Voxel-based Network for Shape Completion by Leveraging Edge Generation

Xiaogang Wang Marcelo H Ang Jr Gim Hee Lee
National University of Singapore
xiaogangw@u.nus.edu {mpeangh,gimhee.lee}@nus.edu.sg

Abstract

Deep learning technique has yielded significant improvements in point cloud completion with the aim of completing missing object shapes from partial inputs. However, most existing methods fail to recover realistic structures due to over-smoothing of fine-grained details. In this paper, we develop a voxel-based network for point cloud completion by leveraging edge generation (VE-PCN). We first embed point clouds into regular voxel grids, and then generate complete objects with the help of the hallucinated shape edges. This decoupled architecture together with a multi-scale grid feature learning is able to generate more realistic on-surface details. We evaluate our model on the publicly available completion datasets and show that it outperforms existing state-of-the-art approaches quantitatively and qualitatively. Our source code is available at https://github.com/xiaogangw/VE-PCN.

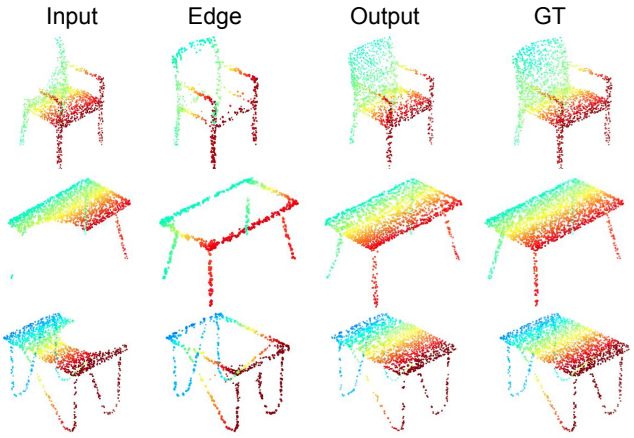

Figure 1: We propose an edge-guiding and voxel-based point cloud completion network to reconstruct complete points from incomplete inputs. Under the guidance of hallucinated edges and the help of gridding structures, we are able to generate fine-grained details for the thin structures.

1. Introduction

3D shape completion is a fundamental problem in computer vision and robotic perception. The aim is to reconstruct complete object topologies from sparse and incomplete observations, *e.g.* raw data collected by RGB-D or LiDAR sensors. Since incompleteness and irregularity of input point clouds impose difficulties on down-stream tasks such as 3D object classification [25, 26, 39, 16], segmentation [25, 26, 17] and detection [3, 24, 44, 49], several point cloud completion methods [46, 31, 36, 20, 41, 28] are proposed to improve the quality of the point clouds. Although existing works have achieved impressive results, they are limited to low-fidelity outputs. In this work, we focus on generating high-quality 3D objects from occluded inputs.

Numerous methods have attempted to achieve shape completion from different representations, *e.g.* meshes [9, 34], implicit fields [4, 18, 21, 30] and point clouds [46, 31, 36, 20, 41, 28, 2, 48, 14, 37, 38, 40, 43, 12, 10]. Meshes represent object shapes by a set of vertices and edges. Despite their ability to reconstruct complex object structures, it is difficult to change shape topologies during training due to the fixed vertex connection patterns. Implicit fields represent shapes by signed distance functions (SDF) [23] that are more flexible to achieve arbitrary resolutions. However, learning an accurate SDF requires a large amount of sampling for a single object. In contrast, point clouds are concise 3D shape representations and easier to add new points during training. Nonetheless, previous point cloud shape completion works [46, 31] are struggling at synthesizing surface details since point clouds are unordered and challenging to control. To alleviate this problem, recent works [20, 36, 43] propose to add a skip connection between the partial input and the decoder and have shown better reconstruction on object details. Despite the effort, the results for complex topologies are still inferior because of unrealistic points generated on the missing part.

In view of these limitations, we propose a voxel-based shape completion network that leverages the generated object edges to recover more fine-grained details. The voxel representation has also been used in GRNet [43], where a Gridding and Gridding Reverse layer are proposed for conversion between unordered points and 3D grids. However, their voxel representation is only used to reconstruct low-

resolution shapes, and additional multi-layer perceptrons (MLPs) [25] are applied for denser point set generation. To fully integrate the voxel representation for completion in an end-to-end manner, we generate points for every grid cell by predicting a binary classification score that indicates a cell being empty or occupied and a probability density value that estimates the number of points in the cell inspired by [19]. This operation is differentiable everywhere and thus makes it easier to generate accurate coordinates according to various shape topologies. Moreover, the end-to-end grid strategy allows us to approximate object shapes without running into memory issues since we can sample an arbitrary number of points for each grid cell. In this way, we are able to use a lower voxel resolution, *e.g.* $32 \times 32 \times 32$ compared to $64 \times 64 \times 64$ in GRNet, and concurrently achieve superior results. In addition, we introduce a multi-scale grid transformation module to learn critical object shapes in different resolutions, while GRNet only considers the full resolutional inputs.

Although the voxel representation is shown to be superior on calculating local features [43, 11, 35], it is challenging to generate arbitrary thin structures for various objects. We thus further introduce an edge generator to enhance our network. Since object structures are well-represented by their edges or contours, it is beneficial to incorporate the edge information when generating complete 3D shapes. Several examples of edge generation and point cloud completion are shown in Figure 1.

We evaluate our model on both the synthetic and real-world datasets. Qualitative and quantitative experiments are compared against existing state-of-the-art schemes. Our key contributions are as follows:

- We design a multi-scale voxel-based network to generate fine-grained details for point cloud completion.
- We incorporate object structure information into the shape completion by leveraging edge generation.
- We achieve state-of-the-art performances on different point cloud completion datasets.

2. Related work

In this section, we discuss the recent developments of 3D learning on point cloud analysis and completion.

2.1. Point Cloud Analysis

The pioneering work PointNet [25] proposes a MLP-based network for shape classification and segmentation. They successfully learn the global shape by applying a max-pooling operation on the point features. However, they ignore the point relationships within a local area. In view of this limitation, PointNet++ [26] proposes a hierarchical point set feature learning module for feature extraction. Apart from the above MLP-based method, several works [17, 32] adopted convolution-based operations in image processing onto point clouds. Although they have achieved impressive performances on classification and segmentation, they are limited to complete and clean point sets. They are thus not applicable to real-world data due to noisy and incompleteness. This motivates us to design point completion networks to improve the performances of down-stream tasks.

2.2. Point Cloud Completion

We roughly categorize point cloud completion into three classes: fully supervised methods, semi-supervised methods and self-supervised methods. The majority of works [46, 31, 36, 43, 22] follow the fully supervised manner. PCN [46] proposes an encoder-decoder architecture for point cloud completion. A following work TopNet [31] shares the same encoder architecture with PCN and proposes a tree-structure decoder to represent different object parts by several tree branches. Although PCN and TopNet have achieved good performances in recovering the object shapes, they are incapable of synthesizing shape details. In view of this, MSN [20] and CRN [36] proposes to preserve the details of the objects from partial inputs and refine the geometric structures for the missing regions. Following work GRNet [43] proposes a similar gridding network for dense point reconstruction. We differ from them as follows: 1) They use differentiable gridding and gridding reverse operations for conversion between points and grids, while our method adopts a different conversion approach and also proposes an edge generation pipeline to improve the completion performance. We also consider partial features from different scales during shape completion (§3.2). 2) GRNet can only obtain coarse outputs by their voxelization strategy and generate dense results by MLPs. In contrast, our method is able to directly generate dense and complete points in an end-to-end manner. Another work SK-PCN [22] proposes a similar thought that adopts skeleton generations to help the shape completion. However, our edges are different from their meso-skeleton. Their meso-skeleton focus on the overall shapes, while our edges focus on high frequency components (*e.g.* thin structures), which are difficult to generate in existing methods. Moreover, SK-PCN generates the complete points by learning displacements from skeletal points with a local adjustment strategy, while we synthesize the complete points by injecting the edge features into the completion decoder with a multi-scale voxelization strategy.

Instead of training the completion models using the fully complete ground truth, some works [10, 2, 38] propose semi-supervised or self-supervised methods to generate complete points. Gu $et\ al.$ [10] propose a weakly supervised completion method, which estimates 3D canonical

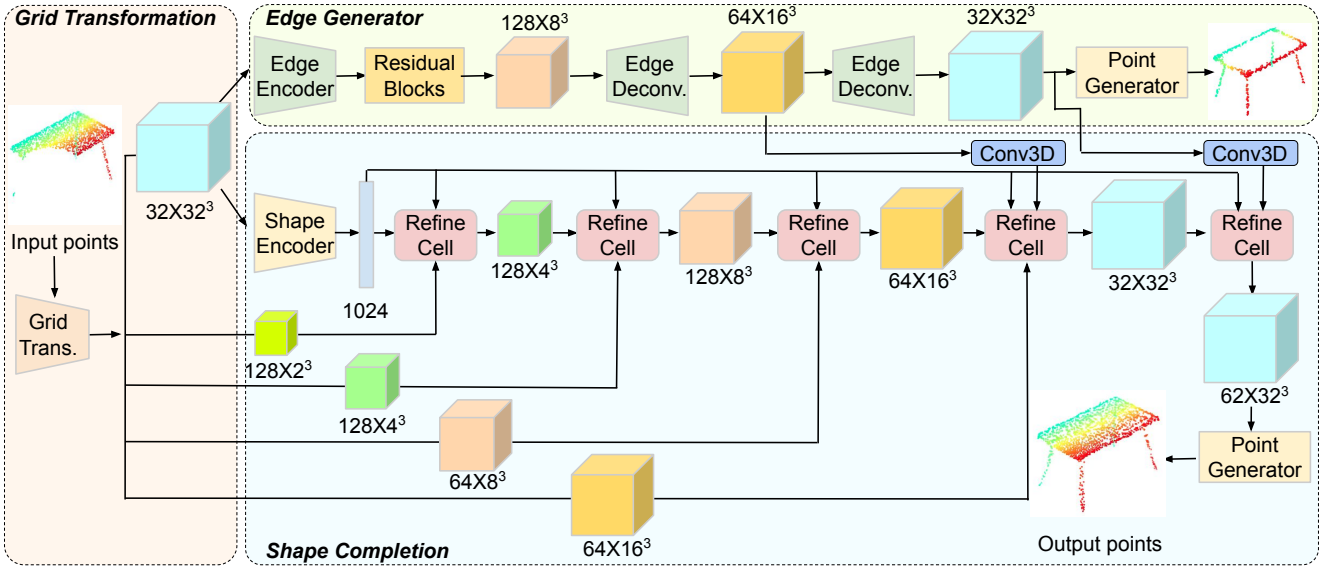

Figure 2: Overview of our network architecture. Given an incomplete point cloud, we first transform it to a stack of 3D features by a grid transformation module in the left part. The top right branch shows the architecture of the edge generator that reconstructs edges from the grid feature $\mathrm{P}_f^0 \in R^{32\times32^3}$. The bottom right branch presents the shape completion part, which includes a shape encoder, refinement modules and a shape decoder. The shape encoder maps the grid feature to a latent embedding $z \in R^{1024}$ that is used to project features from different scales to the output by refinement cells. The shape decoder predicts the complete point clouds by a sampling strategy.

shapes and 6-DoF poses simultaneously from multiple partial observations. They infer the missing regions by optimizing multi-view constraints. To further alleviate the labeling problem, Chen *et al.* [2] propose an unpaired point cloud completion method by the adversarial training. They use generative adversarial networks (GAN) [8] to reconstruct the complete objects for the real-world partial scans. Despite their impressive performance, they need to pre-train two auto-encoders and another GAN on the latent embeddings. Wang *et al.* [38] propose a self-supervised approach for the point cloud completion, which consists of two self-training strategies when the ground truth training data are not available. Although we train our model with a supervised manner, we also evaluate it on the unseen categories to show our generalizability.

3. Our Method

3.1. Overview

Our objective is to reconstruct complete and high-quality 3D objects P' given sparse and corrupted point clouds P. Figure 2 shows the illustration of our pipeline. Most previous works [46, 36, 20] process point sets by MLP-based neural networks with a coarse-to-fine manner and are struggling to reconstruct object details, which mainly due to two reasons: 1) the coarse outputs created from global embeddings lose the high-frequency information for 3D objects; 2) the second stage acts as a point upsampling function that is also incapable of synthesizing complex topologies. To circumvent these problems, we first design an edge generator to generate the object edges. These generated edges which encode the high-frequency information of an object are then used to help the completion network to generate object details. We further propose to adopt grid representations for both the edge generation and shape completion in view of the success of voxel-based techniques [43, 11, 35] on local feature embedding. We consider multi-scale voxel features to enhance the learning of shape structures. Overall, our point completion network consists of three stages: 1) a multi-scale grid transformer to project raw points into voxel representations, 2) an object edge generator to obtain shape edges, and 3) a shape completion network to generate complete point sets. Detailed network architectures are shown in our supplementary materials.

3.2. Grid Transformation

Figure 3 gives the detailed illustration of the grid transformation module shown on the left of Figure 2. The grid transformer is used to project the partial points into 3D grids for the edge generator and shape completion network. Instead of directly obtaining the binary representations that lead to the loss of structure information in previous works [5], we learn a stack of grid features $\mathrm{P}_f^i, \{i = 0, 1, 2, 3, 4\}$ by a multi-scale grid transformation module. Specifically, we first downsample the raw partial point cloud into another four sparser points sets with a downsampling ratio of 2. We then calculate all the point offsets between the point coordinates and their eight nearby grid

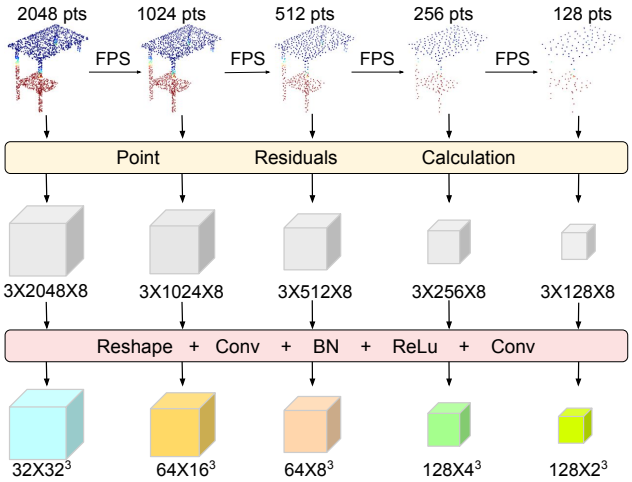

Figure 3: Illustration of the grid transformation architecture. FPS refers to farthest point sampling [26].

vertexes. This results in a tensor with the size of $3 \times \mathrm{N} \times 8$, where N is the point number. Subsequently, two 3D convolutions with batch normalization [15] and ReLU activation between them are used to obtain five grid features P_f^i. The five features represent the shape features from different point resolutions. We take mean values for all the point features that lie in the same grid as the per-grid representations. The size of the five grid features P_f^i are 32×32^3, 64×16^3, 64×8^3, 128×4^3 and 128×2^3, respectively.

3.3. Edge Generator

The top right of Figure 2 shows our encoder-decoder structure of the edge generator. The ground truth edges are obtained by a light-weight algorithm [1].

3.3.1 Edge Encoder

We extract high dimensional voxel features E_f in the edge encoder by two 3D convolutional blocks that downsample P_f^0 by two times. Each convolutional block includes one 3D convolution, instance normalization (IN) [33] and ReLU. Given the success of residual architecture [13] in deep learning, we refine the edge features E_f by three residual blocks. Each block consists of two $3 \times 3 \times 3$ convolutional layers and INs. ReLU activation is added after the first IN layer.

3.3.2 Edge Decoder

The edge decoder includes two deconvolutional layers followed by one IN and ReLU activation. It upscales the feature size by a factor of 2 each time. Another convolution is used to obtain the binary prediction score for each grid cell. The outputs of the edge decoder used in the shape completion module are two grid features, *i.e.*, $\mathrm{P}_e^{16} \in R^{1\times 16^3}$ and $\mathrm{P}_e^{32} \in R^{1\times 32^3}$.

3.3.3 Point Generator

The point generator aims at generating point clouds from the grid representations, *e.g.* P_e^{16} and P_e^{32}. We assume the local shape within a voxel can be approximated by a surface patch [45, 9] and thus adopt the folding mechanism to repeat the point features within a grid cell to obtain numerous point features f_g. Consequently, we generate multiple points for every voxel by predicting p_c and δ_c from f_g for each grid cell instead of generating one point for each cell [43] inspired by [19]. p_c is a non-zero classification score indicating whether the cell is occupied or empty. δ_c is the density value indicating the number of points in a given cell. We predict the residual offsets for each grid cell, and the final complete point clouds P' are obtained by adding the offsets on the corresponding cell centers.

3.4. Shape Completion

The shape completion network is shown in the bottom right of Figure 2, which includes a shape encoder, an iterative refinement module and a point generator. The inputs of the shape completion are the partial grid features P_f^i and the edge representations P_e^{16} and P_e^{32}. The completion synthesis begins at a small resolution of 2^3, and grid features are iteratively refined with the help of Adaptive Instance Normalization (AdaIN) [19].

3.4.1 Shape Encoder

The shape encoder is used to extract global embeddings $z \in R^{1024}$ from the partial features. Given the highest resolutional spatial feature $\mathrm{P}_f^0 \in R^{32\times 32^3}$, four partial grid features C_f^i are obtained by four 3D convolutional blocks. The number of feature channels are doubled and the spatial size are downsampled by a factor of 2^3 as i increases from 0 to 3 in different blocks. These four convolution blocks consist of two 3D convolutions and one batch normalization and a max-pooling layer. $z \in R^{1024}$ is calculated by downsampling the feature $\mathrm{C}_f^3 \in R^{2^3\times 512}$ by another convolutional block followed by a reshape operation. This block downsamples the spatial size by setting the padding value to 0 in the 3D convolution.

3.4.2 Refinement Modules

Refinement modules aim to refine and upsample the grid features during the shape completion process. Every refinement cell operates at a specific spatial size and comprises a series of upsampling, 3D convolution, instance normalization, affine feature transformation and ReLU activation. The input x for each refinement cell includes three elements: the output from the previous layer, the partial point feature P_f^i obtained from the grid transformation module

and the binary edge features calculated from the edge generator. They are concatenated on channel dimensions. Unlike GRNet [43] which only considers the global information in the first layer of the decoder, we incorporate the global embedding in all the subsequent layers of the decoder by AdaIN. We first calculate the channel-wise mean $\mu(y)$ and variance $\sigma(y)$ for input x from the global embedding z by one fully connected layer. The input features x are then projected by $\text{AdaIN}(x, y) = \sigma(y)(\frac{x-\mu(x)}{\sigma(x)}) + \mu(y)$. $\mu(x)$ and $\sigma(x)$ are the channel-wise mean values and variances of input x. The overall number of cascaded refinement modules depends on the output resolution, which is five for the resolution of $32 \times 32 \times 32$ in our work. We generate object points from the grid features by the same point generator in edge generation with different parameters.

3.5. Optimization

Since our edge generator and point completion network are both differentiable, our entire voxel-based network can be optimized in an end-to-end manner. The overall constraints consist of the losses for the complete outputs and edge results.

We calculate the Chamfer Distance (CD) [6] and its sharper version [19] on the output point clouds P' and the ground truth points Q as:

$$\mathcal{L}_{\text{CD}} = \frac{1}{\mathrm{N}} \sum_{p \in \mathrm{P}'} d(p, \mathrm{Q})^2 + \frac{1}{\mathrm{M}} \sum_{q \in \mathrm{Q}} d(q, \mathrm{P}')^2 \tag{1}$$

$$\mathcal{L}^{\text{S}}_{\text{CD}} = \frac{1}{\mathrm{N}} (\sum_{p \in \mathrm{P}'} d(p, \mathrm{Q})^5)^{\frac{1}{5}} + \frac{1}{\mathrm{M}} (\sum_{q \in \mathrm{Q}} d(q, \mathrm{P}')^5)^{\frac{1}{5}}, \tag{2}$$

where N and M are the numbers of the outputs and ground truth points, respectively. Additionally, the Chamfer Distances $\mathcal{L}^E_{\text{CD}}$ between the generated edge points and the ground truth edges are computed.

We also compute the binary cross entropy (BCE) loss between the predicted completion voxels and the ground truth grids as:

$$\mathcal{L}^P_{\text{BCE}} = -\frac{1}{32^3} \sum_c \widehat{p_c} \cdot log(p_c) + (1 - \widehat{p_c}) \cdot log(1 - p_c), \tag{3}$$

where $\widehat{p_c}$ and p_c are the ground truth grid probability and predicted grid probability, respectively. Similarly, $\mathcal{L}^E_{\text{BCE}}$ represents for the edge loss.

Since our final outputs are the offsets to the per-grid centers, we compute the density estimation and the quality of locality [19] to enhance the constraint on the grid cells. The density loss and the locality measurement are given by the mean squared error:

$$\mathcal{L}_d(\delta, \hat{\delta}) = \frac{1}{32^3} \sum_c (\delta_c - \widehat{\delta_c})^2 \tag{4}$$

and

$$\mathcal{L}_o = \sum_c \sum_{p \in \mathrm{P}'} \max(\text{dist}(p, c_o) - \sqrt{3}, 0). \tag{5}$$

c_o is the cell center, δ_c and $\widehat{\delta_c}$ are the predicted grid density and ground truth density, respectively. We also compute $\mathcal{L}^E_d(\delta, \hat{\delta})$ and $\mathcal{L}^E_o$ for the edge reconstruction.

Finally, the overall losses of our network are given by:

$$\begin{aligned} \mathcal{L}_{all} = \lambda_1(\mathcal{L}_{\text{CD}} + \mathcal{L}^E_{\text{CD}}) + \lambda_2 \mathcal{L}^{\text{S}}_{\text{CD}} + \lambda_3(\mathcal{L}^P_{\text{BCE}} + \mathcal{L}^E_{\text{BCE}}) \\ + \lambda_4(\mathcal{L}_d(\delta, \hat{\delta}) + \mathcal{L}^E_d(\delta, \hat{\delta})) + \lambda_5(\mathcal{L}_o + \mathcal{L}^E_o), \end{aligned} \tag{6}$$

where λ_i are weights to balance the different loss terms.

4. Experiments

4.1. Datasets

We first evaluate our method on the widely used benchmarks of 3D point cloud completion, *e.g.* Completion3D [31] and the PCN dataset [46], which are large-scale datasets derived from the ShapeNet dataset. We follow the settings of training, validation and testing splits in the Compeltion3D and PCN dataset for fair comparisons. The incomplete points are obtained by back-projecting 2.5D depth images from a partial view into the 3D space, and the complete points are uniformly sampled from the mesh models. These two datasets contains eight categories: airplane, cabinet, car, chair, lamp, sofa, table and vessel. The results on the Completion3D benchmark are directly obtained from the benchmark [1] and the results of the PCN dataset are shown in the supplementary materials.

To further test the generalization ability on more categories and the capability of reconstructing more complex structures, we create a new dataset consisting of 12 categories from the ShapeNet dataset instead of 8 classes in the Completion3D and PCN datasets. The objects include bag, cap, car, chair, earphone, guitar, lamp, laptop, motorbike, mug, skateboard and table. There are 12,137 training data, 1,870 validation data and 2,874 testing data, respectively. The ground-truth point clouds are obtained by uniformly sampling 2,048 points from 3D meshes. Instead of creating the partial views following the PCN dataset, we explore the other widely used method [14, 28, 27] to randomly select a viewpoint as a center and remove points within a certain radius from complete data to obtain the partial inputs. This is to show the generalization ability and robustness of our algorithm on another type of incompleteness.

We also test on unseen categories of our dataset and the real-world KITTI dataset [7] to verify the robustness of our method.

[1] https://completion3d.stanford.edu/results

Methods	Mean Chamfer Distance per point (10^{-4})								
	Avg	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Vessel
FoldingNet[45]	19.07	12.83	23.01	14.88	25.69	21.79	21.31	20.71	11.51
PCN[46]	18.22	9.79	22.70	12.43	25.14	22.72	20.26	20.27	11.73
PointSetVoting[47]	18.18	6.88	21.18	15.78	22.54	18.78	28.39	19.96	11.16
AtlasNe[9]	17.77	10.36	23.40	13.40	24.16	20.24	20.82	17.52	11.62
TopNet [31]	14.25	7.32	18.77	12.88	19.82	14.60	16.29	14.89	8.82
SoftPoolNet[40]	11.90	4.89	18.86	10.17	15.22	12.34	14.87	11.84	6.48
SA-Net[41]	11.22	5.27	14.45	7.78	13.67	13.53	14.22	11.75	8.84
GRNet[43]	10.64	6.13	16.90	8.27	12.23	10.22	14.93	10.08	5.86
PMP-Net[42]	9.23	3.99	14.70	8.55	10.21	9.27	12.43	8.51	5.77
CRN[36]	9.21	3.38	13.17	8.31	10.62	10.00	12.86	9.16	5.80
SCRN[38]	9.13	**3.35**	12.81	**7.78**	9.88	10.12	12.95	9.77	6.10
VE-PCN	**8.10**	3.83	**12.74**	7.86	**8.66**	**7.24**	**11.47**	**7.88**	**4.75**

Table 1: Quantitative comparison for point cloud completion on eight categories objects of Completion3D benchmark.

Categories	Mean Chamfer Distance per point (10^{-4})								
	PCN [46]	PCN-FC [46]	CDA [19]	TopNet [31]	CRN [36]	GRNet [43]	DPC [48]	MSN [20]	VE-PCN
Bag	11.609	11.722	15.854	13.571	6.253	6.048	4.543	6.118	**3.466**
Cap	9.451	11.089	14.437	13.191	6.709	6.334	3.218	4.918	**3.098**
Car	7.254	7.667	8.344	8.419	4.776	5.704	3.714	8.214	**3.480**
Chair	4.675	5.165	6.446	6.026	3.116	4.286	2.811	2.666	**2.476**
Earphone	10.821	10.437	14.026	11.921	6.117	5.321	4.991	8.309	**3.239**
Guitar	0.996	1.134	1.289	1.403	0.690	1.309	0.869	**0.641**	0.750
Lamp	9.421	10.592	12.225	10.619	5.652	4.443	5.809	5.618	**3.226**
Laptop	3.236	3.464	3.653	3.941	2.534	3.745	1.854	**1.639**	2.197
Motorbike	6.005	6.398	6.706	7.033	3.434	4.237	4.081	4.692	**2.717**
Mug	9.522	10.788	11.877	11.706	6.731	8.022	6.802	5.407	**4.938**
Skateboard	3.504	3.858	4.464	4.321	2.288	2.835	1.804	2.431	**1.780**
Table	5.990	6.789	8.010	6.973	3.805	4.325	3.946	3.199	**2.798**
Average	5.771	6.420	7.577	6.889	3.697	4.261	3.570	3.550	**2.669**

Table 2: Quantitative comparison for point cloud completion on 12 seen categories of our dataset.

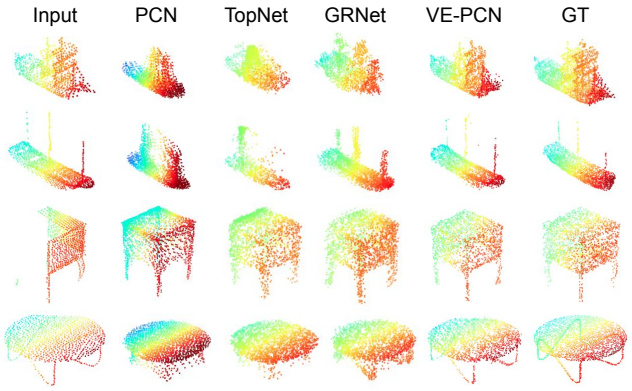

Figure 4: Qualitative results on the Completion3D dataset.

4.2. Implementation Details

We empirically set the weights of the loss terms to $\lambda_1 = 1e4$, $\lambda_2 = 300$, $\lambda_3 = 100$, $\lambda_4 = 1e10$, $\lambda_5 = 0.3$ for the Completion3D dataset and our dataset. $\lambda_2 = \lambda_5 = 0$ for the PCN dataset. We train our model with a learning rate of 0.0007 and a batch size of 32 on one TITAN X GPU.

4.3. Completion3D Dataset

We compare the quantitative results with existing point cloud completion methods in Table 1, where our method VE-PCN achieves the best performance in terms of the average Chamfer Distance across all the categories. Compared with existing state-of-the-art approaches, we achieve better results in six categories, which verifies the effectiveness of our method. The superior performances of object details in Figure 4 prove the superiority of our edge-guiding strategy and voxelization technique.

4.4. Our Dataset

Seen Categories. Qualitative and quantitative results are shown in Figure 5 and Table 2, respectively. We retrain other models on our dataset using their released codes. The number of output points is 2,048, and all the object categories are seen during training. Apart from CD, we adopt Fréchet Point Cloud Distance (FPD) [29] as another evaluation metric. As shown in Figure 5, previous approaches show inferior performances on reconstructing the missing parts or fine-grained object details, while our model syn-

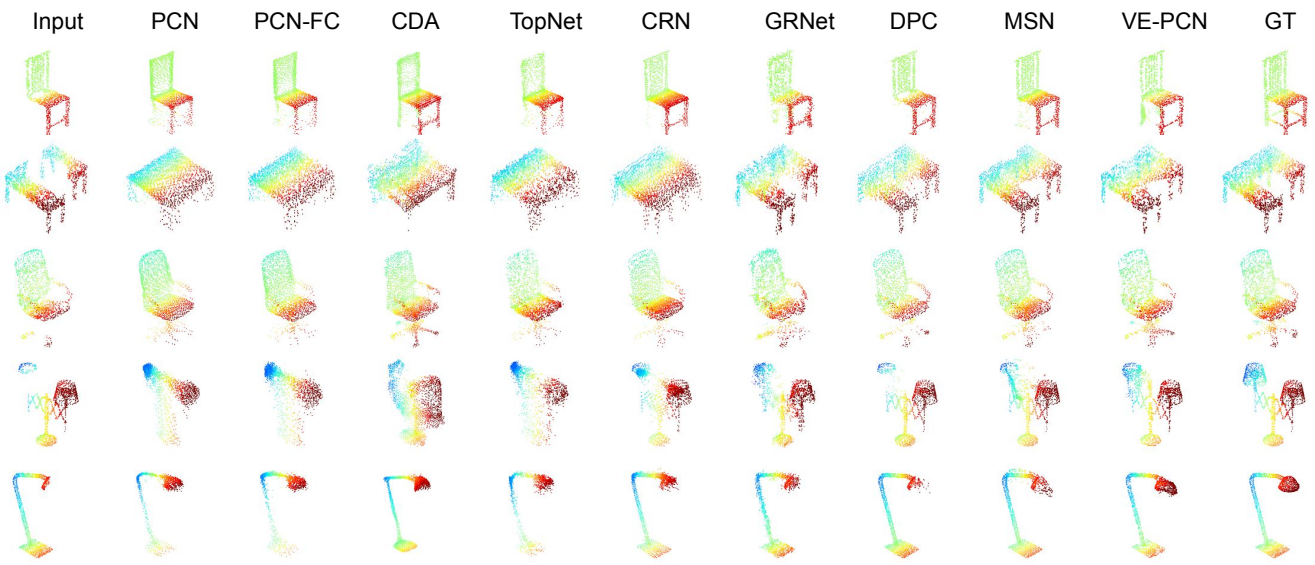

Figure 5: Qualitative comparisons of point cloud completion on seen categories of our dataset.

thesizes more realistic object shapes with accurate surfaces. For example, other methods fail to reconstruct the object details of the lampshade (the last second row). In contrast, our method is able to preserve lamp details in the partial input and concurrently generate details for the missing regions. Other examples such as the thin legs of the chair and table further verify this. The quantitative results shown in Table 2 verify that we achieve superior scores in the coordinates approximation and distribution estimation. We decrease CD error by around 24.8% compared to the second-best result and achieve the best performances on almost all the categories compared to other approaches. Overall, the superior qualitative and quantitative performances verify the effectiveness of our edge generation and voxelization strategies for point cloud completion. As shown in Table 3, we also obtain better results with different resolutions. This indicates that our edge-guiding network and grid representations are more robust to different kinds of completion scenarios. More results are shown in supplementary materials.

Unseen Categories. Table 4 shows the results on unseen categories, which contain airplane, knife, pistol and rocket. We directly use the models trained on the seen categories for testing. Our method achieves lower CD errors compared to other state-of-the-art works and synthesizes high-fidelity object shapes even though our network is not trained on those categories. We obtain 18.4% relative improvement compared to the second-best method GRNet [43], which demonstrates that our method has better generalization.

4.5. KITTI Dataset

We further compare the robustness of our method with other approaches on the real-world KITTI [7] dataset. We directly adopt the models trained on the car category of our dataset for testing since there are no ground truths for

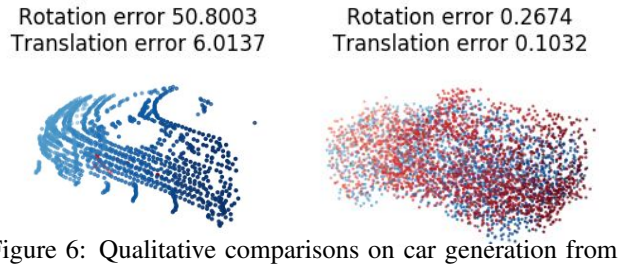

Figure 6: Qualitative comparisons on car generation from the KITTI dataset.

KITTI. We make use of the fidelity distance as the evaluation metric following [46, 43, 48], which measures how well the inputs are preserved in the outputs. Table 5 indicates that our method is capable of reconstructing more realistic cars, although the examples are noisy and severely occluded. The result of our model without edge generation is 0.0279, which is worse than our entire pipeline. This verifies that our edge guidance is beneficial for real-world datasets. We also measure the registration errors between neighboring frames in the same Velodyne sequence following [46]. An example in Figure 6 shows that our method can decrease the rotation and translation errors by completing the points when compared to the rotation and translation errors from raw inputs. More results are in supplementary materials.

4.6. Edge Generation

We compare our voxel-based method to the classical MLP-based method PCN [46] for the point cloud edge generation and show the results in Figure 7. Our edges are cleaner and more accurate compared to the results of PCN, and our output edges cover all the thin structures of an object, while the results from PCN are over-smoothed. This demonstrates that grid representation is desirable for the re-

Resolutions	Mean Chamfer Distance per point (10^{-4})								
	PCN [46]	PCN-FC [46]	CDA [19]	TopNet [31]	CRN [36]	GRNet [43]	DPC [48]	MSN [20]	VE-PCN
2048	5.771	6.420	7.577	6.889	3.697	4.261	3.570	3.550	**2.669**
4096	3.821	4.472	6.721	4.588	2.671	2.956	2.943	3.409	**2.275**
8192	3.264	3.795	6.100	4.319	2.342	2.423	2.920	3.808	**1.880**
16384	2.864	3.251	5.934	3.513	1.953	2.011	2.462	3.588	**1.620**

Table 3: Quantitative comparison for point cloud completion with different resolutions of our dataset.

Categories	Mean Chamfer Distance per point (10^{-4})								
	PCN [46]	PCN-FC [46]	CDA [19]	TopNet [31]	CRN [36]	GRNet [43]	DPC [48]	MSN [20]	VE-PCN
Airplane	26.418	28.518	33.550	17.675	11.047	4.526	10.301	9.906	**3.382**
Knife	2.998	3.110	5.235	3.495	2.015	1.700	3.518	2.604	**1.603**
Pistol	12.168	11.131	27.701	11.038	6.428	**3.482**	6.505	5.524	4.884
Rocket	10.035	10.853	16.029	8.075	4.750	1.999	5.378	4.992	**1.574**
Average	20.764	22.208	27.821	14.44	8.948	3.892	8.690	8.154	**3.176**

Table 4: Quantitative comparison for point cloud completion on unseen categories of our dataset.

Methods	PCN [46]	PCN-FC [46]	CDA [19]	TopNet [31]	CRN [36]	GRNet [43]	DPC [48]	MSN [20]	VE-PCN
Fidelity	0.0436	0.0407	0.0428	0.0438	0.0337	0.0298	0.0347	0.0345	**0.0258**

Table 5: Fidelity evaluations that measure the average distance between each input point to its nearest neighbor in the output (lower is better) on the KITTI dataset.

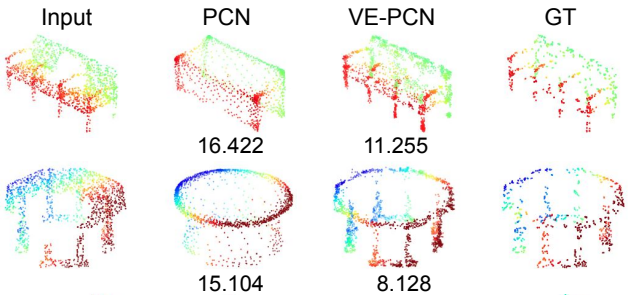

Figure 7: Comparisons of object edge generation with PCN. Bottom values are the CD errors per point (10^{-4}).

construction of thin structures. More results are shown in supplementary materials.

4.7. Ablation Studies

This section explores the effects of different network modules by removing or replacing a specific component. The quantitative results are reported in Table 6. The training of ablation experiments is conducted on our created ShapeNet dataset with an output resolution of 2,048.

MGT represents our multi-grid transformation module. DG represents the differentiable Gridding module in [43]. We replace our MGT with DG for comparison. We further test our model without MGT, where we directly feed the binary voxelizations obtained from raw points to the edge generator and shape completion network. EG is the edge generator and PE represents partial edges directly calculated from [1]. We replace EG with PE when using PE.

As shown in Table 6, the performance drops upon the removal of our MGT (ID: 3 vs 5) or the use of DG [43] (ID: 3 vs 4). This indicates that our multi-scale grid feature learning plays an essential role in edge generation and shape completion. Additionally, we obtain worse results by either removing the edge generator (ID: 1 vs 3) or replacing it with the calculated partial edges (ID: 1 vs 2). This verifies the importance of complete edges hallucinated from our edge generator. More ablation studies on different losses are shown in the supplementary materials.

ID	MGT	DG	EG	PE	CD
1	✓	✗	✓	✗	**2.669**
2	✓	✗	✗	✓	3.586
3	✓	✗	✗	✗	3.600
4	✗	✓	✗	✗	4.768
5	✗	✗	✗	✗	5.792

Table 6: Ablation studies on our dataset. Results are represented by mean CD per point (10^{-4}).

5. Conclusion

We present a voxel-based network for point cloud completion by leveraging the edge generation. To reconstruct the realistic structures with fine-grained details, we propose to generate the object edges and complete point sets with the guidance of the hallucinated edges. Furthermore, we transform the unordered point sets into grid representations to support edge generation and point cloud reconstruction. Our multi-scale grid feature learning further boost the shape completion performance, and we obtain superior performance on different point cloud completion datasets.

Acknowledgments. This work is supported in part by the Singapore MOE Tier 2 grant MOE-T2EP20120-0011, and the National Research Foundation, Prime Ministers Office, Singapore, under its CREATE programme, Singapore-MIT Alliance for Research and Technology (SMART) Future Urban Mobility (FM) IRG.

References

[1] Syeda Mariam Ahmed, Yan Zhi Tan, Chee Meng Chew, Abdullah Al Mamun, and Fook Seng Wong. Edge and corner detection for unorganized 3d point clouds with application to robotic welding. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7350–7355. IEEE, 2018.

[2] Xuelin Chen, Baoquan Chen, and Niloy J Mitra. Unpaired point cloud completion on real scans using adversarial training. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[3] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

[4] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6970–6981, 2020.

[5] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5868–5877, 2017.

[6] 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.

[7] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[9] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry. AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[10] Jiayuan Gu, Wei-Chiu Ma, Sivabalan Manivasagam, Wenyuan Zeng, Zihao Wang, Yuwen Xiong, Hao Su, and Raquel Urtasun. Weakly-supervised 3d shape completion in the wild. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 283–299. Springer, 2020.

[11] Xiaoguang Han, Zhen Li, Haibin Huang, Evangelos Kalogerakis, and Yizhou Yu. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 85–93, 2017.

[12] Zhizhong Han, Xiyang Wang, Yu-Shen Liu, and Matthias Zwicker. Multi-angle point cloud-vae: unsupervised feature learning for 3d point clouds from multiple angles by joint self-reconstruction and half-to-half prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10441–10450. IEEE, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. Pf-net: Point fractal network for 3d point cloud completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7662–7670, 2020.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[16] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9397–9406, 2018.

[17] 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.

[18] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018.

[19] Isaak Lim, Moritz Ibing, and Leif Kobbelt. A convolutional decoder for point clouds using adaptive instance normalization. In *Computer Graphics Forum*, volume 38, pages 99–108. Wiley Online Library, 2019.

[20] 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.

[21] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[22] Yinyu Nie, Yiqun Lin, Xiaoguang Han, Shihui Guo, Jian Chang, Shuguang Cui, and Jian.J Zhang. Skeleton-bridged point completion: From global inference to local adjustment. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16119–16130. Curran Associates, Inc., 2020.

[23] Stanley Osher and Ronald Fedkiw. Signed distance functions. In *Level set methods and dynamic implicit surfaces*, pages 17–22. Springer, 2003.

[24] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgbd data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018.

[25] 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.

[26] 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*, pages 5099–5108, 2017.

[27] Audrey Richard, Ian Cherabier, Martin R Oswald, Marc Pollefeys, and Konrad Schindler. Kaplan: A 3d point descriptor for shape completion. In *2020 International Conference on 3D Vision (3DV)*, pages 101–110. IEEE, 2020.

[28] Muhammad Sarmad, Hyunjoo Jenny Lee, and Young Min Kim. Rl-gan-net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5898–5907, 2019.

[29] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[30] David Stutz and Andreas Geiger. Learning 3d shape completion from laser scan data with weak supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1955–1964, 2018.

[31] Lyne P Tchapmi, Vineet Kosaraju, S. Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[32] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019.

[33] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[34] Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. 3dn: 3d deformation network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1038–1046, 2019.

[35] Weiyue Wang, Qiangui Huang, Suya You, Chao Yang, and Ulrich Neumann. Shape inpainting using 3d generative adversarial network and recurrent convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2298–2306, 2017.

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

[37] Xiaogang Wang, Marcelo H Ang, and Gim Hee Lee. Point cloud completion by learning shape priors. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10719–10726. IEEE, 2020.

[38] 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.

[39] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[40] Yida Wang, David Joseph Tan, Nassir Navab, and Federico Tombari. Softpoolnet: Shape descriptor for point cloud completion and classification. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 70–85. Springer, 2020.

[41] 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*, pages 1939–1948, 2020.

[42] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7443–7452, 2021.

[43] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *European Conference on Computer Vision*, pages 365–381. Springer, 2020.

[44] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018.

[45] 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.

[46] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision*, pages 728–737. IEEE, 2018.

[47] Junming Zhang, Weijia Chen, Yuping Wang, Ram Vasudevan, and Matthew Johnson-Roberson. Point set voting for partial point cloud analysis. *IEEE Robotics and Automation Letters*, 6(2):596–603, 2021.

[48] Wenxiao Zhang, Qingan Yan, and Chunxia Xiao. Detail preserved point cloud completion via separated feature aggregation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 512–528. Springer, 2020.

[49] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.