Snowflake Point Deconvolution for Point Cloud Completion and Generation with Skip-Transformer

Peng Xiang*, Xin Wen*, Yu-Shen Liu *Member, IEEE*, Yan-Pei Cao, Pengfei Wan, Wen Zheng, Zhizhong Han

Abstract—Most existing point cloud completion methods suffered from discrete nature of point clouds and unstructured prediction of points in local regions, which makes it hard to reveal fine local geometric details. To resolve this issue, we propose SnowflakeNet with Snowflake Point Deconvolution (SPD) to generate the complete point clouds. SPD models the generation of complete point clouds as the snowflake-like growth of points, where the child points are progressively generated by splitting their parent points after each SPD. Our insight of revealing detailed geometry is to introduce skip-transformer in SPD to learn point splitting patterns which can fit local regions the best. Skip-transformer leverages attention mechanism to summarize the splitting patterns used in previous SPD layer to produce the splitting in current SPD layer. The locally compact and structured point clouds generated by SPD precisely reveal the structure characteristic of 3D shape in local patches, which enables us to predict highly detailed geometries. Moreover, since SPD is a general operation, which is not limited to completion, we further explore the applications of SPD on other generative tasks, including point cloud auto-encoding, generation, single image reconstruction and upsampling. Our experimental results outperform the state-of-the-art methods under widely used benchmarks.

Index Terms—point clouds, 3D shape completion, generation, reconstruction, upsampling, transformer

1 Introduction

In 3D computer vision [1], [2], [3], [4] applications, raw point clouds captured by 3D scanners and depth cameras are usually sparse and incomplete [5], [6], [7] due to occlusion and limited sensor resolution. Therefore, point cloud completion [5], [8], which aims to predict a complete shape from its partial observation, is vital for various downstream tasks. Benefiting from large-scaled point cloud datasets, deep learning based point cloud completion methods have been attracting more research interests. Current methods either constrain the generation of point clouds using a hierarchical rooted tree structure [8], [9], [10] or assuming a specific topology [5], [11] for the target shape. However, most of these methods suffered from discrete nature of point cloud and unstructured prediction of points in local regions, which makes it hard to preserve a well arranged structure for points in local patches. Therefore, it is still challenging to reveal local geometric details and structure characteristic during completing partial 3D shapes, such as smooth regions, sharp edges and corners, as illustrated in Fig. 1.

- *Peng Xiang and Xin Wen are with the School of Software, Tsinghua University, Beijing, China. Xin Wen is also with JD.com, Beijing, China. E-mail: xp20@mails.tsinghua.edu.cn, wenxin16@jd.com*
- *Yu-Shen Liu is with the School of Software, BNRist, Tsinghua University, Beijing, China. E-mail: liuyushen@tsinghua.edu.cn*
- *Yan-Pei Cao, Pengfei Wan and Wen Zheng are with the Y-tech, Kuaishou Technology, Beijing, China. Email: caoyanpei@gmail.com, {wanpengfei, zhengwen}@kuaishou.com*
- *Zhizhong Han is with the Department of Computer Science, Wayne State University, USA. E-mail: h312h@wayne.edu*

Peng Xiang and Xin Wen contributed equally to this work. Yu-Shen Liu is the corresponding author. This work was supported by National Key R&D Program of China (2020YFF0304100), the National Natural Science Foundation of China (62072268), and in part by Tsinghua-Kuaishou Institute of Future Media Data. Source code is available at https://github.com/AllenXiangX/SnowflakeNet.

In order to address this problem, we propose a novel network called *SnowflakeNet*, especially focusing on the decoding process to complete partial point clouds. SnowflakeNet mainly consists of layers of *Snowflake Point Deconvolution* (SPD), which models the generation of complete point clouds like the snowflake growth of points in 3D space. We progressively generate points by stacking one SPD layer upon another, where each SPD layer produces child points by splitting their parent point with inheriting shape characteristics captured by the parent point. Fig. 2 illustrates the process of SPD and point-wise splitting.

Our insight of revealing detailed geometry is to introduce *skip-transformer* in SPD to learn point splitting patterns which can fit local regions the best. Compared with the previous methods, which often ignore the spatial relationship among points [8], [11], [13] or simply learn through self-attention in a single level of multi-step point cloud decoding [5], [9], [14], our skip-transformer is proposed to integrate the spatial relationships across different levels of decoding. Therefore, it can establish a cross-level spatial relationships between points in different decoding steps, and refine their location to produce more detailed structure. To achieve this, skip-transformer leverages attention mechanism to summarize the splitting patterns used in the previous SPD layer, which aims to produce the splitting in current SPD layer. The skip-transformer can learn the shape context and the spatial relationship between the points in local patches. This enables the network to precisely capture the structure characteristic in each local patches, and predict a higher quality point cloud for both smooth plane and sharp edges in 3D space. We achieved the state-of-the-art completion accuracy under the widely used benchmarks.

Furthermore, since SPD is a general operation for point cloud generation, which is also simple and effective. We take a step forward and explore its performance in more tasks that are closely

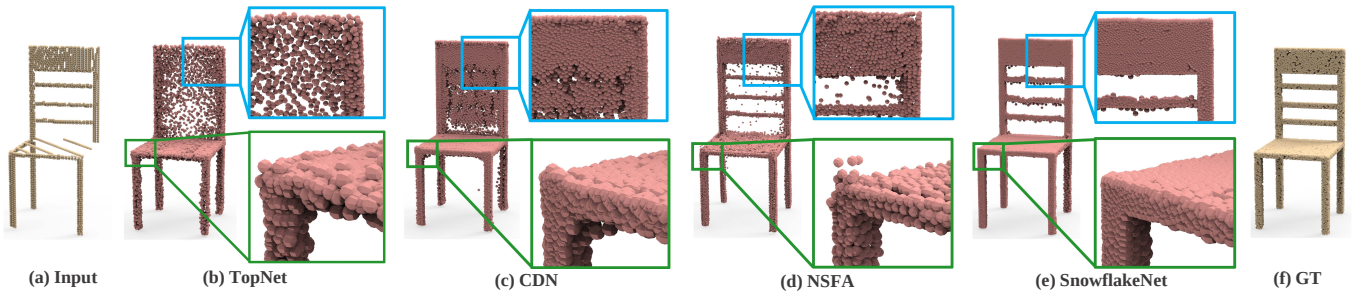


Fig. 1. Visual comparison of point cloud completion results. The input and ground truth have 2048 and 16384 points, respectively. Compared with current completion methods like TopNet [8], CDN [9] and NSFA [12], our SnowflakeNet can generate the complete shape (16384 points) with fine-grained geometric details, such as smooth regions (blue boxes), sharp edges and corners (green boxes).

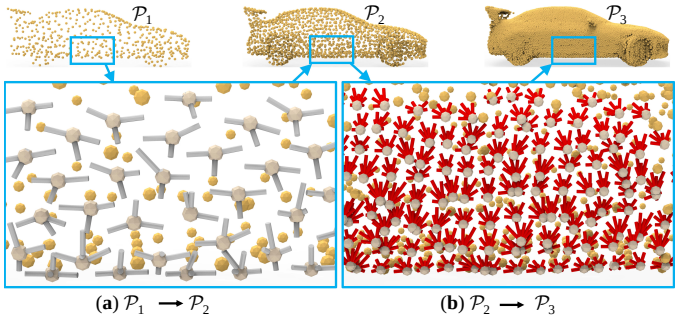


Fig. 2. Illustration of snowflake point deconvolution (SPD) for completing missing parts of a car. To show the local change more clearly, we only illustrate some sample points as parent points in the same patch and demonstrate their splitting paths for child points, which are marked as gray and red lines, respectively. (a) illustrates the SPD of point splitting from a coarse point cloud $\mathcal{P}_1$ (512 points) to its splitting $\mathcal{P}_2$ (2048 points). (b) illustrates the SPD of point splitting from $\mathcal{P}_2$ to dense complete point cloud $\mathcal{P}_3$ (16384 points), where the child points are expanding like the growth process of snowflakes.

related to point cloud generation, including point cloud auto-encoding, novel point cloud generation, single image reconstruction and point cloud upsampling. The generalization ability of SPD is demonstrated by quantitative and qualitative results in our experiments.

Our main contributions can be summarized as follows.

- We propose a novel SnowflakeNet for point cloud completion. Compared with previous methods without considering local generation patterns. SnowflakeNet can interpret point cloud completion as an explicit and structured local pattern generation which greatly improves the performance of 3D shape completion.
- We propose the novel Snowflake Point Deconvolution (SPD) for progressively increasing the number of points. It reformulates the generation of child points from parent points as a growing process of snowflake, where the shape characteristic embedded by the parent point features is extracted and inherited into the child points through a *point-wise splitting* operation.
- We introduce a novel skip-transformer to learn splitting patterns in SPD. It learns shape context and spatial relationship between child points and parent points. This process encourages SPD to produce locally structured and compact point arrangements, and captures the structure characteristic of 3D surface in local patches.
- In addition to point cloud completion, we further generalize SPD to more tasks related to point cloud generation. With a few network arrangements, SPD can be well applied to multiple point cloud generation scenarios. Comprehensive experiments are conducted to verify the effectiveness and generation ability of SPD.

2 Related Work

Point cloud completion methods can be roughly divided into two categories. (1) Traditional point cloud completion methods [15], [16], [17], [18] usually assume a smooth surface of 3D shape, or utilize large-scaled complete shape dataset to infer the missing regions for incomplete shapes. (2) Deep learning based methods [9], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], however, learn to predict a complete shape based on the learned prior from the training data. Our method falls into the second category and focuses on the decoding process of point cloud completion. We briefly review deep learning based methods below.

2.1 Point cloud completion by folding-based decoding

The development of deep learning based 3D point cloud processing techniques [45], [46], [47], [48], [49], [50], [51], [52], [53], [54] have boosted the research of point cloud completion. Suffering from the discrete nature of point cloud data, the generation of high-quality complete shape is one of the major concerns in the point cloud completion research. One of the pioneering work is the FoldingNet [11]. It was proposed for point cloud generation rather than completion. It includes a two-stage generation process and combines with the assumption that 3D object lies on 2D-manifold [8]. Following the similar practice, methods like SA-Net [5] further extended such generation process into multiple stages by proposing hierarchical folding in decoder. However, the problem of these folding-based methods [5], [11], [55], [56] is that the 3-dimensional code generated by intermediate layer of network is an implicit representation of target shape, which is hardly interpreted or constrained in order to help refine the shape in local region. On the other hand, TopNet [8] modeled the point cloud generation process as the growth of rooted tree, where one parent point feature is projected into several child point features in a feature expansion layer. Same as FoldingNet [11], the intermediate generation processes of TopNet and SA-Net are also implicit, where the shape information is only represented by the point features, cannot be constrained or explained explicitly.

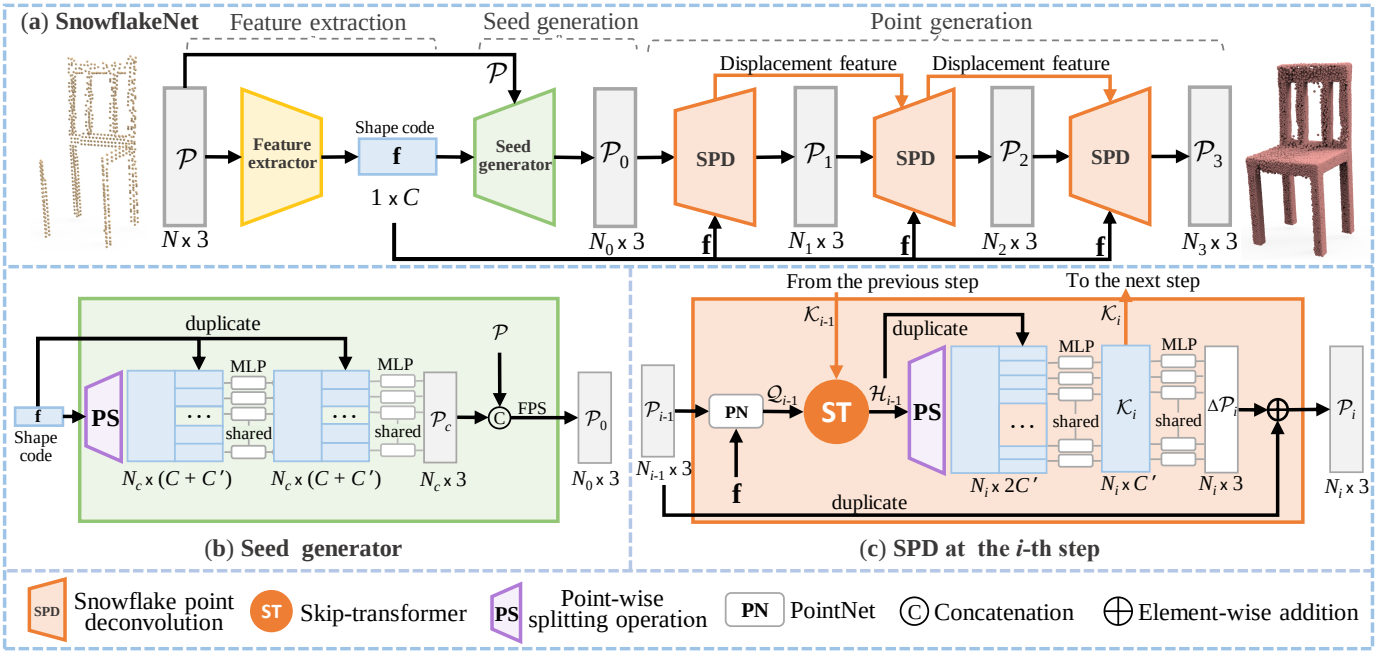


Fig. 3. (a) The overall architecture of SnowflakeNet, which consists of three modules: feature extraction, seed generation and point generation. (b) The details of seed generation module. (c) Snowflake point deconvolution (SPD). Note that N, N_c and N_i are the number of points, C and C' are the number of point feature channels that are 512 and 128, respectively.

2.2 Point cloud completion by coarse-to-fine decoding

Recently, explicit coarse-to-fine completion framework [10], [57], [58], [59], [60] has received an increasing attention, due to its explainable nature and controllable generation process. Typical methods like PCN [61] and NSFA [12] adopted the two-stage generation framework, where a coarse and low resolution point cloud is first generated by the decoder, and then a lifting module is used to increase the density of point clouds. Such kind of methods can achieve better performance since it can impose more constraints on the generation process of point cloud, i.e. the coarse one and the dense one. Followers like CDN [9] and PF-Net [19] further extended the number of generation stages and achieved the currently state-of-the-art performance. Although intriguing performance has been achieved by the studies along this line, most of these methods still cannot preserve a locally structured point splitting pattern, as illustrated in Fig. 1. The biggest problem is that these methods only focus on the expansion of point number and the reconstruction of global shape, while ignoring to preserve a well-structured generation process for points in local regions. This makes these methods difficult to capture local detailed geometries and structures of 3D shape.

Compared with the above-mentioned methods, our SnowflakeNet takes one step further to explore an explicit, explainable and locally structured solution for the generation of complete point clouds. SnowflakeNet models the progressive generation of point clouds as a hierarchical rooted tree structure like TopNet, while keeping the process explainable and explicit like CDN [9] and PF-Net [19]. Moreover, it excels the predecessors by arranging the point splitting in local regions in a locally structured pattern, which enables to precisely capture the detailed geometries and structures of 3D shapes.

2.3 Relation to transformer

Transformer [62] was initially proposed for encoding sentence in natural language processing, and soon gets popular in the research of 2D computer vision (CV) [63], [64]. Then, the success of transformer-based 2D CV studies have drawn the attention of 3D point cloud research, where pioneering studies like Point-Transformer [65], PCT [66] and Pointformer [67] have introduced such framework in the encoding process of point clouds to learn the representation. In our work, instead of only utilizing its representation learning ability, we further extend the application of transformer-based structure into the decoding process of point cloud completion, and reveal its ability for generating high quality 3D shapes through the proposed skip-transformer.

3 SnowflakeNet

The overall architecture of SnowflakeNet is shown in Fig. 3 (a), which consists of three modules: feature extraction, seed generation and point generation. We will detail each module in the following.

3.1 Overview

Feature extraction module. Let $\mathcal{P} = \{\mathbf{p}_j\}$ with a size of $N \times 3$ be an input point cloud, where N is the number of points and each point $\mathbf{p}_j$ indicates a 3D coordinate. The feature extractor aims to extract a shape code $\mathbf{f}$ with a size of $1 \times C$, which captures the global structure and detailed local pattern of the target shape. To achieve this, we adopt three layers of set abstraction from [68] to aggregate point features from local to global, along which point transformer [65] is applied to incorporate local shape context.

Seed generation module. The objective of the seed generator is to produce a coarse but complete point cloud $\mathcal{P}_0$ with a size of $N_0 \times 3$ that captures the geometry and structure of the target shape. As shown in Fig. 3 (b), with the extracted shape code $\mathbf{f}$, the

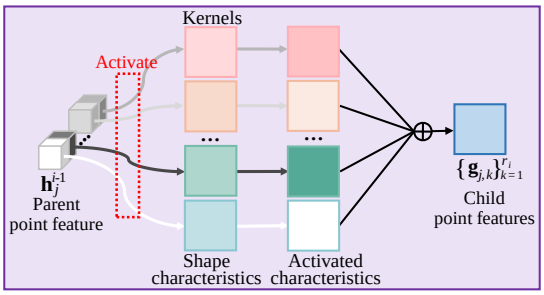


Fig. 4. The point-wise splitting operation. The cubes are logits of the parent point feature that represent activation status of the corresponding shape characteristics (Kernels), and child point features are obtained by adding activated shape characteristics.

seed generator first produces point features that capture both the existing and missing shape through point-wise splitting operation. Next, the per-point features are integrated with the shape code through multi-layer perceptron (MLP) to generate a coarse point cloud $\mathcal{P}_c$ of size $N_c \times 3$. Then, following the previous method [9], $\mathcal{P}_c$ is merged with the input point cloud $\mathcal{P}$ by concatenation, and then the merged point cloud is down-sampled to $\mathcal{P}_0$ through farthest point sampling (FPS) [68]. In this paper, we typically set $N_c = 256$ and $N_0 = 512$, where a sparse point cloud $\mathcal{P}_0$ suffices for representing the underlying shape. $\mathcal{P}_0$ will serve as the seed point cloud for point generation module.

Point generation module. The point generation module consists of three steps of Snowflake Point Deconvolution (SPD), each of which takes the point cloud from the previous step as input and splits it by upsampling factors (denoted by r_1, r_2 and r_3) to obtain $\mathcal{P}_1, \mathcal{P}_2$ and $\mathcal{P}_3$, which have the point sizes of $N_1 \times 3$, $N_2 \times 3$ and $N_3 \times 3$, respectively. SPDs collaborate with each other to generate a rooted tree structure that complies with local pattern for every seed point. The structure of SPD is detailed below.

3.2 Snowflake Point Deconvolution (SPD)

The SPD aims to increase the number of points by splitting each parent point into multiple child points, which can be achieved by first duplicating the parent points and then adding variations. Existing methods [9], [12], [61] usually adopt the folding-based strategy [11] to obtain the variations, which are used for learning different displacements for the duplicated points. However, the folding operation samples the same 2D grids for each parent point, which ignores the local shape characteristics contained in the parent point. Different from the folding-based methods [9], [12], [61], the SPD obtains variations through a *point-wise splitting* operation, which fully leverages the geometric information in parent points and adds variations that comply with local patterns. In order to progressively generate the split points, three SPDs are used in point generation module. In addition, to facilitate consecutive SPDs to split points in a coherent manner, we propose a novel skip-transformer to capture the shape context and the spatial relationship between the parent points and their split points.

Fig. 3 (c) illustrates the structure of the i-th SPD with upsampling factor r_i. We denote a set of parent points obtained from previous step as $\mathcal{P}_{i-1} = \{\mathbf{p}_j^{i-1}\}_{j=1}^{N_{i-1}}$. We split the parent points in $\mathcal{P}_{i-1}$ by duplicating them r_i times to generate a set of child points $\hat{\mathcal{P}}_i$, and then spread $\hat{\mathcal{P}}_i$ to the neighborhood of the parent points. To achieve this, we take the inspiration from [61] to predict the *point displacement* $\Delta\mathcal{P}_i$ of $\hat{\mathcal{P}}_i$. Then, $\hat{\mathcal{P}}_i$ is updated as $\mathcal{P}_i = \hat{\mathcal{P}}_i + \Delta\mathcal{P}_i$, where $\mathcal{P}_i$ is the output of the i-th SPD.

In detail, taking the shape code $\mathbf{f}$ from feature extraction, the SPD first extracts the per-point feature $\mathcal{Q}_{i-1} = \{\mathbf{q}_j^{i-1}\}_{j=1}^{N_{i-1}}$ for $\mathcal{P}_{i-1}$ by adopting the basic PointNet [69] framework. Then, $\mathcal{Q}_{i-1}$ is sent to the skip-transformer to learn the *shape context feature*, denoted as $\mathcal{H}_{i-1} = \{\mathbf{h}_j^{i-1}\}_{j=1}^{N_{i-1}}$. Next, $\mathcal{H}_{i-1}$ is upsampled by *point-wise splitting* operation and duplication, respectively, where the former serves to add variations and the latter preserves shape context information. Finally, the upsampled feature with size of $N_i \times 2C'$ is fed to MLP to produce the *displacement feature* $\mathcal{K}_i = \{\mathbf{k}_j^i\}_{j=1}^{N_i}$ of current step. Here, $\mathcal{K}_i$ is used for generating the point displacement $\Delta\mathcal{P}_i$, and will be fed into the next SPD. $\Delta\mathcal{P}_i$ is formulated as

$$\Delta\mathcal{P}_i = \tanh(\mathrm{MLP}(\mathcal{K}_i)), \tag{1}$$

where $\tanh$ is the hyper-tangent activation.

Point-wise splitting operation. point-wise splitting operation aims to generate multiple child point features for each $\mathbf{h}_j^{i-1} \in \mathcal{H}_{i-1}$. Fig. 4 shows this operation structure used in the i-th SPD (see Fig. 3 (c)). It is a special one-dimensional deconvolution strategy, where the kernel size and stride are both equal to r_i. In practice, each $\mathbf{h}_j^{i-1} \in \mathcal{H}_{i-1}$ shares the same set of kernels, and produces multiple child point features in a point-wise manner. To be clear, we denote the m-th logit of $\mathbf{h}_j^{i-1}$ as $h_{j,m}^{i-1}$, and its corresponding kernel is indicated by K_m. Technically, K_m is a matrix with a size of $r_i \times \mathrm{C}'$, the k-th row of K_m is denoted as $\mathbf{k}_{m,k}$, and the k-th child point feature $\mathbf{g}_{j,k}$ is given by

$$\mathbf{g}_{j,k} = \sum_m h_{j,m}^{i-1} \mathbf{k}_{m,k}. \tag{2}$$

In addition, in Fig. 4, we assume that each learnable kernel K_m indicates a certain shape characteristic, which describes the geometry and structure of 3D shape in local region. Correspondingly, every logit $h_{j,m}^{i-1}$ indicates the activation status of the m-th shape characteristic. The child point features can be generated by adding the activated shape characteristics. Moreover, the point-wise splitting operation is flexible for upsampling points. For example, when $r_i = 1$, it enables the SPD to move the point from previous step to a better position; when $r_i > 1$, it serves to expand the number of points by a factor of r_i.

Collaboration between SPDs. In Fig. 3 (a), we adopt three SPDs to generate the complete point cloud. We first set the upsampling factor $r_1 = 1$ to explicitly rearrange seed point positions. Then, we set $r_2 > 1$ and $r_3 > 1$ to generate a structured tree for every point in $\mathcal{P}_1$. Collaboration between SPDs is crucial for growing the tree in a coherent manner, because information from the previous splitting can be used to guide the current one. Besides, the growth of the rooted trees should also capture the pattern of local patches to avoid overlapping with each other. To achieve this purpose, we propose a novel *skip-transformer* to serve as the cooperation unit between SPDs. In Fig. 5, the skip-transformer takes per-point feature $\mathbf{q}_j^{i-1}$ as input, and combines it with displacement feature $\mathbf{k}_j^{i-1}$ from previous step to produce the shape context feature $\mathbf{h}_j^{i-1}$, which is given by

$$\mathbf{h}_j^{i-1} = \mathrm{ST}(\mathbf{k}_j^{i-1}, \mathbf{q}_j^{i-1}), \tag{3}$$

where ST denotes the skip-transformer. The detailed structure is described as follows.

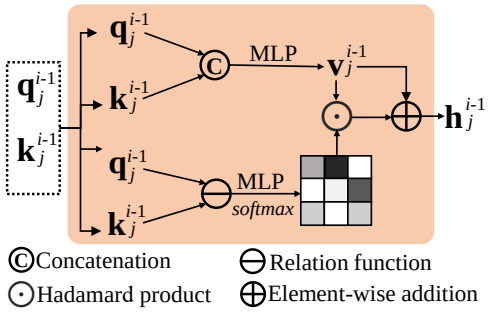


Fig. 5. The detailed structure of skip-transformer.

3.3 Skip-Transformer

Fig. 5 shows the structure of skip-transformer. The skip-transformer is introduced to learn and refine the spatial context between parent points and their child points, where the term "skip" represents the connection between the displacement feature from the previous layer and the point feature of the current layer.

Given per-point feature $\mathbf{q}_j^{i-1}$ and displacement feature $\mathbf{k}_j^{i-1}$, the skip-transformer first concatenates them. Then, the concatenated feature is fed to MLP, which generates the vector $\mathbf{v}_j^{i-1}$. Here, $\mathbf{v}_j^{i-1}$ serves as the value vector which incorporates previous point splitting information. In order to further aggregate local shape context into $\mathbf{v}_j^{i-1}$, the skip-transformer uses $\mathbf{q}_j^{i-1}$ as the query and $\mathbf{k}_j^{i-1}$ as the key to estimate attention vector $\mathbf{a}_j^{i-1}$, where $\mathbf{a}_j^{i-1}$ denotes how much attention the current splitting should pay to the previous one. To enable the skip-transformer to concentrate on local patterns, we calculate attention vectors between each point and its k-nearest neighbors (k-NN). The k-NN strategy also helps to reduce computation cost. Specifically, given the j-th point feature $\mathbf{q}_j^{i-1}$, the attention vector $\mathbf{a}_{j,l}^{i-1}$ between $\mathbf{q}_j^{i-1}$ and displacement features of the k-nearest neighbors $\{\mathbf{k}_{j,l}^{i-1} | l = 1, 2, \ldots, k\}$ can be calculated as

$$\mathbf{a}_{j,l}^{i-1} = \frac{\exp(\mathrm{MLP}((\mathbf{q}_j^{i-1}) \ominus (\mathbf{k}_{j,l}^{i-1})))}{\sum_{l=1}^{k} \exp(\mathrm{MLP}((\mathbf{q}_j^{i-1}) \ominus (\mathbf{k}_{j,l}^{i-1})))}, \tag{4}$$

where $\ominus$ serves as the relation operation, i.e. element-wise subtraction. Finally, the shape context feature $\mathbf{h}_j^{i-1}$ can be obtained by

$$\mathbf{h}_j^{i-1} = \mathbf{v}_j^{i-1} \oplus \sum_{l=1}^{k} \mathbf{a}_{j,l}^{i-1} \odot \mathbf{v}_{j,l}^{i-1}, \tag{5}$$

where $\oplus$ denotes element-wise addition and $\odot$ is Hadamard product. Note that there is no previous displacement feature for the first SPD, of which the skip-transformer takes $\mathbf{q}_j^0$ as both query and key.

3.4 Training Loss

3.4.1 Completion Loss

We leverage Chamfer distance (CD) as the primary loss function. The L_2 version of Chamfer distance (CD) is defined as

$$\mathcal{L}_{\mathrm{CD}}(\mathcal{X}, \mathcal{Y}) = \sum_{\mathbf{x} \in \mathcal{X}} \min_{\mathbf{y} \in \mathcal{Y}} \|\mathbf{x} - \mathbf{y}\|_2 + \sum_{\mathbf{y} \in \mathcal{Y}} \min_{\mathbf{x} \in \mathcal{X}} \|\mathbf{y} - \mathbf{x}\|_2, \tag{6}$$

where $\mathcal{X}$ and $\mathcal{Y}$ are point sets, $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ are point coordinates, respectively. The L_1 version of CD replaces L_2-norm in Eq. (6) with L_1-norm and divide by 2. To explicitly constrain point clouds generated in the seed generation and the subsequent splitting process, we down-sample the ground truth point clouds to the same sampling density as $\{\mathcal{P}_c, \mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3\}$ (see Fig. 3). We define the sum of the four CD losses as the *completion loss*, denoted as

$$\mathcal{L}_{completion} = \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_c, \mathcal{P}_c^{'}) + \sum_{i=1}^{3} \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_i, \mathcal{P}_i^{'}), \tag{7}$$

where $\mathcal{P}_i^{'}$ and $\mathcal{P}_c^{'}$ denote the down-sampled ground truth point clouds that have the same point number as point cloud $\mathcal{P}_i$ and $\mathcal{P}_c$, respectively.

3.4.2 Preservation loss

We exploit the *partial matching loss* from [7] to preserve the shape structure of the incomplete point cloud, which is defined as

$$\mathcal{L}_{partial}(\mathcal{X}, \mathcal{Y}) = \sum_{\mathbf{x} \in \mathcal{X}} \min_{\mathbf{y} \in \mathcal{Y}} \|\mathbf{x} - \mathbf{y}\|_2. \tag{8}$$

The partial matching loss is an unidirectional constraint which aims to match one shape to the other without constraining the opposite direction. Because the partial matching loss only requires the output point cloud to partially match the input, we take it as the *preservation loss* $\mathcal{L}_{\mathrm{preservation}}$, and the total training loss is formulated as

$$\mathcal{L} = \mathcal{L}_{\mathrm{completion}} + \lambda \mathcal{L}_{\mathrm{preservation}}, \tag{9}$$

where we typically set $\lambda = 1$.

4 Experiments

To fully prove the effectiveness of our SnowflakeNet, we first conduct comprehensive experiments under two widely used benchmarks: PCN [61] and Completion3D [8], both of which are subsets of the ShapeNet dataset. The experiments demonstrate that our method has superiority over the state-of-the-art point cloud completion methods. And then, we evaluate our method in other point cloud generation tasks including point cloud auto-encoding, generation, single image reconstruction and upsampling.

4.1 Evaluation on PCN Dataset

4.1.1 Dataset briefs and evaluation metric

The *PCN* dataset [61] is a subset with 8 categories derived from ShapeNet dataset [70]. The incomplete shapes are generated by back-projecting complete shapes into 8 different partial views. For each complete shape, 16384 points are evenly sampled from the shape surface. In order to align with the groundtruth shapes, we typically set $r_1 = 1$, $r_2 = 4$ and $r_3 = 8$ to generate complete point clouds with 16384 points. We follow the same split settings with PCN [61] to fairly compare our SnowflakeNet with other methods. For evaluation, we adopt the L_1 version of Chamfer distance, which follows the same practice as previous methods [61].

4.1.2 Quantitative comparison

Table 1 shows the results of our SnowflakeNet and other completion methods on PCN dataset, from which we can find that SnowflakeNet achieves the best performance over all counterparts. Especially, compared with the result of the second-ranked NSFA [12], SnowflakeNet reduces the average CD by 0.85, which is 10.5% lower than the NSFA's results (8.06 in terms of average CD). Moreover, SnowflakeNet also achieves the best results on all

TABLE 1
Point cloud completion on PCN dataset in terms of per-point L_1 Chamfer distance $\times 10^3$ (lower is better).

Methods	Average	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Boat
FoldingNet [11]	14.31	9.49	15.80	12.61	15.55	16.41	15.97	13.65	14.99
TopNet [8]	12.15	7.61	13.31	10.90	13.82	14.44	14.78	11.22	11.12
AtlasNet [71]	10.85	6.37	11.94	10.10	12.06	12.37	12.99	10.33	10.61
PCN [61]	9.64	5.50	22.70	10.63	8.70	11.00	11.34	11.68	8.59
GRNet [10]	8.83	6.45	10.37	9.45	9.41	7.96	10.51	8.44	8.04
CDN [9]	8.51	4.79	9.97	8.31	9.49	8.94	10.69	7.81	8.05
PMP-Net [6]	8.73	5.65	11.24	9.64	9.51	6.95	10.83	8.72	7.25
NSFA [12]	8.06	4.76	10.18	8.63	8.53	7.03	10.53	7.35	7.48
PoinTr [60]	8.38	4.75	10.47	8.68	9.39	7.75	10.93	7.78	7.29
Ours	**7.21**	**4.29**	**9.16**	**8.08**	**7.89**	**6.07**	**9.23**	**6.55**	**6.40**

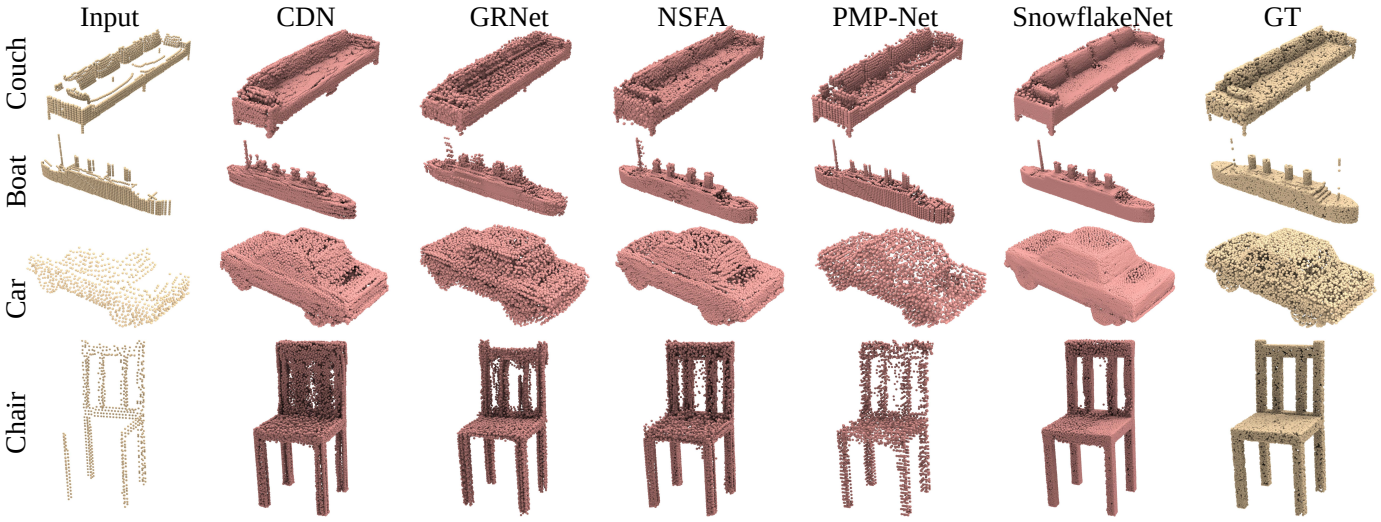


Fig. 6. Visual comparison of point cloud completion on PCN dataset. Our SnowflakeNet can produce smoother surfaces (e.g. car) and more detailed structures (e.g. chair back) compared with the other state-of-the-art point cloud completion methods.

Fig. 7. Visualization of more completion results using our SnowflakeNet on PCN dataset.

categories in terms of CD, which proves the robust generalization ability of SnowflakeNet for completing shapes across different categories. In Table 1, both CDN [9] and NSFA [12] are typical point cloud completion methods, which adopt a coarse-to-fine shape decoding strategy and model the generation of points as a hierarchical rooted tree. Compared with these two methods, our SnowflakeNet also adopts the same decoding strategy but achieves much better results on PCN dataset. Therefore, the improvements should credit to the proposed SPD layers and skip-transformer in SnowflakeNet, which helps to generate points in local regions with a locally structured pattern.

4.1.3 *Visual comparison*

We typically choose top four point cloud completion methods from Table 1, and visually compare SnowflakeNet with these methods in Fig. 6. The visual results show that SnowflakeNet can predict the complete point clouds with a much better shape quality. For example, in the car category, the point distribution on the car's boundary generated by SnowflakeNet is smoother and more uniform than other methods. As for the chair category,

TABLE 2
Point cloud completion on Completion3D in terms of per-point L_2 Chamfer distance $\times 10^4$ (lower is better).

Methods	Average	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Boat
FoldingNet [11]	19.07	12.83	23.01	14.88	25.69	21.79	21.31	20.71	11.51
PCN [61]	18.22	9.79	22.70	12.43	25.14	22.72	20.26	20.27	11.73
PointSetVoting [72]	18.18	6.88	21.18	15.78	22.54	18.78	28.39	19.96	11.16
AtlasNet [71]	17.77	10.36	23.40	13.40	24.16	20.24	20.82	17.52	11.62
SoftPoolNet [73]	16.15	5.81	24.53	11.35	23.63	18.54	20.34	16.89	7.14
TopNet [8]	14.25	7.32	18.77	12.88	19.82	14.60	16.29	14.89	8.82
SA-Net [5]	11.22	5.27	14.45	7.78	13.67	13.53	14.22	11.75	8.84
GRNet [10]	10.64	6.13	16.90	8.27	12.23	10.22	14.93	10.08	5.86
PMP-Net [6]	9.23	3.99	14.70	8.55	10.21	9.27	12.43	8.51	5.77
VRCNet [58]	8.12	3.94	**10.93**	**6.44**	9.32	8.32	11.35	8.60	5.78
VE-PCN [25]	8.10	3.83	12.74	7.86	**8.66**	**7.24**	11.47	7.88	**4.75**
Ours	**7.60**	**3.48**	11.09	6.90	8.75	8.42	**10.15**	**6.46**	5.32

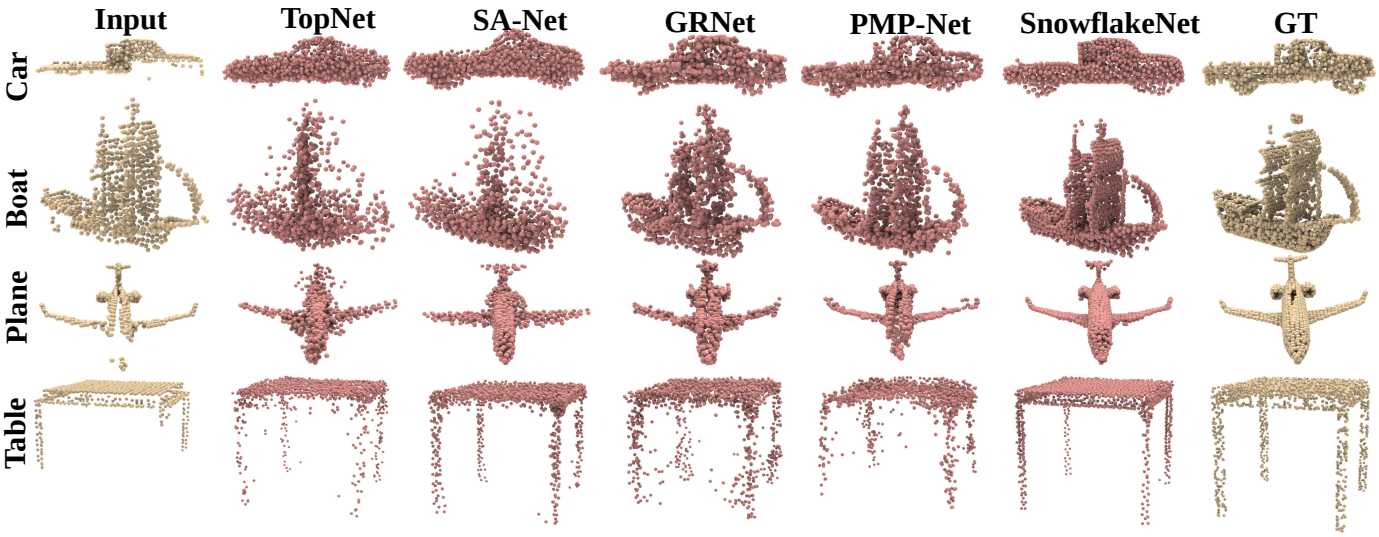


Fig. 8. Visual comparison of point cloud completion on Completion3D dataset. Our SnowflakeNet can produce smoother surfaces (e.g. car and table) and more detailed structures compared with the other state-of-the-art point cloud completion methods.

Fig. 9. Visualization of more completion results using our SnowflakeNet on Completion3D dataset.

SnowflakeNet can predict more detailed and clear structure of the chair back compared with the other methods, where CDN [9] almost fails to preserve the basic structure of the chair back, while the other methods generate lots of noise between the columns of the chair back. In Fig. 7, we additionally present more visual completion results on the PCN dataset, which further demonstrates the completion quality of our method. Especially, SnowflakeNet is able to reveal complex local structures like the back cushion of the couch and table rungs.

4.2 Evaluation on Completion3D Dataset

4.2.1 *Dataset briefs and evaluation metric*

The Completion3D dataset contains 30958 models from 8 categories, of which both partial and ground truth point clouds have 2048 points, here we set $r_1 = 1$, $r_2 = 2$ and $r_3 = 2$ to generate complete point clouds with 2048 points. We follow the

same train/validation/test split of Completion3D to have a fair comparison with the other methods, where the training set contains 28974 models, validation and testing set contain 800 and 1184 models, respectively. For evaluation, we adopt the L_2 version of Chamfer distance on testing set to align with previous studies.

4.2.2 Quantitative comparison

In Table 2, we show the quantitative results of our SnowflakeNet and the other methods on Completion3D dataset. All results are cited from the online public leaderboard of Completion3D[1]. From Table 2, we can find that our SnowflakeNet achieves the best results over all methods listed on the leaderboard. Especially, compared with the latest point cloud completion method like VE-PCN [25], SnowflakeNet can reduce the average CD by 0.50, which is 6.1% lower than the VE-PCN (8.10 in terms of average CD). Meanwhile, SnowflakeNet also outperforms the other methods in multiple categories in terms of per-category CD. Especially in the table category, SnowflakeNet reduces the per-category CD by 1.42 compared with the second-ranked result of VE-PCN. Compared with PCN dataset, the point cloud in Completion3D dataset is much sparser and easier to generate. Therefore, a coarse-to-fine decoding strategy may have less advantages over the other methods. Despite of this, our SnowflakeNet still achieves superior performance over folding-based methods including SA-Net [5] and FoldingNet [11], and we are also the best among the coarse-to-fine methods including TopNet [8] and GRNet [10]. In all, the results on Completion3D dataset demonstrate the capability of SnowflakeNet on predicting high-quality complete shapes using sparse point clouds.

4.2.3 Visual comparison

Same as the practice in PCN dataset, we also visually compare SnowflakeNet with the top four methods in Table 2. Visual comparison in Fig. 8 demonstrates that our SnowflakeNet also achieves much better visual results than the other counterparts on a sparse point cloud completion task. Especially, in plane category, SnowflakeNet predicts the complete plane which is almost the same as the ground truth, while the other methods fail to reveal the complete plane in details. The same conclusion can also be drawn from the observation of car category. In the table and boat categories, SnowflakeNet produces more detailed structures compared with the other methods, e.g. the sails of the boat and the legs of the table. And in Fig. 9, we also visualize more completion results on the Completion3D dataset, where the completed point clouds are highly similar to the ground truth shapes. Especially in the car and boat categories, SnowflakeNet can generate complete point clouds that are even more uniform than the ground truth shapes, this should credit to the skip-transformer in SPD.

4.2.4 Extension to ScanNet chairs

To evaluate the generalization ability of SnowflakeNet on real world scenario, we use the pre-trained model of SnowflakeNet on Completion3D dataset and evaluate its performance on the chair instances in ScanNet dataset without fine-tuning. We compare SnowflakeNet with GRNet [10] and PMP-Net [6] and also use their pre-trained model for testing. The visual comparison is shown in Fig. 10. Even for the shapes that are highly noisy and incomplete, SnowflakeNet completes shapes with less noise than GRNet and PMP-Net, which benefits from the coherent multi-step splittings conducted by consecutive SPDs. And in Fig. 11, we provide more visualization results on ScanNet chairs in the same scene.

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

4.3 Ablation studies

We analyze the effectiveness of each part of SnowflakeNet. For convenience, we conduct all experiments on the validation set of Completion3D dataset. By default, all the experiment settings remain the same as Section 4.2.

4.3.1 Effect of skip-transformer

To evaluate the effectiveness of skip-transformer used in SnowflakeNet, we develop three network variations as follows. (1) The *Self-att* variation replaces the transformer mechanism in skip-transformer with the self-attention mechanism, where the input is the point features of current layer. (2) The *No-att* variation removes the transformer mechanism from skip-transformer, where the features from the previous layer of SPD is directly added to the feature of current SPD layer. (3) The *No-connect* variation removes the whole skip-transformer from the SPD layers, and thus, no feature connection is established between the SPD layers. The experiment results are shown in Table 3. In addition, we denote the original version of SnowflakeNet as *Full* for clear comparison with the performance of each network variations. From Table 3, we can find that the transformer-based Full model achieves the best performance among all compare network variations. The comparison between the No-connect model and the Full model justifies the advantage of skip-transformer between SPD layers, and the comparison between No-att model and Full model further proves the effectiveness of transformer mechanism to learn shape context in local regions. Moreover, the comparison between Self-att model and No-att model shows that the attention based mechanism can also contribute to the completion performance.

TABLE 3
Effect of skip-transformer.

Methods	avg.	Couch	Chair	Car	Lamp
Self-att	8.89	6.04	10.9	9.42	9.12
No-att	9.30	6.15	11.2	10.4	9.38
No-connect	9.39	6.17	11.3	10.5	9.51
Full	**8.48**	**5.89**	**10.6**	**9.32**	**8.12**

4.3.2 Effect of each part in SnowflakeNet

To evaluate the effectiveness of each part in SnowflakeNet, we design four different network variations as follows. (1) The *Folding-expansion* variation replaces the point-wise splitting operation with the folding-based feature expansion method [11], where the features are duplicated several times and concatenated with a 2-dimensional codeword, in order to increase the number of point features. (2) The $\mathrm{E}_{\mathrm{PCN}}$+*SPD* variation employs the PCN encoder and our SnowflakeNet decoder. (3) The *w/o partial matching* variation removes the partial matching loss. (4) The *PCN-baseline* is the performance of original PCN method [61], which is trained and evaluated under the same settings of our ablation study. In Table 4, we report the results of the four network variations along with the default network denoted as *Full*. By comparing $\mathrm{E}_{\mathrm{PCN}}$+SPD with PCN-baseline, we can find that our SPD with skip-transformer based decoder can be potentially applied to other simple encoders, and achieves significant improvement. By

Fig. 10. Visual comparison of the chairs in ScanNet dataset.

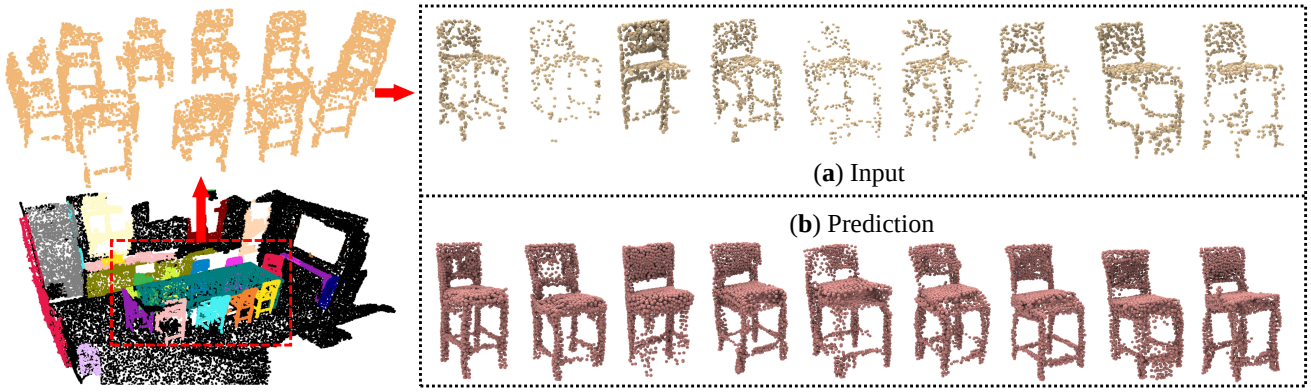


Fig. 11. More visualization of completion results on ScanNet chairs.

TABLE 4
Effect of each part in SnowflakeNet.

Methods	avg.	Couch	Chair	Car	Lamp
Folding-expansion	8.80	8.40	10.80	5.83	10.10
E_{PCN}+SPD	8.93	9.06	11.30	6.14	9.23
w/o partial matching	8.50	8.72	**10.6**	**5.78**	**8.9**
PCN-baseline	13.30	11.50	17.00	6.55	18.20
Full	**8.48**	**8.12**	10.6	5.89	9.32

TABLE 5
Effect of splitting strategy.

Splitting strategy	avg.	Couch	Chair	Car	Lamp
one-step	9.53	6.03	11.0	9.94	9.26
two-steps	8.64	6.00	10.3	9.62	8.66
baseline	**8.48**	**5.89**	10.6	**9.32**	**8.12**
three-steps	8.66	6.04	**10.0**	9.53	9.02

comparing the Folding-expansion with the Full model, the better performance of the Full model proves the advantage of point-wise splitting operation over the folding-based feature expansion methods. By comparing w/o partial matching with the Full model, we find that partial matching loss can slightly improve the average performance of SnowflakeNet.

4.3.3 *The effect of splitting strategy*

The SPD is flexible for increasing the point number. When $r_i = 1$ (r_i is the upsampling factor of the i-th SPD), it serves to move the point from previous step to a better position; when $r_i > 0$, it splits a point by a factor of r_i. In this section, we analyze the influence of different point splitting strategies on the performance of SnowflakeNet, and the other experiment settings are the same as the ablation study. As shown in Table 5, we additionally test three different splitting strategies. The one-step splitting adopts a single SPD (upsampling factor is 4) to split $\mathcal{P}_0$ (512 points) to $\mathcal{P}_1$ (2048 points). The two-step splitting adopts two SPDs (upsampling factors are both equal to 2) and produces three point clouds $\mathcal{P}_0, \mathcal{P}_1$ and $\mathcal{P}_2$ with a size of $512 \times 3, 1024 \times 3$ and 2048×3, respectively. For the three-step strategy (three SPDs of upsampling factor 2), we particularly set $N_c = N_0 = 256$

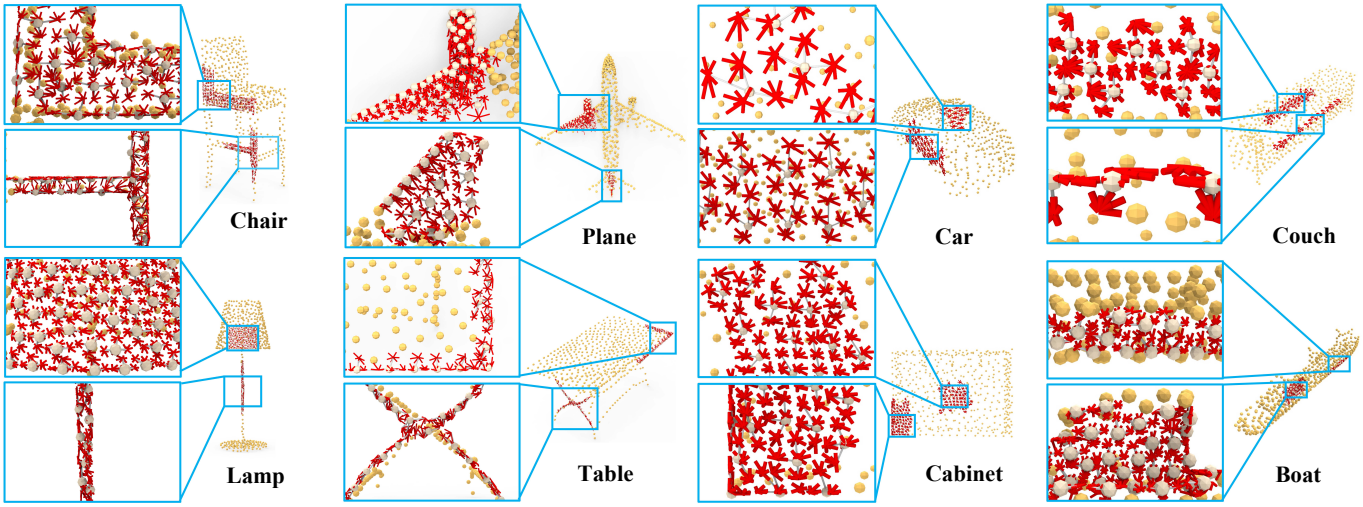


Fig. 12. Visualization of snowflake point deconvolution on different objects. For each object, we sample two patches of points and visualize two layers of point splitting together for each sampled point. The gray lines indicate the paths of the point splitting from $\mathcal{P}_1$ to $\mathcal{P}_2$, and the red lines are splitting paths from $\mathcal{P}_2$ to $\mathcal{P}_3$.

TABLE 6
Comparison of point cloud auto-encoding in terms CD and EMD on ShapeNet. L_2 CD and EMD are multiplied by 10^4 and 10^2, respectively.

Dataset	Metric	Atlas (S1) [71]	Altas (P25) [71]	PointFlow [74]	ShapeGF [75]	DPM [76]	Ours
Airplane	CD	2.00	1.80	2.42	2.10	2.19	**1.51**
	EMD	4.31	4.37	3.31	3.50	2.90	**2.44**
Car	CD	6.91	6.50	5.83	5.47	5.49	**4.68**
	EMD	5.67	5.41	4.39	4.49	3.94	**3.35**
Chair	CD	5.48	4.98	6.80	5.15	5.68	**4.58**
	EMD	5.56	5.28	5.01	4.78	4.15	**3.67**
ShapeNet	CD	5.87	5.42	7.55	5.73	5.25	**4.20**
	EMD	5.46	5.60	5.17	5.05	3.78	**3.44**

(see Section 3.1) so that it outputs 4 points clouds of the size $256 \times 3, 512 \times 3, 1024 \times 3$ and 2048×3. Note that the baseline is two-step splitting with an additional SPD (upsampling factor equals to 1), which serves to rearrange the initial point positions. By comparing one-step splitting with the other strategies, we can find that multiple steps of splitting boosts the performance of SnowflakeNet significantly (in terms of average CD), this should credit to the collaboration between SPDs. By comparing two-step splitting with three-step splitting, we can find that the two-step splitting suffices for generating a sparse point cloud with 2048 points (not necessarily for a dense one with more points), and extra SPDs may not improve the performance but will increase the computational burden. And by comparing the two-step splitting with the baseline, we can find that the additional SPD which adjusts the initial point positions can facilitate the point splitting process.

4.3.4 *Visualization of point generation process of SPD*

In Fig. 12, we visualize the point cloud generation proccess of SPD. We can find that the layers of SPD generate points in a snowflake-like pattern. When generating the smooth plane (e.g. chair and lamp in Fig. 12), we can clearly see the child points are generated around the parent points, and smoothly placed along the plane surface. On the other hand, when generating thin tubes and sharp edges, the child points can precisely capture the geometries.

4.4 Point Cloud Auto-Encoding

The task of point cloud auto-encoding aims to reconstruct a point cloud from its reduced and encoded representation. Since it heavily relies on the generation ability of the decoder, we use the decoding part of SnowflakeNet in point cloud auto-encoding to evaluate the generation ability of snowflake point deconvolution.

4.4.1 *Dataset and evaluation metric*

Dataset. To fairly evaluate the generation ability of SPD, we follow the same experimental settings of DPM [76] and conduct point cloud auto-encoding on the ShapeNet [70] dataset. The ShapeNet [70] dataset contains 51127 shapes from 55 categories, we use the same training, testing and evaluation split of DPM [76], where the ratio of training, testing and validation sets are 80%, 15% and 5%, respectively. Four datasets are used in evaluation, which include three categories (i.e. *car*, *airplane* and *chair*) in ShapeNet and the whole ShapeNet dataset.

Evaluation metric. The commmonly used Chamfer Distance (CD) and Earth Mover's Distance (EMD) are adopted as our evaluation metric.

4.4.2 *Network arrangement and training loss*

Network arrangement. The auto-encoding task aims to evaluate the reconstruction ability of our decoder, we follow DPM and replace our feature extractor with a simple PointNet [69] encoder, so that the encoding settings are the same with other methods.

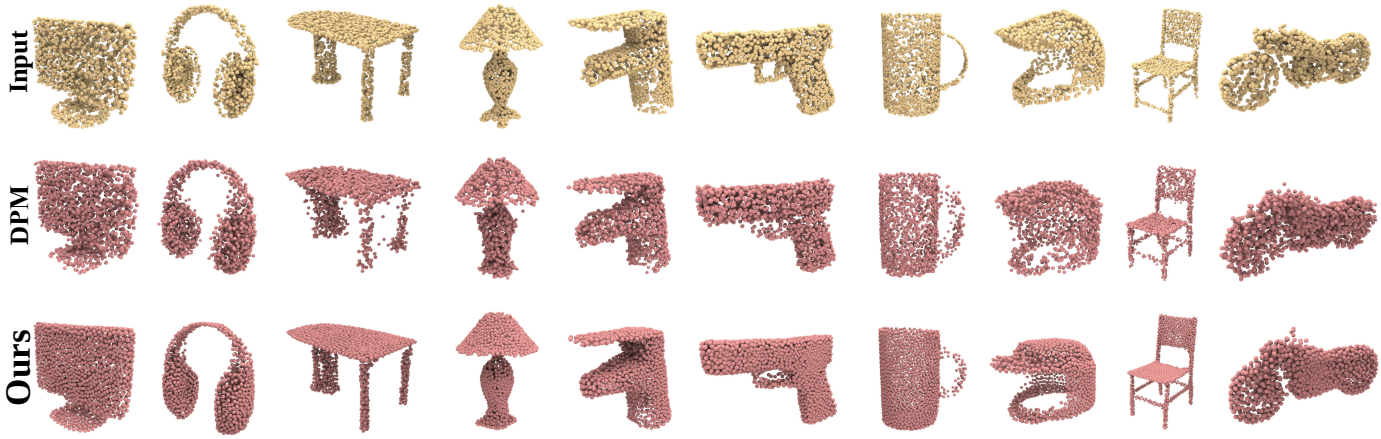


Fig. 13. Visual comparison of reconstructed point clouds between different auto-encoders.

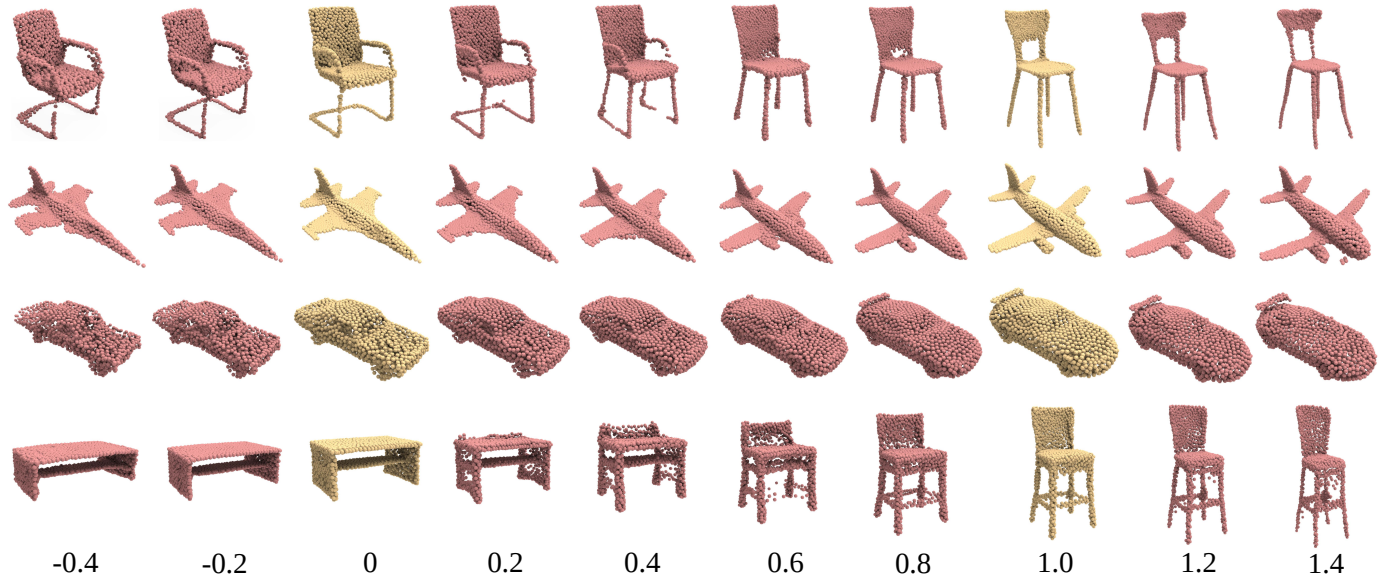


Fig. 14. Latent space interpolation and extrapolation, where the interpolation step is 0.2. The source shapes are in yellow (denoted as "0" and "1.0"); the interpolated and extrapolated shapes are in red.

Besides, the ground truth shapes used in auto-encoding have 2048 points, so we use our seed generator and two stacked SPDs (each of which has a upsampling factor of 2) as decoder. In the seed generator, we set $\mathbf{N}_c$ to 512 and take $\mathcal{P}_c$ as the output of the seed generator, where $\mathcal{P}_c$ and $\mathcal{P}_0$ are the same point cloud. Such arrangement makes sure that the generation of point cloud relies only on the latent code extracted from encoder.

Training loss. We use Chamfer Distance (CD) and Earth Mover's Distance (EMD) as our reconstruction loss, which is given as

$$\mathcal{L}_{recon} = \sum_{i \in \{0,2\}} \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_i, \mathcal{P}_i^{'}) + \mathcal{L}_{\mathrm{EMD}}(\mathcal{P}_i, \mathcal{P}_i^{'}), \qquad (10)$$

where $\mathcal{P}_i^{'}$ is the downsampled ground truth shape that has the same point density of $\mathcal{P}_i$.

4.4.3 Quantitative comparison

The quantitative comparison is given in Table 13. Even though only two SPDs are used in the decoder, our method still achieves the best performance among the compared counterparts in terms of both L_2 CD and EMD. Especially on the whole ShapeNet dataset, SnowflakeNet reduces the average CD by 1.05, which is 20% lower than DPM [76] (5.25 in terms of average CD). At the same time, our method also outperforms the other methods across all categories in terms of per-category CD and EMD. Therefore, the better results on point cloud auto-encoding fully demonstrate the generation ability of the decoder of our SnowflakeNet, especially snowflake point deconvolution.

4.4.4 Qualitative comparison

The qualitative comparison is shown in Fig. 13, we visually compare SnowflakeNet with state-of-the-art generation method DPM. From Fig. 13, we can find that SnowflakeNet is able to generate point clouds with more detailed geometric structures, such as the handle of the cup and the motor tires. Meanwhile, the points generated by SnowflakeNet are much more uniform and tend to cover the surface of the shape evenly, this should credit to the excellent generation ability of SPDs. In addition, we visualize the interpolation and extrapolation between latent

TABLE 7
Comparison of point cloud auto-encoding performance in terms L_2 CD and EMD. CD is multiplied by 10^4 and EMD is multiplied by 10^2.

		COV(%, ↑)		MMD(↓)		1-NNA(%, ↓)		JSD(↓)
Shape	Model	CD	EMD	CD	EMD	CD	EMD	-
Airplane	PC-GAN [77]	42.17	13.84	3.819	1.810	77.59	98.52	6.188
	GCN-GAN [78]	39.04	18.62	4.713	1.650	89.13	98.60	6.669
	TreeGAN [79]	39.37	8.40	4.323	1.953	83.86	99.67	15.646
	PointFlow [74]	44.98	44.65	3.688	1.090	66.39	69.36	1.536
	ShapeGF [75]	**50.41**	**47.12**	3.306	1.027	**61.94**	70.51	**1.059**
	DPM [76]	48.71	45.47	3.276	1.061	64.83	**75.12**	1.067
	Ours	46.29	44.98	**3.271**	**1.002**	69.35	73.15	1.630
Chair	PC-GAN [77]	46.23	22.14	13.436	3.104	69.67	100.00	6.649
	GCN-GAN [78]	39.84	35.09	15.354	2.213	77.86	95.80	21.708
	TreeGAN [79]	38.02	6.77	14.936	3.613	74.92	100.00	13.282
	PointFlow [74]	41.86	43.38	13.631	1.856	66.13	68.40	12.474
	ShapeGF [75]	48.53	46.71	13.175	1.785	**56.17**	**62.69**	**5.996**
	DPM [76]	**48.94**	**47.52**	**12.276**	**1.784**	60.11	69.06	7.797
	Ours	48.83	36.91	12.742	1.830	58.85	75.08	7.579

codes in Fig. 14, where the source shapes are in yellow, the interpolated and extrapolated shapes are in red. From Fig. 14 we can find that the output of SnowflakeNet is able to transit smoothly according to latent interpolation and extrapolation, even though the transition is between different categories (table and chair, the last row). Moreover, the interpolated and extrapolated shapes can still maintain a visually uniform point distribution, which shows the generation stability of our decoder.

4.5 Novel shape Generation

In addition to point cloud auto-encoding, we also apply our network to point cloud generation to demonstrate its ability for novel shape generation.

4.5.1 Dataset and evaluation metric

Dataset. For the task of point cloud generation, we follow DPM [76] and conduct experiments on the ShapeNet dataset. We quantitatively compare our method with several state-of-the-art generative models on the two categories (i.e. *airplane* and *chair*) in ShapeNet. The generated point clouds and the reference point clouds are normalized into a bounding box of $[-1, 1]$.

Evaluation metric. To evaluate the generation quality of our method, we follow prior works [74], [75], [76] and employ the coverage score (COV), the minimum matching distance (MMD), 1-NN classifier accuracy (1-NNA) and the Jenson-Shannon divergence (JSD). The COV score measures whether the generated samples cover all the modes of the data distribution and the MMD score measures the fidelity of the generated samples. The 1-NNA measures the similarity between the distributions of the generated samples and the reference samples. If the accuracy of the 1-NN classifier is closer to 50% (random guess), the generation quality is considered to be better. The JSD score measures the similarity between the marginal point distributions of the generated set and the reference set. Meanwhile, we adopt L_2 CD and EMD to evaluate the reconstruction quality.

4.5.2 Network arrangement and training loss

Network arrangement. The network arrangement in this section is the same as point cloud auto-encoding (Section 4.4.2).

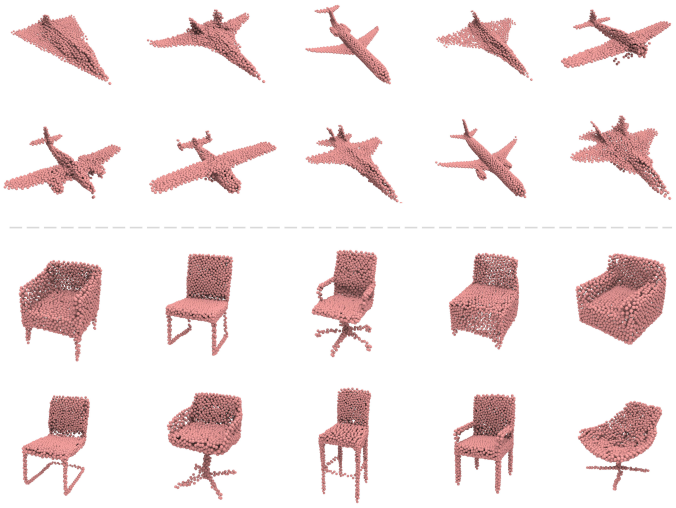


Fig. 15. Examples of novel point clouds generated by our model.

Training loss. To facilitate our decoder to generate novel shapes, we employ normalizing flows [76], [84], [85] to parameterize the prior distribution $p(\mathbf{f})$, here $\mathbf{f}$ is the latent code extracted by encoder. The normalizing flow is essentially a stack of affine coupling layers, which provides a trainable bijector F_α that maps an isotropic Gaussian to a complex distribution. The prior distribution can be computed by:

$$p(\mathbf{f}) = p_{\boldsymbol{w}}(\boldsymbol{w}) \cdot \left| \det \frac{\partial F_\alpha}{\partial \boldsymbol{w}} \right|^{-1}, \tag{11}$$

where $\boldsymbol{w} = F_\alpha^{-1}(\mathbf{f})$ and $p_{\boldsymbol{w}}(\boldsymbol{w})$ is the isotropic Gaussian $\mathcal{N}(\mathbf{0}, I)$. Same as DPM [76], we adopt PointNet [69] as the architecture for μ and Σ of the encoder $q(\mathbf{f}|\mathcal{P})$. Together with the reconstruction loss in Eq. (10), the training loss for point cloud generation is given as:

$$\mathcal{L}_{gen} = D_{\mathrm{KL}}(q(\mathbf{f}|\mathcal{P})||p_{\boldsymbol{w}}(\boldsymbol{w}) \cdot \left| \det \frac{\partial F_\alpha}{\partial \boldsymbol{w}} \right|^{-1}) + \mathcal{L}_{recon}, \tag{12}$$

where D_{KL} is KL divergence. To generate a point cloud, we first obtain the latent code $\mathbf{f}$ by drawing $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, I)$ and pass $\boldsymbol{w}$ through F_α, then pass the latent $\mathbf{f}$ to our decoder.

TABLE 8
Single view reconstruction on ShapeNet dataset in terms of per-point L_1 Chamfer distance $\times 10^2$ (lower is better).

Methods	Average	Plane	Bench	Cabinet	Car	Chair	Display	Lamp	Loud.	Rifle	Sofa	Table	Tele.	Vessel
3DR2N2 [80]	5.41	4.94	4.80	4.25	4.73	5.75	5.85	10.64	5.96	4.02	4.72	5.29	4.37	5.07
PSGN [81]	4.07	2.78	3.73	4.12	3.27	4.68	4.74	5.60	5.62	2.53	4.44	3.81	3.81	3.84
Pixel2mesh [82]	5.27	5.36	5.14	4.85	4.69	5.77	5.28	6.87	6.17	4.21	5.34	5.13	4.22	5.48
AtlasNet [71]	3.59	2.60	3.20	3.66	3.07	4.09	4.16	4.98	4.91	2.20	3.80	3.36	3.20	3.40
OccNet [83]	4.15	3.19	3.31	3.54	3.69	4.08	4.84	7.55	5.47	2.97	3.97	3.74	3.16	4.43
Ours	**2.86**	**1.99**	**2.54**	**2.52**	**2.44**	**3.13**	**3.37**	**4.34**	**3.98**	**1.84**	**3.09**	**2.71**	**2.45**	**2.80**

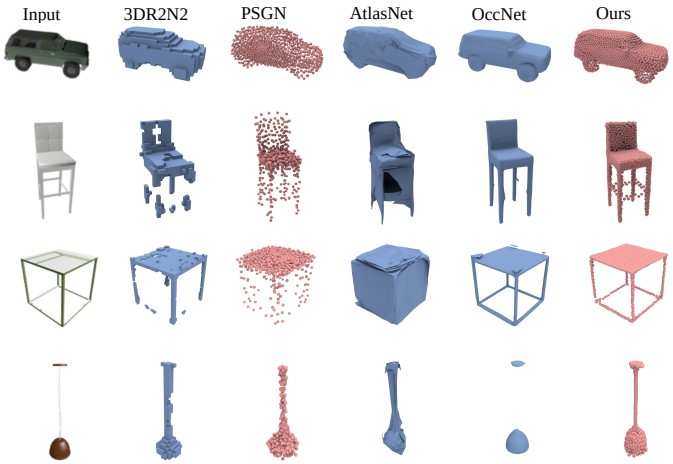


Fig. 16. Visual comparison of single image reconstruction. Mesh is in blue, point cloud is in red.

4.5.3 Quantitative and qualitative results

In Table 7, we quantitatively compare our methods with the following state-of-the-art generative models: DPM [76], ShapeGF [75], PointFlow [74], TreeGAN [79], GCN-GAN [78], PC-GAN [77]. We follow the same experimental settings as DPM [76] and all the other results are cited from DPM [76]. From Table 7, we can find that our method is highly competitive compared to the latest generative models like ShapeGF [75] and DPM [76], and significantly better than the other methods. Moreover, we also achieve the best results on the MMD score of the airplane category, which proves the generation ability of our method. In Fig. 15, we visually present some point cloud samples generated by our method, from which we can find that our method is able to generate novel shapes with good uniformity. Although the latent code is randomly sampled from Gaussian distribution, which leads uncertainty to decoder, our method can still generate diverse novel shapes while maintaining enriched local geometric structures. Therefore, the visual generation results further demonstrate the excellent generation quality of our snowflake point deconvolution.

4.6 Single Image Reconstruction

In this section, we extend our network to the task of single image reconstruction (SVR), of which the goal is to reconstruct a point cloud from an image of the underlying object.

4.6.1 Dataset and evalution metric

Dataset. For the single view reconstruction task, we employ the ShapeNet [70] dataset containing 43783 shapes from 13 categories. The dataset is split into training, testing and validation sets by the ratio of 70%, 20% and 10%, respectively. During training, we track the loss of our method on the validation set to determine when to stop training. And the testing set is used for quantitative and qualitative comparison. The baseline methods include 3D-R2N2 [80], PSGN [81], Pixel2Mesh [82], AtlasNet [71] and OccNet [83], for a fair comparison, we follow [83] and sample 30k points from the watertight mesh as ground truth.

Evaluation metric. We adopt L_1 Chamfer distance (CD) to evaluate the generation quality among all counterparts. In our experiment, we set the output point density of all methods to 2048. The output points of voxel-based (3D-R2N2 [80]) and mesh-based (Pixel2Mesh [82] and OccNet [83]) are sampled on their generated mesh, the sampling strategy is the same as [83].

4.6.2 Network arrangement and training loss

Network arrangement. The network framework of SVR consists of an encoder and a decoder. We use ResNet [86] (ResNet-18) as the encoder to align with the other baseline methods, and the dimension of the latent code is transformed to 128 by a *linear* layer. As for our decoder, the arrangement is the same as auto-encoding (Section 4.4.2).

Training loss. We use only Chamfer-L_1 Distance (CD) as the training loss for single image reconstruction, which is given as

$$\mathcal{L}_{\mathrm{SVR}} = \sum_{i \in \{0,2\}} \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_i, \mathcal{P}_i^i). \tag{13}$$

4.6.3 Quantitative comparison

In Table 8, we quantitatively compare our method with the baseline methods listed in Section 4.6.1. The quantitative results in Table 8 demonstrate that our method has the best reconstruction performance among all compared methods. Especially, by comparing with the second best method (AtlasNet, which is also trained with CD), our method significantly reduces the average CD by 0.73, which is 20.3% lower than AtlasNet (3.59 in terms of average CD). Moreover, our methods also achieves the best per-category CD across all shape categories, which fully demonstrates the generalization ability of our method.

4.6.4 Qualitative comparison

We visually compare the reconstruction results in Fig. 16, which illustrates the best reconstruction quality of our method among all baseline methods. Limited to voxel resolution, the reconstruction results of 3D-R2N2 [80] are very coarse and lack geometric details. PSGN [81] produces too much noise and requires additional denosing post-process steps. AtlasNet [71] is constrained to topological structure, and complex structures like table legs are hard to reveal. By contrast, our method is able to reconstruct the underlying object while recovering smooth surfaces, the SPD enables our decoder to produce shapes with much more uniform

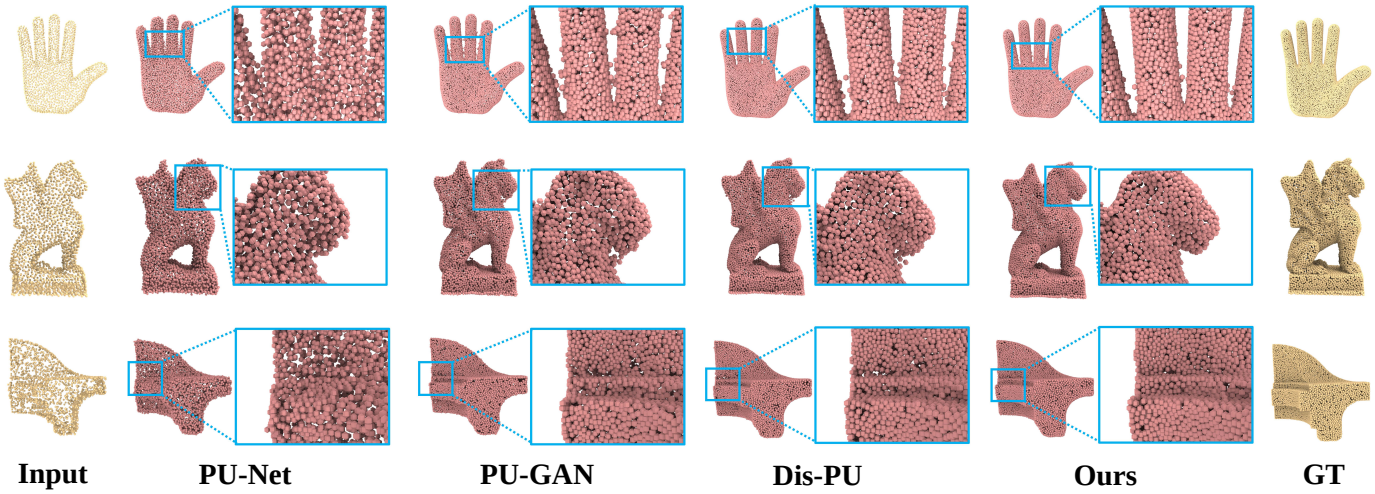


Fig. 17. Visual comparison of point cloud upsampling.

point distribution, and the smoothness of our shapes is even visually close to OccNet [83] (which is based on implicit functions). Moreover, compared with OccNet, our method can capture more complex local structures like the chair legs and the lamppost.

4.7 Point Cloud Upsampling

Given a sparse, noisy and non-uniform point cloud, the task of point cloud upsampling requires to upsample it and generate a dense and uniform point cloud. Because it is also a generation problem, we extend snowflake point deconvolution (SPD) to point cloud upsampling in this section.

TABLE 9
Quantitative comparison on point cloud upsampling task.

Methods	Size	CD (10^{-3})	HD (10^{-3})
PU-Net [14]	10.1M	0.473	4.680
PU-GAN [55]	9.57M	0.246	3.011
Dis-PU [87]	13.2M	0.210	2.771
Ours	**1.37M**	**0.199**	**2.701**

4.7.1 Dataset and evaluation metric

Dataset. To compare the upsampling quality of SPD with existing state-of-the-art methods, we use the benchmark dataset provided by PU-GAN [55] with 120 training and 27 testing objects. Each training object is cropped into 200 overlapped patches, and there are in total 24000 training surface patches. Each surface patch has 1024 points, during training, we randomly sample 256 points as input. For testing objects, we follow [55] and use Poisson disk sampling to sample 8192 points as the ground truth shape, and 2048 points are further sampled as input. Here the input shapes are sampled from the ground truth point cloud by farthest point sampling (FPS) [68] (instead of random sampling), because the size of the testing dataset is relatively small and the quantitative results are highly sensitive to the uniformity of input shapes.

Evaluation metric. We use L_2 Chamfer Distance (CD) and Hausdorff distance (HD) [88] as the evaluation metric for fair comparison with others.

4.7.2 Network arrangement and training loss

Network arrangement. For the task of point cloud upsampling, four stacked SPDs are used to upsample the sparse point cloud into a dense one. The upsampling factors r_1, r_2, r_3 and r_4 of the four SPDs are 1, 2, 2, 1, respectively. Because point cloud upsampling requires the output point cloud to be uniform, we particularly add an additional SPD (of which r_4 = 1) to refine the upsampled point cloud.

Training loss. We use only L_2 Chamfer Distance (CD) to train our network, the training loss is given as

$$\mathcal{L}_{upsampling} = \sum_{i \in \{1,3,4\}} \mathcal{L}_{\mathrm{CD}}(\mathcal{P}_i, \mathcal{P}_i^{'}), \tag{14}$$

where $\mathcal{P}_i^{'}$ is the downsampled ground truth shape that has the same point density of $\mathcal{P}_i$.

4.7.3 Quantitative comparison

In Table 9, we quantitatively compare our method with the following state-of-the-art point cloud upsampling methods: PU-Net [14], PU-GAN [55], Dis-PU [87]. We use the pretrained models provided by these methods for testing on the same test sets. From Table 9, we can find that our method achieves the best performance on both L_2 CD and HD. Even though our method is not trained with repulsion loss [14] or uniform loss [55], our method is able to recover the underlying surface while maintaining good point uniformity. Moreover, by comparing model size with the other methods in terms of parameter number, we can find that our method is significantly smaller than the others, which further proves the superior generation ability of snowflake point deconvolution.

4.7.4 Qualitative comparison

In Fig. 17, we visually compare our method with the state-of-the-art methods listed in Table 9. From Fig. 17, we can find that our method can produce the most similar visual results to the ground truth shape. Whether it is complex tiny local structures (second shape) or simple flat regions (third shape), our method can preserve an uniform point distribution.

5 Conclusion, Limitation and Future Work

In this paper, we propose a novel neural network for point cloud completion, named SnowflakeNet. The SnowflakeNet models the generation of complete point clouds as the snowflake-like growth of points in 3D space using multiple layers of Snowflake Point Deconvolution. By further introducing skip-transformer in Snowflake Point Deconvolution (SPD), SnowflakeNet learns to generate locally compact and structured point cloud with highly detailed geometries. We conduct comprehensive experiments on sparse (Completion3D) and dense (PCN) point cloud completion datasets, which shows our superior performance over the current SOTA point cloud completion methods. We further extend Snowflake Point Deconvolution to more point cloud generation tasks like point cloud auto-encoding, novel point cloud generation, single image reconstruction and upsampling. The generation ability of Snowflake Point Deconvolution is fully demonstrated by both quantitive and qualitative experimental results.

The main limitation of SnowflakeNet is that the skip-transformer in SPD merely aggregates information from the k-nearest neighbors (k-NN) of each point. While the k-NN strategy enables the SPD to focus on local structure, it also disenables the SPD to capture long-range shape context. Therefore, when the input point cloud contains too much noise, the SPD can hardly arrange point splitting properly. In our opinion, a potential and promising future work to this problem is to further explore the local and global feature fusion in SPD, so that SPD can well adapt to more complex shape context.

Another limitation is that SPD generates the same number of child points for all seed points, despite different numbers of points are needed to represent different parts of the shape. For example, it requires more points to represent the lampshade than the lamppost. It would be ideal if the SPD can adaptively split seed points based on the complexity of local shape context; this may require significant changes to the network.

References

[1] Z. Han, C. Chen, Y.-S. Liu, and M. Zwicker, "ShapeCaptioner: Generative caption network for 3D shapes by learning a mapping from parts detected in multiple views to sentences," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 1018–1027. 1

[2] Z. Han, M. Shang, Y.-S. Liu, and M. Zwicker, "View inter-prediction GAN: Unsupervised representation learning for 3D shapes by learning global shape memories to support local view predictions," in *The 33rd AAAI Conference on Artificial Intelligence (AAAI)*, 2019. 1

[3] Z. Han, M. Shang, X. Wang, Y.-S. Liu, and M. Zwicker, "Y2Seq2Seq: Cross-modal representation learning for 3D shape and text by joint reconstruction and prediction of view and word sequences," *The 33th AAAI Conference on Artificial Intelligence (AAAI)*, 2019. 1

[4] Z. Han, Z. Liu, C.-M. Vong, Y.-S. Liu, S. Bu, J. Han, and C. P. Chen, "BoSCC: Bag of spatial context correlations for spatially enhanced 3D shape representation," *IEEE Transactions on Image Processing*, vol. 26, no. 8, pp. 3707–3720, 2017. 1

[5] X. Wen, T. Li, Z. Han, and Y.-S. Liu, "Point cloud completion by skip-attention network with hierarchical folding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 1939–1948. 1, 2, 7, 8

[6] X. Wen, P. Xiang, Z. Han, Y.-P. Cao, P. Wan, W. Zheng, and Y.-S. Liu, "PMP-Net: Point cloud completion by learning multi-step point moving paths," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 6, 7, 8

[7] X. Wen, Z. Han, Y.-P. Cao, P. Wan, W. Zheng, and Y.-S. Liu, "Cycle4Completion: Unpaired point cloud completion using cycle transformation with missing region coding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 5

[8] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese, "TopNet: Structural point cloud decoder," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 383–392. 1, 2, 5, 6, 7, 8

[9] X. Wang, M. H. Ang Jr, and G. H. Lee, "Cascaded refinement network for point cloud completion," in *Proceedings of the IEEEF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 790–799. 1, 2, 3, 4, 6, 7

[10] H. Xie, H. Yao, S. Zhou, J. Mao, S. Zhang, and W. Sun, "GRNet: Gridding residual network for dense point cloud completion," in *European Conference on Computer Vision (ECCV)*, 2020. 1, 3, 6, 7, 8

[11] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 206–215. 1, 2, 4, 6, 7, 8

[12] W. Zhang, Q. Yan, and C. Xiao, "Detail preserved point cloud completion via separated feature aggregation," in *European Conference on Computer Vision (ECCV)*, 2020, pp. 512–528. 2, 3, 4, 5, 6

[13] M. Liu, L. Sheng, S. Yang, J. Shao, and S.-M. Hu, "Morphing and sampling network for dense point cloud completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 11 596–11 603. 1

[14] Y. Lequan, L. Xianzhi, F. Chi-Wing, C.-O. Daniel, and H. Pheng-Ann, "PU-Net: Point cloud upsampling network," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2018. 1, 14

[15] M. Sung, V. G. Kim, R. Angst, and L. Guibas, "Data-driven structural priors for shape completion," *ACM Transactions on Graphics*, vol. 34, no. 6, p. 175, 2015. 2

[16] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, J. Levine, A. Sharf, and C. Silva, "State of the art in surface reconstruction from point clouds," in *Proceedings of Eurographics*, vol. 1, no. 1, 2014, pp. 161–185. 2

[17] D. Thanh Nguyen, B.-S. Hua, K. Tran, Q.-H. Pham, and S.-K. Yeung, "A field model for repairing 3D shapes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5676–5684. 2

[18] W. Hu, Z. Fu, and Z. Guo, "Local frequency interpretation and non-local self-similarity on graph for point cloud inpainting," *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 4087–4100, 2019. 2

[19] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "PF-Net: Point fractal network for 3D point cloud completion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 7662–7670. 2, 3

[20] X. Chen, B. Chen, and N. J. Mitra, "Unpaired point cloud completion on real scans using adversarial training," in *International Conference on Learning Representations*, 2019. 2

[21] J. Gu, W.-C. Ma, S. Manivasagam, W. Zeng, Z. Wang, Y. Xiong, H. Su, and R. Urtasun, "Weakly-supervised 3D shape completion in the wild," in *Proc. of the European Conf. on Computer Vision (ECCV)*. Springer, 2020. 2

[22] Y. Nie, Y. Lin, X. Han, S. Guo, J. Chang, S. Cui, and J. Zhang, "Skeleton-bridged point completion: From global inference to local adjustment," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 16 119–16 130. 2

[23] T. Hu, Z. Han, and M. Zwicker, "3D shape completion with multi-view consistent inference," in *AAAI*, 2020. 2

[24] T. Hu, Z. Han, A. Shrivastava, and M. Zwicker, "Render4Completion: Synthesizing multi-view depth maps for 3D shape completion," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2019. 2

[25] X. Wang, M. H. Ang, and G. H. Lee, "Voxel-based network for shape completion by leveraging edge generation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 13 189–13 198. 2, 7, 8

[26] A. Alliegro, D. Valsesia, G. Fracastoro, E. Magli, and T. Tommasi, "Denoise and contrast for category agnostic shape completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4629–4638. 2

[27] T. Huang, H. Zou, J. Cui, X. Yang, M. Wang, X. Zhao, J. Zhang, Y. Yuan, Y. Xu, and Y. Liu, "Rfnet: Recurrent forward network for dense point cloud completion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 508–12 517. 2

[28] J. Zhang, X. Chen, Z. Cai, L. Pan, H. Zhao, S. Yi, C. K. Yeo, B. Dai, and C. C. Loy, "Unsupervised 3d shape completion through gan inversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1768–1777. 2

[29] B. Gong, Y. Nie, Y. Lin, X. Han, and Y. Yu, "Me-pcn: Point completion conditioned on mask emptiness," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 488–12 497. 2

[30] L. Zhou, Y. Du, and J. Wu, "3d shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 5826–5835. 2

[31] X. Zhang, Y. Feng, S. Li, C. Zou, H. Wan, X. Zhao, Y. Guo, and Y. Gao, "View-guided point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 890–15 899. 2

[32] C. Xie, C. Wang, B. Zhang, H. Yang, D. Chen, and F. Wen, "Style-based point generator with adversarial rendering for point cloud completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 4619–4628. 2

[33] X. Wang, M. H. Ang, and G. Lee, "Cascaded refinement network for point cloud completion with self-supervision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[34] Y. Jiang, D. Ji, Z. Han, and M. Zwicker, "SDFDiff: Differentiable rendering of signed distance fields for 3D shape optimization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2

[35] Z. Han, X. Liu, Y.-S. Liu, and M. Zwicker, "Parts4Feature: Learning 3D global features from generally semantic parts in multiple views," in *International Joint Conference on Artificial Intelligence*, 2019. 2

[36] Z. Han, H. Lu, Z. Liu, C.-M. Vong, Y.-S. Liu, M. Zwicker, J. Han, and C. P. Chen, "3D2SeqViews: Aggregating sequential views for 3D global feature learning by CNN with hierarchical attention aggregation," *IEEE Transactions on Image Processing*, vol. 28, no. 8, pp. 3986–3999, 2019. 2

[37] Z. Han, X. Wang, C.-M. Vong, Y.-S. Liu, M. Zwicker, and C. Chen, "3DViewGraph: Learning global features for 3D shapes from a graph of unordered views with attention," in *International Joint Conference on Artificial Intelligence*, 2019. 2

[38] Z. Han, M. Shang, Z. Liu, C.-M. Vong, Y.-S. Liu, M. Zwicker, J. Han, and C. P. Chen, "SeqViews2SeqLabels: Learning 3D global features via aggregating sequential views by RNN with attention," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 658–672, 2018. 2

[39] Z. Han, G. Qiao, Y.-S. Liu, and M. Zwicker, "SeqXY2SeqZ: Structure learning for 3D shapes by sequentially predicting 1D occupancy segments from 2D coordinates," in *European Conference on Computer Vision*. Springer, 2020, pp. 607–625. 2

[40] Z. Han, B. Ma, Y.-S. Liu, and M. Zwicker, "Reconstructing 3D shapes from multiple sketches using direct shape optimization," *IEEE Transactions on Image Processing*, vol. 29, pp. 8721–8734, 2020. 2

[41] Z. Han, X. Wang, Y.-S. Liu, and M. Zwicker, "Hierarchical view predictor: Unsupervised 3D global feature learning through hierarchical prediction among unordered views," in *ACM International Conference on Multimedia*, 2021. 2

[42] X. Wen, Z. Han, and Y.-S. Liu, "CMPD: Using cross memory network with pair discrimination for image-text retrieval," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 6, pp. 2427–2437, 2020. 2

[43] X. Wen, Z. Han, X. Yin, and Y.-S. Liu, "Adversarial cross-modal retrieval via learning and transferring single-modal similarities," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2019, pp. 478–483. 2

[44] Z. Han, Z. Liu, C.-M. Vong, Y.-S. Liu, S. Bu, J. Han, and C. P. Chen, "Deep Spatiality: Unsupervised learning of spatially-enhanced global and local 3D features by deep neural network with coupled softmax," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 3049–3063, 2018. 2

[45] Z. Han, C. Chen, Y.-S. Liu, and M. Zwicker, "DRWR: A differentiable renderer without rendering for unsupervised 3D structure learning from silhouette images," in *International Conference on Machine Learning (ICML)*, 2020. 2

[46] X. Wen, Z. Han, G. Youk, and Y.-S. Liu, "CF-SIS: Semantic-instance segmentation of 3D point clouds by context fusion with self-attention," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 1661–1669. 2

[47] X. Wen, Z. Han, X. Liu, and Y.-S. Liu, "Point2SpatialCapsule: Aggregating features and spatial relationships of local regions on point clouds using spatial-aware capsules," *IEEE Transactions on Image Processing*, vol. 29, pp. 8855–8869, 2020. 2

[48] X. Liu, Z. Han, X. Wen, Y.-S. Liu, and M. Zwicker, "L2G Auto-encoder: Understanding point clouds by local-to-global reconstruction with hierarchical self-attention," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 989–997. 2

[49] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2Sequence: Learning the shape representation of 3D point clouds with an attention-based sequence to sequence network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8778–8785. 2

[50] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Fine-grained 3D shape classification with hierarchical part-view attention," *IEEE Transactions on Image Processing*, vol. 30, pp. 1744–1758, 2021. 2

[51] X. Liu, Z. Han, F. Hong, Y.-S. Liu, and M. Zwicker, "LRC-Net: Learning discriminative features on point clouds by encoding local region contexts," *Computer Aided Geometric Design*, vol. 79, p. 101859, 2020. 2

[52] Z. Han, X. Wang, Y.-S. Liu, and M. Zwicker, "Multi-angle point cloud-vae: Unsupervised feature learning for 3D point clouds from multiple angles by joint self-reconstruction and half-to-half prediction," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2019, pp. 10 441–10 450. 2

[53] C. Chen, Z. Han, Y.-S. Liu, and M. Zwicker, "Unsupervised learning of fine structure generation for 3D point clouds by 2D projection matching," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 2

[54] B. Ma, Z. Han, Y.-S. Liu, and M. Zwicker, "Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces," in *International Conference on Machine Learning (ICML)*, 2021. 2

[55] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-GAN: A point cloud upsampling adversarial network," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 7203–7212. 2, 14

[56] D. Zong, S. Sun, and J. Zhao, "Ashf-net: Adaptive sampling and hierarchical folding network for robust point cloud completion," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 4, 2021, pp. 3625–3632. 2

[57] A. Dai, C. Ruizhongtai Qi, and M. Nießner, "Shape completion using 3D-encoder-predictor CNNs and shape synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5868–5877. 3

[58] L. Pan, X. Chen, Z. Cai, J. Zhang, H. Zhao, S. Yi, and Z. Liu, "Variational relational point completion network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 8524–8533. 3, 7

[59] X. Yaqi, X. Yan, L. Wei, S. Rui, C. Kailang, and S. Uwe, "ASFM-Net: Asymmetrical siamese feature matching network for point completion," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 3

[60] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou, "Pointr: Diverse point cloud completion with geometry-aware transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 12 498–12 507. 3, 6

[61] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 728–737. 3, 4, 5, 6, 7, 8

[62] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6000–6010. 3

[63] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. 3

[64] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4055–4064. 3

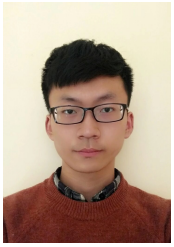
[65] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *ICCV*, 2021. 3

[66] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "PCT: Point cloud transformer," *Computational Visual Media*, vol. 7, no. 2, p. 187–199, Apr 2021. 3

[67] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, "3D object detection with Pointformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 7463–7472. 3

[68] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural*

Information Processing Systems (NeurIPS), 2017, pp. 5099–5108. 3, 4, 14
[69] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1063–6919. 4, 10, 12
[70] A. X. Chang, T. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "ShapeNet: An information-rich 3D model repository," *arXiv:1512.03012*, 2015. 5, 10, 13
[71] T. Groueix, M. Fisher, V. Kim, B. Russell, and M. Aubry, "A papier-mâché approach to learning 3D surface generation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 216–224. 6, 7, 10, 13
[72] J. Zhang, W. Chen, Y. Wang, R. Vasudevan, and M. Johnson-Roberson, "Point set voting for partial point cloud analysis," *IEEE Robotics and Automation Letters*, 2021. 7
[73] Y. Wang, D. J. Tan, N. Navab, and F. Tombari, "SoftPoolNet: Shape descriptor for point cloud completion and classification," in *European Conference on Computer Vision (ECCV)*, 2020. 7
[74] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, "Pointflow: 3d point cloud generation with continuous normalizing flows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 10, 12, 13
[75] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan, "Learning gradient fields for shape generation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 10, 12, 13
[76] S. Luo and W. Hu, "Diffusion probabilistic models for 3d point cloud generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 10, 11, 12, 13
[77] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *International conference on machine learning*. PMLR, 2018, pp. 40–49. 12, 13
[78] D. Valsesia, G. Fracastoro, and E. Magli, "Learning localized generative models for 3D point clouds via graph convolution," in *International conference on learning representations*, 2018. 12, 13
[79] D. W. Shu, S. W. Park, and J. Kwon, "3D point cloud generative adversarial network based on tree structured graph convolutions," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3859–3868. 12, 13
[80] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," in *European conference on computer vision*. Springer, 2016, pp. 628–644. 13
[81] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613. 13
[82] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 52–67. 13
[83] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4460–4470. 13, 14
[84] X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel, "Variational lossy autoencoder," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 12
[85] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538. 12
[86] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 13
[87] R. Li, X. Li, P.-A. Heng, and C.-W. Fu, "Point cloud upsampling via disentangled refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 344–353. 14
[88] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, "A benchmark for surface reconstruction," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 2, pp. 1–17, 2013. 14

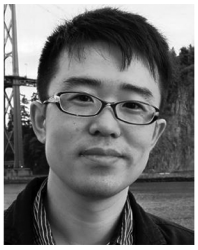


Peng Xiang received the B.S. degree in software engineering from Chongqing University, China, in 2019. He is currently the graduate student with the School of Software, Tsinghua University. His research interests include deep learning, 3D shape analysis and pattern recognition.

Xin Wen received the B.S. degree in engineering management from the Tsinghua University, China, in 2012. He is currently the PhD student with the School of Software, Tsinghua University. His research interests include deep learning, shape analysis and pattern recognition, and NLP.

Yu-Shen Liu (M'18) received the B.S. degree in mathematics from Jilin University, China, in 2000, and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2006. From 2006 to 2009, he was a Post-Doctoral Researcher with Purdue University. He is currently an Associate Professor with the School of Software, Tsinghua University. His research interests include shape analysis, pattern recognition, machine learning, and semantic search.

Yan-Pei Cao received the bachelor's and Ph.D. degrees in computer science from Tsinghua University in 2013 and 2018, respectively. He is currently a research engineer at Y-tech, Kuaishou Technology. His research interests include geometric modeling and processing, 3D reconstruction, and 3D computer vision.



Pengfei Wan received the B.E. degree in electronic engineering and information science from the University of Science and Technology of China, Hefei, China, and the Ph.D. degree in electronic and computer engineering from the Hong Kong University of Science and Technology, Hong Kong. His research interests include image/video signal processing, computational photography, and computer vision.

Wen Zheng received the bachelor's and master's degrees from Tsinghua University, Beijing, China, and the Ph.D. degree in computer science from Stanford University, in 2014. He is currently the Head of Y-tech at Kuaishou Technology. His research interests include computer vision, augmented reality, machine learning, and computer graphics.

Zhizhong Han received the Ph.D. degree from Northwestern Polytechnical University, China, 2017. He was a Post-Doctoral Researcher with the Department of Computer Science, at the University of Maryland, College Park, USA. Currently, he is an Assistant Professor of Computer Science at Wayne State University, USA. His research interests include 3D computer vision, digital geometry processing and artificial intelligence.