CasFusionNet: A Cascaded Network for Point Cloud Semantic Scene Completion by Dense Feature Fusion

Jinfeng Xu, Xianzhi Li*, Yuan Tang, Qiao Yu, Yixue Hao, Long Hu, Min Chen

Huazhong University of Science and Technology
jinfengxu.edu@gmail.com,{xzli, yuan_tang, qiaoyu_epic, yixuehao, hulong, minchen2012}@hust.edu.cn

Abstract

Semantic scene completion (SSC) aims to complete a partial 3D scene and predict its semantics simultaneously. Most existing works adopt the voxel representations, thus suffering from the growth of memory and computation cost as the voxel resolution increases. Though a few works attempt to solve SSC from the perspective of 3D point clouds, they have not fully exploited the correlation and complementarity between the two tasks of scene completion and semantic segmentation. In our work, we present CasFusionNet, a novel cascaded network for point cloud semantic scene completion by dense feature fusion. Specifically, we design (i) a global completion module (GCM) to produce an upsampled and completed but coarse point set, (ii) a semantic segmentation module (SSM) to predict the per-point semantic labels of the completed points generated by GCM, and (iii) a local refinement module (LRM) to further refine the coarse completed points and the associated labels from a local perspective. We organize the above three modules via dense feature fusion in each level, and cascade a total of four levels, where we also employ feature fusion between each level for sufficient information usage. Both quantitative and qualitative results on our compiled two point-based datasets validate the effectiveness and superiority of our CasFusionNet compared to state-of-the-art methods in terms of both scene completion and semantic segmentation. The codes and datasets are available at: https://github.com/JinfengX/CasFusionNet.

Introduction

Humans can infer the complete shapes and semantics from only a partial observation of a 3D scene based on experience. To enable an intelligent agent to behave like humans in 3D physical world, semantic scene completion (SSC) is gaining more attention with the goal of simultaneously reconstructing complete 3D scenes and predicting associate semantics from RGB-D images of a given viewpoint. As SSC supplies complete 3D information of scenes, it benefits diverse applications, e.g., robot navigation and grasping (Gupta et al. 2017; Varley et al. 2017; Liang, Chen, and Song 2021), automatic driving, high-quality visualization, etc.

Followed by the pioneer work SSCNet (Song et al. 2017), most of existing methods (Guo and Tong 2018; Li et al. 2020b, 2021; Cai et al. 2021) utilize 3D CNN to predict the volumetric occupancy and semantics based on voxel representation. Yet, these voxel-based networks suffer from tremendous computation and memory cost when the voxel resolution is high. In contrast, point clouds, or scattered collections of points in 3D, are arguably the simplest shape representation, especially for large-scale 3D scenes with complex structures and fine details. Thus, recent few works (Zhang et al. 2021a; Wang et al. 2022) attempt to consume only points as network input to handle SSC, where two branches are commonly designed, one for point-based scene completion and the other for point-based semantic segmentation. Despite notable achievements, existing works tend to ignore the high correlation and complementarity between scene completion and semantic segmentation, so there is still a lot of room for performance improvement.

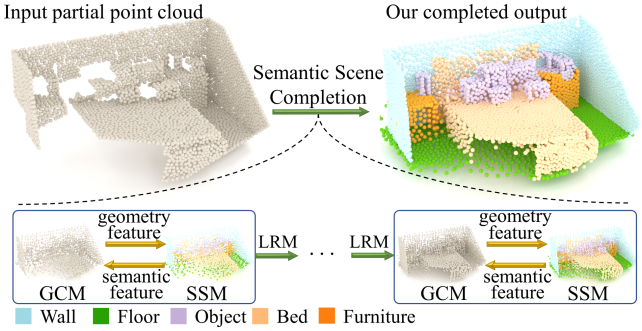

Figure 1: Given a partial point-based scene (left), our CasFusionNet completes and segments the scene (right) by cascaded levels. Each level is composed of GCM, SSM and LRM, which are connected by dense feature fusion.

In our work, we present CasFusionNet, a novel cascaded network for point-based SSC that associates scene completion with semantic segmentation by dense feature fusion. Figure 1 shows the key idea. Specifically, we design (i) a global completion module (GCM) to produce an upsampled and completed but maybe coarse point set from a partial 3D scene represented by points, (ii) a semantic segmentation module (SSM) to predict per-point semantic labels of the completed points generated by GCM, and (iii) a local refinement module (LRM) to further refine the coarse completed

*Corresponding author

points and associated labels from a local perspective. We organize the three modules via dense feature fusion in each level, and cascade a total of four levels, where we also employ skip connection and feature fusion between each level for sufficient information usage.

In our network, both the geometric features extracted from 3D points and the semantic features extracted from semantic label distribution are running through all the three designed modules in each level, which closely ties the tasks of scene completion and segmentation. To validate the effectiveness of our CasFusionNet, we prepare two point-based datasets based on existing SSC datasets, i.e. NYUCAD (Firman et al. 2016) and PCSSC-Net (Zhang et al. 2021a). Extensive quantitative and qualitative results show the superiority and effectiveness of our network compared to state-of-the-arts; see Figure 1 for an example output of our method. Overall, our contributions are summarized as follows:

- We propose a novel cascaded network (CasFusionNet) for point cloud semantic scene completion. It completes the partial scene and predicts the semantics simultaneously in a progressive manner by dense feature fusion.
- We design three novel modules, i.e. the global completion module (GCM), the semantic segmentation module (SSM) and the local refinement module (LRM), to complete, segment, and locally refine the scene, respectively.
- We contribute two point-based datasets for SSC task, and extensive experiments validate that our network outperforms previous works significantly.

Related Work

Semantic Scene Completion. With recent advances in 3D deep learning, semantic scene completion (SSC) has been widely explored, where both semantics and geometry are jointly inferred from a partial 3D scene. To directly adopt 3D CNNs, most recent works encode a 3D scene as a 3D grid, in which cells describe semantic occupancy of the space. For example, SSCNet (Song et al. 2017) was first proposed to tackle SSC with an end-to-end 3D convolution network, which predicts volumetric occupancy and semantic labels of scenes. Later, some followers (Guo and Tong 2018; Liu et al. 2018; Garbade et al. 2019; Li et al. 2020b) leveraged the 2D semantic priors of color images via feature projection to improve the performance. Recent works (Dourado et al. 2021; Dourado, Guth, and de Campos 2022; Wang, Lin, and Wan 2022) further fused complex features that are extracted from depth or color images to 2D semantic network. In addition, IMENet (Li, Ding, and Huang 2021) proposed an iterative fusion scheme to ensure the branches of 2D segmentation and 3D scene completion fully benefit each other. However, the cubic growth of computational and memory requirements of 3D CNN blocks the depth of networks, which limits the task performance. To relieve the computational cost, various efficient network designs were introduced, such as spatial group convolution network (Zhang et al. 2018), octree-based network (Wang, Liu, and Tong 2020), lightweight consecutive dimensional convolutions (Li et al. 2019, 2020a, 2021), and efficient depth information embedding (Chen et al. 2020).

Opposed to occupancy grids (voxels), 3D point cloud is a convenient and memory-efficient representation, which expresses geometry with fine details. Only a few works have in fact explored point-based SSC. Early method SPC-Net (Zhong and Zeng 2020) applied point encoder-decoder architecture on "pointlized" voxels. To use both voxels and points, SISNet (Cai et al. 2021) converted voxels into points to recover detailed 3D shapes in scene-to-instance completion stage. A recent point-voxel aggregation strategy (Tang et al. 2022) used point-based network as the mainstream to improve the learning efficiency and capability of the framework. Yet, these methods are still confined by the low resolution and high operating cost of voxel representation.

Very recently, PCSSC-Net (Zhang et al. 2021a) tackled SSC solely based on point clouds and directly concatenated predicted semantic labels with the point features for better completion. Another method (Wang et al. 2022) obtained semantic labels by attaching an extra point cloud segmentation network. However, the two existing point-based methods do not fully exploit the connection between point completion and segmentation. In our work, we argue that point completion and segmentation are complementary and highly correlated, thus motivating us to design CasFusionNet to encourage the features in completion and segmentation modules to fully communicate and merge.

Point Cloud Completion on Single Object. Though our work focuses on semantic scene completion, we here still briefly summarize the related works on single point cloud completion. PCN (Yuan et al. 2018) was the pioneering work to address shape completion by folding a small patch of 2D grids for each point to represent the local geometry. Some followers (Liu et al. 2020; Wen et al. 2020; Zong, Sun, and Zhao 2021) further improved the performance based on a similar folding-based strategy. On the other hand, coarse-to-fine methods (Xie et al. 2020; Deng et al. 2021; Pan et al. 2021; Zhang et al. 2021b) completed objects in an explicit and controllable way, which gradually increase density or change distribution of the point set. For instance, RFNet (Huang et al. 2021), PMP-Net (Wen et al. 2021) and PMP-Net++ (Wen et al. 2022) completed the points level by level, where the recurrent neural network was utilized to reserve useful information of previous level. ASFM-Net (Xia et al. 2021) and SnowflakeNet (Xiang et al. 2021) upsampled and moved the points at each refinement iteration. Recently, a two-path network (Zhao et al. 2021b) for pairwise completion was proposed to separately complete objects which bear a strong spatial relation.

Nevertheless, the above completion networks focus on small or single objects, which can hardly directly handle a large-scale incomplete 3D scene with occlusion and multiple kinds of objects. Moreover, these single point cloud completion methods have no function of semantic segmentation.

Method

Overview of Network Architecture

Given a partial point cloud $\mathcal{P}_{in} \in \mathbb{R}^{N\times 3}$ with N points as input, which represents the incomplete 3D scene, our target is to attain a complete 3D scene $\mathcal{P}_{out} \in \mathbb{R}^{M\times 3}$ with M points

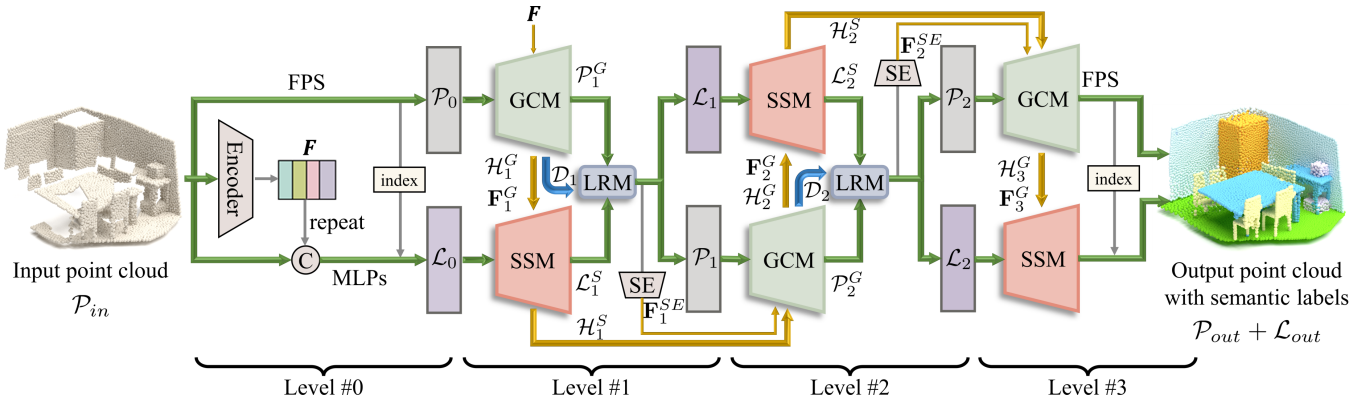

Figure 2: Illustrating the architecture of our CasFusionNet. Given a partial point cloud $\mathcal{P}_{in}$, our network semantically completes it via four successive levels in a coarse-to-fine manner and outputs a complete scene $\mathcal{P}_{out}$ with the associated per-point semantic labels $\mathcal{L}_{out}$. Each level is composed of a global completion module (GCM), a semantic segmentation module (SSM) and a local refinement module (LRM) to complete, segment, and refine the scene, which are closely connected by dense feature fusion.

($M \geq N$), as well as the predicted per-point semantic labels $\mathcal{L}_{out} \in \mathbb{R}^{M \times C}$, where C is the total number of semantic classes. Figure 2 shows the overall pipeline of CasFusionNet, which semantically completes a scene via four successive levels in a coarse-to-fine manner. The outputs of previous level are the inputs of next level; see the green arrows for cascading the four levels. Further, the scene completion and semantic segmentation tasks are closely communicated and complement each other by dense feature fusion, both within and between levels; see the yellow arrows for features flow. Below, we shall elaborate on the details of each level.

In level #0, we first feed $\mathcal{P}_{in}$ into an encoder to extract the hierarchical scene-wise features $\mathbf{F}$. Empirically, the encoder is designed by using the set abstraction layer (Qi et al. 2017) and the point transformer layer (Zhao et al. 2021a). Please refer to our supplementary material for the detailed architecture of the encoder. To realize semantic completion, a naive following step is to decode $\mathbf{F}$ into a complete scene and its associated semantic labels. However, the one-step operation is often difficult to yield high-quality and fine-grained predictions. Hence, we propose to adopt the coarse-to-fine strategy by repeating semantic completion multiple times, thus allowing network to refine its predictions steadily. To ensure enough repeating times (or levels), the computation consumption in level #0 should not be too large, and it should not cause a huge calculation in the subsequent levels. To this end, in the initial level, instead of directly upsampling and completing $\mathcal{P}_{in}$, we downsample it using the farthest point sampling (FPS) to obtain a sparse point cloud $\mathcal{P}_0$, which will be completed by subsequent levels. The advantage of this operation is that, $\mathcal{P}_0$ has fewer points than $\mathcal{P}_{in}$, thus reducing the computation of following levels, but still preserving necessary geometrical structures. In this level, we also predict the coarse per-point semantic labels $\mathcal{L}_0$ associated with $\mathcal{P}_0$ by using the duplicated $\mathbf{F}$ via multi-layer perceptions (MLPs); see Figure 2 for details.

In level #1, given $\mathcal{P}_0$ and $\mathcal{L}_0$, our purpose is to obtain the refined completed scene $\mathcal{P}_1$ and its semantic prediction $\mathcal{L}_1$. As shown in Figure 2, this level is mainly composed of three modules, i.e. Global Completion Module (GCM), Semantic Segmentation Module (SSM), and Local Refinement Module (LRM). Specifically, GCM consumes the point cloud produced by previous level as input and generates a coarse but completed scene by point displacements and upsampling (e.g., $\mathcal{P}_0 \rightarrow \mathcal{P}_1^G$ in level #1). Here, the upsampling ratio is λ and set as 2 by default. SSM aims to predict semantic labels of the completed scene (i.e. $\mathcal{P}_1^G$), which takes the hidden layer output $\mathcal{H}_1^G$ and the feature vector $\mathbf{F}_1^G$ produced by GCM, as well as the $\mathcal{L}_0$ as inputs. We shall explain $\mathcal{H}_1^G$ and $\mathbf{F}_1^G$ in the following subsection. To better recover a 3D scene with fine-grained geometric details, we further design LRM (to be detailed later), which conducts refinement locally according to the point displacement vectors $\mathcal{D}_1$ generated by GCM, and finally outputs $\mathcal{P}_1$ and $\mathcal{L}_1$.

The level #2 continues to upsample and complete $\mathcal{P}_1$ into a finer result $\mathcal{P}_2$ and its associated labels $\mathcal{L}_2$. Most operations are the same as in level #1, except that we further feed the hidden layer output $\mathcal{H}_1^S$ of SSM and the scene-wise features $\mathbf{F}_1^{SE}$ extracted by shared encoder (SE) in the first level to the second level's GCM as well. Intuitively, $\mathcal{H}_1^S$ encodes the semantic features extracted in the first level, while $\mathbf{F}_1^{SE}$ encodes the scene-wise global features embedded from the refined completed point cloud produced by the first level's LRM. In this way, we can fully utilize the information of the first level. Note that, we implement SE by using the same structure as the encoder in level #0.

Further, level #3 repeats the above process again. As shown in the right part of Figure 2, since the completion and segmentation results after the third level's GCM and SSM are good enough, we thus remove LRM in this level and use FPS to keep M points as the final outputs. Note that, FPS operation is optional, which can be used only when we have a specific requirement on the output point number.

Global Completion Module

The purpose of global completion module (GCM) is to produce a completed point cloud $\mathcal{P}_l^G$ with the upsampling ratio of λ in the l-th level, given the previous level's output $\mathcal{P}_{l-1}$.

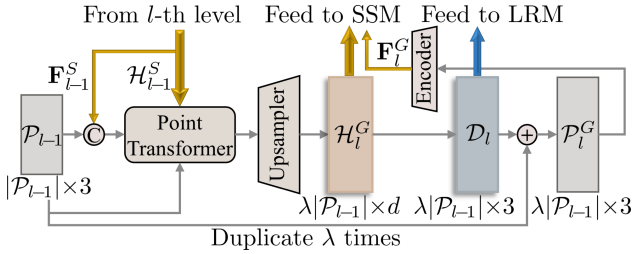

Figure 3: The detailed architecture of global completion module (GCM). Given $\mathcal{P}_{l-1}$ from previous level, GCM generates the completed scene $\mathcal{P}_l^G$ via upsampling and displacement regression. Note that, when $l = 1$, the inputs $\mathbf{F}_{l-1}^{SE}$ and $\mathcal{H}_{l-1}^{S}$ are replaced by $\mathbf{F}$ of level #0.

Figure 3 shows the detailed architecture, which is inspired by the snowflake point deconvolution (Xiang et al. 2021).

Specifically, when $l \geq 2$, besides using $\mathcal{P}_{l-1}$ as the input in the l-th level, GCM also consumes $\mathbf{F}_{l-1}^{SE}$ and $\mathcal{H}_{l-1}^{S}$ as inputs, which encodes the scene-wise features and the semantic features of previous level, respectively. Then, we employ a point transformer (Zhao et al. 2021a) to fuse the three kinds of inputs together. In this way, the obtained features encode not only the input point features (brought by $\mathcal{P}_{l-1}$), but also the rich spatial and semantic information of previous level (brought by $\mathbf{F}_{l-1}^{SE}$ and $\mathcal{H}_{l-1}^{S}$). Note that, when l=1, there is no $\mathbf{F}_{0}^{SE}$ and $\mathcal{H}_{0}^{S}$; see Figure 2 as an illustration. Hence, in the first level, besides $\mathcal{P}_0$, we further input $\mathbf{F}$ from level #0 to GCM. Next, an upsampler (Xiang et al. 2021) is employed to upsample the fused features with $|\mathcal{P}_{l-1}|$ elements to get features $\mathcal{H}_l^G \in \mathbb{R}^{\lambda|\mathcal{P}_{l-1}|\times d}$, where $|\cdot|$ denotes the number of points and d is the number of feature channels. Considering the fact that it is difficult for the network to directly predict the absolute point coordinates with diverse and wide distribution in 3D space, we here choose to regress per-point displacements $\mathcal{D}_l$ from $\mathcal{H}_l^G$, which are then added to the duplicated input points to obtain the final output $\mathcal{P}_l^G$.

Note that, to facilitate accurate semantic segmentation in each level, we further employ an encoder to extract features $\mathbf{F}_l^G$ from $\mathcal{P}_l^G$. We design this encoder to be the same as the encoder in level #0. Both $\mathcal{H}_l^G$ and $\mathbf{F}_l^G$ will be fed into this level's SSM, and $\mathcal{D}_l$ will be fed into this level's LRM.

Semantic Segmentation Module

The purpose of semantic segmentation module (SSM) is to predict the per-point semantic labels $\mathcal{L}_l^S$ for $\mathcal{P}_l^G$. Recall that, we generate $\mathcal{P}_l^G$ in GCM by adding the displacement vectors $\mathcal{D}_l$ to the duplicated $\mathcal{P}_{l-1}$. A common case is that a point may be moved from one object to another, while the two objects belong to different classes. In this case, if we still follow the common routine to treat each point label as a scalar value, the ground-truth semantic labels associated with some points may produce abrupt situations that cause instability in network training. To avoid this case, instead of encouraging our network to predict a scalar semantic value for each point, we propose to regress the probability of each class. Specifically, $\mathcal{L}_l^S = \{L_l^S(i) = [\mathbf{p}_{i,1}, \cdots, \mathbf{p}_{i,c}, \cdots, \mathbf{p}_{i,C}]\}$, where

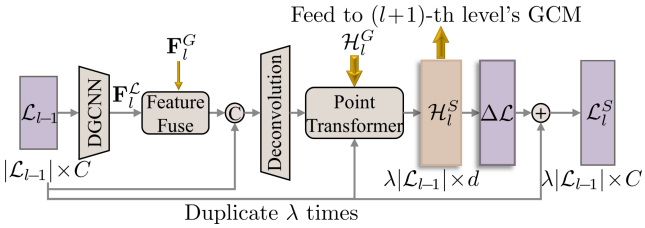

Figure 4: The detailed architecture of semantic segmentation module (SSM). Given $\mathcal{L}_{l-1}$ from previous level, SSM outputs the per-point labels for current level's completed point $\mathcal{P}_l^G$ by predicting label transition probability.

$L_l^S(i)$ denotes the semantic label of the i-th point in the l-th level, and $\mathbf{p}_{i,c} \in \mathbb{R}$ represents the probability of the i-th point belonging to the c-th semantic class.

Figure 4 shows the detailed architecture of our designed SSM. Given the semantic labels (probability) $\mathcal{L}_{l-1}$ from previous level, we first extract features $\mathbf{F}_l^{\mathcal{L}}$ by using DGCNN (Wang et al. 2019). Intuitively, $\mathbf{F}_l^{\mathcal{L}}$ encodes the probability distribution of semantics. Next, we fuse $\mathbf{F}_l^{\mathcal{L}}$ and the geometry features $\mathbf{F}_l^G$ produced by GCM together via attention-based feature fusion (Vaswani et al. 2017). The fused features provide global scene information, which can be used to obtain coarse per-point class probability by concatenating with $\mathcal{L}_{l-1}$ and passing through MLPs. After that, we employ deconvolution to upsample the coarse features with ratio λ. Considering that the label prediction highly relates to the point moving process during completion, we thus inject $\mathcal{H}_l^G$ which contains the hidden information of $\mathcal{D}_l$ into a point transformer to implicitly introduce the displacement information of this level's GCM to SSM. Further, the transformer also takes the duplicated $\mathcal{L}_{l-1}$ as an extra input to generate the upsampled per-point class probability features $\mathcal{H}_l^S \in \mathbb{R}^{\lambda|\mathcal{L}_{l-1}|\times d}$. With $\mathcal{H}_l^S$ in hand, traditional semantic segmentation models directly predict per-point labels. However, this manner ignores the relationship between point displacements and label changes. Thus, similar to GCM, we regress the label transition probability $\Delta\mathcal{L} \in \mathbb{R}^{\lambda|\mathcal{L}_{l-1}|\times C}$ from $\mathcal{H}_l^S$ and obtain the per-point labels by adding $\Delta\mathcal{L}$ to the duplicated $\mathcal{L}_{l-1}$. In addition, the hidden layer output $\mathcal{H}_l^S$ will be further fused into next level's GCM for finer points.

Local Refinement Module

The purpose of local refinement module (LRM) is to further refine the coarse completed points $\mathcal{P}_l^G$ and its associated labels $\mathcal{L}_l^S$ from a local perspective. As shown in Figure 5, LRM takes $\mathcal{P}_l^G$ or its labels $\mathcal{L}_l^S$ as well as the displacement vectors $\mathcal{D}_l$ as inputs and outputs the refined points $\mathcal{P}_l$ or refined labels $\mathcal{L}_l$. The key idea behind LRM comes from the assumption that a large displacement means that its corresponding point is most likely moved to the missing area. Thus, $\mathcal{D}_l$ can supply inductive knowledge and help locate large missing areas, which is also verified in our Experiment section. Hence, we design LRM to predict minor offsets again to tune the coarse completed points and their labels in the located missing regions, thereby optimizing the

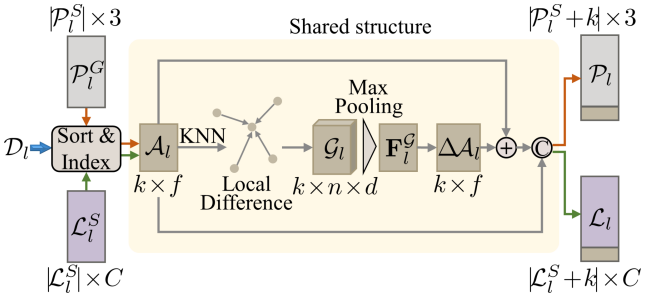

Figure 5: The detailed architecture of local refinement module (LRM). Given $\mathcal{P}_l^G$ or its associated labels $\mathcal{L}_l^S$, LRM regards the regions where points have large displacements as missing regions, then predicts minor offsets $\Delta\mathcal{A}_l$ on these points to refine the missing local regions again, and finally outputs the refined results $\mathcal{P}_l$ or $\mathcal{L}_l$.

associated local structures and labels with finer details.

Let's take $\mathcal{P}_l^G$ as an example to explain the local refinement process, and the process of refining $\mathcal{L}_l^S$ is the same with shared structure. Specifically, as shown in Figure 5, we first compute and sort the $L2$-norm of every displacement vector in $\mathcal{D}_l$. Then, we index $\mathcal{P}_l^G$ corresponding to the top k biggest displacement vectors. We here use agents $\mathcal{A}_l \in \mathbb{R}^{k\times f}$ to denote the selected points from $\mathcal{P}_l^G$, thus $f = 3$. Note that, $f{=}C$ when refining labels. To get the local structure around each reference point in $\mathcal{A}_l$, we employ KNN to group its n nearest points in $\mathcal{P}_l^G$. Next, we follow PointNet++ (Qi et al. 2017) to translate the coordinates of points in a local region into a local frame relative to the reference point, and further employ MLPs to extract per-point features in each group, which is denoted as $\mathcal{G}_l \in \mathbb{R}^{k\times n\times d}$. Then we employ max pooling to obtain the local structure features $\mathbf{F}_l^{\mathcal{G}}$. Similar to GCM, we also regress the offsets $\Delta\mathcal{A}_l \in \mathbb{R}^{k\times f}$ for agents, i.e. displacements of points or transition probability of labels. Then $\mathcal{A}_l$ is tuned by adding $\Delta\mathcal{A}_l$. Finally, the refined output $\mathcal{P}_l$ is obtained by concatenating the tuned $\mathcal{A}_l$ to the input $\mathcal{P}_l^G$. Naturally, the size of $\mathcal{P}_l$ is increased by k.

Loss Function

To fully guide network training in an end-to-end manner, we supervise both the completion and segmentation results in every level, i.e. from level #1 to level #3. In each level, we uniformly downsample ground truth points and their labels from the given ground truths in a dataset to the same number as each level's output, which is denoted as $\overline{\mathcal{P}}_l$ and $\overline{\mathcal{L}}_l$, respectively. For the completion task, we employ the widely-used Chamfer Distance (CD) to measure the difference between the completed points $\mathcal{P}_l$ and the ground truth points $\overline{\mathcal{P}}_l$:

$$\mathcal{L}_{CD} = \frac{1}{|\mathcal{P}_l|}\sum_{x\in\mathcal{P}_l}\min_{y\in\overline{\mathcal{P}}_l}\|x-y\| + \frac{1}{|\overline{\mathcal{P}}_l|}\sum_{y\in\overline{\mathcal{P}}_l}\min_{x\in\mathcal{P}_l}\|y-x\|, \tag{1}$$

where x and y are the points in $\mathcal{P}_l$ and $\overline{\mathcal{P}}_l$, respectively.

For the semantic segmentation task, we should note that there is no one-to-one correspondence between $\mathcal{P}_l$ and $\overline{\mathcal{P}}_l$. Thus, we cannot directly use the traditional cross-entropy loss as other segmentation works. However, the lucky thing is that such one-to-one mapping has already been established when calculating Eq. 1. We thus leverage such correspondence to locate the nearest ground-truth point for each predicted point and use their labels for semantic loss computation. Further, it is common that objects in a scene have various sizes, thus leading to different numbers of points on different objects. For example, the point number in class of Floor or Wall is much larger than that in class of Table or Chair. Hence, the semantic segmentation task often suffers from critical class imbalance. To relieve the overfitting to easy examples (e.g., Floor or Wall), we employ focal loss (Lin et al. 2017) as our semantic segmentation loss:

$$\mathcal{L}_{sem} = \frac{1}{|\mathcal{L}_l|}\sum_{l\in\mathcal{L}_l} -(1-l)^{\gamma}\log(l), \tag{2}$$

where l denotes the predicted labels gathered by ground-truth labels $\bar{l} \in \overline{\mathcal{L}}_l$, and γ is focusing parameter (Lin et al. 2017), which is increased during training to gradually increase the importance of hard examples with fewer points.

The overall loss $\mathcal{L}_{SSC}$ consists of both $\mathcal{L}_{CD}$ and $\mathcal{L}_{sem}$, which are balanced with parameter α:

$$\mathcal{L}_{SSC} = \sum_{l=1}^{3}\mathcal{L}_{CD}(\mathcal{P}_l, \overline{\mathcal{P}}_l) + \alpha\mathcal{L}_{sem}(\mathcal{L}_l, \overline{\mathcal{L}}_l). \tag{3}$$

Experiments and Results

Experimental Settings

Dataset preparation. As there is no public dataset for point cloud semantic scene completion, we prepared two datasets, namely SSC-PC and NYUCAD-PC. Specifically, SSC-PC is compiled based on the dataset provided in (Zhang et al. 2021a). This work provides a total of 1941 scenes covering 16 object categories in the data format of RGB-D images with associated ground-truth point clouds. However, they do not provide camera intrinsics, so input partial point clouds cannot be obtained directly from RGB-D images. Hence, we generate the input point clouds from ground truth points by approximating the visibility of point clouds (Katz, Tal, and Basri 2007) from given views. We follow (Zhang et al. 2021a) to keep 4096 points for both network input and output. Finally, we randomly split the 1941 scenes with 1543 for training and the remaining 398 for testing.

NYUCAD-PC is derived from NYUCAD dataset (Firman et al. 2016), which contains 1449 RGB-D images with 795 training samples and 654 testing samples. The partial input with 4096 points are generated from depth images by using camera intrinsics. The ground truths with 8192 points are sampled from CAD mesh annotations (Guo, Zou, and Hoiem 2015) with Poisson sampling algorithm. The object categories in ground truths are mapped following (Handa et al. 2016). Note that the "empty" class is excluded, thus the number of categories is 11. Because of severe occlusion between objects and incomplete vision of scenes, NYUCAD-PC is much more challenging than SSC-PC.

Evaluation metrics. To evaluate the scene completion performance, we employ the widely-used Chamfer Distance

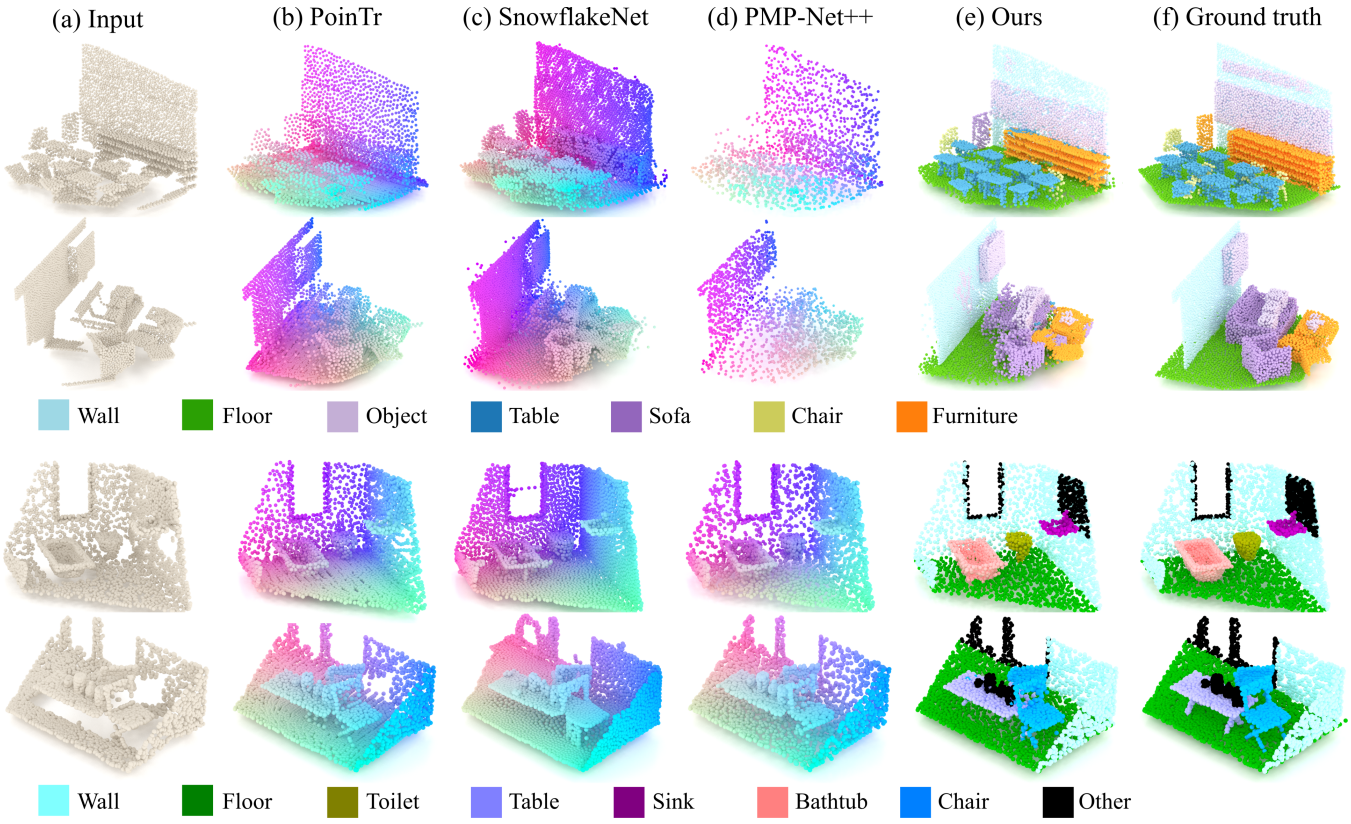

Figure 6: Comparing the scene completion results of our method (e) and recent single object point completion methods (b-d).

(CD) as evaluation metric. To evaluate the semantic segmentation performance, we employ the mean class IoU (mIoU) and the mean class accuracy (mAcc) as metrics.

Implementation details. We implement our network in PyTorch and train it on a single Nvidia RTX 3090 GPU for 400 epochs with batch size of 4. We use Adam optimizer. The learning rate is initialized as 0.001 and decayed every two steps with 0.99 rate. The modulating factor γ in Eq. 2 ranges from 0 to 5 with step size of 0.5 every 30 epochs. The balancing parameter α in Eq. 3 is 0.1. In experiments, the number of points to be refined in LRM is 96, 64 in level #1 and #2, respectively. The input is downsampled to 2048 in level #0. In NYUCAD-PC dataset, we cascade five levels and set the refined point number as 32 in LRM of level #3.

Quantitative and Qualitative Comparisons

Comparing with semantic scene completion methods. We first compared our CasFusionNet with all existing point-based SSC networks, including PCSSC-Net (Zhang et al. 2021a) and (Wang et al. 2022). For (Wang et al. 2022), we directly trained their released code on our prepared two datasets. For PCSSC-Net, it demands a 12-dimension feature vector including point coordinates, RGB values, normal vector, and differential coordinates as network input, so their released code cannot be directly trained on our prepared datasets with only point coordinates. However, our SSC-PC dataset is prepared from the original paper of PCSSC-Net, that shares the same 3D scenes, but with different point sampling. We thus directly used the evaluation values published in their original paper for comparison. Strictly speaking, such comparison is actually unfair for our method, since the CD value calculated between our prepared input and ground truth is much larger than that in (Zhang et al. 2021a), indicating that our prepared SSC-PC is more difficult.

Method	SSC-PC			NYUCAD-PC		
	CD	mIoU	mAcc	CD	mIoU	mAcc
PCSSC-Net	1.58	88.2	-	-	-	-
Wang et al.2022	6.72	7.4	10.1	23.45	10.2	14.0
Ours	0.70	**91.3**	**94.8**	10.28	**49.5**	59.7
Ours (no FPS)	**0.67**	91.2	94.8	**9.99**	49.5	**59.8**

Table 1: Comparing semantic scene completion results of our method against recent SSC methods. Ours (no FPS) means we directly use the output of last level without FPS for evaluation. L1 and L2 Chamfer Distance (multiplied by 10^3) are used in NYUCAD and SSC-PC, respectively.

Table 1 shows the comparison results. Clearly, our method (last two rows) achieves the best values across all metrics on both datasets. Particularly, when we remove FPS in the last level (see Figure 2) and directly use the raw output for evaluation, the performance is further improved; see the bot-

Method	SSC-PC	NYUCAD-PC
PoinTr (Yu et al. 2021)	11.85	14.91
SnowflakeNet (Xiang et al. 2021)	13.72	11.83
PMP-Net++ (Wen et al. 2022)	10.74	13.84
Ours	8.96	10.28
Ours (no FPS)	**8.73**	**9.99**

Table 2: Comparing scene completion results with recent single object completion methods in terms of L1 Chamfer Distance (multiplied by 10^3).

tom row. Note that, the method (Wang et al. 2022) fails to produce a satisfying performance. We think that this is because our prepared inputs are quite incomplete and sparse, thus preventing their proposed encoder-decoder architecture to capture a representative feature of a scene.

Comparing with single object completion methods. Since most point cloud completion methods focus on a single object, we thus compared our CasFusionNet with newly proposed single object point completion methods, i.e. PoinTr (Yu et al. 2021), PMP-Net++ (Wen et al. 2022) and SnowflakeNet (Xiang et al. 2021), in terms of the scene completion performance. In detail, we trained their networks on our prepared datasets. Though our method further performs semantic segmentation, we only compare the CD value. As shown in Table 2, even with FPS, our method still achieves the lowest CD values with a significant margin compared to others, showing that directly employing single object completion methods to complete a 3D scene is not workable.

Figure 6 further shows the qualitative comparisons on two datasets, where the top two scenes are from NYUCAD-PC and the bottom two scenes are from SSC-PC. Clearly, regardless of the segmentation results, the completion results produced by our method (e) is the closest to the ground truths (f) against others (b-d), especially on the local details.

Ablation Study

To evaluate the effectiveness of the major components in our method, we conducted an ablation study by simplifying CasFusionNet in the following cases.

- *Model A*: we remove both LRM and SSM in all levels.
- *Model B*: we remove only LRM in all levels.
- *Model C*: we remove the feature fusion between GCM and SSM. More specifically, the features of previous level's GCM are fed into the next level's GCM, and the features of previous level's SSM are fed into the next level's SSM.
- *Model D*: we remove levels #2 & #3, and only keep levels #0 & #1.

We re-trained the network model separately for each case using the same training dataset of NYUCAD-PC, and the evaluation results are summarized in Table 3. By comparing Model A & B vs. our full pipeline, we can see that both LRM and SSM contribute to better performance, particularly on the scene completion task. By comparing Model C with our full pipeline and Model B, we can see that removing feature fusion results in poor performance on both scene completion and semantic segmentation. Especially, the performance of semantic segmentation in Model C has a serious setback, thus validating the importance of our designed feature fusion. By comparing Model D with our full pipeline, it shows that our cascaded network design with multiple levels certainly improves task learning ability.

Model	Scene completion	Semantic segmentation	
	CD ($\times 10^{-3}$)	mIoU	mAcc
A	10.73	-	-
B	10.50	49.0	59.8
C	10.61	40.3	51.9
D	11.60	41.5	52.2
Ours	**10.28**	**49.5**	**59.8**

Table 3: Ablation analysis of our network on NYUCAD-PC.

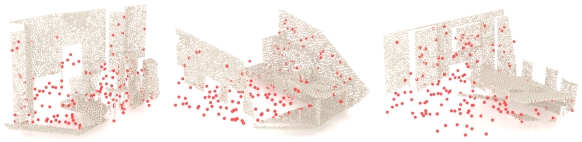

Figure 7: The points with large displacement values are mostly located in missing areas; see these red points.

Network Analysis and Discussions

Visualization of the located missing regions. As mentioned in Method section, we regard the points with large displacement values as missing regions. To validate this, we visualize the top k points back in the partial input scene; see Figure 7 as an illustration, where the k points are rendered in red color. Clearly, most of the red points are located in missing areas, thus validating the rationality of our assumption.

Limitations. First, as a common drawback of existing works, our CasFusionNet still fails to recover local details for heavily occluded objects. Second, our network extracts both scene-wise and point-wise features, but the object-wise features are not fully utilized, which might be useful for better completing objects. At last, our current network can handle about 6 scenes each with 4096 points in one second. The time performance may be further improved if we design a lightweight feature extractor in the future.

Conclusion

In this work, we present CasFusionNet, a novel point cloud semantic scene completion network by dense feature fusion. By cascaded fusing the geometry and semantic information, our network jointly completes the missing areas and predicts the per-point semantic labels of scenes. Considering that there is no public dataset for point-based SSC, we thus prepared two datasets. Both quantitative and qualitative results show that our network outperforms state-of-the-arts significantly. In the future, we shall explore the possibility of utilizing object-level features based on the predicted semantic labels for precise object completion.

Acknowledgments

This work is supported by the China National Natural Science Foundation No. 62202182, No. 62276109, No. 62176101.

References

Cai, Y.; Chen, X.; Zhang, C.; Lin, K.-Y.; Wang, X.; and Li, H. 2021. Semantic Scene Completion via Integrating Instances and Scene in-the-Loop. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 324–333.

Chen, X.; Lin, K.-Y.; Qian, C.; Zeng, G.; and Li, H. 2020. 3D Sketch-aware Semantic Scene Completion via Semi-supervised Structure Prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4193–4202.

Deng, X.; Hu, X.; Buris, N. E.; An, P.; and Chen, Y. 2021. 3D GRID TRANSFORMATION NETWORK FOR POINT CLOUD COMPLETION. In *2021 IEEE International Conference on Image Processing (ICIP)*, 3642–3646. IEEE.

Dourado, A.; De Campos, T. E.; Kim, H.; and Hilton, A. 2021. EdgeNet: Semantic Scene Completion from a Single RGB-D Image. In *Proceedings of International Conference on Pattern Recognition*, 503–510. IEEE.

Dourado, A.; Guth, F.; and de Campos, T. 2022. Data Augmented 3D Semantic Scene Completion with 2D Segmentation Priors. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3781–3790.

Firman, M.; Mac Aodha, O.; Julier, S.; and Brostow, G. J. 2016. Structured Prediction of Unobserved Voxels From a Single Depth Image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5431–5440.

Garbade, M.; Chen, Y.-T.; Sawatzky, J.; and Gall, J. 2019. Two Stream 3D Semantic Scene Completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 416–425.

Guo, R.; Zou, C.; and Hoiem, D. 2015. Predicting Complete 3D Models of Indoor Scenes. *arXiv preprint arXiv:1504.02437*.

Guo, Y.; and Tong, X. 2018. View-Volume Network for Semantic Scene Completion from a Single Depth Image. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 726–732.

Gupta, S.; Davidson, J.; Levine, S.; Sukthankar, R.; and Malik, J. 2017. Cognitive Mapping and Planning for Visual Navigation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2616–2625.

Handa, A.; Patraucean, V.; Badrinarayanan, V.; Stent, S.; and Cipolla, R. 2016. Understanding Real World Indoor Scenes With Synthetic Data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4077–4085.

Huang, T.; Zou, H.; Cui, J.; Yang, X.; Wang, M.; Zhao, X.; Zhang, J.; Yuan, Y.; Xu, Y.; and Liu, Y. 2021. RFNet: Recurrent Forward Network for Dense Point Cloud Completion. In *IEEE International Conference on Computer Vision (ICCV)*, 12508–12517.

Katz, S.; Tal, A.; and Basri, R. 2007. Direct Visibility of Point Sets. In *ACM Transactions on Graphics (SIGGRAPH)*, 24–es.

Li, J.; Ding, L.; and Huang, R. 2021. IMENet: Joint 3D Semantic Scene Completion and 2D Semantic Segmentation through Iterative Mutual Enhancement. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 793–799.

Li, J.; Han, K.; Wang, P.; Liu, Y.; and Yuan, X. 2020a. Anisotropic Convolutional Networks for 3D Semantic Scene Completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3351–3359.

Li, J.; Liu, Y.; Gong, D.; Shi, Q.; Yuan, X.; Zhao, C.; and Reid, I. 2019. RGBD Based Dimensional Decomposition Residual Network for 3D Semantic Scene Completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7693–7702.

Li, J.; Wang, P.; Han, K.; and Liu, Y. 2021. Anisotropic Convolutional Neural Networks for RGB-D based Semantic Scene Completion. *IEEE Transactions Pattern Analysis & Machine Intelligence*.

Li, S.; Zou, C.; Li, Y.; Zhao, X.; and Gao, Y. 2020b. Attention-based Multi-modal Fusion Network for Semantic Scene Completion. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, 11402–11409.

Liang, Y.; Chen, B.; and Song, S. 2021. SSCNav: Confidence-Aware Semantic Scene Completion for Visual Semantic Navigation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 13194–13200. IEEE.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal Loss for Dense Object Detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2980–2988.

Liu, M.; Sheng, L.; Yang, S.; Shao, J.; and Hu, S.-M. 2020. Morphing and Sampling Network for Dense Point Cloud Completion. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, 11596–11603.

Liu, S.; Hu, Y.; Zeng, Y.; Tang, Q.; Jin, B.; Han, Y.; and Li, X. 2018. See and Think: Disentangling Semantic Scene Completion. In *Advances in Neural Information Processing Systems*, 261–272.

Pan, L.; Chen, X.; Cai, Z.; Zhang, J.; Zhao, H.; Yi, S.; and Liu, Z. 2021. Variational Relational Point Completion Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 8524–8533.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, volume 30.

Song, S.; Yu, F.; Zeng, A.; Chang, A. X.; Savva, M.; and Funkhouser, T. 2017. Semantic Scene Completion from a Single Depth Image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1746–1754.

Tang, J.; Chen, X.; Wang, J.; and Zeng, G. 2022. Not All Voxels Are Equal: Semantic Scene Completion from the

Point-Voxel Perspective. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 36, 2352–2360.

Varley, J.; DeChant, C.; Richardson, A.; Ruales, J.; and Allen, P. 2017. Shape Completion Enabled Robotic Grasping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2442–2447. IEEE.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30.

Wang, P.-S.; Liu, Y.; and Tong, X. 2020. Deep Octree-based CNNs with Output-Guided Skip Connections for 3D Shape and Scene Completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 266–267.

Wang, X.; Lin, D.; and Wan, L. 2022. FFNet: Frequency Fusion Network for Semantic Scene Completion. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 36, 2550–2557.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions on Graphics*, 38(5): 1–12.

Wang, Y.; Tan, D. J.; Navab, N.; and Tombari, F. 2022. Learning Local Displacements for Point Cloud Completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1568–1577.

Wen, X.; Li, T.; Han, Z.; and Liu, Y.-S. 2020. Point Cloud Completion by Skip-attention Network with Hierarchical Folding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1939–1948.

Wen, X.; Xiang, P.; Han, Z.; Cao, Y.-P.; Wan, P.; Zheng, W.; and Liu, Y.-S. 2021. PMP-Net: Point Cloud Completion by Learning Multi-step Point Moving Paths. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7443–7452.

Wen, X.; Xiang, P.; Han, Z.; Cao, Y.-P.; Wan, P.; Zheng, W.; and Liu, Y.-S. 2022. PMP-Net++: Point Cloud Completion by Transformer-Enhanced Multi-step Point Moving Paths. *IEEE Transactions Pattern Analysis & Machine Intelligence*.

Xia, Y.; Xia, Y.; Li, W.; Song, R.; Cao, K.; and Stilla, U. 2021. ASFM-Net: Asymmetrical Siamese Feature Matching Network for Point Completion. In *Proceedings of the 29th ACM International Conference on Multimedia*, 1938–1947.

Xiang, P.; Wen, X.; Liu, Y.-S.; Cao, Y.-P.; Wan, P.; Zheng, W.; and Han, Z. 2021. Snowflake Point Deconvolution for Point Cloud Completion and Generation with Skip-Transformer. In *IEEE International Conference on Computer Vision (ICCV)*, 5499–5509.

Xie, H.; Yao, H.; Zhou, S.; Mao, J.; Zhang, S.; and Sun, W. 2020. GRNet: Gridding Residual Network for Dense Point Cloud Completion. In *European Conference on Computer Vision (ECCV)*, 365–381.

Yu, X.; Rao, Y.; Wang, Z.; Liu, Z.; Lu, J.; and Zhou, J. 2021. PoinTr: Diverse Point Cloud Completion with Geometry-Aware Transformers. In *IEEE International Conference on Computer Vision (ICCV)*, 12498–12507.

Yuan, W.; Khot, T.; Held, D.; Mertz, C.; and Hebert, M. 2018. PCN: Point Completion Network. In *2018 International Conference on 3D Vision (3DV)*, 728–737. IEEE.

Zhang, J.; Zhao, H.; Yao, A.; Chen, Y.; Zhang, L.; and Liao, H. 2018. Efficient Semantic Scene Completion Network with Spatial Group Convolution. In *European Conference on Computer Vision (ECCV)*, 733–749.

Zhang, S.; Li, S.; Hao, A.; and Qin, H. 2021a. Point Cloud Semantic Scene Completion from RGB-D Images. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, 3385–3393.

Zhang, X.; Feng, Y.; Li, S.; Zou, C.; Wan, H.; Zhao, X.; Guo, Y.; and Gao, Y. 2021b. View-Guided Point Cloud Completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 15890–15899.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021a. Point Transformer. In *IEEE International Conference on Computer Vision (ICCV)*, 16259–16268.

Zhao, X.; Zhang, B.; Wu, J.; Hu, R.; and Komura, T. 2021b. Relationship-based Point Cloud Completion. *IEEE Transactions Visualization & Computer Graphics*.

Zhong, M.; and Zeng, G. 2020. Semantic Point Completion Network for 3D Semantic Scene Completion. In *24th European Conference on Artificial Intelligence*, 2824–2831. IOS Press.

Zong, D.; Sun, S.; and Zhao, J. 2021. ASHF-Net: Adaptive Sampling and Hierarchical Folding Network for Robust Point Cloud Completion. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, 3625–3632.