DEEP POINT CLOUD RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Point cloud obtained from 3D scanning is often sparse, noisy, and irregular. To cope with these issues, recent studies have been separately conducted to densify, denoise, and complete inaccurate point cloud. In this paper, we advocate that jointly solving these tasks leads to significant improvement for *point cloud reconstruction*. To this end, we propose a deep point cloud reconstruction network consisting of two stages: 1) a 3D sparse stacked-hourglass network as for the initial densification and denoising, 2) a refinement via transformers converting the discrete voxels into continuous 3D points. In particular, we further improve the performance of the transformers by a newly proposed module called amplified positional encoding. This module has been designed to differently amplify the magnitude of positional encoding vectors based on the points' distances for adaptive refinements. Extensive experiments demonstrate that our network achieves state-of-the-art performance among the recent studies in the ScanNet, ICL-NUIM, and ShapeNet datasets. Moreover, we underline the ability of our network to generalize toward real-world and unmet scenes.

1 INTRODUCTION

3D scanning devices, such as LiDAR and RGB-D sensors, allows to quickly and accurately reconstruct a scene as a 3D point cloud. This compact 3D representation is commonly used to achieve various tasks in autonomous driving (Yang et al., 2018; Qi et al., 2018; Shi et al., 2020; Lang et al., 2019), robotics (He et al., 2021; 2020; Wang et al., 2019), or 3D mapping (Choe et al., 2021; Choy et al., 2020). However, processing raw point cloud happens to be particularly challenging due to their sparsity, irregularity and sensitivity to noise (Xiang et al., 2019; Guo et al., 2020).

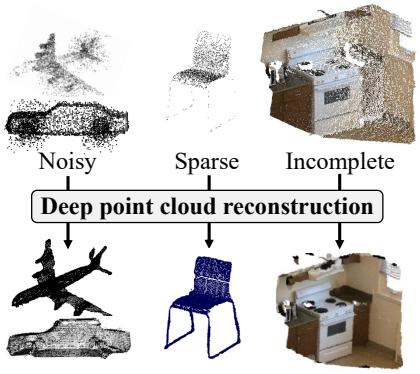


Figure 1: **Point cloud reconstruction.** We propose a novel neural architecture that jointly solves inherent shortcomings in raw point cloud, such as noise, sparsity, and incompleteness.

To address these issues, recent deep learning-based studies have proposed to improve the quality of point cloud. Specifically, prior works that are dedicated to point cloud refinement can be categorized into three distinct subfields: (1) point upsampling (Yu et al., 2018b), (2) point denoising (Rakotosaona et al., 2020), and (3) point completion (Yuan et al., 2018). Each of these tasks has its own benefits and inconveniences. For instance, the point upsampling task usually handles noisy input points but do not explicitly deal with strong outliers. On the other hand, point denoising techniques have been designed to filter noisy data but cannot densify the point cloud. Finally, most point completion methods focus on object-level full 3D completion but are known for their weak generalization to unmet environments or objects. Though there exist differences among the tasks, their common goal is to improve proximity-to-surface in a continuous 3D point representation.

Thus, we believe in the complementary of these three sub-tasks and hypothesize that a joint framework is greatly beneficial to deal with deteriorated and incomplete 3D point cloud. Accordingly, this paper challenges to resolve all these issues as a single task, called *point cloud reconstruction* as depicted in Fig. 1. To the best of our knowledge, this paper is the first attempt to jointly resolve the inherent shortcomings of point cloud obtained from 3D scanning devices: sparsity, noise, irregular-

ity, and outliers. To this end, we propose a *deep point cloud reconstruction network* that consists in two stages: a voxel generation network and a point re-localization network.

In the first stage, the voxel generation network (Sec. 3.1) aims to densify voxels and remove outliers via sparse convolution layer (Choy et al., 2019). Previous point cloud densification strategies Luo & Hu (2021); Xiang et al. (2021); Li et al. (2021) rely on the k-Nearest Neighbors algorithm (kNN) to compute local responses from the raw inaccurate point cloud. However, these strategies lack robustness when dealing with irregular point cloud where each point's cluster size is varying depending upon the local density of points (Mao et al., 2019; Thomas et al., 2019). To cope with this limitation, we propose to first convert the point cloud into a set of sparse voxels such that it can be processed through our sparse 3D stacked hourglass network for coarse-to-fine voxel generation. Processing voxelized data offers multiple advantages (discussed in Sec. 3.1) allowing to robustly densify and denoise the data. However, due to their discrete nature, voxelized representation is not appropriate for a detailed reconstruction of a scene. Therefore, we propose a second stage to convert this initial voxel volume into a refined 3D point cloud.

The second stage network, point re-localization network (Sec. 3.2), consists of transformers (Vaswani et al., 2017). Its goal is to convert the generated voxels into continuous points and to refine the distribution of voxels with respect to their neighbors. We further increase the performance of the transformers by introducing *amplified positional encoding*. Positional encoding has proven to be of primary importance to synthesize sharp images (Mildenhall et al., 2020), similarly, our new positional encoding strategy is useful to infer descriptive and detailed point cloud. Indeed, in the context of voxel-to-point conversion, spatial frequency plays an important role that can be accounted for by our positional encoding. For this purpose, we propose to control the amplitude of positional encoding vectors with respect to the voxel's distance. As a result, amplified positional encoding provides an adaptive point cloud refinement by amplifying high-frequency signals based on the local voxel density. Our contributions are summarized as follows:

- New problem formulation: point cloud reconstruction.
- Novel two-stage architecture for voxel-to-point refinement.
- Unique embedding scheme for point cloud, amplified positional encoding.
- A large number of experiments illustrating the generalization ability of our point cloud reconstruction to real 3D scans.

2 Related Works

Point cloud obtained from 3D scanning devices are known to contain various artifacts (Berger et al., 2017), such as noise, outliers, irregularity, and sparsity. Point cloud reconstruction aims at resolving these aforementioned issues in order to provide fine-grained 3D reconstruction. In this section, we introduce different approaches related to point cloud refinement.

Point Cloud Denoising. Without extra information, such as RGB images, point cloud denoising purely based on the input point distribution is a challenging task (Lee, 2000; Avron et al., 2010). Recent deep learning-based strategies (Rakotosaona et al., 2020; Roveri et al., 2018; Hermosilla et al., 2019; Pistilli et al., 2020; Luo & Hu, 2021) demonstrate promising results. However, despite the large improvement provided by these architectures, these methods do not have the ability to densify the reconstruction.

Point Cloud Completion. A family of solutions called point cloud completion have recently been proposed to alleviate the problem of incomplete reconstruction. One of the pioneer papers (Yuan et al., 2018) proposes to overcome the point incompleteness problem by generating points in both non-observed and visible areas. To improve the completion process, recent approaches (Huang et al., 2020; Zhang et al., 2020; Wen et al., 2021; Xiang et al., 2021) propose to include prior structural information of the objects. While these object-specific point completion techniques can lead to accurate results, they suffer from a lack of versatility since the object class is assumed to be known beforehand. Without such semantic level a priori, the entire completion in occluded surface (*e.g.*, behind objects) is a doubtful task. Thus, in this paper, we focus on a reliable and generic object-agnostic approach that is able to complete visible areas within an arbitrary point cloud and to cope with sparse, noisy, and unordered point cloud.

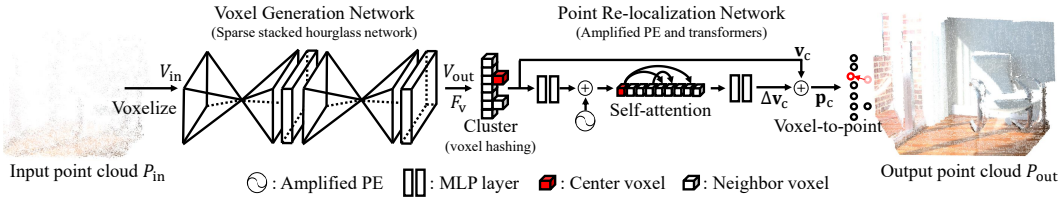


Figure 2: **Voxel-to-point network.** Our pipeline consists in two stages. First the point cloud $\mathcal{P}_{in}$ is converted to a voxelized volume V_{in} which is processed by our voxel generation network to generate a refined and densified 3D volume V_{out} of the same size as the input. This densified volume is finally fed to our point re-location network which jointly denoises and converts V_{out} to a 3D point cloud $\mathcal{P}_{out}$. In this figure, point cloud has been colorized for visualization purpose.

Point Cloud Upsampling. Given a sparse 3D reconstruction, the goal of point cloud upsampling is to densify the point cloud distribution. Conventional methods (Alexa et al., 2003; Huang et al., 2013) rely on point clustering techniques to re-sample points within the 3D space. Recent deep learning-based studies (Li et al., 2021; Yu et al., 2018a; Yifan et al., 2019; Yu et al., 2018b) propose point densification networks under the supervision of dense ground truth points. Though some papers internally conduct point denoising (Yu et al., 2018a; Yifan et al., 2019), most of these works mainly assume that there are no strong outliers among the input points input points, such as flying-point cloud (You et al., 2019).

In contrast to previous studies, our method aims to perform object-agnostic and well generalized point cloud reconstruction that is robust to overall shortcomings in the real-world 3D point cloud. To this end, we introduce our deep point cloud reconstruction network that consists of the voxel generation network and the point re-localization network.

3 Deep Point Cloud Reconstruction

Given a noisy and sparse point cloud $\mathcal{P}_{in}$, point cloud reconstruction aims at generating a dense and accurate set of points $\mathcal{P}_{out}$. For this purpose, we propose a deep point cloud reconstruction network composed of two stages: voxel generation network (Sec. 3.1) and point re-localization network (Sec. 3.2). The first network has been designed to perform a densification and outliers removal in the sparse voxel space. The second network complements and improves this first stage by denoising the data further and by converting voxels back to 3D points. The overall architecture is illustrated in Fig. 2.

3.1 Voxel Generation Network

The roles of the voxel generation network are to simultaneously densify the sparse point reconstruction and to remove outliers. Most previous studies (Li et al., 2021; Yifan et al., 2019; Yu et al., 2018a; Luo & Hu, 2021) attempts to resolve these problems by directly processing a raw point cloud. However, it is known to be a challenging problem due to the unordered nature of the 3D point cloud. To tackle this issue, we propose to convert the 3D point cloud $\mathcal{P}_{in}$ into a sparse voxel representation $\mathcal{V}_{in}$ that can be used as an input for our voxel generation network. From this data, our network generates a refined and densified volume $\mathcal{V}_{out}$. A noticeable advantage of sparse voxel representation is its ability to preserve the neighbor connectivity of point cloud in a uniform and regular 3D grids (Guo et al., 2020) via voxel hashing (Choy et al., 2019; Graham & van der Maaten, 2017). Voxel hashing keeps track of the geometric positions of sparse tensors so that we can trace back to find the voxels' neighbors to apply sparse convolution. Moreover, sparse convolution layers can cover the consistent scope of the local region and hierarchically capture its large receptive fields. Such property is then particularly beneficial to deal with an inaccurate and unordered input point cloud $\mathcal{P}_{in}$.

In order to denoise and densify $\mathcal{V}_{in}$, our voxel generation network consists of a sparse stacked hourglass network as shown in Fig. 3. Our network architecture is close to the previous studies (Newell et al., 2016; Chang & Chen, 2018; Im et al., 2019; Choe et al., 2019) in that it is composed of several hourglass networks for cascaded refinements of images or dense volumes. In the context of this paper, scanning devices can only reconstruct the surface of the structure, thus, the resulting voxelized volume is inherently sparse. Therefore, to process this 3D volume efficiently, our voxel generation

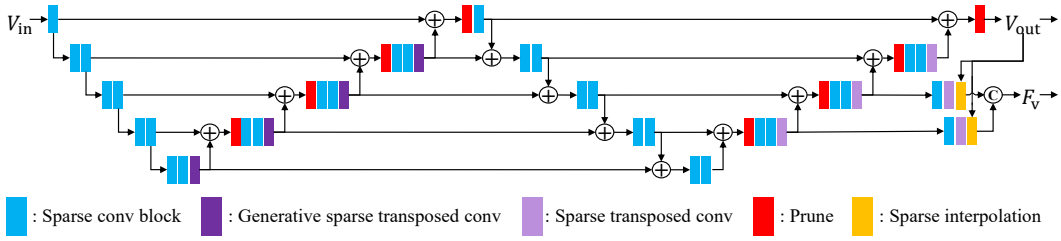


Figure 3: **Voxel generation network.** We design sparse stacked hourglass network that hierarchically and sequentially generates and prunes voxels in a coarse-to-fine manner.

network takes advantage of a sparse 3D convolution strategy inspired by recent works (Choy et al., 2019; Graham & van der Maaten, 2017).

It should be noted that the specificity of sparse convolution operation is particularly desirable for the tasks at hand. First, generative sparse transposed convolution layer (Gwak et al., 2020) has the ability to create new voxels given a pre-defined upsampling rate r, which can be seen as a point upsampling methodology (Yu et al., 2018b). Secondly, voxel pruning is widely used for fine-grained volume rendering (Liu et al., 2020) and efficient computation (Choy et al., 2019). In our context, voxel pruning is closely related to outlier removal. Last, the combined use of generation and pruning enables our network to create an arbitrary number of points unlike previous upsampling papers (Yu et al., 2018b; Li et al., 2021) that generate a static number of points by a factor of r (*i.e.*, upsampling ratio) regardless of the shape of the 3D structures.

As illustrated in Fig. 3, we first compute local voxel features using sparse convolutional blocks in an encoder part. The aggregated voxel features undergo the voxel generation layers and pruning layers in a decoder part. We use two hourglass networks to densify and prune voxels. To train the voxel generation network, we adopt a Binary Cross-Entropy loss (BCE) to classify the status of estimated voxels into an occupied/empty state as:

$$\mathcal{L}_{\text{BCE}}(\mathcal{V})=\frac{1}{\mathcal{N}_{\text{v}}}\sum_{\mathbf{v}\in\mathcal{V}_{\text{out}}} y_{\mathbf{v}}\cdot\log(x_{\mathbf{v}})+(1-y_{\mathbf{v}})\cdot\log(1-x_{\mathbf{v}}), \tag{1}$$

where $x_{\mathbf{v}}$ is an estimated class of a generated voxel $\mathbf{v}$ from the network, and $y_{\mathbf{v}}$ is the ground truth class at the generated voxel $\mathbf{v}$. $\mathcal{N}_{\text{v}}$ is the total number of voxels that we used to compute the BCE loss. If $\mathbf{v}$ is on the 3D surface, $y_{\mathbf{v}}$ becomes true class (*i.e.*, $y_{\mathbf{v}}$=1). Furthermore, following the original loss design (Newell et al., 2016), we compute the BCE Loss $\mathcal{L}_{\text{BCE}}$ for both the final voxel prediction $\mathcal{V}_{\text{out}}$ and the intermediate results $\mathcal{V}_{\text{mid}}$. In total, we compute the voxel loss $\mathcal{L}_{\text{vox}}$ as:

$$\mathcal{L}_{\text{vox}}=\mathcal{L}_{\text{BCE}}(\mathcal{V}_{\text{out}})+0.5\cdot\frac{1}{\mathcal{N}_{\text{v}}}\sum_{\mathcal{V}_{\text{i}}\in\mathcal{V}_{\text{mid}}}\mathcal{L}_{\text{BCE}}(\mathcal{V}_{\text{i}}), \tag{2}$$

where $\mathcal{N}_{\text{v}}$ is the total number of voxels in the intermediate inference volume $\mathcal{V}_{\text{mid}}$ that we used to calculate the BCE loss. Along with the predicted voxels $\mathcal{V}_{\text{out}}$, we extract voxel features $\mathcal{F}_{\mathbf{v}}$ using sparse interpolation as illustrated in Fig. 3. Thanks to the encoder-decoder style of our network, the voxel feature vectors $\mathcal{F}_{\mathbf{v}}$ involve information from wide receptive fields and will be used for voxel-to-point conversion in the following stage.

3.2 Point Re-localization Network

In the previous stage, an input 3D point cloud is converted into sparse voxels and processed by our voxel generation network to increase the voxel density and to remove outliers. This process in itself noticeably improves the quality of the reconstruction, however, the voxelization inevitably leads to information loss due to the discretization process. To provide a fine-grained reconstruction, we propose a point re-localization network that is designed to convert discrete voxels into continuous 3D points. Our network estimates the local 3D shape using neighbor voxels. Thus, to compute the relation between each center voxel and its neighbor, we use transformers (Vaswani et al., 2017).

Let us define a 3D point $\mathbf{p}$ as $\mathbf{p}=[\text{x},\text{y},\text{z}]^\top\in\mathcal{P}_{\text{out}}$ where x,y,z$\in\mathbb{R}$ and a voxel $\mathbf{v}$ as $\mathbf{v}=[\text{i},\text{j},\text{k}]^\top\in\mathcal{V}_{\text{out}}$ where i,j,k$\in\mathbb{Z}$. The 2nd stage network aims to regress offset $\Delta\mathbf{v}=[\Delta\text{i},\Delta\text{j},\Delta\text{k}]^\top$ (Δi,Δj,Δk$\in\mathbb{R}$) where we add the offset $\Delta\mathbf{v}$ to the regular 3D grid (voxel $\mathbf{v}$) for voxel-to-point conversion. Accordingly, the relation of a continuous 3D point $\mathbf{p}$ and a discrete voxel $\mathbf{v}$ is described as $\mathbf{p}=\mathbf{v}+\Delta\mathbf{v}$.

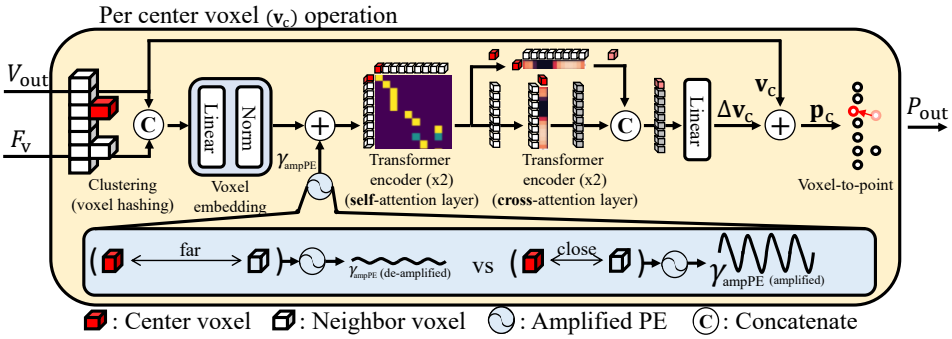


Figure 4: **Point re-localization network.** The 2nd stage network involves transformers for voxel-to-point refinement. In particular, transformers are applied into self/cross-attention layers with our amplified positional encoding to compute the relation between a center voxel and its neighbor voxels.

To perform voxel-to-point refinement, we group a center voxel $\mathbf{v}_c$ and its neighbor voxel set $\{\mathbf{v}_n^1, ..., \mathbf{v}_n^k\}$. This clustering scheme is simply conducted by voxel hashing that has been already formulated by the 1st stage network. To encode their geometric relationship, we utilize two information, a directional vector $\mathbf{d}$ $(=\mathbf{v}_n - \mathbf{v}_c)$ and voxel features $\mathcal{F}_v$ from the voxel generation network. This geometric information is used for the estimation of the offset $\Delta\mathbf{v}$ through self/cross-attention layers as shown in Fig. 4. Note that we hash eight neighbor voxels for each center voxel.

While this voxel-to-point conversion can be directly trained in a supervised manner using the ground-truth point cloud $\mathcal{P}_{GT}$, we propose to provide additional information to the network by our novel amplified positional encoding. Originally, positional encoding has been introduced to help transformers to understand the relative positions between words in a given sentence (Vaswani et al., 2017). When applied to other domains of applications, positional encoding is often used to preserve high-frequency signals which tend to be disregarded by neural networks (Rahaman et al., 2019). For example, positional encoding is proven to be essential to synthesize sharp images from an implicit 3D neural representation (NeRF (Mildenhall et al., 2020)). In our purpose, we need to consider spatial frequency to coherently re-locate a center voxel $\mathbf{v}_c$ with respect to its neighbor voxels $\mathbf{v}_N$.

In contrast to the previous study about natural language process (Vaswani et al., 2017), the 3D spatial relation among point cloud is already known Zhao et al. (2021). Based on geometric priors, we can expect the necessity of spatial high-frequency information from a given voxel group. For example, if a set of clustered voxels is closely located, a high-frequency signal should be strongly considered to re-locate center voxel $\mathbf{v}_c$ in a subtle manner as illustrated in Fig. 4. To this end, our amplified positional encoding changes the amplitude of the positional encoding vector based on the directional vector $\mathbf{d}$. We define our amplified positional encoding vector γ_{ampPE} as:

$$\gamma_{ampPE}(\mathbf{d}, pos) = e^{-\|\mathbf{d}\|_1^1} \cdot \gamma_{PE}(pos), \tag{3}$$

$$\gamma_{PE}(pos) = [..., \sin(pos/10000^{i/C}), \cos(pos/10000^{i/C})], \tag{4}$$

where γ_{ampPE} is an amplified positional encoding vectors and $\|\cdot\|_1^1$ is the L1-norm. Following the conventions (Vaswani et al., 2017), γ_{PE} is 1D positional encoding vectors, pos represents position, i means dimension, and C is channel dimensions of voxel embedding. After we encode γ_{ampPE} into the voxel embedding, we aggregate this embedding through self-attention layers and cross-attention layers to find relationships among the center voxel and its neighbors. Finally, the regressed offset $\Delta\mathbf{v}$ is added to the original location of center voxel $\mathbf{v}_c$ to update its location. To train the second stage network, we calculate a chamfer distance loss $\mathcal{L}_{dist}$ as:

$$\mathcal{L}_{dist} = \frac{1}{\mathcal{P}_{pred}} \sum_{\tilde{\mathbf{p}} \in \mathcal{P}_{pred}} \min_{\mathbf{p} \in \mathcal{P}_{gt}} \|\tilde{\mathbf{p}} - \mathbf{p}\|_2^2 + \frac{1}{\mathcal{P}_{gt}} \sum_{\mathbf{p} \in \mathcal{P}_{gt}} \min_{\tilde{\mathbf{p}} \in \mathcal{P}_{pred}} \|\mathbf{p} - \tilde{\mathbf{p}}\|_2^2 . \tag{5}$$

where $\mathcal{P}_{pred}$ is the predicted point cloud, $\mathcal{P}_{gt}$ indicates ground truth point cloud, $\tilde{\mathbf{p}}$ represents a predicted point, and $\mathbf{p}$ means a ground truth point. Through our two-stage pipelines, our whole network performs point cloud reconstruction. By jointly generating, removing, and refining points, our network can robustly recover accurate point cloud and increase the level of proximity-to-surface.

4 EXPERIMENT

In this section, we describe the implementation details of our network and we present a large series of assessments underlying the effectiveness of our approach. Our point reconstruction network has been solely trained on the ShapeNet-Part (Yi et al., 2016) dataset but tested on other real and synthetic datasets such as ScanNet (Dai et al., 2017) and ICL-NUIM (Handa et al., 2014). These experiments highlight the ability of our solution to generalize well to other scenarios involving various levels of noise and different densities.

4.1 IMPLEMENTATION

We first train our voxel generation network for 10 epochs using the Adam optimizer (Kingma & Ba, 2014) with initial learning of $1e$-3 and a batch size of 4. We decrease the learning rate by half for every 2 epochs. After that, we freeze the voxel generation network and initiate the training of the point re-localization using the inferred voxels $\mathcal{V}_{\text{out}}$ from the voxel generation network. The point re-location network is trained using identical hyper-parameters used for training the first stage network. Further details regarding the training rules are provided in Sec. 4.3.

4.2 BASELINE APPROACHES

We conduct a thorough comparison against previous state-of-the-art techniques: Dis-PU for point cloud upsampling (Li et al., 2021), Score-PD for point cloud denoising (Luo & Hu, 2021), and Snowflake-PC for point cloud completion (Xiang et al., 2021). These three papers are currently positioned at the highest rank for each task and we utilize their official implementation code for training and evaluation. Contrary to these task-specific solutions, our point cloud reconstruction network constitutes an object-agnostic and unified solution to these problems. Our network has been trained with the same data augmentation as competing approaches and its performances have been measured on various datasets: ShapeNet-part (Yi et al., 2016), ScanNet (Dai et al., 2017), and ICL-NUIM (Handa et al., 2014) datasets. For a fair comparison, we measure the quality of the reconstruction under various values of upsampling ratio r to similarly set the number of generated points $\mathcal{P}_{\text{out}}$. Furthermore, to analyze the efficacy of joint refinement by our network, we sequentially combine the use of the denoising network with estimations from the completion network (Snowflake-PC→Score-PD) or the upsampling network (Dis-PU→Score-PD) as in Table 1 and Fig. 5. Since the point completion task also puts weight on the point densification on a visible surface, we also take into account this task.

4.3 DATASET

ShapeNet dataset. We utilize 13 categories of different objects categories from the ShapeNet-Part dataset (Chang et al., 2015; Yi et al., 2016). We extract dense and uniform 10K ground truth point cloud per object by using the Poisson Disk Sampling algorithm (Yuksel, 2015) provided by Open3D (Zhou et al., 2018). Among these ground truth 3D points, we randomly sample 2048 points as an input $\mathcal{P}_{\text{in}}$ for the training networks. For the sake of fairness, we adjust our hyper-parameter l_{vox} to 0.0200 for training our network. This is to produces a comparable number of points than previous studies. We follow a data augmentation scheme proposed by the previous point upsampling study (Li et al., 2021), such as random noise addition and random re-scaling. Additionally, we include random outliers that constitute less that 5 percent of input points. To train and validate the networks, we carefully follow the official train/val/test split provided by (Yi et al., 2016).

ScanNet / ICL-NUIM dataset. To demonstrate the generalization capabilities of our point cloud reconstruction network, we propose to test our method and previous studies in unmet datasets. In particular, we propose to reflect the versatility of our work via the evaluation process proposed in image-based multi-view stereo studies (Yao et al., 2018)[1]. In particular, we choose real-world datasets to show the applicability of the point cloud reconstruction task. ScanNet (Dai et al., 2017) is an indoor dataset that consists of 800 real-world scenes. Each scene provides RGB-D sequences that are captured by 3D sensors. Following the official guideline, we utilize 100 test scenes from which we randomly sample 200 frames for evaluation.

For integrity, we also propose an extensive evaluation on the ICL-NUIM dataset (Handa et al., 2014) which provides synthetic ground truth RGB-D sequences. Since the quality of the ground truth depth

[1] A pioneer study (Yao et al., 2018) trains a deep network in object-scale dataset (DTU (Jensen et al., 2014)), then measures metrics in unmet scenes, Tanks-and-Temples dataset (Knapitsch et al., 2017) without fine-tunes.

ShapeNet-Part dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k Nearest Chamf. (↓)			Percentage (↑) (< 0.5)			Percentage (↑) (< 1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
Snowflake-PC (r=4)	2048	8192	**1.14**	**1.29**	**1.51**	72.28	**42.71**	51.51	92.41	**88.48**	**90.16**
Dis-PU (r=4)	2048	8192	1.56	1.69	1.90	60.71	26.91	36.04	86.67	68.19	75.60
Ours (l_{vox}=0.0200)	2048	8732	1.19	1.33	1.52	**81.02**	40.41	**53.48**	**96.95**	81.23	88.08
Snowflake-PC (r=8)	2048	16384	1.01	1.13	1.31	70.89	60.66	64.17	92.16	93.56	92.76
Snowflake-PC (r=8) → Score-PD	2048	16384	1.25	1.34	1.49	61.11	42.98	49.74	91.73	81.01	85.83
Dis-PU (r=8)	2048	16384	1.02	1.14	1.32	80.84	50.36	60.74	95.69	82.21	88.05
Dis-PU (r=8) → Score-PD	2048	16384	1.29	1.39	1.56	71.79	43.88	53.57	94.68	75.41	83.55
Ours (l_{vox}=0.0150)	2048	15863	**0.90**	**0.94**	**1.12**	**85.77**	**69.01**	**78.14**	**97.11**	**94.98**	**96.16**
ScanNet dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k Nearest Chamf. (↓)			Percentage (↑) (< 0.5)			Percentage (↑) (< 1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
Snowflake-PC (r=8)	4098	32768	3.64	3.98	4.43	19.50	8.06	10.53	45.32	40.23	41.19
Snowflake-PC (r=8) → Score-PD	4098	32768	3.61	3.84	4.19	24.58	10.33	13.82	55.21	38.18	44.47
Dis-PU (r=8)	4096	32768	3.20	3.49	3.92	21.52	9.323	11.94	48.30	36.10	39.72
Dis-PU (r=8) → Score-PD	4098	32768	3.01	3.29	3.71	30.24	14.07	18.04	63.36	42.15	49.68
Ours (l_{vox}=0.0100)	4096	27740	**1.51**	**1.64**	**1.87**	**70.49**	**40.59**	**51.29**	**93.63**	**73.24**	**82.02**
Ours (l_{vox}=0.0085)	4096	41022	1.73	1.86	2.05	63.67	38.55	47.77	91.03	70.67	79.36
Snowflake-PC (r=16)	4098	65536	3.79	4.08	4.48	16.91	11.96	13.12	40.45	53.70	45.45
Snowflake-PC (r=16) → Score-PD	4098	65536	3.71	3.93	4.23	21.93	15.05	17.33	51.57	47.38	49.12
Dis-PU (r=16)	4096	65536	3.68	3.94	4.32	16.81	13.48	13.90	37.89	44.60	40.04
Dis-PU (r=16) → Score-PD	4098	65536	2.86	3.10	3.44	26.06	21.04	22.36	55.73	53.56	54.04
Ours (l_{vox}=0.0075)	4096	55188	**1.91**	**2.04**	**2.24**	**57.88**	**36.83**	**44.72**	**88.06**	**68.34**	**76.72**
Ours (l_{vox}=0.0065)	4096	76182	2.17	2.32	2.32	51.55	34.78	41.16	84.47	65.28	73.37
Ours (l_{vox}=0.0050)	4096	116266	2.86	3.02	3.26	40.42	30.24	34.08	76.37	58.12	65.61
ICL-NUIM dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k Nearest Chamf. (↓)			Percentage (↑) (< 0.5)			Percentage (↑) (< 1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
Snowflake-PC (r=8)	4098	32768	3.32	3.59	3.97	21.61	12.28	13.87	44.32	50.18	45.98
Snowflake-PC (r=8) → Score-PD	4098	32768	3.21	3.40	3.69	28.84	15.54	18.89	56.27	48.97	51.88
Dis-PU (r=8)	4096	32768	3.25	3.49	3.86	21.00	13.05	14.16	43.31	42.44	41.46
Dis-PU (r=8) → Score-PD	4098	32768	2.85	3.09	3.45	32.34	20.03	22.75	61.38	51.12	54.96
Ours (l_{vox}=0.0100)	4096	21395	**1.50**	**1.61**	**1.79**	**75.81**	**48.13**	**58.44**	**92.99**	**74.57**	**82.40**
Ours (l_{vox}=0.0085)	4096	30986	1.70	1.81	2.00	71.65	46.43	55.74	91.01	72.36	80.12
Snowflake-PC (r=16)	4098	65536	3.41	3.63	3.95	18.69	17.53	16.65	38.26	63.02	47.20
Snowflake-PC (r=16) → Score-PD	4098	65536	3.18	3.34	3.59	24.46	20.88	21.53	50.98	57.43	53.84
Dis-PU (r=16)	4096	65536	3.83	4.03	4.33	17.78	18.19	16.83	35.45	49.93	41.03
Dis-PU (r=16) → Score-PD	4098	65536	2.80	3.00	3.29	27.46	27.11	26.18	53.15	60.50	56.26
Ours (l_{vox}=0.0075)	4096	41322	**1.87**	**1.99**	**2.18**	**67.78**	**44.97**	**53.36**	**89.09**	**70.52**	**78.141**
Ours (l_{vox}=0.0065)	4096	56902	2.14	2.26	2.45	63.40	42.85	80.26	86.84	67.78	75.41
Ours (l_{vox}=0.0050)	4096	99321	2.78	2.92	3.10	54.11	38.09	43.55	81.76	61.88	69.45

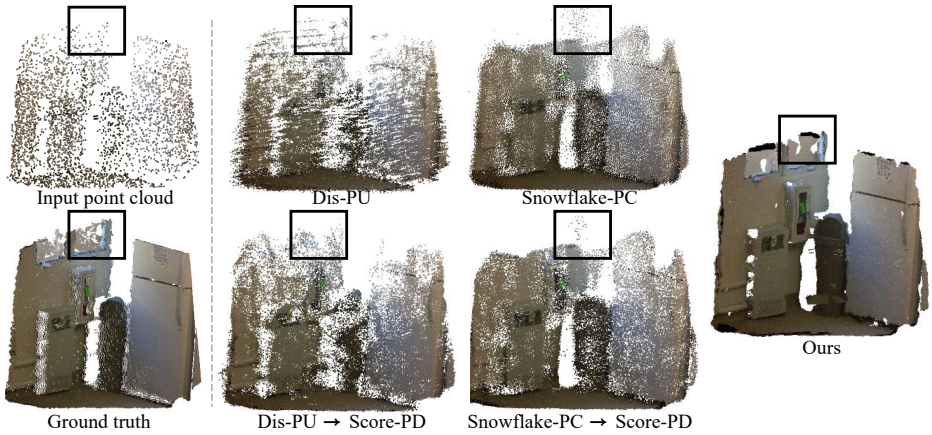
Table 1: **Quantitative results in ShapeNet-Part, ScanNet, and ICL-NUIM datasets.** We measure the quality of point reconstruction from ours, Snowflake-PC (Xiang et al., 2021), Score-PD (Luo & Hu, 2021) and Dis-PU (Li et al., 2021) using typical criteria: chamfer distance (Fan et al., 2017) and percentage metric that consists of accuracy (Acc.) and completeness (Comp.) and f-score (Jensen et al., 2014). For fair comparison, we test various conditions by changing the number of input point cloud $\mathcal{P}_{\text{in}}$, the upsample ratio r, and voxel size l_{vox}. Note that identical colors mean the results from the same pre-trained weights for each method.

maps in ScanNet dataset are measured from low-quality 3D sensors, we utilize ICL-NUIM dataset for the precise quality comparison as in Table 1. Within ICL-NUIM dataset, we use four scenes (livingroom1, livingroom2, office1, office2), and totally sample 200 RGB-D pairs for evaluation. Note that we apply identical conditions as used in ScanNet.

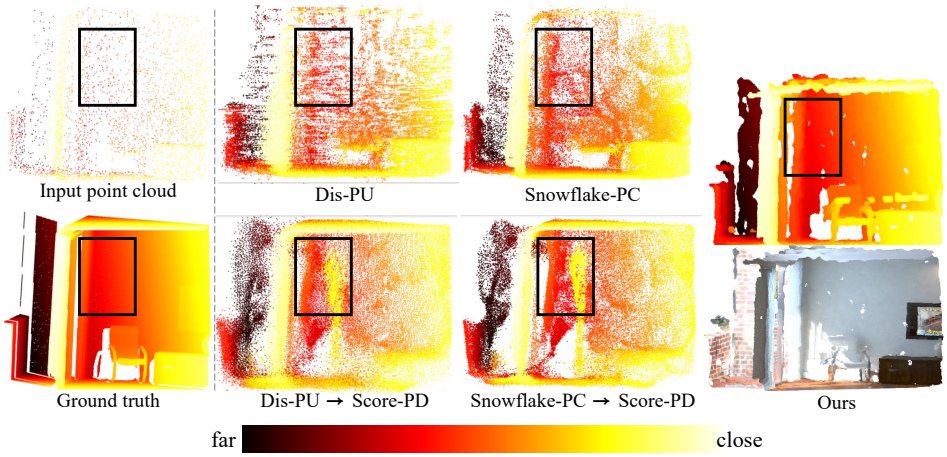
4.4 Comparison

For metric computation, we Chamfer distance and percentage metrics (*e.g.*, accuracy, completeness, and f-score) (Yao et al., 2018; Jensen et al., 2014), as shown in Table 1. In particular, we extend to measure the k-Nearest Chamfer distance that can much efficiently compute point-wise distance than Earth Mover's distance and can consider more nearest neighbor points than the original Chamfer distance (*i.e.*, k=1). Based on these criteria, we calculate the metrics of the whole methods.

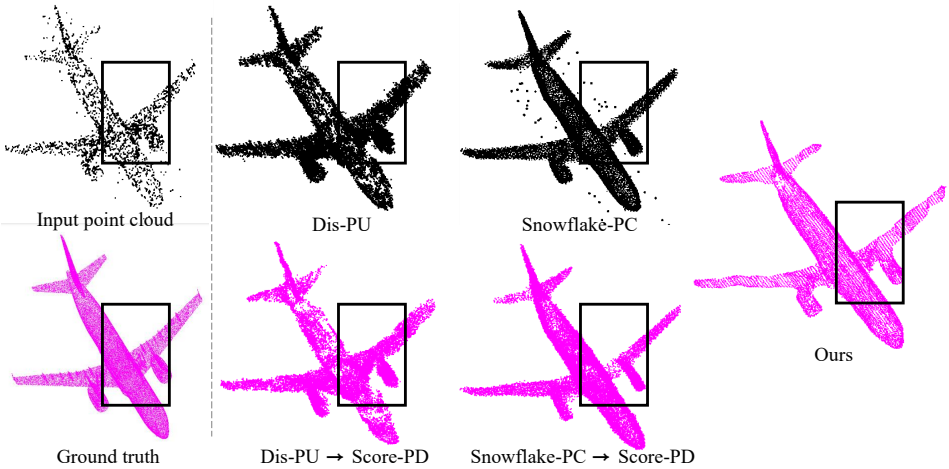
As shown in Table 1, our network achieves state-of-the-art performance compared to the previous studies (Li et al., 2021; Luo & Hu, 2021; Xiang et al., 2021). Even in the various conditions of number of points $\mathcal{P}_{\text{in}}$ with different upsampling ratio r for the previous studies, still our method consistently outperforms in a view of point reconstruction. Moreover, the performance gap increases when we conduct generalization test in ScanNet dataset and ICL-NUIM dataset. Mainly, we deduce the proposed sparse stacked hourglass networks well function in general cases as an initial voxel generation. Sparse convolution layer inherently has consistent scope of 3D space via convolutional kernels and our voxel generation network can hierarchically generate and prune the voxels in a coarse-to-fine manner. As shown in Fig. 5, qualitative results constantly support the



(a) Point cloud reconstruction results in ScanNet dataset.



(b) Point cloud reconstruction results in ICL-NUIM dataset.



(c) Point cloud reconstruction results in ShapeNet-Part dataset.

Figure 5: **Illustration of point cloud reconstruction results from ours and previous studies.** Point cloud has been colorized for visualization purpose.

quality gap between ours and competitors. In particular, we measure the quality of our network using the identical pre-trained weights while previous studies are re-trained because of pre-defined hyper-parameter, *i.e.*, upsampling ratio r. For clearance in Table 1, we apply the same color for the performance that are obtained from the identical network weights.

Type of positional encoding				Performance	
No PE	γ_{PE}	PE*	γ_{ampPE}	Chamf.	f-score
✓				1.231	52.011
	✓			1.224	52.434
		✓		1.226	52.954
			✓	1.195	55.488

(a) Ablation study for amplified positional encoding.

preserve (✓)			Performance	
S-Hourglass	Self-attn	Cross-attn	Chamf.	f-score
✓			1.42	49.56
✓	✓		1.209	53.374
✓		✓	1.223	52.692
✓	✓	✓	1.195	55.488

(b) Ablation study for point re-localization.

Table 2: **Ablation study of our voxel-to-point network.** In (a), PE* indicates positional encoding from NeRF Mildenhall et al. (2020). In (b), S-hourglass represents the voxel generation network. Self-attn and Cross-attn mean self-attention layers and cross-attention layers, respectively. The evaluation is conduced in ShapeNet-Part dataset under l_{vox}=0.02, $\mathcal{P}_{\text{in}}$=2048.

Moreover, when we sequentially apply point densification and point denoising, performance are not constantly improved in Table 1. In qualitative results (Fig. 5), point denoising effectively remove the remaining outliers from results of Snowflake-PC. But, it also erase part of the airplane's wing. In contrast, our voxel generation network conduct both voxel pruning and generation so that our network is more robust than the combined usage of the previous studies.

Based on the evaluation in various datasets, our method demonstrates high-quality results with generalized performance in point cloud reconstruction. Such strength derives from our two-stage pipeline that consists of novel sparse voxel based reconstruction (voxel generation network) and voxel-to-point refinement (point re-localization network).

4.5 Ablation Study

In this section, we propose to evaluate the contribution of each proposed component (amplified positional encoding and point re-localization network), through an extensive ablation study in Table 2.

In Table 2-(a), we validate our amplified positional encoding in comparison with different styles of positional encodings, γ_{PE} (Vaswani et al., 2017) and PE* (Mildenhall et al., 2020). Also, we include an additional experiment that does not use any positional encodings (*i.e.*, No PE in Table 2-(a)). We measure the accuracy by calculating the Chamfer distance and f-score. It demonstrates that our method outperforms the previous embedding schemes. We deduce that by using the geometric position priors of point cloud, we can intentionally magnify or suppress the position encoding vectors that are usually beneficial for the control of high-frequency embedding in 3D spatial domain.

In Table 2-(b), we conduct experiments on different composition of point re-localization network. Our network involves sparse stacked hourglass network (*i.e.*, S-Hourglass), self-attention layers (*i.e.*, Self-attn), and cross-attention layers (*i.e.*, Cross-attn). By intentionally omitting certain modules in the point re-localization network, we measure the quality of point reconstruction using the Chamfer distance and f-scores. Despite reasonable results from the voxel generation network (*i.e.* S-Hourglass), it turns out that self- and cross-attention layers further increase the accuracy of of the point cloud reconstruction. In particular, the combined usage of self- and cross-layers outperforms other strategies. To ensure a fair comparison, the self-attention only method and cross-attention solely network adopts more transformer layers to balance with our final network design.

So far, we demonstrate the effectiveness of our network design through a fair comparison with previous studies and an extensive ablation studies. Our network aims at point cloud reconstruction that can be applied in general cases to achieve high-quality point refinement.

5 Conclusion

Contrary to previous approaches – individually addressing the problems of densification and denoising, we propose an elegant two-stage pipeline to jointly resolves these tasks. As a result of this unified framework, the quality of sparse, noisy and irregular point cloud can be drastically improved. This claim has been validated through a large series of experiments underlying the relevance and efficiency of this joint refinement strategy. Specifically, qualitative and quantitative results demonstrate significant improvements in terms of reconstruction's accuracy and generalization when compared with prior techniques. Our point cloud reconstruction strategy paves the way towards more flexible, multi-tasks and effective architectures to refine point cloud's quality.

REFERENCES

Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on visualization and computer graphics*, 9(1):3–15, 2003.

Haim Avron, Andrei Sharf, Chen Greif, and Daniel Cohen-Or. l1-sparse reconstruction of sharp point set surfaces. *ACM Trans. Graph.*, 2010.

Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction from point clouds. In *Computer Graphics Forum*. Wiley Online Library, 2017.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

Jaesung Choe, Kyungdon Joo, Francois Rameau, Gyumin Shim, and In So Kweon. Segment2regress: Monocular 3d vehicle localization in two stages. In *Robotics: Science and Systems*, 2019.

Jaesung Choe, Kyungdon Joo, Tooba Imtiaz, and In So Kweon. Volumetric propagation network: Stereo-lidar fusion for long-range depth estimation. *IEEE Robotics and Automation Letters*, 6(3): 4672–4679, 2021.

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5828–5839, 2017.

Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017.

Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

JunYoung Gwak, Christopher Choy, and Silvio Savarese. Generative sparse detection networks for 3d single-shot object detection. In *Eur. Conf. Comput. Vis.*, 2020.

Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *IEEE Int. Conf. on Robotics and Automation.*, 2014.

Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021.

Pedro Hermosilla, Tobias Ritschel, and Timo Ropinski. Total denoising: Unsupervised learning of 3d point cloud cleaning. In *Int. Conf. Comput. Vis.*, 2019.

Hui Huang, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri Ascher, and Hao Zhang. Edge-aware point set resampling. *ACM Trans. Graph.*, 2013.

Zitian Huang, Yikuan Yu, Jiawen Xu, Feng Ni, and Xinyi Le. Pf-net: Point fractal network for 3d point cloud completion. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020.

Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. Dpsnet: End-to-end deep plane sweep stereo. In *Int. Conf. Learn. Represent.*, 2019.

Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 2017.

Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12697–12705, 2019.

In-Kwon Lee. Curve reconstruction from unorganized points. *Computer aided geometric design*, 17(2):161–177, 2000.

Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu. Point cloud upsampling via disentangled refinement. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021.

Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Adv. Neural Inform. Process. Syst.*, 2020.

Shitong Luo and Wei Hu. Score-based point cloud denoising. In *Int. Conf. Comput. Vis.*, 2021.

Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Int. Conf. Comput. Vis.*, 2019.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Eur. Conf. Comput. Vis.*, 2020.

Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Eur. Conf. Comput. Vis.*, 2016.

Francesca Pistilli, Giulia Fracastoro, Diego Valsesia, and Enrico Magli. Learning graph-convolutional representations for point cloud denoising. In *Eur. Conf. Comput. Vis.*, 2020.

Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *Int. Conf. on Machine Learning.*, 2019.

Marie-Julie Rakotosaona, Vittorio La Barbera, Paul Guerrero, Niloy J Mitra, and Maks Ovsjanikov. Pointcleannet: Learning to denoise and remove outliers from dense point clouds. In *Computer Graphics Forum*. Wiley Online Library, 2020.

Riccardo Roveri, A Cengiz Öztireli, Ioana Pandele, and Markus Gross. Pointpronets: Consolidation of point clouds with convolutional neural networks. In *Computer Graphics Forum*. Wiley Online Library, 2018.

Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10529–10538, 2020.

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 *Int. Conf. Comput. Vis.*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Adv. Neural Inform. Process. Syst.*, 2017.

Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3343–3352, 2019.

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 *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021.

Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

Peng Xiang, Xin Wen, Yu-Shen Liu, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Zhizhong Han. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In *Int. Conf. Comput. Vis.*, 2021.

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*, pp. 7652–7660, 2018.

Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Eur. Conf. Comput. Vis.*, 2018.

Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung. Patch-based progressive 3d point set upsampling. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. In *Int. Conf. Learn. Represent.*, 2019.

Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Ec-net: an edge-aware point set consolidation network. In *Eur. Conf. Comput. Vis.*, 2018a.

Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018b.

Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, 2018.

Cem Yuksel. Sample elimination for generating poisson disk sample sets. In *Computer Graphics Forum*, 2015.

Wenxiao Zhang, Qingan Yan, and Chunxia Xiao. Detail preserved point cloud completion via separated feature aggregation. In *Eur. Conf. Comput. Vis.*, 2020.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. In *Int. Conf. Comput. Vis.*, 2021.

Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018.