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 3D points. In particular, we further improve the performance of transformer 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 ShapeNetPart datasets. Moreover, we underline the ability of our network to generalize toward real-world and unmet scenes. Implementation will be released after the review.

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., 2021b; 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).

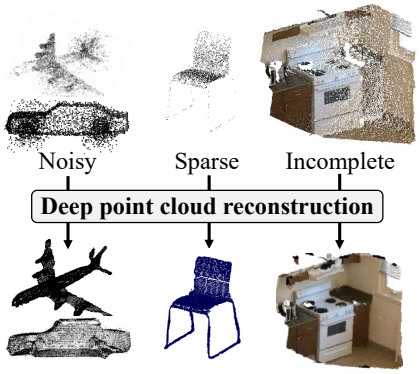


Figure 1: **Point cloud reconstruction.** We propose a novel neural architecture that jointly solves inherent shortcomings in raw point cloud, such as noise, sparsity, and incompleteness.

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, point completion methods mainly focus on object-level full 3D completion but are known for their weak generalization to unmet environments or unknown category of objects. Though there exist differences among the tasks, their common goal is to improve proximity-to-surface in the 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). Rather than to use k-Nearest Neighbor that is sensitive to points' density (Mao et al., 2019), we utilize voxel hashing with sparse convolution layer to understand absolute-scale 3D structures for densification and denoising. Despite its success, voxelization inevitably leads to information loss due to the discretization process. To provide a fine-grained reconstruction, in the 2nd stage, we propose a point re-localization network that converts discrete voxels into 3D points using transformer. Additionally, we increase the performance of transformer by the *amplified positional encoding*. In our analysis, spatial frequency plays an important role in the context of voxel-to-point conversion. Our new positional encoding strategy is useful to infer descriptive and detailed point cloud by simply changing the amplitude of the encoding vector. Our contributions can be 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.

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).

Surface Reconstruction. Point cloud meshing is a common solution for surface reconstruction (Hoppe et al., 1992; Curless & Levoy, 1996). For instance, this process can be achieved by converting point cloud to an implicit surface representation via signed distance function (SDF) and to apply Marching Cubes (Lorensen & Cline, 1987) or Dual Contouring (Ju et al., 2002) for meshing (*e.g.*, Poisson surface reconstruction (Kazhdan et al., 2006)). This strategies is known to be effec-

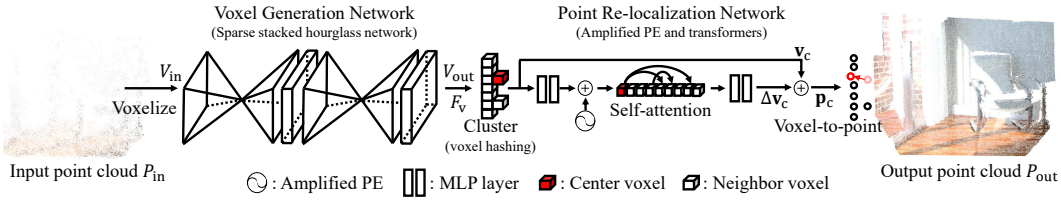


Figure 2: **Two-stage architecture for point cloud reconstruction.** In this figure, point cloud has been colorized for visualization purpose and we set the identical size of visualized points in $\mathcal{P}_{in}$ and $\mathcal{P}_{out}$ for fair comparison.

tive under favorable conditions but can hardly handle sparse and non-uniform points' distribution. Moreover, such technique tends to lead to over-smoothed and undetailed reconstruction (Berger et al., 2017). Similar limitations can be observed with recent deep learning-based SDF representation (Park et al., 2019). Many other solution have been developed for the problem of surface reconstruction, such as, Delaunay triangulation (Rakotosaona et al., 2021; Boissonnat, 1984; Kolluri et al., 2004; Boissonnat & Oudot, 2005; Gopi et al., 2000), alpha shapes (Edelsbrunner & Mücke, 1994), or ball pivoting (Bernardini et al., 1999). Despite their compelling performances, these approaches require uniform, dense, and accurate inputs since they usually preserve the original distribution of a given point cloud. This assumption still holds for the recent studies (Sharp & Ovsjanikov, 2020; Rakotosaona et al., 2021).

Compared to surface reconstruction, point cloud reconstruction can be viewed as a prior-process to overcome natural artifacts: sparsity, noise, irregularity. Furthermore, in contrast to previous studies about point cloud reconstruction, 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 to generate 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

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 $\mathcal{V}_{in}$ through 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 tracking of the geometric positions of sparse tensors so that we can find the voxels' neighbors to apply sparse convolution. Moreover, sparse convolution layer can cover the consistent and absolute-scale 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., 2016b; 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

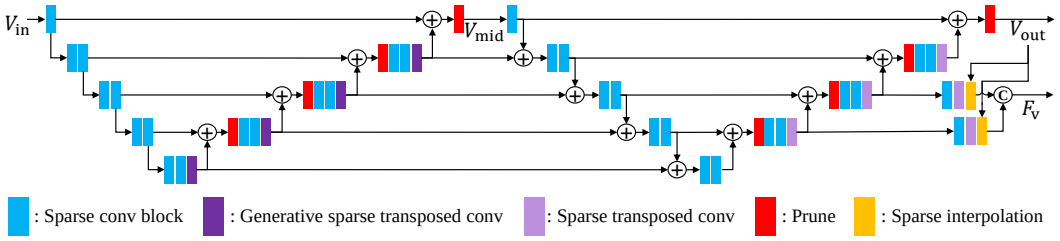


Figure 3: **Voxel generation network.** We design sparse stacked hourglass network that hierarchically and sequentially generates and prunes voxels in a coarse-to-fine manner.

volume is inherently sparse. Therefore, to process this 3D volume efficiently, our voxel generation 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 joint use of generation and pruning enables our network to create an arbitrary number of points.

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:

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

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., 2016b), 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.

3.2 Point Re-localization Network

Despite the denoising and densification offered by the 1st stage network, voxelization inevitably leads to information loss. Thus, in this section, we propose a point re-localization network that converts the discrete voxels $\mathcal{V}_{\text{out}} \in \mathbb{R}^{N \times 3}$ into 3D points $\mathcal{P}_{\text{out}} \in \mathbb{R}^{N \times 3}$. This conversion requires an understanding of local geometry to describe the 3D surface as a group of point sets. Inspired by (Zhao et al., 2021), we utilize self-attention as a point set operator.

Let us describe the detailed process in the point re-localization network, which is illustrated in Fig. 4. Given output voxels $\mathcal{V}_{\text{out}}$, we collect the $K(=8)$ closest voxels to each voxel $\mathbf{v}_i \in \mathbb{R}^{1 \times 3}$ using the hash table that we used in the 1st stage network. Then, we obtain a voxel set $\mathcal{V}_i = \{\mathbf{v}_k\}_{k=1}^{K}$ that is referenced to the query voxel $\mathbf{v}_i$. Similarly, we cluster a set of voxel features $\mathcal{F}_i = \{\mathbf{f}_k\}_{k=1}^{K}$ that correspond to $\mathcal{P}_i$. As in Fig. 4, the 2nd stage network regresses the location of a query point $\mathbf{p}_i$ (red voxel) using the geometric relation between the query voxel $\mathbf{v}_i$ and its neighbor $\mathcal{V}_i$ (white voxels) through self/cross attention layers.

While this formulation is close to PointTransformer (Zhao et al., 2021), our point re-localization network has three distinctive differences. First, PointTransformer utilizes k-Nearest Neighbors to

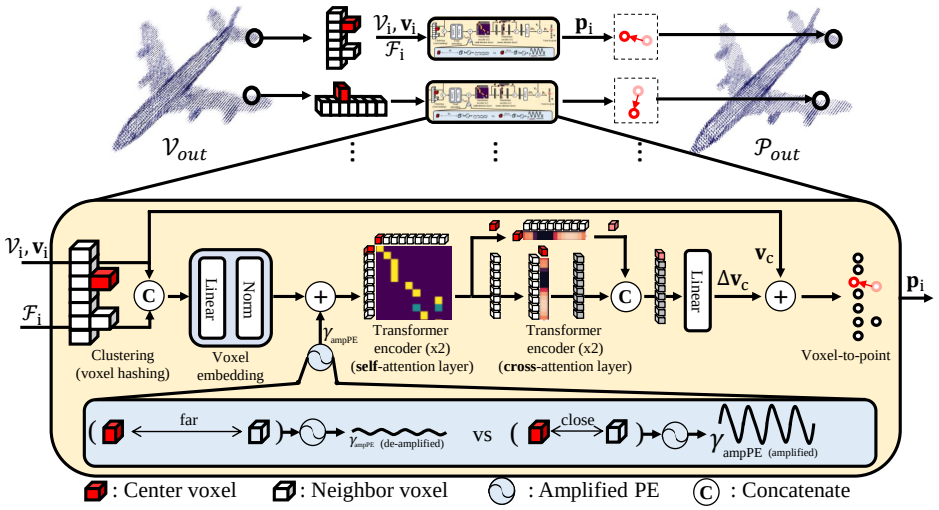


Figure 4: **Point re-localization network.** The 2nd stage network involves transformers for voxel-to-point refinement. In particular, transformers are applied into self/cross-attention layers with our amplified positional encoding to compute the relation between a center voxel and its neighbor voxels.

group 3D points but we re-use the voxel hashing that covers absolute-scale local geometry. Second, our network consists of both self-attention and cross-attention. In our analysis, it is better to use two different attention layer than to solely adopt self-attention. Third, we design our own positional encoding system that controls the necessity of high-frequency signal by changing the amplitude of the encoding vector, called *amplified positional encoding*.

Originally, positional encoding has been introduced to help transformer to understand the relative positions between tokens (*e.g.*, word embeddings) 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 query voxel $\mathbf{v}_i$ by using its neighbor set $\mathcal{V}_i$. Our intuition is simple. Since positional encodings are mainly effective for the high-frequency embedding. As in Fig. 4, we increase the amplitude of positional encoding when we need to move a query voxel by a small amount of distance (*i.e.*, high-frequency in 3D spatial domain). If a query voxel is far away from its neighbor, we decrease the amplitude of the positional encoding. Based on the relation between high frequency signals and our point re-location scheme, we define our amplified positional encoding γ_{ampPE} as:

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

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

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 $\mathcal{L}_{\text{dist}}$ as:

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

where $\mathcal{P}_{\text{pred}}$ is the predicted point cloud, $\mathcal{P}_{\text{gt}}$ indicates ground truth point cloud, $\tilde{\mathbf{p}}$ represents a predicted point, and $\mathbf{p}$ means a ground truth point. $\mathcal{N}_{\mathcal{P}_{\text{pred}}}$ and $\mathcal{N}_{\mathcal{P}_{\text{gt}}}$ are the number of prediction and ground truth point cloud, respectively. 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.

ShapeNet-Part dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k-Chamf. (↓)			Percentage (↑) (< 0.5)			Percentage (↑) (< 1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
PC (r=4)	2048	8192	**1.14**	**1.29**	**1.51**	72.28	**42.71**	51.51	92.41	**88.48**	**90.16**
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
PC (r=4) → PU (r=4) → PD	2048	32987	1.14	1.48	1.60	56.55	49.72	52.43	87.13	82.73	84.67
PU (r=4) → PC (r=4) → PD	2048	32723	1.51	1.59	1.71	56.63	59.90	57.63	83.13	89.47	85.97
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-Chamf. (↓)			Percentage (↑) (< 0.5)			Percentage (↑) (< 1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
PC (r=16)	4098	65536	3.79	4.08	4.48	16.91	11.96	13.12	40.45	53.70	45.45
PC (r=4) → PU (r=4) → PD	4096	64427	3.69	3.91	4.23	23.05	12.59	15.31	49.75	42.24	44.84
PU (r=16)	4096	65536	3.68	3.94	4.32	16.81	13.48	13.90	37.89	44.60	40.04
PU (r=4) → PC (r=4) → PD	4096	64578	2.66	2.86	3.14	22.89	23.30	21.86	49.64	64.11	55.40
PU (r=4) → PD → PC (r=4)	4096	63118	3.75	3.99	4.33	15.99	15.43	14.50	36.60	54.70	43.10
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**
ICL-NUIM dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k-Chamf. (↓)			Percentage (↑) (< 0.5)			Percentage (↑) (< 1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
PC (r=16)	4098	65536	3.41	3.63	3.95	18.69	17.53	16.65	38.26	63.02	47.20
PC (r=4) → PU (r=4) → PD	4096	64451	3.61	3.81	4.11	21.62	11.41	14.05	45.34	42.57	43.32
PU (r=16)	4096	65536	3.83	4.03	4.33	17.78	18.19	16.83	35.45	49.93	41.03
PU (r=4) → PC (r=4) → PD	4096	64872	3.61	3.94	4.42	27.55	8.64	12.42	54.42	33.58	40.60
PU (r=4) → PD → PC (r=4)	4096	63521	4.00	4.21	4.53	10.09	11.64	9.51	22.70	48.52	30.25
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 of point cloud reconstruction.** Note that PC, PU, PD represent point completion (Xiang et al., 2021), point upsampling (Li et al., 2021), and point denoising (Luo & Hu, 2021), respectively. Please refer to the appendix for more results.

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

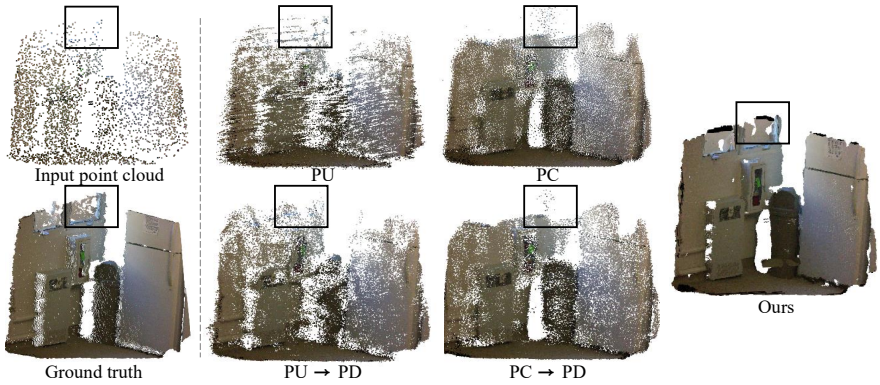
We first train our voxel generation network for 10 epochs using the Adam optimizer (Kingma & Ba, 2014) with initial learning of $1e$-3 and a batch size of 4. We decrease the learning rate by half for every 2 epochs. After that, we freeze the voxel generation network and initiate the training of the point re-localization using the inferred voxels $\mathcal{V}_{\text{out}}$ from the voxel generation network. The point relocation 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

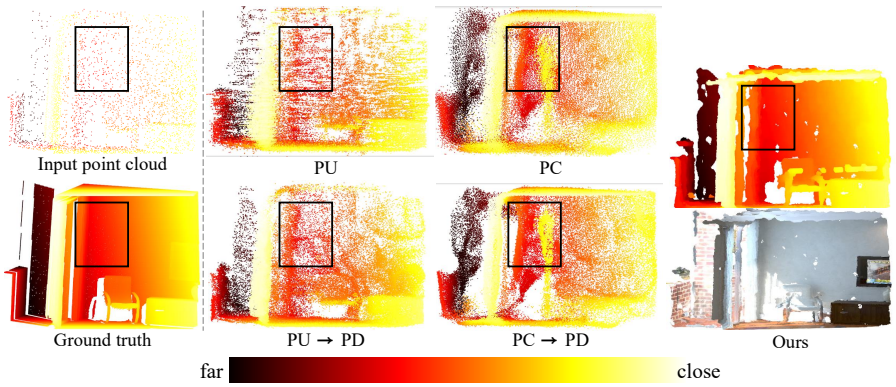
We conduct a thorough comparison against previous state-of-the-art techniques: **PU** for point cloud upsampling (Li et al., 2021), **PD** for point cloud denoising (Luo & Hu, 2021), and **PC** for point cloud completion (Xiang et al., 2021). These three papers are currently positioned at the highest rank for each task and we utilize their official implementation for training and evaluation. all methods are 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. In particular, we analyze the efficacy of ours compared to the combination of the existing studies. For instance in Table 1, *PC(r=4)→PU(r=4)→PD* represents consecutive operation of point completion, and point upsampling and point denoising under upsampling ratio r=4. Note that we also take into account the point completion task since this task also considers point densification on visible surface.

4.3 DATASET

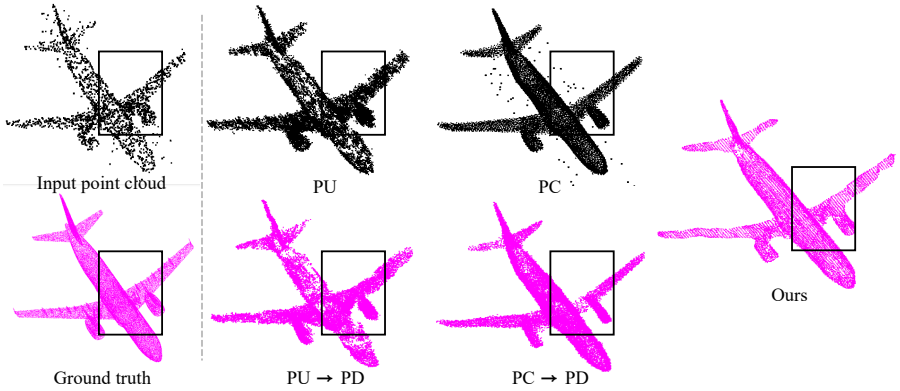
ShapeNet dataset. We utilize 13 categories of different objects categories from the ShapeNet-Part dataset (Chang et al., 2015; Yi et al., 2016). We extract dense and uniform 10K ground truth



(a) Point cloud reconstruction results in ScanNet dataset.



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



(c) Point cloud reconstruction results in ShapeNetPart dataset.

Figure 5: **Point cloud reconstruction results using two baselines.** Note that point cloud has been colorized for a visualization purpose.

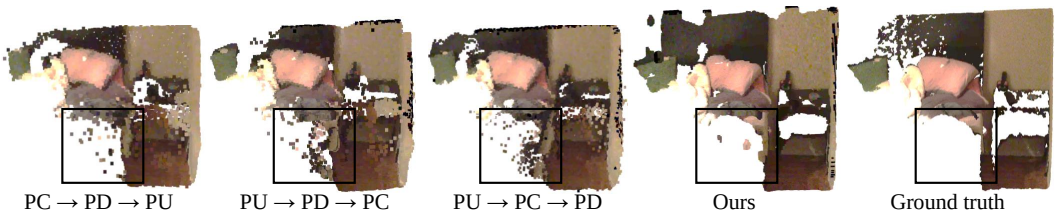


Figure 6: **Point cloud reconstruction results using three baselines.** Note that point cloud has been colorized for a visualization purpose.

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 maps in ScanNet dataset are measured from low-quality 3D sensors, we utilize ICL-NUIM dataset for the precise comparison. 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 (*i.e.*, k-Chamf.[2]) that efficiently computes point-wise distance than Earth Mover's distance. Also, this formulation considers 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). Moreover, the performance gap increases when we conduct generalization test in ScanNet dataset and ICL-NUIM dataset. There are two dominant reasons behind the better generalization and transferability. First, each sparse convolution layer has consistent and absolute-scale receptive fields in 3D space. This local voxel neighborhood appears to be more robust than raw point cloud processing via k-Nearest Neighbor (Mao et al., 2019) as adopted in the three baselines (Xiang et al., 2021; Luo & Hu, 2021; Li et al., 2021). This is also one of the strength of using voxelization as an initial processing of point cloud reconstruction, which is conducted by our 1st network. Second, our network does not have task-specific header networks or task-oriented losses for completion/upsampling/denoising tasks. Our network purely focuses on the inherent problems resided in raw point cloud. Such joint approach trains our network to improve the overall proximity-to-surface, not the task-specific quality.

Regardless the order in which we apply the baselines, our method systematically outperforms them. When we sequentially apply point densification and point denoising, the performances are not constantly improved in Table 1. This observation is supported by the qualitative results in Fig. 5. While point denoising can effectively remove the remaining outliers that are generated from (Xiang et al., 2021) or (Li et al., 2021), it has chance to remove the wrong points sets. For instance in Fig. 4, the point denoising method erases part of the airplane's wing. In contrast, our voxel generation network conducts both voxel pruning and generation in each hourglass network so that it can effectively circumvent such case.

[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.

[2] Precise equations of the metric computation are described in the appendix.

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	
1st stage	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 point cloud reconstruction 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.

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 in Table 1), and cross-attention layers (*i.e.*, Cross-attn in Table 1). 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.* 1st stage in Table 1), 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 transformers 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. Despite our success, there exist remaining issues, such as noise modeling for better denoising and point-voxel fusion design.

References

Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2): 153–168, 2016.

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.

Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Claudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999.

Jean-Daniel Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Trans. Graph.*, 1984.

Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67(5):405–451, 2005.

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, Sunghoon Im, Francois Rameau, Minjun Kang, and In So Kweon. Volumefusion: Deep depth fusion for 3d scene reconstruction. In *Int. Conf. Comput. Vis.*, 2021a.

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, 2021b.

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.

Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pp. 303–312, 1996.

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.

Herbert Edelsbrunner and Ernst P Mücke. Three-dimensional alpha shapes. *ACM Trans. Graph.*, 1994.

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.

Zan Gojcic, Caifa Zhou, Jan D Wegner, Leonidas J Guibas, and Tolga Birdal. Learning multiview 3d point cloud registration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1759–1769, 2020.

Meenakshisundaram Gopi, Shankar Krishnan, and Cláudio T Silva. Surface reconstruction based on lower dimensional localized delaunay triangulation. In *Computer Graphics Forum*. Wiley Online Library, 2000.

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.

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques*, pp. 71–78, 1992.

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.

Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pp. 339–346, 2002.

Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

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.

Ravikrishna Kolluri, Jonathan Richard Shewchuk, and James F O'Brien. Spectral surface reconstruction from noisy point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 11–21, 2004.

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.

William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987.

Weixin Lu, Guowei Wan, Yao Zhou, Xiangyu Fu, Pengfei Yuan, and Shiyu Song. Deepvcp: An end-to-end deep neural network for point cloud registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–21, 2019.

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, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Adv. Neural Inform. Process. Syst.*, 2016a.

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

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019.

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.

Marie-Julie Rakotosaona, Paul Guerrero, Noam Aigerman, Niloy J Mitra, and Maks Ovsjanikov. Learning delaunay surface elements for mesh reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021.

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.

Nicholas Sharp and Maks Ovsjanikov. Pointtrinet: Learned triangulation of 3d point sets. In *European Conference on Computer Vision*, pp. 762–778. Springer, 2020.

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.

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.

Haofei Xu and Juyong Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1959–1968, 2020.

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.

Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 185–194, 2019.

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.

APPENDIX

In this appendix, we describe further details of the proposed methodology, deep point cloud reconstruction. First, we provide the additional quantitative results (Sec. A). Second, we describe precise equations of the metrics that we used for evaluation (Sec. B). Third, we present qualitative comparisons of ours and baseline methods (Sec. C). Last, we include our response letter for the reviewers' convenience (Sec. D).

A QUANTITATIVE RESULTS

Along with Table 1 of the manuscript, we provide additional results that we did not include in the manuscript.

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
PC (r=4)	2048	8192	**1.14**	**1.29**	**1.51**	72.28	**42.71**	51.51	92.41	**88.48**	**90.16**
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
PC (r=8)	2048	16384	1.01	1.13	1.31	70.89	60.66	64.17	92.16	93.56	92.76
PC (r=8) → PD	2048	16384	1.25	1.34	1.49	61.11	42.98	49.74	91.73	81.01	85.83
PC (r=4) → PU (r=4) → PD	2048		1.14	1.48	1.60	56.55	49.72	52.43	87.13	82.73	84.67
PU (r=8)	2048	16384	1.02	1.14	1.32	80.84	50.36	60.74	95.69	82.21	88.05
PU (r=8) → PD	2048	16384	1.29	1.39	1.56	71.79	43.88	53.57	94.68	75.41	83.55
PU (r=4) → PC (r=4) → PD	2048		1.51	1.59	1.71	56.63	59.90	57.63	83.13	89.47	85.97
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
PC (r=8)	4098	32768	3.64	3.98	4.43	19.50	8.06	10.53	45.32	40.23	41.19
PC (r=8) → PD	4098	32768	3.61	3.84	4.19	24.58	10.33	13.82	55.21	38.18	44.47
PU (r=8)	4096	32768	3.20	3.49	3.92	21.52	9.323	11.94	48.30	36.10	39.72
PU (r=8) → 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
PC (r=16)	4098	65536	3.79	4.08	4.48	16.91	11.96	13.12	40.45	53.70	45.45
PC (r=16) → PD	4098	65536	3.71	3.93	4.23	21.93	15.05	17.33	51.57	47.38	49.12
PC (r=4) → PU (r=4) → PD	4096		3.69	3.91	4.23	23.05	12.59	15.31	49.75	42.24	44.84
PU (r=16)	4096	65536	3.68	3.94	4.32	16.81	13.48	13.90	37.89	44.60	40.04
PU (r=16) → PD	4098	65536	2.86	3.10	3.44	26.06	21.04	22.36	55.73	53.56	54.04
PU (r=4) → PC (r=4) → PD	4096		2.66	2.86	3.14	22.89	23.30	21.86	49.64	64.11	55.40
PU (r=4) → PD → PC (r=4)	4096		3.75	3.99	4.33	15.99	15.43	14.50	36.60	54.70	43.10
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
PC (r=8)	4098	32768	3.32	3.59	3.97	21.61	12.28	13.87	44.32	50.18	45.98
PC (r=8) → PD	4098	32768	3.21	3.40	3.69	28.84	15.54	18.89	56.27	48.97	51.88
PU (r=8)	4096	32768	3.25	3.49	3.86	21.00	13.05	14.16	43.31	42.44	41.46
PU (r=8) → 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
PC (r=16)	4098	65536	3.41	3.63	3.95	18.69	17.53	16.65	38.26	63.02	47.20
PC (r=16) → PD	4098	65536	3.18	3.34	3.59	24.46	20.88	21.53	50.98	57.43	53.84
PC (r=4) → PU (r=4) → PD	4096		3.61	3.81	4.11	21.62	11.41	14.05	45.34	42.57	43.32
PU (r=16)	4096	65536	3.83	4.03	4.33	17.78	18.19	16.83	35.45	49.93	41.03
PU (r=16) → PD	4098	65536	2.80	3.00	3.29	27.46	27.11	26.18	53.15	60.50	56.26
PU (r=4) → PC (r=4) → PD	4096		3.61	3.94	4.42	27.55	8.64	12.42	54.42	33.58	40.60
PU (r=4) → PD → PC (r=4)	4096		4.00	4.21	4.53	10.09	11.64	9.51	22.70	48.52	30.25
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 3: **Additional quantitative results.** We measure the quality of point reconstruction from ours, point completion (PC) (Xiang et al., 2021), point denoising (PD) (Luo & Hu, 2021) and point upsampling (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.

B METRICS

B.1 PERCENTAGE METRICS

Let us describe the precise equation of the metrics that we used in Table 1 of the manuscript and Table 3 of the appendix. Let $\mathcal{P}_{\text{GT}}$ be ground truth point cloud, $\mathcal{P}_{\text{pred}}$ a reconstructed points. For each inferred point $\mathbf{p}\in\mathcal{P}_{\text{pred}}$, its distance to ground truth is defined as:

$$d_{\mathbf{p}\rightarrow\mathcal{P}_{\text{GT}}}=\min_{\mathbf{g}\in\mathcal{P}_{\text{GT}}}\|\mathbf{p}-\mathbf{g}\|_2^2\,, \tag{6}$$

where $\mathbf{g}$ is a single point included in the ground truth point cloud $\mathcal{P}_{\text{GT}}$. These distances are gathered to calculate Accuracy (*i.e.*, Acc.) as:

$$\text{Accuracy}(d_{\text{thresh}})=\frac{100}{\mathcal{N}_{\mathcal{P}_{\text{pred}}}}\sum_{\mathbf{p}\in\mathcal{P}_{\text{pred}}}\left[d_{\mathbf{p}\rightarrow\mathcal{P}_{\text{GT}}}\leq d_{\text{thresh}}\right], \tag{7}$$

where $\mathcal{N}_{\mathcal{P}_{\text{pred}}}$ is the number of reconstructed points $\mathcal{P}_{\text{pred}}$, $[\cdot]$ is the Iverson bracket, and d_{thresh} is a threshold distance. In a similar manner, we calculate the distance from ground truth point $\mathbf{g}\in\mathcal{P}_{\text{GT}}$ to the reconstructed points $\mathcal{P}_{\text{pred}}$ as:

$$d_{\mathbf{g}\rightarrow\mathcal{P}_{\text{pred}}}=\min_{\mathbf{p}\in\mathcal{P}_{\text{pred}}}\|\mathbf{g}-\mathbf{p}\|_2^2\,, \tag{8}$$

These distances are gathered to calculate Completeness (*i.e.*, Comp.) as:

$$\text{Completeness}(d_{\text{thresh}})=\frac{100}{\mathcal{N}_{\mathcal{P}_{\text{GT}}}}\sum_{\mathbf{g}\in\mathcal{P}_{\text{GT}}}\left[d_{\mathbf{g}\rightarrow\mathcal{P}_{\text{pred}}}\leq d_{\text{thresh}}\right] \tag{9}$$

where $\mathcal{N}_{\mathcal{P}_{\text{GT}}}$ is the number of ground truth point clouds. To calculate f1-score, we use Accuracy and Completeness as below:

$$\text{f1-score}(d_{\text{thresh}})=\frac{2\cdot\text{Accuracy}(d_{\text{thresh}})\cdot\text{Completeness}(d_{\text{thresh}})}{\text{Accuracy}(d_{\text{thresh}})+\text{Completeness}(d_{\text{thresh}})}, \tag{10}$$

Note that in ShapeNet dataset, we set d_{thresh}=0.0200. For ScanNet and ICL-NUIM dataset, we set d_{thresh}=0.0050 for evaluation. We mostly follow these metrics from (Jensen et al., 2014; Aanæs et al., 2016; Yao et al., 2018; Knapitsch et al., 2017).

B.2 CHAMFER DISTANCE

Before we explain the k-Nearest Chamfer distance, let us revisit the original Chamfer distance. Based on the equations Eq. 8 and Eq. 6, we describe the Chamfer distance (*i.e.*, Chamf.) as:

$$\text{Chamf.}=\frac{1}{\mathcal{N}_{\mathcal{P}_{\text{pred}}}}\sum_{\mathbf{p}\in\mathcal{P}_{\text{pred}}}(d_{\mathbf{p}\rightarrow\mathcal{P}_{\text{GT}}})+\frac{1}{\mathcal{N}_{\mathcal{P}_{\text{GT}}}}\sum_{\mathbf{g}\in\mathcal{P}_{\text{GT}}}(d_{\mathbf{g}\rightarrow\mathcal{P}_{\text{pred}}}), \tag{11}$$

Since the original Chamfer distance only considers the closest distances, it has difficulty in describing the optimal distance between two point sets having different number of points (Fan et al., 2017). However, rather than to use Earth Mover distance that requires quadratic computational power and memory consumption, we formulate k-Nearest Chamfer distance by considering the k closest points instead of the single closest point (*i.e.*, k=1). Let us define the distance from a inferred point $\mathbf{p}\in\mathcal{P}_{\text{pred}}$ to the k closest ground truth points $\mathbf{g}_k\in\mathcal{P}_{\text{GT}}$ as follow:

$$d^k_{\mathbf{p}\rightarrow\mathcal{P}_{\text{GT}}}=\frac{1}{k}\sum_k\|\mathbf{p}-\mathbf{g}_k\|_2^2\,, \tag{12}$$

where $\mathbf{g}_k$ is the k-th closest point to the inferred point $\mathbf{p}$. Similarly, we calculate the distance from a ground truth point $\mathbf{g}$ to the k-closest reconstructed points $\mathbf{p}_k$ as follows:

$$d^k_{\mathbf{g}\rightarrow\mathcal{P}_{\text{pred}}}=\frac{1}{k}\sum_k\|\mathbf{g}-\mathbf{p}_k\|_2^2\,, \tag{13}$$

Based on these equations, we compute k-Nearest Chamfer distance (*i.e.*, k-Chamf.) as below:

$$k\text{-Chamf.}=\frac{1}{\mathcal{N}_{\mathcal{P}_{\text{pred}}}}\sum_{\mathbf{p}\in\mathcal{P}_{\text{pred}}}(d^k_{\mathbf{p}\rightarrow\mathcal{P}_{\text{GT}}})+\frac{1}{\mathcal{N}_{\mathcal{P}_{\text{GT}}}}\sum_{\mathbf{g}\in\mathcal{P}_{\text{GT}}}(d^k_{\mathbf{g}\rightarrow\mathcal{P}_{\text{pred}}}). \tag{14}$$

C QUALITATIVE RESULTS

Figure 7: **Point cloud reconstruction results.** Note that point cloud has been colorized for a visualization purpose.

D RESPONSE LETTER

Thank you for the precious comments. Following the reviewer's advice, we revise our initial manuscript as follows:

- Elaborate the related works section about a surface reconstruction task.
- Clarify the Sec. 3.2.
- Conduct experiments about possible combinations of three previous papers, which is added in Table 1 of the manuscript and Table 3 in the appendix.
- Include description in Sec. 4.4 about the strength of the proposed method, generalization and transferability.
- Add more qualitative results in the appendix.
- Describe the precise equations of metrics in the appendix.
- Implementation will be released after the review.

With this revision, the initial manuscript becomes more reasonable, analytic and feasible. We hope that all reviewers are satisfied with the improved quality of the paper. For the reviewers' convenience, we include our response letter in the appendix.

D.1 REVIEWER:VEKS

[Writings] "Q.1-1. The high-level idea makes lots of sense to me. Using voxelization to denoise and complete irregular point cloud is reasonable and smart. Besides, we can sample as many points as we want from a voxel grid to get a dense surface. If possible, the authors could further elaborate on this contributions."

Thank you very much for this positive remark regarding using voxelization for this task. As suggested, we underlined the flexibility offered by our voxel based technique in Sec. 4.4 of the manuscript. This advantage is also highlighted by the quantitative results presented in Table 1 of the manuscript and Table 1 of the appendix. These results demonstrate that the number of output points $\mathcal{P}_{\text{out}}$ can be adjusted by the hyper-parameters while preserving the overall reconstruction quality.

[Baseline] "Q.1-2. The presented results are great, but the baselines are not very considerate. This paper is arguing that the three tasks should be unified and hence multi-task joint-learning baselines should be considered. The simplest setting is to ensemble all three baselines together in an optimized order. Alternatives contain training each baseline methods on all three tasks jointly."

Admittedly, our initial experimental setup was lacking fairness, therefore, we have re-conducted a large set of experiments where all possible combinations between the three baselines have been investigated. The obtained results as visible in Table 1 of the manuscript and Table 3 of the appendix. Please take an example with some notations used in the tables in Table 1 of the manuscript. For instance, *PC*(r=4)→*PU*(r=4)→*PD* represents consecutive operation of point completion, and point upsampling and point denoising under upsampling ratio r=4. Note that we also take into account the point completion task since this task also considers point densification on visible surface.

None of the tested configurations is qualitatively nor quantitatively competitive with our point reconstruction technique. It should however be noted that since only 2048 points have been used as input, the point denoising method (Luo & Hu, 2021) could not be used before upsampling (because (Luo & Hu, 2021) requires at least 10,000 points to operate). Thus, we cannot provide results for every single possible combinations. Nonetheless, our method still achieves state-of-the-art performance among the possible orders of point denoising/completion/upsample networks.

[Multi-task learning] "Q.1-3. The proposed method should also be evaluated for each single task to demonstrate advantages."

We understand the intention of the reviewer veKS. We guess that the worrying points derive from the reviewer veKS's initial description, *"I will use the word **multi-task** to refer to the problem*

setting in this work, in contrast to the single-task baselines." If this paper can be viewed as multi-task learning of three different tasks, it is important to evaluate the results on each single task, such as denoise, densification and completion task.

However, our point reconstruction technique cannot truly be considered as a multi-task learning pipeline. There are two reasons. First, the series of problems (sparsity, noise, and irregularity) are inherently resided in raw point clouds. In real-world point clouds, these problems are completely mixed. Second, we solves the problems in a completely joined manner. As a result, we cannot isolate each individual task since our network does not have dedicated to design task-specific header networks nor task-oriented losses. Moreover, even if we forcefully obtain the results in each single task, it will lack fairness and would not reflect our intention nor the capability of our point reconstruction technique. We hope to have clarified this particular point and that the reviewer will understand the context for which we have designed this point reconstruction strategy combining three tasks at once to resolve the inherent shortcomings of 3D point clouds.

[Amplified positional encoding] "Q.1-4. I am not fully convinced by the story of positional encoding. In the original form, the distance information is already expressed as the frequency of the signal. The magnitude is just redundant information. Why is it important? And what kind of effect it will cause precisely?"

We apologize for the confusion. The intuition behind our amplitude positional encoding is the following, **when to embed high frequency signal?** We are inspired by NeRF (Mildenhall et al., 2020) where positional encoding results in preserving the high-frequency image signal. In our case, we are facing a different situation. high-frequency signal is not always welcomed. When a voxel needs to move to a distant location, it is better to keep the low frequency signal, which is simply operated by decreasing the amplitude of the positional encoding vector as described in Fig. 4 of the manuscript.

Such control is possible since we already know the relative distance between voxels that are estimated from the 1st stage network. As underlined by the reviewer G4Aq, we believe that our amplified encoding is an important contribution of this work. To better highlight the necessity of this component, we further elaborate the paragraphs about our amplified positional encoding in Sec. 3.2 of the manuscript.

[Generalization] "Q.1-5. For the tested dataset, especially this synthetic ones (ICL-NUIM), how does the input noise/sparsity/incompleteness are created? ... Also, regular synthetic noise (*e.g.*, Gaussian noise) are much simple than real-world noise. How does the proposed framework deal with real-world noise? ... It is unclear how does this framework work for real-world sparse point clouds (*e.g.*, the remote areas in LiDAR data)."

Thank you very much for this remark regarding the transferability of the approach. As the reviewer veKS mentioned, we train our network only in synthetic/object-scale dataset (ShapeNetPart). Then, without fine-tuning, we utilize the pre-trained weights to measure the quality of reconstruction using unmet point clouds in ScanNet or ICL-NUIM datasets. Despite the domain gap between datasets, we think our network generalizes well toward various real datasets. We conjecture that this transferability can be explained by two dominant reasons.

First, we optimize the nature of the objective function. Indeed improving the proximity-to-surface instead of task-specific losses (such as denoising, densification, and completion) seems to be a more generic and generalizable problem. Regardless of the difference in synthetic and real-world noises, our network is trained to improve the quality of reconstruction. Also, this is clearly different from the point completion task (Yuan et al., 2018) that concentrates on category-specific datasets for generating points in occluded regions of objects.

Second, we utilize the hash table for sparse convolution layer instead of k Nearest Neighbor. To handle noisy and sparse point cloud, kNN is not a proper choice since it is sensitive to the density of point clouds (Mao et al., 2019). Nonetheless, previous studies (Li et al., 2021; Xiang et al., 2021; Yu et al., 2018b) mainly rely on this clustering algorithm. In contrast, sparse convolution layer takes absolute scale of receptive field regardless of the input distribution, which results in better transferability toward unmet and real-world scenes.

Note that we apply random Gaussian noise into ICL-NUIM dataset following (Li et al., 2021). This synthetic noise is truely much simpler than the real-world noise, as addressed by the reviewer. Since

we did not precisely analyze the noise modeling, we have improved the exposition in the paper to clarify the discussion.

[Sparse stacked hourglass network] "Q.1-6. 3D sparse stacked hourglass network is not a super popular network. It is also not implemented in popular sparse conv net framework (*e.g.*, Minkowski engine). Why this network is adopted? Maybe the authors can share some related works to justify the advantage. It is not a common choice."

In our understanding, stacked hourglass has unique properties and widely used for various application. First of all, we found that the total number of citation of stacked hourglass network is **3364** (Newell et al., 2016b) and **541** (Newell et al., 2016a). Note that the original stacked hourglass network is proposed in (Newell et al., 2016b) and the modified version of the network is introduced in (Newell et al., 2016a) by the same first author.

Moreover, the variants of stacked hourglass network largely used in various tasks. For instance in stereo matching network, PSMNet (cite: **706**) (Chang & Chen, 2018), GANet (cite: **246**) (Zhang et al., 2019), AANet (cite: **76**) (Xu & Zhang, 2020), utilizes a volumetric version of hourglass network using 3D CNNs. This paper clearly mentions their stacked hourglass network in their abstract, *"The 3D CNN learns to regularize cost volume using stacked multiple hourglass networks in conjunction with intermediate supervision."* In an another example, multi-view stereo task, DPSNet (cite: **98**) (Im et al., 2019) also adopts 3D stacked hourglass network to aggregate cost volumes. Different from deep and wide U-shape network, such as Minkowski-UNet (Choy et al., 2019), using multiple light encoder-decoder networks (*i.e.*, stacked hourglass network) aims to sequentially refine feature vectors by intermediate losses.

In summary, our 1st stage network aims to sequentially refine the given point cloud through the proposed sparse stacked hourglass network. Similar to the idea of voxel carving, hourglass networks re-shape the sparse points in coarse-to-fine manner. We hope that the reviewer veKS is satisfied with our understanding of stacked hourglass network (Newell et al., 2016b). Also, we expect the future studies about point cloud reconstruction will widely use this sparse version of stacked hourglass network as stereo matching methods (Chang & Chen, 2018; Xu & Zhang, 2020; Zhang et al., 2019) and multi-view stereo (Im et al., 2019; Choe et al., 2021a) studies do.

[Missing details] "Q.1-7. How many voxels are processed in the transformer on average?"

We apologize for the missing details. We set $k = 8$ as an input of our transformer. This information is included in Sec. 3.2 of the manuscript and Table 3 of the appendix.

[Missing details] "Q.1-8. How is the dimension of $\mathcal{P}_{\text{out}}$ decided?"

Since our 1st stage network adaptively generates and prunes voxels, we cannot precisely decide for the dimension of $\mathcal{P}_{\text{out}} \in \mathbb{R}^{N' \times 3}$ where N' is determined after the inference. Thus, we report the average cardinality of the generated point clouds in Table 1 of the manuscript.

[Related works] "Q.1-9. The difference between point upsampling and point completion is not very clear. Very specific definition should be made in the very early stage of the writing so that there is no ambiguity between the challenges that these two problems pose. "

The difference between the different tasks have been described in the "related works" section. The goal of point upsampling is to densify sparse but homogeneously distributed input points. On the other hand the point completion task, as described in the pioneering paper (Yuan et al., 2018), aims at extrapolating entire parts of the objects resulting from occlusions or the limited sensor resolution. Regarding the experiments validating each task, we invite the reviewer to refer to our response entitled "Multi-task learning" in this rebuttal.

[Title] "Q.1-10. If find it misleading to their this task *point cloud reconstruction*. A maybe improper analogy is that we do not call joint image denoising and super-resolution and image generation. If the authors really want to use the term reconstruction, maybe call it *conditioned point cloud reconstruction*? Or something else that is more accurate like *point cloud refinement*?"

We are pleased for the reviewer's recommendation. We also think of this naming issue before the submission. Our conclusion is that we put more weights on the quality of 3D reconstruction using 3D points, so called point cloud reconstruction. As addressed by the reviewer **dx4y**, this task can be similar to surface reconstruction task because of the common goals, high-quality 3D description.

However, without point triangulation or mesh formulation, this task truely focus on the point itself to describe 3D environments. We hope that the reviewer is satisfied with our opinion.

[Missing details] "Q.1-11. Where are the cluster information used in the later framework? If I understand correctly, this used because the voxel-to-point step only operates on center voxels and hence we want a locally augmented feature representation?"

To be more informative, we revise the paragraphs in Sec. 3.2.

Let us describe the detailed process in the point re-localization network, which is illustrated in Fig. 4. Given output voxels $\mathcal{V}_{\text{out}}$, we collect the $K(=8)$ closest voxels to each voxel $\mathbf{v}_i \in \mathbb{R}^{1\times 3}$ using the hash table that we used in the 1st stage network. Then, we obtain a voxel set $\mathcal{V}_i = \{\mathbf{v}_k\}_{k=1}^{K}$ that is referenced to the query voxel $\mathbf{v}_i$. Similarly, we cluster a set of voxel features $\mathcal{F}_i = \{\mathbf{f}_k\}_{k=1}^{K}$ that correspond to $\mathcal{P}_i$. As in Fig. 4, the 2nd stage network regresses the location of a query point $\mathbf{p}_i$ (red voxel) using the geometric relation between the query voxel $\mathbf{v}_i$ and its neighbor $\mathcal{V}_i$ (white voxels) through self/cross attention layers.

While this formulation is close to PointTransformer (Zhao et al., 2021), our point re-localization network has three distinctive differences. First, PointTransformer utilizes k-Nearest Neighbors to group 3D points, we re-use the voxel hashing that is computed in the previous stage. Second, our network consists of both self-attention and cross-attention. In our analysis, it is better to use two different attention layers than to solely adopt self-attention.

[Figure] "Q.1-12. Fig2 input figure is barely visible."

We are sincerely thankful regarding this comment on the editorial quality of the paper. In Fig. 2 of the manuscript, to faithfully highlight the ability of our network, we intentionally set the same point's size for the visualization of the input and output point clouds. This clearly emphasizes how precisely and densely our network reconstructs the given sparse points. To clarify this point, we include more details about visualization in Fig. 2 of the manuscript.

[highlight] "Q.1-13. In Tab.1, maybe the light color can be used to highlight the 2nd best among the baseline method. Highlighting the 2nd best among all including **ours** cannot fully demonstrate the advantages."
Following the request by the reviewer, we remove the 2nd best highlight in Table 1 of the manuscript.

D.2 Reviewer:1Mes

[More qualitative results] "Q2-1. Qualitative examples are good and would like to see more of them in the supplemental material."

We newly include qualitative results in the appendix. Also, we update the results from sequential uses of three baselines in Fig. 5 of the manuscript.

[Amplified positional encoding] "Q2-2. The initial section of the paper until sec.3.1 is clear and well explained, while section 3.2 needs a better explanation. I still do not understand the exact intuition behind the amplitude-based positional embedding."

We newly revise section 3.2 of the manuscript following the reviewer's comment. The intuition behind our amplitude positional encoding is clear, **when to embed high frequency signal?** We are inspired by NeRF (Mildenhall et al., 2020) where positional encoding results in preserving the high-frequency image signal. In our case, we are in different situation. high-frequency signal is not always welcomed. When a voxel needs to move to a distant location, it is better to keep the low frequency signal, which is simply conducted by decreasing the amplitude of the positional encoding vector as in Fig. 4 of the manuscript.

Such control is possible since we already know the relative distance between voxels that are estimated from the 1st stage network. As underlined by the reviewer G4Aq, we believe that our amplified encoding is an important contribution of this work. To better highlight the necessity of this component, we further elaborate the paragraphs about our amplified positional encoding in Sec. 3.2 of the manuscript.

[Writing] "Q2-3. The baselines are not explained well, the exact characteristics of the Snowflake-PC $\rightarrow$ Score-PD are not explained, likewise for Dis-PU. These sections need rewriting."

As requested by the reviewer, we revised Sec. 4.2 and Sec. 2 of the manuscript. Also, we included additional experiments about ensembles of three related papers in Table 1 of the manuscript and Table 3 in the appendix. Let us present the revised paragraphs in Sec. 4.2 below.

In particular, we analyze the efficacy of ours compared to the combination of the existing studies. For instance in Table 1, *PC*(r=4)→*PU*(r=4)→*PD* represents consecutive operation of point completion, and point upsampling and point denoising under upsampling ratio r=4. Note that we also take into account the point completion task since this task also considers point densification on visible surface.

[Details] "Q2-4. There is no evaluation independently for Stage-1 all evaluations. ... I understand that maybe doing the full evaluation on Stage-1 is not possible."

The evaluation for our 1st network is already included in Table 2-(b) of the manuscript. For clarity, we denote the name as 1st stage.

[Details] "Q2-5. The paper claims like past work when they upsample they do not have this scaling-up factor of r but I think in this case, r just manifests as the voxel density in stage 1."

It is true that the *generative sparse transposed convolution layer* generates voxels in a fixed ratio r. Following the reviewer's comment, we revise the sentences in Sec. 3.1 of the manuscript.

[Future direction] "Q2-6. I am also not clear what happens when multiple voxel centers move to the same location in 3D points."

We are pleased to answer to this very well spotted observation. This is, indeed, a limitation of our architectural design. Specifically, when multiple points are closer than the voxel size, they are merged into a single voxel during the voxelization process. Although this was not clearly visible in our experiments, we agree that the raised concerns can harm the results. In the future, we think of a new network design that utilize both voxel and point-level understanding to overcome such limitation.

D.3 REVIEWER:G4AQ

[Generalization] "Q3-01. It would be better if the paper could explain why the proposed new method can achieve such strong generalization ability, and why the generalization ability of the existing methods is poor."

It is our pleasure for the reviewer to recognize the strength of our method, generalization. We conjecture that this transferability can be explained by two dominant reasons.

First, we optimize the nature of the objective function. Indeed improving the proximity-to-surface instead of task-specific losses (such as denoising, densification, and completion) seems to be a more generic and generalizable problem. Regardless of the difference in synthetic and real-world noises, our network is trained to improve the quality of reconstruction. Also, this is clearly different from the point completion task (Yuan et al., 2018) that concentrates on category-specific datasets for generating points in occluded regions of objects.

Second, we utilize the hash table for sparse convolution layer instead of k Nearest Neighbor. To handle noisy and sparse point cloud, kNN is not a proper choice since it is sensitive to the density of point clouds (Mao et al., 2019). Nonetheless, previous studies (Li et al., 2021; Xiang et al., 2021; Yu et al., 2018b) mainly rely on this clustering algorithm. In contrast, sparse convolution layer takes absolute scale of receptive field regardless of the input distribution, which results in better transferability toward unmet and real-world scenes.

These are our analysis of our strength in better generalization and transferability. With our sparse stacked hourglass network, we can achieve well-generalized reconstruction performance. These voxelized results are further refined by the 2nd stage network that converts discrete voxels into 3D points.

Finally, we revise Sec. 4.4 of the manuscript to clarify the discussion about generalization and transferability of our deep point cloud reconstruction network.

D.4 REVIEWER:DX4Y

[Point cloud reconstruction] "Q4-1. Since many scans are combined into a single scan, there is another challenge that is entirely overlooked, misalignment. If the idea is to combine all challenges into one, this should another challenge to address."

Point cloud registration is one another topic that is widely addressed in previous studies (Choy et al., 2020; Lu et al., 2019; Gojcic et al., 2020). However, in our opinion, point cloud registration (misalignment) is not an inherent problem of point clouds. Misalignment usually derives from the extrinsic parameters between multiple 3D scans. In our point of view, this problem happens when we fuse multiple 3D scans that are taken in different views or time stamps. We think that it will be a wonderful future direction that extends the applicability of point cloud reconstruction toward multiple 3D scans.

[Related works] "Q4-02. Missing related work. There is a very large body of recent works on neural shape implicit representation which focus on the surface reconstruction task. ..., one could simply sample points on the zero level set and achieve point cloud reconstruction."

We apologize for the missing references about surface reconstruction. Following the reviewer's advice, we included a paragraph in Sec. 2. However, we claim that the point cloud reconstruction task is clearly different from surface reconstruction. Let us briefly describe their difference.

First, mesh triangulation is not related to the inherent shortcomings of point cloud. According to the recent papers, PointTriNet (Sharp & Ovsjanikov, 2020) and (Rakotosaona et al., 2021), conventional studies about point triangulation often requires dense and uniform 3D points. It implies that the surface reconstruction task puts more weight on mesh generation from given 3D points.

Second, according to the DeepSDF (Park et al., 2019), this paper also utilizes ShapeNet dataset (Chang et al., 2015), but relies on a large number of points for training, which is described as *"To prepare data, we start by normalizing each mesh to a unit sphere and sampling **500,000** spatial points x's: we sample more aggressively near the surface of the object as we want to capture a more detailed SDF near the surface"*. Therefore, while being different, we believe that our research can be very complementary with such type of work. For instance, point cloud reconstruction can be applied as a pre-processing to provide a dense and well distributed point cloud to train DeepSDF."

Third, point sampling from mesh cannot be ahead of point cloud reconstruction task. In our case, we utilize Poisson Disk Sampling (Yuksel, 2015; Zhou et al., 2018) to obtain ground truth point cloud from ground truth mesh provided in the ShapeNet dataset (Chang et al., 2015; Yi et al., 2016).

Based on these three reasons, we claim that point cloud reconstruction has clear characteristics compared to the surface reconstruct task. Moreover, to obtain such large amount of points to train recent implicit methods (Park et al., 2019), point cloud reconstruction should be taken as a pre-processing stage, ahead of surface reconstruction. We hope the reviewer is satisfied with the necessity of point cloud reconstruction task.

[Details] "Q4-03. It was not clear to me how the final number of points is specified. Also, not clear how color information is determined for the newly generated points."

As we described in Sec. 3.1, our network jointly generates and prunes voxels in an adaptive manner. Even though transposed generative convolution layer upsample voxels in a fixed ratio of r, voxel pruning layer remove outliers based on their signals. Thus, we cannot strictly say the number of outputs and report the average of $\mathcal{P}_{\text{out}}$ in Table 1.

[Evaluation] "Q4-04. I find the choices of the baseline questionable. First, there was no unified approach used (denoising + completion + sampling). Second, It would make more sense to me compare the proposed method on the other's task since it is expected that a denoising method will not do well for upsampling. Third, in the combined baselines (Snowflake-PC $\rightarrow$ Score-PD and Dis-PU $\rightarrow$ score-PD), it is not clear the order of application (it makes sens to first denoise and upsample/complete but the text suggest the opposite which should bee reported). Finally, as mentioned in the related work, the paper lacks comparison to implicit representations surface reconstruction.."

Following the request of the reviewers, we conduct additional experiments about the combination of three models, denoising, completion and upsampling. It turns out that the different orders of three models do not improve the quality of point reconstruction compared to the results in Table 1 of the

manuscript. We claim that these results consistently support the necessity of joint solution for point cloud reconstruction.

We think that our point reconstruction technique cannot truly be considered as a multi-task learning pipeline since it solves the problems in a completely joined manner. As a result, we cannot isolate each individual task since our network does not have dedicated to design task-specific header networks nor task-oriented losses. Instead, we assume that these problems are inherent to point cloud obtained by scanning devices and should be resolved together. Moreover, even if we forcefully obtain the results in each single task, it will lack fairness and would not reflect our intention nor the capability of our point reconstruction technique. We hope to have clarified this particular point and that the reviewer will understand the context for which we have designed this point reconstruction strategy combining three tasks at once to resolve the inherent shortcomings of 3D point clouds.

[Novelty] "Q4-05. This paper is missing an important subsection that provides some insight into what the different modules have learned and what is that beneficial to the task. This raises some question about novelty - how is this approach better than the sum of its technical parts?"

As noted by three reviewers (veKS, 1Mes, and G4Aq), our contributions derive from two-stage pipeline that aims to handle raw point clouds via voxelization. This module is very simple and straightforward, but as we describe in Sec. 3.1, each layer are deeply related to the point cloud reconstruction task. These architectural design leads to better generalization and transferability toward unmet and real-world 3D scans as in Table 4. Let us briefly explain the reasons behind our strength of generalization and transferability. Throughout this paper, we conjecture that this transferability can be explained by two dominant reasons.

First, we optimize the nature of the objective function. Indeed improving the proximity-to-surface instead of task-specific losses (such as denoising, densification, and completion) seems to be a more generic and generalizable problem. Regardless of the difference in synthetic and real-world noises, our network is trained to improve the quality of reconstruction. Also, this is clearly different from the point completion task (Yuan et al., 2018) that concentrates on category-specific datasets for generating points in occluded regions of objects.

Second, we utilize the hash table for sparse convolution layer instead of k Nearest Neighbor. To handle noisy and sparse point cloud, kNN is not a proper choice since it is sensitive to the density

ShapeNet-Part dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k-Chamf. (↓)			Percentage (↑) (<0.5)			Percentage (↑) (<1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
PC (r=4)	2048	8192	**1.14**	**1.29**	**1.51**	72.28	**42.71**	51.51	92.41	**88.48**	**90.16**
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
PC (r=4) → PU (r=4) → PD	2048	32987	1.14	1.48	1.60	56.55	49.72	52.43	87.13	82.73	84.67
PU (r=4) → PC (r=4) → PD	2048	32723	1.51	1.59	1.71	56.63	59.90	57.63	83.13	89.47	85.97
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-Chamf. (↓)			Percentage (↑) (<0.5)			Percentage (↑) (<1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
PC (r=16)	4098	65536	3.79	4.08	4.48	16.91	11.96	13.12	40.45	53.70	45.45
PC (r=4) → PU (r=4) → PD	4096	64427	3.69	3.91	4.23	23.05	12.59	15.31	49.75	42.24	44.84
PU (r=16)	4096	65536	3.68	3.94	4.32	16.81	13.48	13.90	37.89	44.60	40.04
PU (r=4) → PC (r=4) → PD	4096	64578	2.66	2.86	3.14	22.89	23.30	21.86	49.64	64.11	55.40
PU (r=4) → PD → PC (r=4)	4096	63118	3.75	3.99	4.33	15.99	15.43	14.50	36.60	54.70	43.10
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**
ICL-NUIM dataset											
Methods	$\mathcal{P}_{\text{in}}$	$\mathcal{P}_{\text{out}}$	k-Chamf. (↓)			Percentage (↑) (<0.5)			Percentage (↑) (<1.0)		
			k=1	k=2	k=4	Acc.	Comp.	f-score	Acc.	Comp.	f-score
PC (r=16)	4098	65536	3.41	3.63	3.95	18.69	17.53	16.65	38.26	63.02	47.20
PC (r=4) → PU (r=4) → PD	4096	64451	3.61	3.81	4.11	21.62	11.41	14.05	45.34	42.57	43.32
PU (r=16)	4096	65536	3.83	4.03	4.33	17.78	18.19	16.83	35.45	49.93	41.03
PU (r=4) → PC (r=4) → PD	4096	64872	3.61	3.94	4.42	27.55	8.64	12.42	54.42	33.58	40.60
PU (r=4) → PD → PC (r=4)	4096	63521	4.00	4.21	4.53	10.09	11.64	9.51	22.70	48.52	30.25
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 4: **Quantitative results of point cloud reconstruction.** Note that PC, PU, PD represent point completion (Xiang et al., 2021), point upsampling (Li et al., 2021), and point denoising (Luo & Hu, 2021), respectively. Please refer to the appendix for more results.

of point clouds (Mao et al., 2019). Nonetheless, previous studies (Li et al., 2021; Xiang et al., 2021; Yu et al., 2018b) mainly rely on this clustering algorithm. In contrast, sparse convolution layer takes absolute scale of receptive field regardless of the input distribution, which results in better transferability toward unmet and real-world scenes.

Nonetheless, we find that we need to further provide the reasonable analysis of our strength in generalization ability and transferability. Following the advice from the reviewer dx4y, we elaborate the strength of our method in Sec. 4.4 of the manuscript.

[Writing] "Q4-06. In its current for, the paper is not self-contained and the method is not reproducible due to a lack of technical details"

Following the request from the reviewer, we newly update Fig. 3 and Fig. 4. Every detail of channel dimensions and connectivity between layers are precisely described. Moreover, we are planning to release the code after the review. Since this is a pioneering paper that opens a new task, as a researcher, we think it is our responsibility to clearly open everything for the community.

[Minor-Writing] "Q4-07. There is an issue with the term "continuous 3D points" which repeats itself in the paper. Point clouds are not continuous, they are sampled on a continuous surface."

We apologize for the confusing terminology. We revise all these words in the manuscript.

[Minor-Metrics] "Q4-08. The evaluation metrics were not properly specified. When performing standardized experiments it may be sometimes acceptable to commit this, however, in these cases, the paper proposes a new task, therefore the evaluation metrics should be specified."

Following the request by the reviewer, we describe the precise equations of all metrics that we used in the appendix.

[Minor-Metrics] "Q4-9. $\mathcal{V}_{\text{mid}}$ is missing."

We apologize for the missing details. Following the reviewer's request, we newly update the Fig. 3 of the manuscript. Note that $\mathcal{V}_{\text{mid}}$ is the intermediate results from each hourglass network in the 1st stage network.