we added is significant to the overall performance. To evaluate this, we disentangle the points-to-tokens and coarse-to-fine blocks. In practice, we separate the backbone, which takes points in the partial scan as input and outputs a coarse point cloud, from the coarse-to-fine strategy. Evidently, in our approach, the points-to-tokens block is part of the backbone.

Since most methods can also be separated in this manner, we then compose Table 4 to mix-and-match different backbones with different coarse-to-fine methods for object and scene completion. In both tables, we classified the other coarse-to-fine methods as: (1) *deform* which includes the operation in deforming 3D grids; (2) *deconv* which processes with MLP, 1D or 2D deconvolutions; and, (3) Edge-aware Feature Expansion (*EFE*) [19]. We then highlight the originally proposed architectures in yellow.

For any given backbone in every row, our coarse-to-fine method produces the best results. Moreover, for any given coarse-to-fine strategy in every column, our backbone performs the best. Therefore, this study essentially proves that each of the proposed components in our transformer architecture has a significant role in the overall performance.

8. Conclusion

We propose three novel operators for point cloud processing. To bring out the value of these operators, we apply them on two novel architectures that are designed for object completion and semantic scene completion. The first assembles together the proposed operators in an encoder-decoder fashion, while the second incorporates them in the context of transformers. Notably, both architectures produce highly competitive results, with the latter achieving the state of the art in point cloud completion for both objects and scenes.

OBJECT COMPLETION

Backbone	Coarse-to-Fine			
	deform	deconv	EFE	**Ours**
MSN [16]	7.28	9.34	7.15	6.91
PoinTr [44]	5.48	5.71	4.91	3.76
SoftPoolNet [33]	10.08	8.27	7.65	7.63
GRNet [39]	9.25	5.61	5.26	4.90
VRCNet [20]	8.09	8.88	5.08	4.21
Ours	4.93	4.99	4.12	**3.04**

SCENE COMPLETION

Backbone	Coarse-to-Fine			
	deform	deconv	EFE	**Ours**
MSN [16]	9.97	12.31	9.26	9.08
PoinTr [44]	8.38	8.49	8.31	8.13
TreeGAN [24]	14.26	9.72	9.12	9.05
SoftPoolNet [33]	11.73	9.20	8.75	8.64
GRNet [39]	9.12	8.83	8.73	8.51
VRCNet [20]	10.03	10.20	8.52	8.26
Ours	8.19	8.30	8.07	**7.96**

Table 4. Mix-and-match evaluation on different backbone attached to different coarse-to-fine methods for object and scene completion. The originally proposed combinations are marked in yellow.

References

[1] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 3

[2] Yingjie Cai, Xuesong Chen, Chao Zhang, Kwan-Yee Lin, Xiaogang Wang, and Hongsheng Li. Semantic scene completion via integrating instances and scene in-the-loop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 324–333, 2021. 1, 7

[3] 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. 1, 2, 6

[4] Xiaokang Chen, Kwan-Yee Lin, Chen Qian, Gang Zeng, and Hongsheng Li. 3d sketch-aware semantic scene completion via semi-supervised structure prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4193–4202, 2020. 2, 7

[5] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 3

[6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 1, 7, 8

[7] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2017. 1, 2

[8] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jürgen Sturm, and Matthias Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2018. 1

[9] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 5

[10] Andreas Geiger and Chaohui Wang. Joint 3d object and layout inference from a single rgb-d image. In *German Conference on Pattern Recognition*, pages 183–195. Springer, 2015. 7

[11] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2, 8

[12] Yuxiao Guo and Xin Tong. View-volume network for semantic scene completion from a single depth image. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 2018. 2, 7

[13] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018. 1

[14] Dahua Lin, Sanja Fidler, and Raquel Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In *Proceedings of the IEEE international conference on computer vision*, pages 1417–1424, 2013. 7

[15] Zhi-Hao Lin, Sheng-Yu Huang, and Yu-Chiang Frank Wang. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1800–1809, 2020. 3, 5

[16] Minghua Liu, Lu Sheng, Sheng Yang, Jing Shao, and Shi-Min Hu. Morphing and sampling network for dense point cloud completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11596–11603, 2020. 1, 2, 3, 6, 7, 8

[17] Shice Liu, YU HU, Yiming Zeng, Qiankun Tang, Beibei Jin, Yinhe Han, and Xiaowei Li. See and think: Disentangling semantic scene completion. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 263–274. Curran Associates, Inc., 2018. 7

[18] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[19] Liang Pan. Ecg: Edge-aware point cloud completion with graph convolution. *IEEE Robotics and Automation Letters*, 5(3):4392–4398, 2020. 7, 8

[20] Liang Pan, Xinyi Chen, Zhongang Cai, Junzhe Zhang, Haiyu Zhao, Shuai Yi, and Ziwei Liu. Variational relational point completion network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8524–8533, 2021. 1, 2, 3, 6, 7, 8

[21] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 1

[22] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017. 1, 2

[23] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 1, 2

[24] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3859–3868, 2019. 8

[25] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pages 746–760. Springer, 2012. 2, 7, 8

[26] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020. 4

[27] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017. 1, 2, 7

[28] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 4

[29] Lyne P Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 383–392, 2019. 6, 7

[30] Xiaogang Wang, Marcelo H. Ang Jr. , and Gim Hee Lee. Cascaded refinement network for point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 7

[31] Xiaogang Wang, Marcelo H Ang Jr, and Gim Hee Lee. A self-supervised cascaded refinement network for point cloud completion. *arXiv preprint arXiv:2010.08719*, 2020. 7

[32] Yida Wang, David Joseph Tan, Nassir Navab, and Federico Tombari. Forknet: Multi-branch volumetric semantic completion from a single depth image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8608–8617, 2019. 1, 2, 7

[33] Yida Wang, David Joseph Tan, Nassir Navab, and Federico Tombari. Softpoolnet: Shape descriptor for point cloud completion and classification. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 70–85, Cham, 2020. Springer International Publishing. 1, 2, 3, 6, 7, 8

[34] Xin Wen, Tianyang Li, Zhizhong Han, and Yu-Shen Liu. Point cloud completion by skip-attention network with hierarchical folding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[35] Xin Wen, Peng Xiang, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. Pmp-net: Point cloud completion by learning multi-step point moving paths. *arXiv preprint arXiv:2012.03408*, 2020. 1, 2

[36] Shun-Cheng Wu, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scfusion: Real-time incremental scene reconstruction with semantic completion. *arXiv preprint arXiv:2010.13662*, 2020. 2, 7, 8

[37] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 1

[38] Yaqi Xia, Yan Xia, Wei Li, Rui Song, Kailang Cao, and Uwe Stilla. Asfm-net: Asymmetrical siamese feature matching network for point completion. *arXiv preprint arXiv:2104.09587*, 2021. 2, 7

[39] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 365–381, Cham, 2020. Springer International Publishing. 1, 2, 7, 8

[40] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 492–502. Curran Associates, Inc., 2019. 1

[41] Bo Yang, Stefano Rosa, Andrew Markham, Niki Trigoni, and Hongkai Wen. Dense 3d object reconstruction from a single depth view. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 1, 2

[42] Bo Yang, Hongkai Wen, Sen Wang, Ronald Clark, Andrew Markham, and Niki Trigoni. 3d object reconstruction from a single depth view with adversarial learning. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 679–688, 2017. 1, 2

[43] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018. 1, 2, 3, 6, 7, 8

[44] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12498–12507, 2021. 1, 2, 4, 5, 6, 7, 8

[45] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737. IEEE, 2018. 1, 2, 3, 6, 7, 8

[46] Pingping Zhang, Wei Liu, Yinjie Lei, Huchuan Lu, and Xiaoyun Yang. Cascaded context pyramid for full-resolution 3d semantic scene completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7801–7810, 2019. 7

[47] Wenxiao Zhang, Qingan Yan, and Chunxia Xiao. Detail preserved point cloud completion via separated feature aggregation. *arXiv preprint arXiv:2007.02374*, 2020. 7