PMP-Net++: Point Cloud Completion by Transformer-Enhanced Multi-step Point Moving Paths

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

Abstract—Point cloud completion concerns to predict missing part for incomplete 3D shapes. A common strategy is to generate complete shape according to incomplete input. However, unordered nature of point clouds will degrade generation of high-quality 3D shapes, as detailed topology and structure of unordered points are hard to be captured during the generative process using an extracted latent code. We address this problem by formulating completion as point cloud deformation process. Specifically, we design a novel neural network, named PMP-Net++, to mimic behavior of an earth mover. It moves each point of incomplete input to obtain a complete point cloud, where total distance of point moving paths (PMPs) should be the shortest. Therefore, PMP-Net++ predicts unique PMP for each point according to constraint of point moving distances. The network learns a strict and unique correspondence on point-level, and thus improves quality of predicted complete shape. Moreover, since moving points heavily relies on per-point features learned by network, we further introduce a transformer-enhanced representation learning network, which significantly improves completion performance of PMP-Net++. We conduct comprehensive experiments in shape completion, and further explore application on point cloud up-sampling, which demonstrate non-trivial improvement of PMP-Net++ over state-of-the-art point cloud completion/up-sampling methods.

Index Terms—point clouds, 3D shape completion, transformer, up-sampling

1 Introduction

As one of the widely used 3D shape representations, point clouds can be easily obtained through depth cameras or other 3D scanning devices. Due to the limitations of view-angles or occlusions of 3D scanning devices, the raw point clouds are usually sparse and incomplete [1]. Therefore, a shape completion/consolidation process is usually required to generate the missing regions of 3D shape for the downstream 3D computer vision applications like classification [2], [3], [4], [5], [6], [7], segmentation [8], [9] and other visual analysis [10].

In this paper, we focus on the completion task for 3D objects represented by point clouds, where the missing parts are caused by self-occlusion due to the view angle of scanner. Most of the previous methods formulate the point cloud completion as a point cloud generation problem [1], [11], [12], [13], where an encoder-decoder framework is usually adopted to extract a latent code from the input incomplete point cloud, and decode the extracted latent code into a complete point cloud. Benefiting from the deep neural network based point cloud learning methods, the point cloud completion methods along this line have made huge progress in the last few years [1], [13]. However, the generation of point clouds remains a difficult task using deep neural network, because the unordered nature of point clouds makes the generative model difficult to capture the detailed topology or structure among discrete points [13]. Therefore, the performance of generative models based point clouds completion is still unsatisfactory.

To improve the point cloud completion performance, in this paper, we propose a novel deep neural network, named PMP-Net++, to formulate the task of point cloud completion from a new perspective. Different from the generative model that directly predicts the coordinations of all points in 3D space, the PMP-Net++ learns to move the points from the source 3D shape to the target one. Through the point moving process, the PMP-Net++ learns the point-level correspondences between the source point cloud and the target, which captures the detailed topology and structure relationships between the two point clouds. On the other hand, there are many possible solutions to move points from the source to the target, which will make the network difficult to train well. Therefore, in order to encourage the network to learn a unique optimal arrangement of point moving path, we take the inspiration from the Earth Mover's Distance (EMD) and propose to regularize a transformer-enhanced Point-Moving-Path Network (named PMP-Net++) under the constraint of the total point moving distances (PMDs), which guarantees the uniqueness of path arrangement between the source point cloud and the target one.

A detailed illustration is given in Figure 1. Taking the task of completing short line AB to the long line $A'B'$ for example, the generation based neural network aims to predict the coordinates of $A'B'$, which is usually optimized by the chamfer distance (CD) or earth mover's distance (EMD). However, due to the discrete

- *Xin Wen and Peng Xiang are with the School of Software, Tsinghua University, Beijing, China. Xin Wen is also with JD Logistics, JD.com, China. E-mail: wenxin16@jd.com, xp20@mails.tsinghua.edu.cn*
- *Zhizhong Han is with the Department of Computer Science, Wayne State University, USA. E-mail: h312h@wayne.edu*
- *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*

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. Code is available at https://github.com/diviswen/PMP-Net.

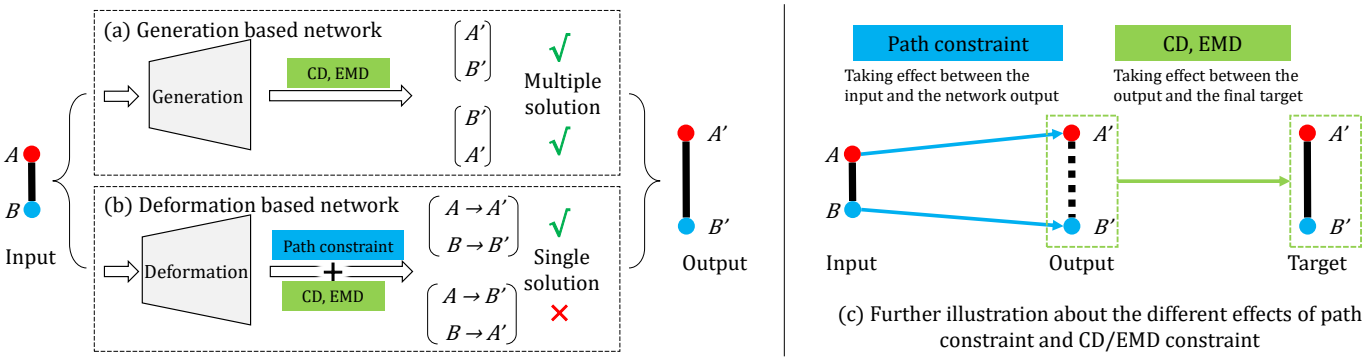


Fig. 1. Illustration of the differences between the generation based methods and the deformation based methods, where the task is to complete a short line AB to a long line $A'B'$ (in (a) and (b)). The differences of the effect between the path constraint and the widely used CD/EMD is further illustrated in (c).

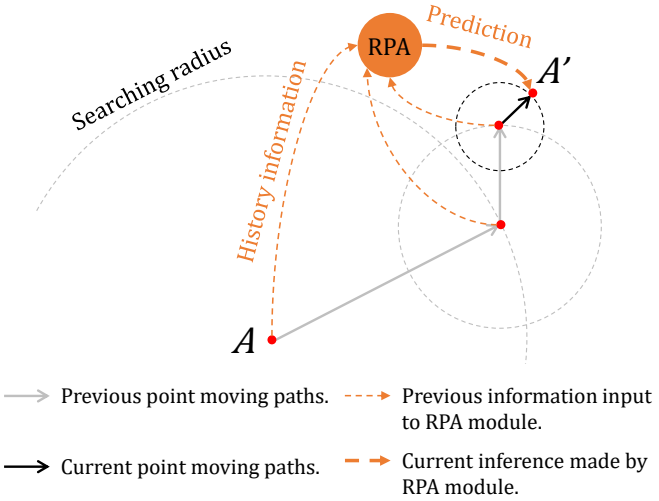


Fig. 2. Illustration of path searching with multiple steps under the coarse-to-fine searching radius. The PMP-Net++ moves point A to point A' by three steps, with each step reducing its searching radius, and looking back to consider the moving history in order to decide the next place to move.

nature of point cloud data, the target matrix representing the line $A'B'$ has multiple arrangements, all of which meet the minimum loss of CD and EMD constraints. As a result, there are multiple optimal targets for the network, which cannot guide the network to learn the detailed shape correspondence between the short line AB and the long line $A'B'$. In contrast, shape deformation based neural network can potentially establish a direct and point-wise correspondence between the source input (AB) and the target output ($A'B'$), with the guidance of path constraint. The reason is further illustrated in Figure 1 (c). The path constraint takes effect between the input and the output of the network. It regularizes the point moving path and optimizes the network to predict a unique displacement for each point. Under such circumstance, the output of network is the displacement, which is locally related to each start point and end point. On the other hand, the output of generation based network is the coordinate matrix representing line $A'B'$, and its only source of supervision is the overall shape constraints CD/EMD, which takes effect between the output and the target ground truth.

Moreover, in order to predict the point moving path more accurately, we propose a multi-step path searching strategy to continuously refine the point moving path under multi-scaled searching radius. Specifically, as shown in Figure 2, the path searching is repeated for multiple steps in a coarse-to-fine manner. Each step will take the previously predicted paths into consideration, and then plan its next move path according to the previous paths. To record and aggregate the history information of point moving path, we take the inspiration from Gated Recurrent Unit (GRU) to propose a novel Recurrent Path Aggregation (RPA) module. It can memorize and aggregate the route sequence for each point, and combine the previous information with the current location of point to predict the direction and the length for the next move. By reducing the searching radius step-by-step, PMP-Net++ can consistently refine a more and more accurate path for each point to move from its original position on the incomplete point cloud to the target position on the complete point cloud.

The PMP-Net++ proposed in this paper is an enhanced extension of our latest work PMP-Net [14]. We find that the point features learned in the moving procedure plays the key role during the prediction of high quality complete shape. And the PointNet++ [15] based backbone used in PMP-Net cannot provide more discriminative point features, due to its max-pooling based feature aggregation strategy [4]. Therefore, inspired by the recent success of Transformer in point cloud representation learning [16], we introduce a novel transformer based framework into PMP-Net++ to enhance the point features learned by our network, which aims to predict a more accurate displacement for each point. In all, the main contributions of our work can be summarized as follows.

- We propose a novel network for point cloud completion task, named PMP-Net++, to move each point on the incomplete shape to the complete one to achieve a highly accurate point cloud completion. Compared with previous generative completion methods, PMP-Net++ has the ability to learn more detailed topology and structure relationships between incomplete shapes and complete ones, by learning the point-level correspondence through point moving path prediction.
- We propose to learn a unique point moving path arrangement between incomplete and complete point clouds, by regularizing the network using the constraint of Earth Mover's Distance. As a result, the network will not be confused by multiple solutions of moving points, and finally predicts a meaningful point-wise correspondence

between the source and target point clouds.
- We propose to search point moving path with multiple steps in a coarse-to-fine manner. Each step will decide the next move based on the aggregated information from the previous paths and its current location, using the proposed Recurrent Path Aggregation (RPA) module.
- Compared with our latest PMP-Net, we further introduce a transformer-enhanced point cloud network into PMP-Net++ to improve the point feature learning. We conduct comprehensive experiments on Completion3D [13] and PCN [12] datasets, and further explore the application on point cloud up-sampling, all of which demonstrate the non-trivial improvement of PMP-Net++ over the state-of-the-art point cloud completion/up-sampling methods (including our latest PMP-Net).

2 Related Work

The deep learning technology in 3D reconstruction [17], [18], [18], [19], [20] and representation learning [21], [22], [23], [24] have boosted the research of 3D shape completion, which can be roughly divided into two categories. (1) Traditional 3D shape completion methods [25], [26], [27], [28] usually formulate hand-crafted features such as surface smoothness or symmetry axes to infer the missing regions, while some other methods [29], [30], [31], [32] consider the aid of large-scale complete 3D shape datasets, and perform searching to find the similar patches to fill the incomplete regions of 3D shapes. (2) Deep learning based methods [33], [34], [35], [36], on the other hand, exploit the powerful representation learning ability to extract geometric features from the incomplete input shapes, and directly infer the complete shape according to the extracted features. Those learnable methods do not require the predefined hand-crafted features in contrast with traditional completion methods, and can better utilize the abundant shape information lying in the large-scale completion datasets. The proposed PMP-Net++ also belongs to the deep learning based method, where the methods along this line can be further categorized and detailed as below.

2.1 Volumetric aided shape completion

The representation learning ability of convolutional neural network (CNN) has been widely used in 2D computer vision research, and the studies concerning application of 2D image inpainting have been continuously surging in recent years. A intuitive idea for 3D shape completion can be directly borrowed from the success of 2D CNN in image inpainting research [37], [38], [39], extending it into 3D space. Recently, several volumetric aided shape completion methods, which are based on 3D CNN structure, have been developed. Note that we use the term *"volumetric aided"* to describe this kind of methods, because the 3D voxel is usually not the final output of the network. Instead, the predicted voxel will be further refined and converted into other representations like mesh [11] or point cloud [40], in order to produce more detailed 3D shapes. Therefore, the voxel is more like an intermediate aid to help the completion network infer the complete shape. Notable works along this line like 3D-EPN [11] and GRNet [40] have been proposed to reconstruct the complete 3D voxel in a coarse-to-fine manner. They first predict a coarse complete shape using 3D CNN under an encoder-decoder framework, and then refine the output using similar patches selected from a complete shape dataset [11] or by further reconstructing the detailed point cloud according to the output voxel [40]. Also, there are some studies that consider purely volumetric data for shape completion task. For example, Han et. al [41] proposed to directly generate the high-resolution 3D volumetric shape, by simultaneously inferring global structure and local geometries to predict the detailed complete shape. Stutz et. al [42] proposed a variational auto-encoder based method to complete the 3D voxel under weak supervision. Despite the fascinating ability of 3D CNN for feature learning, the computational cost which is cubic to the resolution of input voxel data makes it difficult to process fine-grained shapes [1].

2.2 Point cloud based shape completion

There is a growing attention on the task of point cloud based shape completion [1], [13], [43], [44] in recent years. Since point cloud is a direct output form of many 3D scanning devices, and the storage and process of point clouds require much less computational cost than volumetric data, many recent studies consider to perform direct completion on 3D point clouds. Enlighten from the improvement of point cloud representation learning [15], [45], previous methods like TopNet [13], PCN [12] and SA-Net [1] formulate the solution as a generative model under an encoder-decoder framework. They adopted encoder like PointNet [45] or PointNet++ [15] to extract the global feature from the incomplete point cloud, and use a decoder to infer the complete point cloud according to the extracted features. Compare to PCN [12], TopNet [13] improved the structure of decoder in order to implicitly model and generate point cloud in a rooted tree architecture [13]. SA-Net [1] took one step further to preserve and convey the detailed geometric information of incomplete shape into the generation of complete shape through skip-attention mechanism. Other notable methods like RL-GAN-Net [34], Render4Completion [44] and VRCNet [46] focused on the framework of adversarial learning and variational auto-encoder to improve the reality and consistency of the generated complete shapes. More recently, the progressive refinement of 3D shape has become a popular idea in the research of point cloud completion (e.g. CRN [47], PF-Net [35]), as it can help the network generate 3D shapes with detailed structures.

In all, most of the above methods are generative solution for point cloud completion task, and inevitably suffer from the unordered nature of point clouds, which makes it difficult to reconstruct the detailed typology or structure using a generative decoder. Therefore, in order to avoid the problem of predicting unordered data, PMP-Net++ uses a different way to reconstruct the complete point cloud, which learns to move all points from the initial input instead of directly generating the final point cloud from a latent code.

The idea of PMP-Net++ is also related to the research of 3D shape deformation [48], which mainly considered one-step deformation. However, the deformation between the incomplete and complete shapes is more challenging, which requires the inference of totally unknown geometries in missing regions without any other prior information. In contrast, we propose multi-step searching to encourage PMP-Net++ to infer more detailed geometric information for missing region, along with point moving distance regularization to guarantee the efficiency of multi-step inference.

3 Architecture of PMP-Net

An overview of the proposed PMP-Net++ is shown in Figure 4. The network basically consists of three parts: (1) the encoder to

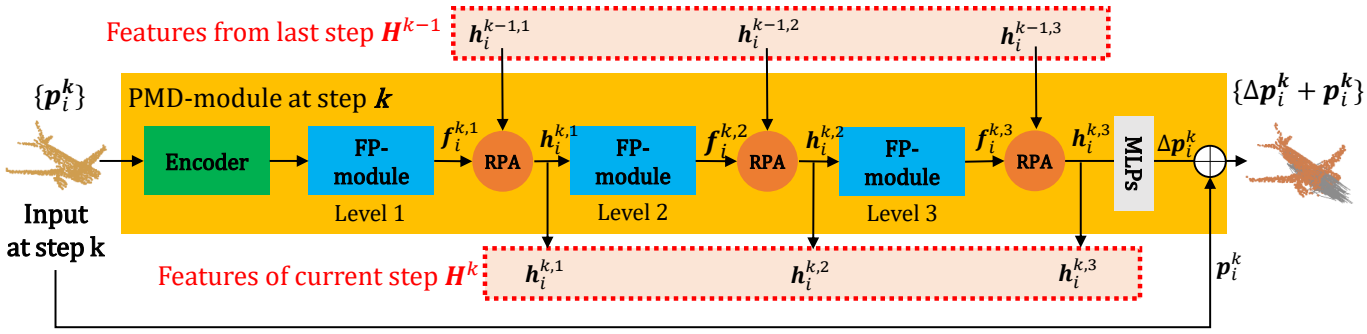


Fig. 3. Detailed structure of PMD-module at step k. It mainly consists of three parts: (1) point cloud encoder and (2) feature prorogation module (FP-module) to extract per-point features; (3) RPA module to recurrently learn and forget the path searching information from the previous steps.

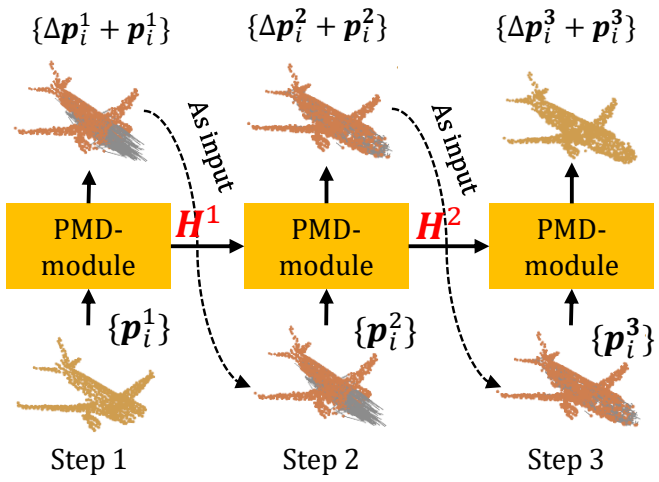


Fig. 4. Illustration of path searching with multiple steps under the coarse-to-fine searching radius. The PMP-Net++ moves point A to point A' by three steps, with each step reducing its searching radius, and looking back to consider the moving history in order to decide the next place to move.

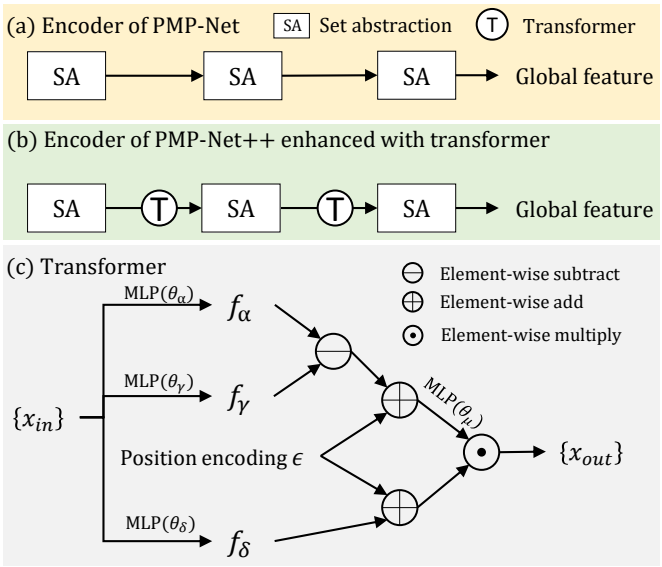


Fig. 5. The architecture of encoder used in PMD-module. Moreover, we also show the comparison with previous work and the detailed structure of transformer.

extract point cloud features; (2) the feature propagation module (FP-Module) to predict the point moving path for each point; (c) the RPA module recurrently fuses and integrates the current step's point features with the previous steps' path information. The details of each part will be described as below.

3.1 Point Displacement Prediction

3.1.1 Multi-step framework

An overview of the proposed PMP-Net++ is shown in Figure 4. Given an input point cloud $P = \{\boldsymbol{p}_i\}$ and a target point cloud $P' = \{\boldsymbol{p'}_j\}$. The objective of PMP-Net++ is to predict a displacement vector set $\Delta P = \{\Delta \boldsymbol{p}_i\}$, which can move each point from P into the position of P' such that $\{(\boldsymbol{p}_i + \Delta \boldsymbol{p}_i)\} = \{\boldsymbol{p'}_j\}$. PMP-Net++ moves each point $\boldsymbol{p}_i$ for $K = 3$ steps in total. The displacement vector for step k is denoted by $\Delta \boldsymbol{p}_i^k$, so $\Delta \boldsymbol{p}_i = \sum_{k=1}^{3} \Delta \boldsymbol{p}_i^k$. For step k, the network takes the deformed point cloud $\{\boldsymbol{p}_i^{k-1}\} = \{\boldsymbol{p}_i + \sum_{j=1}^{k-1} \Delta \boldsymbol{p}_i^j\}$ from the last step $k-1$ as input, and calculates the new displacement vector according to the input point cloud. Therefore, the predicted shape will be consistently refined step-by-step, which finally produces a complete shape with high quality.

3.1.2 Transformer-enhanced displacement prediction

At step k, in order to predict the displacement vector $\Delta \boldsymbol{p}_i^k$ for each point, we first extract per-point features from the point cloud. In the previous implementation of our PMP-Net [14], this is achieved by first adopting the basic framework of PointNet++ [15] to extract the global feature of input the 3D shape, and then using the feature propagation module to propagate the global feature to each point in the 3D space, and finally producing per-point feature $\boldsymbol{h}_i^{k,l}$ for point $\boldsymbol{p}_i^k$. In PMP-Net++, we adopt the recent implementation success of transformer [49] to enhance the point feature learned by the PointNet++, where we follow the practice of Point Transformer [16], and add an additional transformer module between each set abstraction (SA) layer of PointNet++ based encoder. The detailed structure and comparison between PMP-Net and PMP-Net++ is shown in Figure 5. Specifically, in PMP-Net++, the local features (denoted as $\{x_{in}\}$ for convenience) learned by the previous layer of set abstraction is input to the next transformer module. In transformer module, $\{x_{in}\}$ serves as both key and query for the calculation of self-attention, according to which a set of new local features $\{x_{out}\}$ are produced through

several MLPs and element-wise operations. Note that the position encoding ϵ in Figure 5 (c) is used to guide the network to learn the spatial relationships between different local features. We follow the same practice of previous work [16] to adopt the learnable position encoding, which depends on the 3D coordinates between two points p_i and p_j:

$$\epsilon = \text{MLP}(p_i - p_j|\theta_\epsilon) \tag{1}$$

Since our experimental implementation applies three levels of feature propagation to hierarchically produce per-point features (see Figure 3), we use superscript k to denote the step and the subscript l to denote the level in $\boldsymbol{h}_i^{k,l}$. The per-point feature $\boldsymbol{h}_i^{k,l}$ is then concatenated with a random noise vector $\hat{\boldsymbol{x}}$, which according to [48] can give point tiny disturbances and force it to leave its original place. Then, the final point feature $\boldsymbol{h}_i^{k,3}$ at step k and level 3 is fed into a multi-layer perceptron (MLP) followed by a hyper-tangent activation (tanh), to produce a 3-dimensional vector as the displacement vector $\Delta \boldsymbol{p}_i^k$ for point $\boldsymbol{p}_i^k$ as

$$\Delta \boldsymbol{p}_i^k = \tanh(\text{MLP}([\boldsymbol{h}_i^{k,3} : \hat{\boldsymbol{x}}])), \tag{2}$$

where ":" denotes the concatenation operation.

3.1.3 Recurrent information flow between steps

The information of previous moves is crucial for network to decide the current move, because the previous paths can be used to infer the location of the final destination for a single point. Moreover, such information can guide the network to find the direction and distance of next move, and prevent it from changing destination during multiple steps of point moving path searching. In order to achieve this target, we propose to use a special RPA unit between each step and each level of feature propagation module, which is used to memorize the information of previous path and to infer the next position of each point. As shown in Figure 3, the RPA module in (step k, level l) takes the output $\boldsymbol{f}_i^{k,l-1}$ from the last level i-1 as input, and combines it with the feature $\boldsymbol{h}_i^{k-1,l}$ from the previous step $k-1$ at the same level l to produce the feature of current level $\boldsymbol{h}_i^{k,l}$, denoted as

$$\boldsymbol{h}_i^{k,l} = \text{RPA}(\boldsymbol{f}_i^{k,l-1}, \boldsymbol{h}_i^{k-1,l}). \tag{3}$$

The detailed structure of RPA module is described below.

3.2 Recurrent Path Aggregation

The detailed structure of recurrent path aggregation module is shown in Figure 6. The previous paths of point moving can be regarded as the sequential data, where the information of each move should be selectively memorized or forgotten during the process. Following this idea, we take the inspiration from the recurrent neural network, where we mimic the behavior of gated recurrent unit (GRU) to calculate an update gate $\boldsymbol{z}$ and reset gate $\boldsymbol{r}$ to encode and forget information, which is according to the point feature $\boldsymbol{h}_i^{k-1,l}$ from the last step $k-1$ and the point feature $\boldsymbol{f}_i^{k,l-1}$ of current step k. The calculation of two gates can be formulated as

$$\boldsymbol{z} = \sigma(W_z[\boldsymbol{f}_i^{k,l-1} : \boldsymbol{h}_i^{k-1,l}] + \boldsymbol{b}_z), \tag{4}$$

$$\boldsymbol{r} = \sigma(W_r[\boldsymbol{f}_i^{k,l-1} : \boldsymbol{h}_i^{k-1,l}] + \boldsymbol{b}_r), \tag{5}$$

where W_z, W_r are weight matrix and $\boldsymbol{b}_z, \boldsymbol{b}_z$ are biases. σ is the *sigmoid* activation function, which predicts a value between 0 and 1 to indicate the ratio of information that allowed to pass the gate. ":" denotes the concatenation of two features.

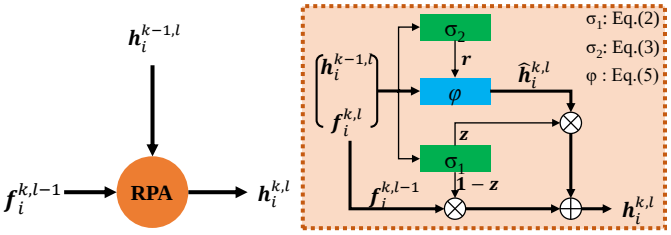


Fig. 6. Detailed structure of RPA module at step k, level l.

Different from the standard GRU, which emphasizes more importance on the preservation of previous information when calculating the output feature $\boldsymbol{h}_i^{k,l}$ at current step, in RPA, we address more importance on the preservation of current input information, and propose to calculate the output feature $\boldsymbol{h}_i^{k,l}$ as

$$\boldsymbol{h}_i^{k,l} = \boldsymbol{z} \odot \hat{\boldsymbol{h}}_i^{k,l} + (1 - \boldsymbol{z}) \odot \boldsymbol{f}_i^{k,l-1}, \tag{6}$$

where $\hat{\boldsymbol{h}}_i^{k,l}$ is the intermediate feature of current step. It contains the preserved information from the past, which is calculated according to the current input feature. The formulation of $\hat{\boldsymbol{h}}_i^{k,l}$ is given as

$$\hat{\boldsymbol{h}}_i^{k,l} = \varphi(W_h[\boldsymbol{r} \odot \boldsymbol{h}_i^{k-1,l} : \boldsymbol{f}_i^{k,l-1}] + \boldsymbol{b}_h), \tag{7}$$

where φ is *relu* activation in our implementation.

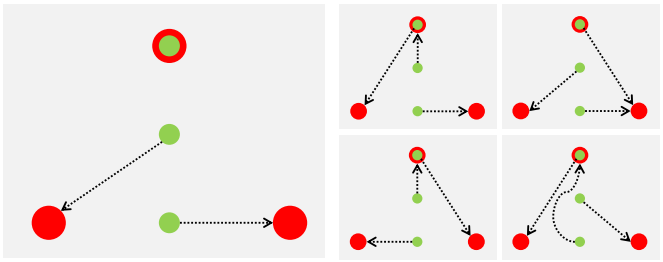
The reason of fusing $\hat{\boldsymbol{h}}_i^{k,l}$ with $\boldsymbol{f}_i^{k,l-1}$ instead of $\boldsymbol{h}_i^{k-1,l}$ is that, compared with standard unit in RNN unit, the current location of point should have greater influence to the decision of next move. Especially, when RPA module needs to ignore the previous information which is not important in the current decision making, Eq.(6) can easily allow RPA model to forget all history by simply pressing the update gate $\boldsymbol{z}$ to a zero-vector, and thus enables the RPA module fully focus on the information of current input $\boldsymbol{f}_i^{k,l-1}$.

3.3 Optimized Searching for Unique Paths

3.3.1 Minimizing moving distance

As shown in Figure 7, the unordered nature of point cloud allows multiple solutions to deform the input shape into the target one, and the direct constraint (e.g. chamfer distance) on the deformed shape and its ground truth cannot guarantee the uniqueness of correspondence established between the input point set and the target point set. Otherwise, the network will be confused by the multiple solutions of point moving, which may lead to the failure of capturing detailed topology and structure relationships between incomplete shapes and complete ones. In order to establish a unique and meaningful point-wise correspondence between input point cloud and target point cloud, we take the inspiration from Earth Mover's Distance [50], and propose to train PMP-Net++ to learn the path arrangement ϕ between source and target point clouds under the constraint of total point moving path distance. Specifically, given the source point clouds $\hat{X} = \{\hat{\boldsymbol{x}}_i | i = 1, 2, 3, ..., N\}$ and the target point cloud $X = \{\boldsymbol{x}_i | i = 1, 2, 3, ..., N\}$, we follow EMD to learn an arrangement ϕ which meets the constraint below

$$\mathcal{L}_{\text{EMD}}(\hat{X}, X) = \min_{\phi:\hat{X}\to X} \frac{1}{\hat{X}} \sum_{\hat{\boldsymbol{x}}\in\hat{X}} \|\hat{\boldsymbol{x}} - \phi(\hat{\boldsymbol{x}})\|, \tag{8}$$



(a) PMP-constrained solution. (b) Other solutions.

Fig. 7. Illustration of multiple solutions when deforming input point cloud (green) into target point cloud (red). The PMD-constraint guarantees the uniqueness of point level correspondence (a) between input and target point cloud, and filter out various redundant solutions for moving points (b).

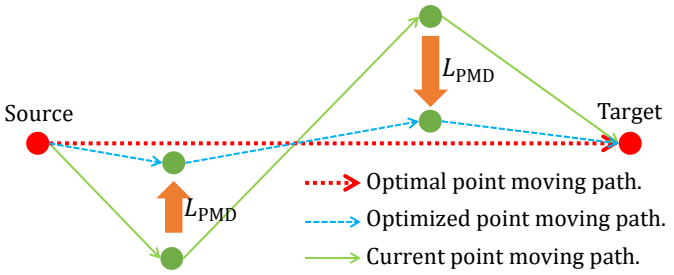


Fig. 8. Illustration of the effectiveness of $\mathcal{L}_{PMD}$. By minimizing the point moving distance, the network is encouraged to learn more consistent paths from source to target, which will reduce redundant searching in each step and improve the efficiency.

In Eq.(8), ϕ is considered as a bijection that minimizes the average distance between corresponding points in $\hat{X}$ and X.

Here, a unique correspondence between two point clouds can be guaranteed by the definition of EMD (Eq.(8)), which can be explained as follows: (1) deforming point cloud A into the shape of point cloud B (with the same number of points) equals to establish a bijection ϕ between point clouds A and B; (2) the bijection ϕ is unique when the sum of point displacement reaches the minimum value; (3) the optimization of Eq.(8) is to minimize the sum of point displacement, which will yield a unique correspondence, following the data order determined by the input point cloud.

According to Eq.(8), bijection ϕ established by the network should achieve the minimum moving distance to move points from input shape to target shape. However, even if the correspondence between input and target point clouds is unique, there still exist various paths between source and target points, as shown in Figure 8. Therefore, in order to encourage the network to learn an optimal point moving path, we choose to minimize the point moving distance loss ($\mathcal{L}_{\text{PMD}}$), which is the sum of all displacement vector $\{\Delta \boldsymbol{p}_i^k\}$ output by all three steps in PMP-Net. The *Point Moving Distance* loss is formulated as

$$\mathcal{L}_{\text{PMD}} = \sum_k \sum_i \|\Delta \boldsymbol{p}_i^k\|_2. \tag{9}$$

Eq.(9) is more strict than EMD constraint. It requires not only the overall displacements of all point achieve the shortest distance, but also limits the point moving paths in each step to be the shortest one. Therefore, in each step, the network will be encouraged to search new path following the previous direction, as shown in Figure 8, which will lead to less redundant moving decision and improve the searching efficiency.

3.3.2 *Multi-scaled searching radius*

PMP-Net++ searches the point moving path in a coarse-to-fine manner. For each step, PMP-Net++ reduces the maximum stride to move a point by the power of 10, which is, for step k, the displacement $\Delta \boldsymbol{p}_i^k$ calculated in Eq.(2) is limited to $10^{-k+1}\Delta \boldsymbol{p}_i^k$. This allows the network converges more quickly during training. And also, the reduced searching range will guarantee the network at next step not to overturn its decision made in the previous step, especially for the long range movements. Therefore, it can prevent the network from making redundant decision during path searching process.

3.4 Extension to Dense Point Cloud Completion

The deformation of point cloud cannot be directly used for increasing the number of points. Therefore, the input and the output point cloud must have the same resolution. Such characteristic of deformation based PMP-Net++ may become a problem when tackling 3D shapes with various number of points. In order to solve this problem, we propose to extend the PMP-Net++ to dense point cloud completion scenarios by adding noise to the input of each step. To achieve this goal, in step k, the input point cloud $\{\boldsymbol{p}_i^k\}$ is concatenated with a noise vector $\hat{\boldsymbol{n}}$ sampled from a standard normal distribution $\mathrm{N}(0, 1)$ as:

$$\{\boldsymbol{p}_i^k\} \leftarrow \{[\boldsymbol{p}_i^k : \hat{\boldsymbol{n}}]\}, \hat{\boldsymbol{n}} \sim \mathrm{N}(0, 1) \tag{10}$$

As a result, in each time, the point cloud varies from its previous data when input to the network. Then, by deforming point clouds multiple times and overlapping the deformation results together, we can get a dense point cloud with more points than the original input.

Note that different from the noise in Eq. 2, which aims to push the points away from their original position in 3D space, the noise used in the input of each step aims to vary the final deformation results.

More discussion. The key idea of extension to dense point cloud completion is to increase the number of points. Such practice is commonly adopted in generation based methods. For PMP-Net, we duplicate points and concatenate with random noise to increase the number of points. And for the other methods like TopNet [13] and PCN [12], they also need to duplicate features and concatenate with 2D grid code to increase points. The difference between PMP-Net++ and the other methods is that the duplication operation is settled at different stages in the network.

Note that if we simply repeat point coordinates, and move them without additional operations, the duplicated points will produce exactly the same displacement. Therefore, to solve this problem, in PMP-Net++, we add noise on each point features to force them moving to different places. Because the noise-enhanced input points have already been located at different spatial locations before the movement, these duplications will not be moved to the same target points.

3.5 Training Loss

The deformed shape is regularized by the complete ground truth point cloud through Chamfer distance (CD) and Earth Mover

Distance (EMD). Following the same notations in Eq.(8), the Chamfer distance is defined as:

$$\mathcal{L}_{CD}(X, \hat{X}) = \sum_{\boldsymbol{x} \in X} \min_{\hat{\boldsymbol{x}} \in \hat{X}} \|\boldsymbol{x} - \hat{\boldsymbol{x}}\| + \sum_{\hat{\boldsymbol{x}} \in \hat{X}} \min_{\boldsymbol{x} \in X} \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|. \quad (11)$$

The total loss for training is then given as

$$\mathcal{L} = \sum_{k} \mathcal{L}_{\mathrm{CD}}(P^k, P') + \mathcal{L}_{\mathrm{PMD}}, \quad (12)$$

where P^k and P' denote the point cloud output by step k and the target complete point cloud, respectively. Note that finding the optimal ϕ is extremely computational expensive. In experiments, we follow the simplified algorithm in [12] to estimate an approximation of ϕ.

4 Experiments

In this section, we first evaluate PMP-Net++ on general completion benchmark *PCN* [12] and *Completion3D* [13]. Then we further explore the potential application of PMP-Net++ on the point cloud up-sampling task. Finally, the effectiveness of each part of PMP-Net++ will be quantitatively evaluated through comprehensive ablation studies.

4.1 Detailed Settings

We use the single scale grouping (SSG) version of PointNet++ and its feature propagation module as the basic framework of PMP-Net. The detailed architecture of each part is described in Table 1 and Table 2, respectively.

TABLE 1
The detailed structure of encoder.

Level	#Points	Radius	#Sample	MLPs
1	512	0.2	32	[64, 64, 128]
2	128	0.4	32	[128, 128, 256]
3	-	-	-	[256, 512, 1024]

In Table 1, "#Points" denotes the number of down-sampled points, "Radius" denotes the radius of ball query, "#Sample" denotes the number of neighbors points sampled for each center point, "MLPs" denotes the number of output channels for MLPs in each level of encoder.

TABLE 2
The detailed architecture of feature propagation module.

Level	1	2	3
MLPs	[256, 256]	[256, 128]	[128, 128, 128]

We use AdamOptimizer to train PMP-Net++ with an initial learning rate 10^{-3}, and exponentially decay it by 0.5 for every 20 epochs. The training process is accomplished using a single NVDIA GTX 2080TI GPU with a batch size of 24. PMP-Net++ takes 150 epochs to converge on both PCN and Completion3D dataset. We scale all input training shapes of Completion3D by 0.9 to avoid points out of the range of *tanh* activation.

Note that currently the transformer is only involved in encoder, and we do not observe significant performance gain when adding transformer in decoder network. Since the key operation of feature propagation module (FP-module in decoder) is the trilinear interpolation (which is not learnable), the quality of point features produced by the decoder are mainly relies on the encoder. Therefore, additional transformer in decoder may not help the network too much to enhance the quality of point features.

4.2 Point Cloud Completion on PCN Dataset

4.2.1 Dataset and evaluation metric

We show that PMP-Net++ learned on sparse point cloud can be directly applied to the dense point cloud completion. Specifically, we keep training PMP-Net++ on sparse shape with 2,048 points, and reveal its generalization ability by predicting dense complete shape with 16,384 points on PCN dataset [12]. PCN dataset is derived from ShapeNet dataset, in which each complete shape contains 16,384 points. The partial shapes have various point numbers, so we first down-sample shapes with more than 2,048 points to 2,048, and up-sample shapes with less than 2048 points to 2048 by randomly copying points. Since PMP-Net++ learns to move points instead of generating points, it requires the same number of points in incomplete point cloud and complete one. In order to predict complete shape of 16,384 points, we repeat 8 times of prediction for each shape during testing, with each time moving 2,048 points. Note that PMP-Net++ is still trained on sparse point clouds with 2,048 points, which are sampled from the dense point clouds of PCN dataset.

On PCN dataset, we use the per-point L1 Chamfer distance (CD) as the evaluation metric, which is the CD in Eq.(11) averaged by the point number.

4.2.2 Quantitative comparison

The comparison in Table 3 shows that PMP-Net++ yields a comparable performance to the state-of-the-art method [47], and ranks first on PCN dataset. The result of Wang et al. [47] is cited from its original paper, while the results of other compared methods are all cited from [40]. Note that most generation based methods (like Wang et al. [47] and GRNet [40] in Table 3) specially designed a coarse-to-fine generation process in order to obtain better performance on dense point cloud completion. In contrast, our PMP-Net++ trained on 2,048 points can directly generate arbitrary number of dense points by simply repeating the point moving process, and still achieves comparable results to the counterpart methods. Moreover, we further discuss the effectiveness of PMD loss by comparing the baseline PMP-Net++ with *no PMD* variation, which we remove the PMD loss during training. From Table 3, we can find that PMD loss effectively improves the performance of PMP-Net++, which is in accordance with our opinion that point moving path should be regularized to better capture the detailed topology and structure of 3D shapes.

4.2.3 Qualitative comparison

In Figure 9 and Figure 10, we further demonstrate the advantage of PMP-Net++ over other methods, by visually compare the completion results on PCN dataset. Specifically, in Figure 9, we compare our PMP-Net++ with other counterparts cross different object categories. For example, in the third row of Figure 9, the task is to predict the complete shape of an incomplete lamp. In this task, most of the evaluated methods failed to preserve the detailed geometries of the lamppost, where many noise points are distributed around the lamppost. Compared with the generative methods in Figure 9, our PMP-Net++ can reveal a high quality lamppost. Moreover, when comparing the PMP-Net++ with PMP-Net, we can find that PMP-Net++ achieves a better shape prediction at the bottom and the top of the lamppost. The advantages of

TABLE 3
Point cloud completion on PCN dataset in terms of per-point L1 Chamfer distance $\times 10^3$ (lower is better).

Methods	Average	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Watercraft
FoldingNet [51]	14.31	9.49	15.80	12.61	15.55	16.41	15.97	13.65	14.99
TopNet [13]	12.15	7.61	13.31	10.90	13.82	14.44	14.78	11.22	11.12
AtlasNet [52]	10.85	6.37	11.94	10.10	12.06	12.37	12.99	10.33	10.61
PCN [12]	9.64	5.50	22.70	10.63	8.70	11.00	11.34	11.68	8.59
GRNet [40]	8.83	6.45	10.37	9.45	9.41	7.96	10.5	8.44	8.04
CRN [47]	8.51	4.79	9.97	**8.31**	9.49	8.94	10.69	7.81	8.05
NSFA [33]	8.06	4.76	10.18	8.63	8.53	7.03	10.53	7.35	7.48
PMP-Net [14]	8.66	5.50	11.10	9.62	9.47	6.89	10.74	8.77	7.19
PMP-Net++(no PMD, Ours)	7.74	4.69	10.15	8.78	8.30	6.08	10.28	7.23	6.61
PMP-Net++(Ours)	**7.56**	**4.39**	**9.96**	8.53	**8.09**	**6.06**	**9.82**	**7.17**	**6.52**

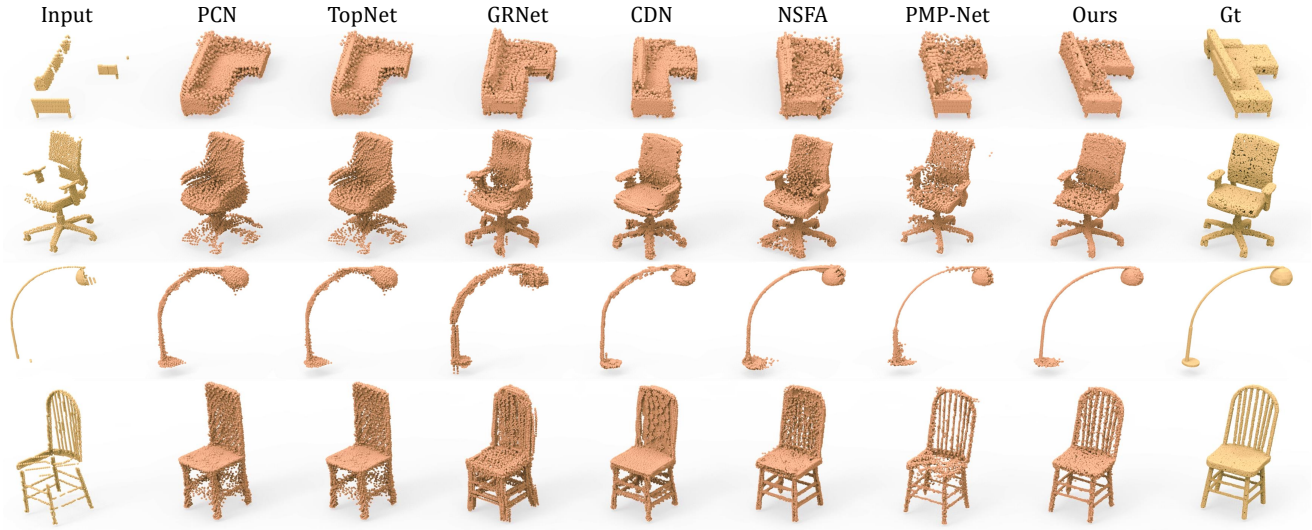


Fig. 9. Visualization of point cloud completion comparison with previous methods on PCN dataset.

Fig. 10. Visualization of more completion results using our PMP-Net++ on PCN dataset.

inferring and preserving detailed shapes of PMP-Net++ can also be well proved by the observation of the fourth row: by comparing the reconstruction results of the chair back, we can find that PMP-Net++ can clearly preserve the detailed shapes of each beam on the chairback; by comparing the completion results of the chair legs, the results of PMP-Net++ is also closer to the ground truth than its any other counterparts. Moreover, the comparison of the sofa between the PMP-Net and PMP-Net++ visually reveals the improvements of PMP-Net++ over the PMP-Net. The complete sofa predicted by PMP-Net distributes less points in the missing region of the sofa than the PMP-Net++.

4.3 Point Cloud Completion on Completion3D Dataset

4.3.1 *Dataset and evaluation metric*

We evaluate our PMP-Net++ on the widely used benchmark of 3D point cloud completion, i.e. *Completion3D* [13], which is a large-scaled 3D object dataset derived from the ShapeNet dataset. The partial 3D shapes are generated by back-projecting 2.5D depth images from partial views into 3D space. Completion3D dataset concerns the completion task of sparse point cloud completion, where it generates only one partial view for each complete 3D object in ShapeNet dataset, and samples 2,048 points from the mesh surface for both the complete and partial shapes. We follow

TABLE 4
Point cloud completion on Completion3D dataset in terms of per-point L2 Chamfer distance $\times 10^4$ (lower is better).

Methods	Average	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Watercraft
FoldingNet [51]	19.07	12.83	23.01	14.88	25.69	21.79	21.31	20.71	11.51
PCN [12]	18.22	9.79	22.70	12.43	25.14	22.72	20.26	20.27	11.73
PointSetVoting [53]	18.18	6.88	21.18	15.78	22.54	18.78	28.39	19.96	11.16
AtlasNet [52]	17.77	10.36	23.40	13.40	24.16	20.24	20.82	17.52	11.62
SoftPoolNet [54]	16.15	5.81	24.53	11.35	23.63	18.54	20.34	16.89	7.14
TopNet [13]	14.25	7.32	18.77	12.88	19.82	14.60	16.29	14.89	8.82
SA-Net [1]	11.22	5.27	14.45	7.78	13.67	13.53	14.22	11.75	8.84
GRNet [40]	10.64	6.13	16.90	8.27	12.23	10.22	14.93	10.08	5.86
CRN [47]	9.21	3.38	13.17	8.31	10.62	10.00	12.86	9.16	5.80
PMP-Net [14]	9.23	3.99	14.70	8.55	10.21	9.27	12.43	8.51	5.77
VRCNet [46]	8.12	3.94	13.46	**6.72**	10.35	9.87	12.48	7.73	6.14
PMP-Net++(Ours)	**7.97**	**3.25**	**12.25**	7.62	**8.71**	**7.64**	**11.6**	**7.06**	**5.38**

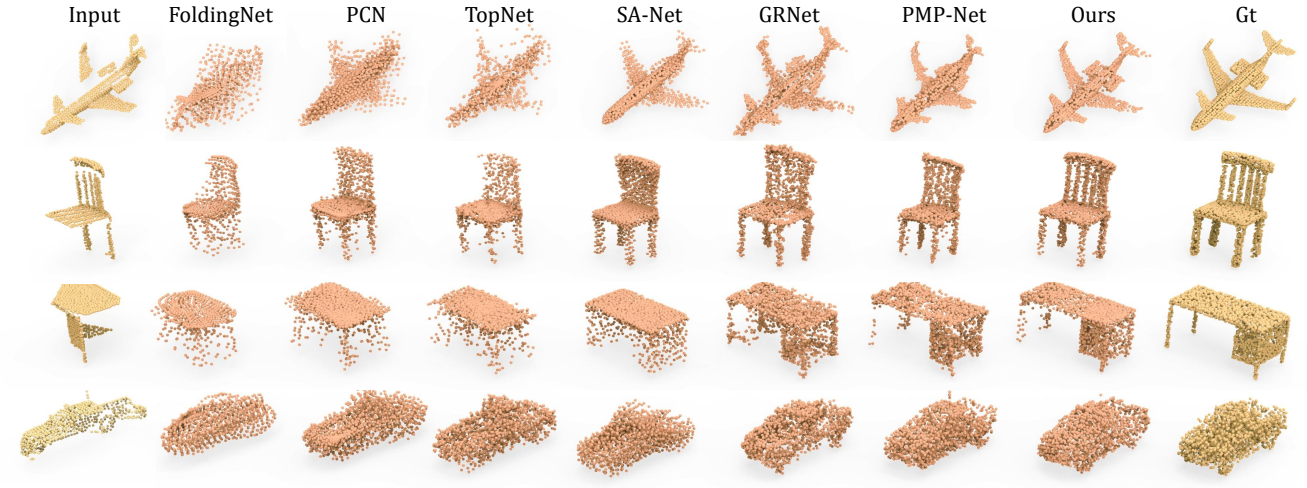


Fig. 11. Visualization of point cloud completion comparison with previous methods on Completion3D dataset.

the settings of training/validation/test splits in Completion3D for fair comparison with the other methods.

Following previous studies [1], [12], [13], [40], we use the per-point L2 Chamfer distance (CD) as the evaluation metric on Completion3D dataset. L2 Chamfer distance is the CD in Eq.(11) averaged by the point number, where the L1-norm in Eq.(11) is replaced by L2-norm.

4.3.2 Quantitative comparison

The quantitative comparison results[1] of PMP-Net++ with the other state-of-the-art point cloud completion methods are shown in Table 4, in which the PMP-Net++ achieves the best performance in terms of average chamfer distance across all categories. The second best published method on Completion3D leaderboard is VRCNet [46], which achieves 8.12 in terms of average CD, and PMP-Net++ improves such state-of-the-art performance by 0.17 (also in terms of average CD). When considering per-category performance, PMP-Net++ achieves the best results in 7 out of 8 categories across all compared counterpart methods, which justifies the better generalization ability of PMP-Net++ across different shape categories. Note that, compared with PMP-Net, PMP-Net++ significantly reduces the average CD loss on Completion3D dataset by 13.8%, and outperforms the PMP-Net on all 8 categories in terms of per-category CD. As we discussed in Section 2, GRNet [40] is a voxel aided shape completion method, where the completion process utilizes information from two different data modalities (i.e point cloud and voxels). The better performance of PMP-Net++ over GRNet proves the effectiveness of our method, which can exploit the abundant geometric information lying in the point clouds. Other methods like SA-Net [1] in Table 4 are typical generative completion methods which are fully based on point clouds, and the nontrivial improvement of PMP-Net++ over these methods justifies the effectiveness of deformation based solution in point cloud completion task.

4.3.3 Qualitative comparison

In Figure 11, we visually compare PMP-Net++ with the other completion methods on Completion3D dataset, from which we can find that PMP-Net++ predicts much more accurate complete shapes on various shape categories, while other methods may output some failure cases for certain input shapes. For example, the input *table* in Figure 11 (the third row) loses half of its legs and surface. The completion results made by GRNet and PMP-Net almost predict the right overall shape of the complete desktop, but fail to reconstruct the detailed structure like clean legs and smooth surface. On the other hand, methods like FoldingNet [51], PCN [12], TopNet [13] and SA-Net [1] intend to repair the desktop but fail to predict a complete overall shape. Moreover, the advantage of deformation based PMP-Net++ over the generative methods can be well proved by the case of *chair* in Figure 11 (the second row). Generative methods, especially like GRNet, successfully learn the complete structure of the input chair, but fail

1. Results are cited from https://completion3d.stanford.edu/results

Fig. 12. More completion results using our PMP-Net++ on Completion3D dataset.

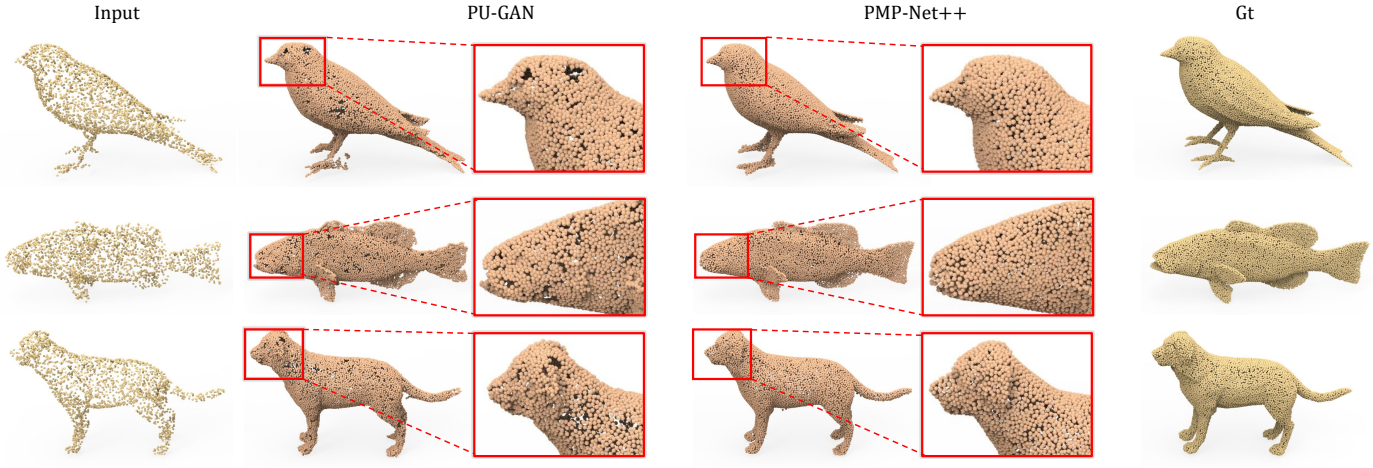


Fig. 13. Visualization of point cloud up-sampling. We typically compare our PMP-Net++ with the state-of-the-art up-sampling method PU-GAN [55], and demonstrate the advantages of PMP-Net++ for generating better detailed shapes.

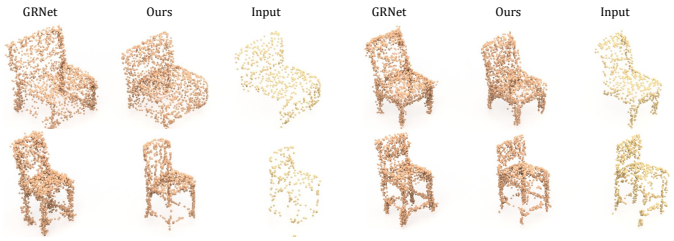


Fig. 14. Visual comparison of PMP-Net++ and GRNet on ScanNet chairs.

to reconstruct the columns on the chair back, which is the residual part of input shape. On the other hand, the deformation based PMP-Net++ can directly preserve the input shape by moving just a small amount of point to perform completion on certain areas, and keep the input shape unchanged. In Figure 12, we further visualize more completion results of PMP-Net.

4.3.4 *Extension to ScanNet chairs*

To evaluate the generalization ability of PMP-Net++ on point cloud completion task, we pre-train PMP-Net++ on Completion3D dataset and evaluate its performance on the chair instances in ScanNet dataset without finetuning, and compare with GRNet (which is the second best method in Table 4, and also pre-trained on Completion3D). The visual comparison is shown in Figure 14. Since there is no ground truth for ScanNet dataset, we typically follow [12] to report partial metrics (Fidelity and MMD) in Table 5 based on the selected chairs of Figure 14. The PMP-Net++ completes shapes with less noise than GRNet, which benefits from its point moving practice. Because the differences in data distribution between Completion3D and ScanNet will inevitably confuse the network, and the point moving based PMP-Net++ can simply choose to leave those points in residual part of an object to stay at their own place to preserve a better shape, in contrast, generation based GRNet has to predict new points for both residual and missing part of an object.

TABLE 5
Quantitative evaluation of ScanNet chairs.

Methods	Fidelity(10^4)	MMD(10^3)
GRNet [40]	2.95	3.17
PMP-Net++	2.05	2.65

4.4 Point Cloud Up-sampling

4.4.1 *Dataset and evaluation metric*

In this section, we experimentally demonstrate the effectiveness of PMP-Net++ on other similar task, i.e. the point cloud up-sampling task. The scenario encountered by PMP-Net++ on this task is the same as the dense point cloud completion, where the difficulty is to increase the number of points in order to reveal more detailed geometric information of 3D model. Therefore, we adopt the dense

completion version of PMP-Net++ to the up-sampling task. For fair comparison with the previous counterpart methods, we follow the same practice of PU-GAN to use its dataset and experimental settings for evaluation. According to PU-GAN, the dataset is a collection of 147 models from the dataset of PU-Net, MPU and Vision-air repository, where 120 models are selected for training and the rest 27 models are used for testing. The commonly used Chamfer distance (CD) and Hausdorff distance are adopted as our evaluation metric.

4.4.2 Quantitative and qualitative comparison

The quantitative comparison is given in Table 6, from which we can find that our PMP-Net++ achieves the best performance among the compared counterparts. Note that, in point cloud up-sampling task, we use exact the same network settings and structures as the point cloud completion on PCN dataset. Therefore, the better results on point cloud up-sampling of our network proves the generalization ability of PMP-Net++ to various tasks.

TABLE 6
Quantitative comparison on point cloud up-sampling task.

Methods	CD(10^{-3})	HD(10^{-3})
EAR [56]	0.52	7.37
PU-Net [57]	0.72	8.94
MPU [58]	0.49	6.11
PU-GAN [55]	0.28	4.64
PMP-Net++(Ours)	**0.26**	**4.63**

In Figure 13, we further illustrate the better performance of PMP-Net++ by visually compare our network with PU-GAN. For example, in the completion of the head of the bird (first row of Figure 13), the up-sampling results of PMP-Net++ distribute the points more evenly than the PU-GAN. From the results of PU-GAN, we can still observe several holes on the bird's head, while on the result of PMP-Net++, there is no such incomplete region appearing on the surface of the bird's head. Moreover, the up-sampled points using PMP-Net++ are distributed more evenly compared with the PU-GAN, especially on the head of the fish in the second row of Figure 13.

4.5 Model Analysis

In this subsection, we analyze the influence of different parts in the PMP-Net++, and compared it with PMP-Net to analysis the effectiveness of the incremental contribution. By default, we use the same network sittings in PMP-Net [14] for all experiments, where all studies are typically conducted on the validation set of Completion3D dataset under four categories (i.e. plane, car, chair and table) for convenience.

4.5.1 Analysis of RPA module and PMP loss

We analyze the effectiveness of RPA module by replacing it with other units in PMP-Net++. And for PMP loss, we analyze its effectiveness by removing PMP loss from the network. Specifically, we develop six different variations for comparison: (1) *NoPath* is the variation that removes the RPA module from the network; (2) *Add* is the variation that replaces RPA module with element-wise add layer in the network; (3) RNN, (4) LSTM and (5) GRU are variations that replace RPA module with different recurrent unit.

The shape completion results are shown in Table 7, in which we report the results of both PMP-Net and PMP-Net++ for a comprehensive comparison. From Table 7, we can find that two baseline variations (which uses the RPA module) achieve the best performance on two backbones, respectively, which justifies the effectiveness of the proposed RPA module across different network structure. The worst result across different kinds of unit is yielded by Add variation under the PMP-Net backbone, while under the PMP-Net++ backbone, the worst result is yielded by the RNN unit. The different performance of Add variation and GRU variation cross different backbones indicates that, recurrent network structures cannot provide a robust aggregation of sequential information, which is generated during the path searching process of PMP-Net++ framework. Moreover, since the RPA module is originated from the GRU unit, the comparisons between RPA baseline and GRU variation on two backbones justify the effectiveness of our designation of RPA module, which can give more consideration to the information from current step than GRU unit, and help the network to make more precise decision for point moving.

The effectiveness of newly added transformer unit can be fully justified by the comparison between the backbones of PMP-Net and PMP-Net++, where the best performance with the PMP-Net backbones is still worse than the worst performance of PMP-Net++, no matter which kind of variation they use.

TABLE 7
Analysis of RPA and PMP loss (baseline marked by "*").

Backbone	Unit.	avg.	plane	chair	car	table
PMP-Net	NoPath	11.95	3.55	8.30	16.15	19.79
	Add	12.23	**3.32**	16.47	8.10	21.05
	RNN	12.12	3.55	16.19	8.14	20.58
	LSTM	11.99	3.79	15.37	8.09	20.72
	GRU	11.87	3.44	15.44	7.85	20.72
	baseline*	**11.58**	3.42	**15.88**	**7.87**	**19.15**
PMP-Net++	NoPath	8.09	2.47	10.8	6.65	12.31
	Add	8.07	**2.39**	10.90	6.54	12.40
	RNN	8.54	2.46	11.70	6.86	13.00
	LSTM	8.30	2.47	10.92	6.69	13.1
	GRU	8.27	2.40	11.31	6.63	12.62
	baseline*	**8.03**	2.58	**10.70**	**6.44**	**12.30**

4.5.2 Effect of multi-step path searching

In Table 8, we analyze the effect of different steps for point cloud deformation, and we also compare the performance between PMP-Net and PMP-Net++. Specifically, the ratio of searching radius between each step is fixed to 10, and then, the number of steps to deform the point clouds is set to 1, 2 and 4, respectively. For example, when the step is set to 4, the corresponding searching radius is $\{1.0, 10^{-1}, 10^{-2}, 10^{-3}\}$. And the searching radius for step=2 is set to $\{1.0, 10^{-1}\}$. By comparing the results of step 1, 2 and 3 from Table 8, we can find that under both backbones, deforming point cloud by multiple steps effectively improves the completion performance. On the other hand, the comparison between step 3 and step 4 shows that the performance of multi-step path searching will reach its limitation, because too many steps may cause information redundancy in path searching. It can also be noticed that the gap between 1 step v.s. 2 step for PMP-Net++ (9.45 v.s. 8.36) is much larger than PMP-Net (12.26 v.s. 11.90). This can be dedicated to the point transformer enhanced module, which enables PMP-Net++ to predict more accurate shape completion than PMP-Net. Therefore, based on the output of step 1, where PMP-Net++ yields a CD of 9.45 and PMP-Net is 12.26

according to Table 8, at step 2, the point transformer enhanced module of PMP-Net++ can learn more efficient geometric information, which achieves better performance (9.45 vs. 8.36) than PMP-Net (12.26 vs. 11.90).

TABLE 8
The effect of different steps (baseline marked by "*").

Backbone	Steps.	avg.	plane	chair	car	table
PMP-Net	1	12.26	3.71	15.59	8.27	21.48
	2	11.90	3.47	15.66	7.95	20.53
	3*	**11.58**	3.42	**15.88**	**7.87**	**19.15**
	4	11.67	**3.39**	15.89	7.91	19.48
PMP-Net++	1	9.45	2.64	12.90	7.26	14.90
	2	8.36	2.61	11.63	6.66	12.50
	3*	**8.03**	2.58	10.70	**6.44**	**12.30**
	4	8.05	**2.32**	**10.61**	6.51	12.60

4.5.3 *Visual analysis of multi-step searching*

We visualize the point deformation process under different searching step sittings in Figure 15. Comparing the 3-step searching in the top-row with the other two sittings, the empty space on the chair back is shaped cleaner as highlighted by rectangles, which justifies the effectiveness of multi-step searching to consistently refine the shape. Moreover, from the visualization of Figure 15, we also find that the network can actually use some edge information to infer the destination of moving point cloud. Take the chair in Figure 15(b) as an example, we can draw two conclusions as follows.

- In Region 1 of Figure 15(b), the network moves the points on the chair back to complete the missing upper part, where the points on the edge of chair back are moved above, because they are geometrically closer to the missing part. Such point moving pattern utilizes the edge information of incomplete shape.
- In Region 2 of Figure 15(b), the network moves the point from another chair legs to complete the missing legs. We can find that the point moving paths are almost parallel to each other, where each point on one leg is moved to the same place on the other leg. The point moving pattern in Region 2 clearly follows the one-to-one geometric correspondence between the points on two legs.

In all, the network not only learns the one-to-one geometric correspondence between the points, but also takes the geometric distance between the border of incomplete shape and its missing part into account during completion. Considering that the PMD loss focuses on regularizing the geometric correspondence, and there is no constraints for learning the border information, the improvements of PMP-Net++ mainly comes from the geometric correspondence.

4.5.4 *The shape to apply deformation*

Deforming shapes from incomplete point cloud is not the only choice for PMP-Net++. Therefore, we provide and discuss one more possible choice of shape deformation, which is to deform 512 grid points (aranged as $8 \times 8 \times 8$ cubic) into a complete shape. Specifically, we use one-step version of PMP-Net++ for convenience, and use trilinear interpolation to propagate point cloud features onto these grid points. In order to generate 2048 points, we duplicate each grid point 4 times and concatenate each duplication with a random noise. Thus, each grid point will be encouraged to be split into 4 sub-points, and the all 512 grid points will finally output 2048 points. The performance and visualization are given in Table 9 and Figure 16, respectively.

Table 9 shows that grid points yield 9.98 of the average CD, which is lower but relatively comparable to PMP-Net++. We believe the effectiveness of grid points comes from the ordered data predictions, and the better performance of PMP-Net++ comes from utilizing more geometric information, which is provided by the deformation process from the incomplete point cloud.

TABLE 9
Comparison with deformation from grid points.

Methods.	avg.	plane	chair	car	table
PMP-Net++(one step)	9.45	2.64	12.90	7.26	14.90
Grid	9.98	3.34	13.71	7.45	15.42

4.5.5 *Analysis of searching radius*

By default, we decrease the searching radius for each step by the ratio of 10. In Table 10, we analyze different sittings of searching radius and evaluate their influence to the performance of PMP-Net++ framework. We additionally test two different strategies to perform point moving path searching, i.e. the strategy without decreasing searching radius ($[1.0, 1.0, 1.0]$ for each step), and the strategy with smaller decreasing ratio ($[1.0, 0.5, 0.25]$). The baseline result is the default setting of PMP-Net++ ($[1.0, 0.1, 0.01]$ for each step). Table 10 shows that both PMP-Net and PMP-Net++ achieve the worst performance at $[1.0, 1.0, 1.0]$. This is a non-decrease strategy where the searching radius in each step weighs the same to the network, and the better experimental results achieved by the other two variations justify the effectiveness of the strategy to decrease searching radius. And when comparing strategy of $[1.0, 0.5, 0.25]$ with $[1.0, 0.1, 0.01]$, we can find that decreasing searching radius with larger ratio can improve the model performance, because larger ratio can better prevent the network from overturning the decisions in previous steps. We also note that when the decreasing ratio becomes large, the PMP-Net++ will approximate the behavior of network with step=1 in Table 8, which can in return be harmful to the performance of shape completion.

TABLE 10
The effect of searching radius (baseline marked by "*").

Backbone	Radius.	Avg.	Plane	Chair	Car	Table
PMP-Net	$[1.0, 1.0, 1.0]$	12.01	3.61	16.44	8.22	19.79
	$[1.0, 0.5, 0.25]$	11.77	**3.36**	15.92	8.01	19.79
	$[1.0, 0.1, 0.01]$*	**11.58**	3.42	**15.88**	**7.87**	**19.15**
PMP-Net	$[1.0, 1.0, 1.0]$	8.69	2.58	12.38	6.86	12.90
	$[1.0, 0.5, 0.25]$	8.54	**2.33**	11.33	6.66	13.79
	$[1.0, 0.1, 0.01]$*	**8.03**	2.58	**10.70**	**6.44**	**12.30**

4.5.6 *Visual analysis of point moving path under different radius*

In Figure 17, we visualize the searching process under different strategies of searching radius in Table 8. By analyzing the deformation output in step 1, we can find that PMP-Net++ with a coarse-to-fine searching strategy can learn to predict a better shape at early step, where the output of step 1 in Figure 17 (a) is more complete and tidy than the ones in Figure 17 (b) and Figure 17 (c).

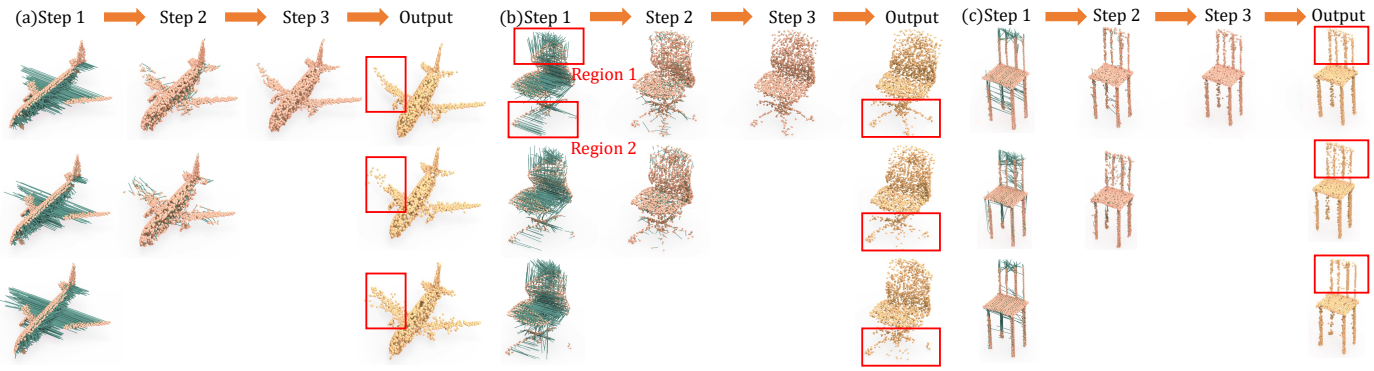


Fig. 15. Illustration of multi-step searching under different searching steps. The first row is 3-step completion, and the second row is 2-step completion, and so on. The 4-step completion have the similar visual effects as 3-step completion. Due to very short searching radius for visualizing the displacement at 4-th step, we only illustrate step 1, 2 and 3 in this figure.

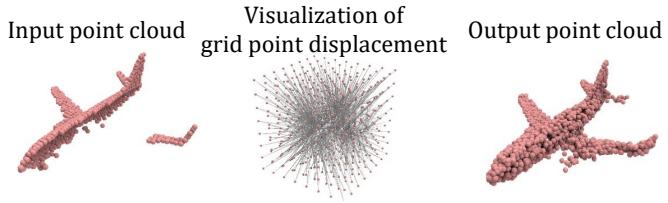


Fig. 16. Illustration of deformation from grid point of size $8 \times 8 \times 8$.

Moreover, a better overall shape predicted in the early stage will enable the network focus on refining a better detailed structure of point cloud, which can be concluded from the comparison of step 3 in Figure 17, where the region highlighted by red rectangles in Figure 17 (a) is much better than the other two subfigures.

4.5.7 Dimension of noise vector

The noise vector in Eq.(1) in our paper is used to push the points to leave their original place. In this section, we analyze the dimension and the standard deviation of the noise, which may potentially decide the influence of the noise to the points. Because either the dimension or the standard deviation of the noise vector decreases to 0, there will be no disturbance in the network. On the other hand, larger vector dimension or standard deviation will cause larger disturbance in the network. In Table 11, we first analyze the influence of dimension of noise vector. By comparing 0-dimension result with others, we can draw conclusion that the disturbance caused by noise vector is important to learn the point deformation. And by analyzing the performance of different length of noise vector, we can find that the influence of vector length is relatively small, compared with the existence of noise vector.

TABLE 11
The effect of noise dimension (baseline marked by "*").

Backbone	Dim.	Avg.	Plane	Car	Chair	Table
PMP-Net	0	14.56	4.39	10.48	19.01	24.33
	8	11.85	3.28	7.95	15.65	20.50
	16	11.68	3.44	**7.86**	**15.22**	20.19
	32*	**11.58**	3.42	7.87	15.88	**19.15**
	64	**11.58**	**3.14**	7.96	16.01	19.17
PMP-Net++	0	11.10	3.33	8.66	14.63	17.68
	8	8.20	2.23	6.53	10.80	13.10
	16	8.24	2.46	6.66	10.72	13.13
	32*	**8.03**	2.58	**6.44**	**10.70**	**12.30**
	64	8.11	**2.39**	6.50	10.80	12.69

4.5.8 Standard deviation of noise distribution

In Table 12, we show the completion results of PMP-Net++ under different standard deviations of noise vector. Similar to the analysis of vector dimension, we can draw conclusion that larger disturbance caused by bigger standard deviation will help the network achieve better completion performance. The influence of noise vector becomes weak when the standard deviation reaches certain threshold (around 10^{-1} according to Table 12).

TABLE 12
The effect of standard deviation (baseline marked by "*").

Backbone	Stddev.	Avg.	Plane	Car	Chair	Table
PMP-Net	10^{-2}	11.89	3.32	8.15	16.42	19.58
	10^{-1}	**11.56**	3.58	7.78	15.47	19.41
	1.0*	11.58	3.42	**7.87**	15.88	**19.15**
	10	11.62	**3.35**	7.88	**15.29**	19.95
PMP-Net++	10^{-2}	8.08	2.74	6.54	**10.60**	12.31
	10^{-1}	8.43	2.35	6.56	10.81	13.89
	1.0*	**8.03**	2.58	**6.44**	10.70	**12.30**
	10	8.25	**2.27**	6.49	11.30	12.82

4.5.9 Efficiency analysis

The efficiency analysis on Completion3D dataset is given in Table 13 We can find that PMP-Net++ is more efficient than GRNet in terms of both parameters and FLOPs, which proves our opinion that voxel-aided methods may suffer from the computational inefficiency problem. Moreover, the efficiency of PMP-Net++ is also comparable with PCN. Since PCN adopts a linear layer to generate coarse point cloud in its network designation, it still requires lots of computational resources even in single-step point cloud completion

TABLE 13
Comparison with deformation from grid points.

Methods	PCN	GRNet	PMP-Net++(3 steps)
Params(M)	5.29	76.7	5.89
FLOPs(G)	3.31	18.6	4.48

5 Conclusions

In this paper, we propose a novel PMP-Net++ for point cloud completion by multi-step shape deformation. By moving points from the source to the target point clouds with multiple steps,

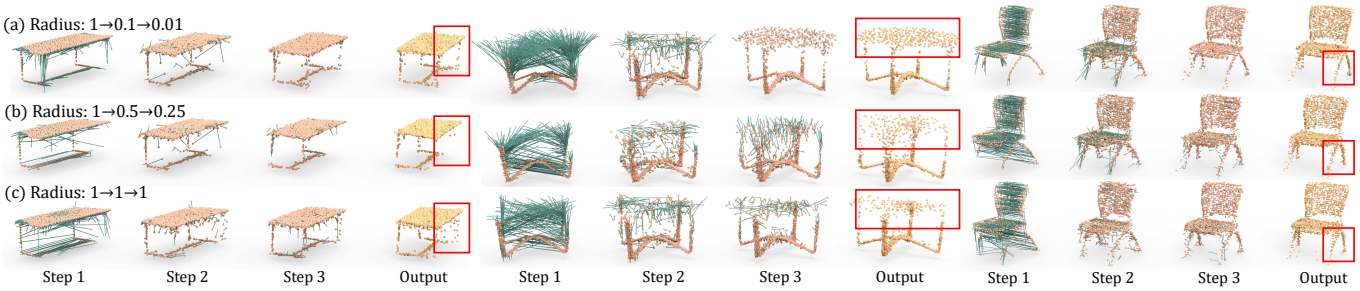


Fig. 17. Illustration of deformation process in each step under different strategies of searching radius.

PMP-Net++ can consistently refine the detailed structure and topology of the predicted shape, and establish the point-level shape correspondence between the incomplete and the complete shape. In experiments, we show the superiority of PMP-Net++ by comparing with other methods on the Completion3D benchmark and PCN dataset, and also demonstrate its good performance in the point cloud up-sampling task.

In all, the research of PMP-Net++ is somewhat limited by the lack of effective constraints to the deformation process. In this paper, although the PMD loss has been proposed to regularize the moving distances of all points in 3D space, the results may not be satisfactory in some examples due to the lack of supervision like point moving directions or final destinations. Therefore, the network may have difficulty to learn the optimal solution during the training process. In our opinion, a potential solution to this problem is to further explore the PMD loss, which we plan to try in our future work. This solution not only constrains the total moving distance, but also can guide the network to learn the directions of each move.

REFERENCES

[1] X. Wen, T. Li, Z. Han, and Y.-S. Liu, "Point cloud completion by skip-attention network with hierarchical folding," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1939–1948. 1, 3, 9

[2] 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. 1

[3] 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. 1

[4] 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. 1, 2

[5] Z. Han, M. Shang, Z. Liu, C.-M. Vong, Y.-S. Liu, J. Han, M. Zwicker, 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, 2019. 1

[6] 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 *AAAI Conference on Artificial Intelligence*, 2019. 1

[7] 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," in *International Conference on Geometric Modeling and Processing*, 2020. 1

[8] 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 *ACM International Conference on Multimedia*, 2020. 1

[9] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4867–4876. 1

[10] J. Yuan, C. Chen, W. Yang, M. Liu, J. Xia, and S. Liu, "A survey of visual analytics techniques for machine learning," *Computational Visual Media*, vol. 7, no. 1, 2021. 1

[11] A. Dai, C. Ruizhongtai Qi, and M. Nießner, "Shape completion using 3D-encoder-predictor CNNs and shape synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5868–5877. 1, 3

[12] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in *2018 International Conference on 3D Vision*. IEEE, 2018, pp. 728–737. 1, 3, 6, 7, 8, 9, 10

[13] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese, "TopNet: Structural point cloud decoder," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 383–392. 1, 3, 6, 7, 8, 9

[14] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 4, 8, 9, 11

[15] 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*, 2017, pp. 5099–5108. 2, 3, 4

[16] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," *arXiv preprint arXiv:2012.09164*, 2020. 2, 4, 5

[17] 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*, 2020. 3

[18] 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*, 2020. 3

[19] 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*, 2020. 3

[20] 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. 3

[21] 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. 3

[22] 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 *IEEE International Conference on Computer Vision*, 2019, pp. 10 442–10 451. 3

[23] 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 *ACM International Conference on Multimedia*. ACM, 2019, pp. 989–997. 3

[24] 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. 3

[25] 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. 3

[26] 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 *Conference of the European Association for Computer Graphics*, vol. 1, no. 1, 2014, pp. 161–185. 3

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

[28] 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. 3

[29] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo, "An interactive approach to semantic modeling of indoor scenes with an RGB-D camera," *ACM Transactions on Graphics*, vol. 31, no. 6, p. 136, 2012. 3

[30] E. Kalogerakis, S. Chaudhuri, D. Koller, and V. Koltun, "A probabilistic model for component-based shape synthesis," *ACM Transactions on Graphics*, vol. 31, no. 4, p. 55, 2012. 3

[31] A. Martinovic and L. Van Gool, "Bayesian grammar learning for inverse procedural modeling," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 201–208. 3

[32] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu, "Structure recovery by part assembly," *ACM Transactions on Graphics*, vol. 31, no. 6, p. 180, 2012. 3

[33] W. Zhang, Q. Yan, and C. Xiao, "Detail preserved point cloud completion via separated feature aggregation," in *European Conference on Computer Vision*, 2020. 3, 8

[34] M. Sarmad, H. J. Lee, and Y. M. Kim, "RL-GAN-Net: A reinforcement learning agent controlled gan network for real-time point cloud shape completion," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5898–5907. 3

[35] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "PF-Net: Point fractal network for 3D point cloud completion," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7662–7670. 3

[36] T. Hu, Z. Han, and M. Zwicker, "3D shape completion with multi-view consistent inference," in *AAAI Conference on Artificial Intelligence*, 2020. 3

[37] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5505–5514. 3

[38] R. A. Yeh, C. Chen, T. Yian Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5485–5493. 3

[39] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *European Conference on Computer Vision*, 2018, pp. 85–100. 3

[40] 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*, 2020. 3, 7, 8, 9, 10

[41] X. Han, Z. Li, H. Huang, E. Kalogerakis, and Y. Yu, "High-resolution shape completion using deep neural networks for global structure and local geometry inference," in *IEEE International Conference on Computer Vision*, 2017, pp. 85–93. 3

[42] D. Stutz and A. Geiger, "Learning 3D shape completion from laser scan data with weak supervision," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1955–1964. 3

[43] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 3

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

[45] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3

[46] L. Pan, X. Chen, Z. Cai, J. Zhang, H. Zhao, S. Yi, and Z. Liu, "Variational relational point completion network," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 3, 9

[47] X. Wang, M. H. Ang Jr, and G. H. Lee, "Cascaded refinement network for point cloud completion," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 790–799. 3, 7, 8, 9

[48] K. Yin, H. Huang, D. Cohen-Or, and H. Zhang, "P2P-Net: Bidirectional point displacement net for shape transform," *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 1–13, 2018. 3, 5

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017. 4

[50] Y. Rubner, C. Tomasi, and L. J. Guibas, "The Earth Mover's Distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000. 5

[51] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 206–215. 8, 9

[52] T. Groueix, M. Fisher, V. Kim, B. Russell, and M. Aubry, "A papier-mâché approach to learning 3D surface generation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 8, 9

[53] 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*, vol. 6, no. 2, pp. 596–603, 2021. 9

[54] 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*, 2020. 9

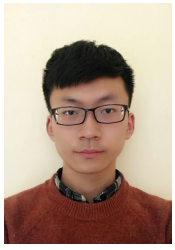
[55] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-GAN: A point cloud upsampling adversarial network," in *IEEE International Conference on Computer Vision*, 2019, pp. 7203–7212. 10, 11

[56] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. Zhang, "Edge-aware point set resampling," *ACM Transactions on Graphics*, vol. 32, no. 1, pp. 1–12, 2013. 11

[57] Y. Lequan, L. Xianzhi, F. Chi-Wing, C.-O. Daniel, and H. Pheng-Ann, "PU-Net: Point cloud upsampling network," in *IEEE International Conference on Computer Vision*, 2018. 11

[58] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3D point set upsampling," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5958–5967. 11

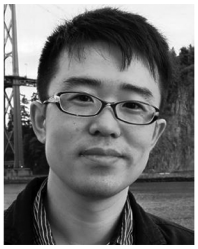
Xin Wen received the B.S. degree in engineering management from the Tsinghua University, China, in 2012. He is currently the PhD student with the School of Software, Tsinghua University. His research interests include deep learning, shape analysis and pattern recognition, and NLP.



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.

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.

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.

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.