PU-Mask: 3D Point Cloud Upsampling via an Implicit Virtual Mask

Hao Liu, Hui Yuan, *Senior Member, IEEE,* Raouf Hamzaoui, *Senior Member, IEEE,*
Qi Liu, Shuai Li, *Member, IEEE*

***Abstract*—We present PU-Mask, a virtual mask-based network for 3D point cloud upsampling. Unlike existing upsampling methods, which treat point cloud upsampling as an "unconstrained generative" problem, we propose to address it from the perspective of "local filling", i.e., we assume that the sparse input point cloud (i.e., the unmasked point set) is obtained by locally masking the original dense point cloud with virtual masks. Therefore, given the unmasked point set and virtual masks, our goal is to fill the point set hidden by the virtual masks. Specifically, because the masks do not actually exist, we first locate and form each virtual mask by a virtual mask generation module. Then, we propose a mask-guided transformer-style asymmetric autoencoder (MTAA) to restore the upsampled features. Moreover, we introduce a second-order unfolding attention mechanism to enhance the interaction between the feature channels of MTAA. Next, we generate a coarse upsampled point cloud using a pooling technique that is specific to the virtual masks. Finally, we design a learnable pseudo Laplacian operator to calibrate the coarse upsampled point cloud and generate a refined upsampled point cloud. Extensive experiments demonstrate that PU-Mask is superior to the state-of-the-art methods. Our code will be made available at: https://github.com/liuhaoyun/PU-Mask.**

***Index Terms*—Point cloud upsampling, implicit virtual mask, local filling, graph Laplacian.**

I. Introduction

Three-dimensional (3D) point clouds are sets of discrete and unordered points sampled from the underlying surface of an object. 3D point clouds are an emerging and popular data representation of 3D scenes. With the advancement of low-cost 3D scanning sensors, point clouds have become widely used in many applications, such as autonomous driving [1], 3D reconstruction [2], and virtual/augmented reality [3], [4]. However, due to the limitation of sensor accuracy and noise interference, the collected raw point clouds are usually sparse, non-uniformly distributed, and noisy, which may affect downstream tasks, e.g., surface reconstruction. Therefore, it is necessary to focus on point cloud upsampling. In general, given a sparse, noisy and non-uniform low resolution (LR) point cloud, the upsampling method aims to generate a dense, noise-free and uniform high resolution (HR) point cloud.

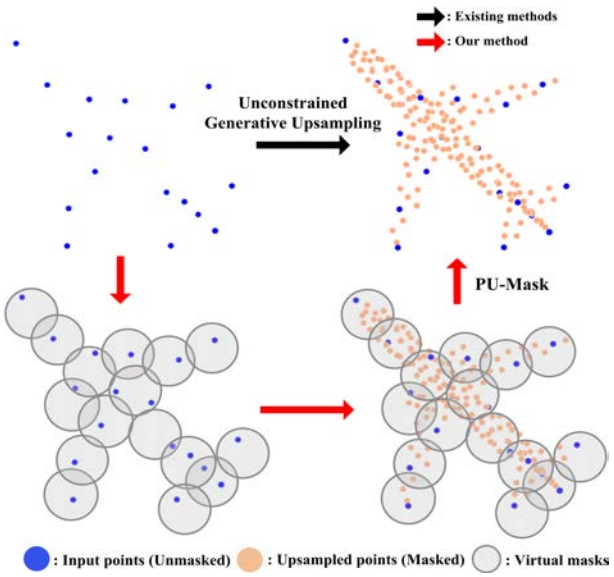


Fig. 1. Proposed method versus existing upsampling methods.

The two primary types of point cloud upsampling methods are handcrafted methods [5]–[13] and deep learning-based methods [14]–[31]. With the aid of geometry priors (e.g., curvature, normal, and density), handcrafted methods work well for smooth regions but struggle in relatively complex sharp areas. Advanced deep learning-based methods can process point-wise features and overcome the constraints imposed by geometry priors, leading to superior upsampling performance. Existing deep learning-based upsampling methods usually treat point cloud upsampling as a conventional "unconstrained generative" problem. That is, during the upsampling process, the upsampled features are obtained without restrictions from the features of the sparse input points. However, since the individual sparse input points do not well characterize the local neighborhoods of the underlying surface, the upsampled points may be generated at arbitrary locations, resulting in random

This work was supported in part by the National Natural Science Foundation of China under Grants 62222110, 62172259, and the High-end Foreign Experts Recruitment Plan of Chinese Ministry of Science and Technology under Grant G2023150003L, the exchange project of Royal Society and the National Natural Science Foundation of China under Grant 62311530104 (IEC\NSFC\223076), the Taishan Scholar Project of Shandong Province (tsqn202103001), the Natural Science Foundation of Shandong Province under Grant ZR2022ZD38, ZR2023QF111, the Shandong Provincial Higher Education Youth Innovation Team Project, 2022KJ268.

Hao Liu is with the School of Computer and Control Engineering, Yantai University, Yantai, 264005, China. E-mail: liuhaoxb@gmail.com

Hui Yuan is with School of Control Science and Engineering, Shandong University, and also with Shandong Key Lab. of Wireless Communication Technologies, Shandong University, Jinan, 250061, China. E-mail: huiyuan@sdu.edu.cn

Raouf Hamzaoui is with the School of Engineering and Sustainable Development, De Montfort University, Leicester, UK. E-mail: rhamzaoui@dmu.ac.uk

Qi Liu is with the School of Electronic Information, Qingdao University, Qingdao, 266071, China. E-mail: sdqi.liu@gmail.com

Shuai Li is with the School of Control Science and Engineering, Shandong University, Jinan, 250061, China. E-mail: shuaili@sdu.edu.cn

Hui Yuan is the corresponding author.

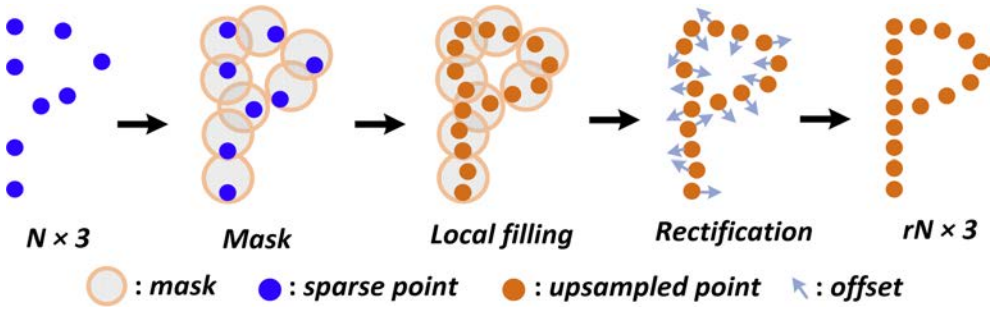


Fig. 2. Simplified flowchart of the proposed method.

perturbations and introducing artifacts in the upsampled points.

Like point cloud upsampling, point cloud completion [32] can accurately infer a complete point cloud from a partially missing point cloud, showing that global features facilitate local reconstruction. However, it is not appropriate to simply apply this completion strategy to upsample LR point clouds. The main reason for this is that point cloud completion aims at adding missing points in the damaged regions by analyzing global properties of the point cloud. It specifically targets incomplete areas, while leaving other areas unchanged. In contrast, point cloud upsampling aims to interpolate a specific number of new points between the neighbors of each original point. This leads to slight fluctuations at each point on the local underlying surface. Therefore, point cloud completion is a global filling process, while point cloud upsampling is a local filling one.

Inspired by point cloud completion and masked autoencoders (MAE) [33], we formulate the point cloud upsampling problem as a "local filling" problem and propose a virtual mask-guided point cloud upsampling method, namely PU-Mask. We assume that the LR point cloud (i.e., the unmasked point set) is produced by locally blocking the HR point cloud (ground truth) with some virtual masks. Given the LR point cloud and the virtual masks, the objective of PU-Mask is to fill the point sets in the virtual masks (Fig. 1). The mask associated with an arbitrary point in the LR point can be viewed as its global representation and can therefore help with the regression of local upsampled points. The virtual mask can regulate the upsampled points to closely match/regress the local underlying surface. This feature significantly distinguishes our approach from existing upsampling methods.

Fig. 2 and Fig. 3 show simplified and detailed frameworks of PU-Mask, respectively. First, we propose a virtual mask generation (VMG) module to locate and form a virtual mask for each LR point. Then, we devise a mask-guided transformer-style asymmetric auto-encoder (MTAA) to restore the upsampled features. This effectively captures the local dependencies between the LR point cloud and the virtual masks. Next, we introduce a second-order unfolding attention (SUA) mechanism to boost the interaction between the feature channels of MTAA. Afterwards, we obtain the coarse upsampled point set using a virtual mask-based pooling design. Finally, exploiting the local smoothing properties of the Laplacian operator, we propose a learnable pseudo Laplacian operator to refine the upsampled point cloud by calibrating the previously obtained coarse one.

To summarize, our work makes the four following contributions:

- We propose a virtual mask-guided "local filling" method for point cloud upsampling that effectively leverages the local representation of the mask to accurately predict the distribution of the local point set and thus generate a realistic upsampled point cloud.
- We propose a VMG module to locate and build a virtual mask for each point in the LR point cloud.
- We propose an efficient MTAA to predict the distribution of upsampled points hidden in the virtual masks. We also introduce an SUA mechanism to enrich the feature interaction of MTAA.
- We devise a virtual mask-based local pooling technique to interpolate the coarse dense point cloud. We also construct a pseudo Laplacian operator to adaptively adjust the coarse dense point cloud and generate a high-quality upsampled point cloud.

The remainder of this paper is organized as follows. In section II, we briefly review the related work on point cloud upsampling. In section III, we present the proposed PU-Mask. In Section IV, we report and analyze our experimental results. Finally, in Section V, we give concluding remarks and suggest future work.

II. RELATED WORK

Handcrafted upsampling. In their pioneering work, Alexa *et al*. [5] proposed to upsample a set of points by adding points at the vertices of a structured Voronoi diagram. Lipman *et al*. [6] presented a locally optimal projection (LOP) to resample points. Preiner *et al*. [7] devised a modified LOP for fast resampling. Huang *et al*. [8] introduced an edge-aware point cloud interpolation method. A point set consolidation strategy [9] was proposed to fill some local holes caused by partially missing points. Sabbadin *et al*. [4] proposed to infill a color point cloud in the gradient domain. Dinesh *et al*. [10] proposed a graph total variation-based point cloud interpolation method with surface normals. Subsequently, they [11] also proposed a color point cloud super-resolution strategy via a fast graph total variation. Heimann *et al*. [12] presented a mesh-to-mesh resampling technique in 2D plane for colored point clouds. Recently, Borges *et al*. [13] proposed a point set upsampling scheme that exploits the self-similarity within the point cloud.

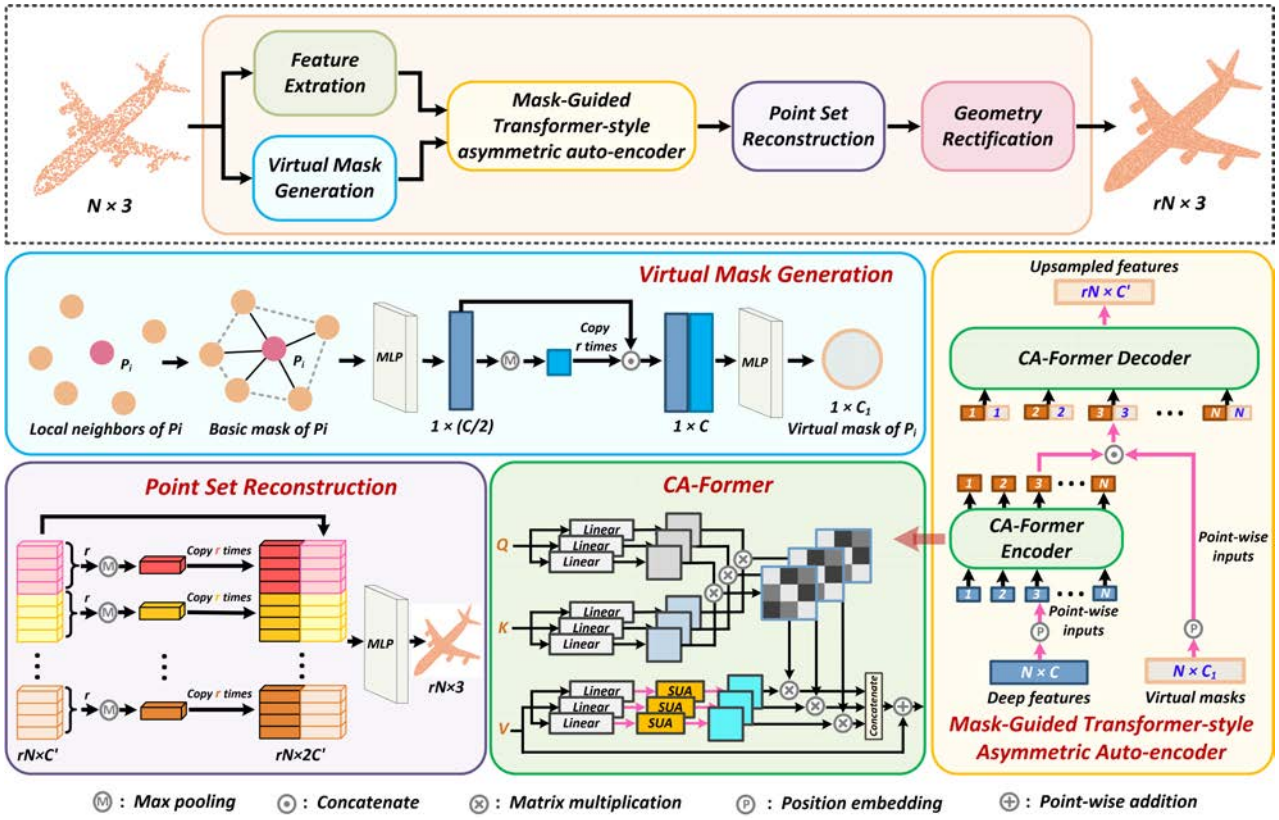


Fig. 3. Architecture of PU-Mask. The proposed method contains five modules: feature extraction, virtual mask generation, mask-guided tansformer-style asymmetric auto-encoder, point set reconstruction and geometry rectification. The feature extraction and virtual mask generation modules generate point-wise features and mask, respectively. Then, the mask-guided tansformer-style asymmetric auto-encoder interpolates the upsampled features with rich details. Next, the coarse upsampled point cloud is regressed by the point set reconstruction module. Finally, the geometry rectification module refines the coarse upsampled point cloud and outputs a more detailed upsampled point set.

Deep learning-based upsampling. PU-Net [14] uses PointNet++ [34] to extract multi-resolution features. Then a number of independent multilayer perceptrons (MLP) are introduced to generate the desired dense point clouds. EC-Net [15] can preserve the sharp features of the original point cloud by interpolating points close to the edges. MPU [16] is a progressive point cloud upsampling method that achieves hierarchical reconstruction via recursive upsampling. Li *et al*. [17] proposed a point cloud upsampling method with adversarial learning. Qian *et al*. [18] proposed a deep learning network that learns a local 2D parametric domain for each point, generates new samples in the parametric domain, and uses a linear transform to map them onto the 3D tangent plane. The same authors [19] proposed a learnable linear estimation theory to adaptively interpolate local point sets. Qian *et al*. [20] used a dilated graph convolution and node shuffle strategy to upsample the 3D point cloud. Li *et al*. [21] use two cascaded networks that split the upsampling process into two tasks. The first network generates a dense but coarse point set, while the second one is a spatial refiner that adjusts the position of the points in the coarse point set to better represent the underlying surface. Feng *et al*. [22] proposed a method that enables upsampling with an arbitrary upsampling ratio. The method uses a deep learning network to build a continuous surface patch around each point. These local patches are then combined into a global surface from which point clouds of arbitrary resolutions can be obtained. Delléva *et al*. [23] proposed another flexible upsampling strategy with arbitrary upsampling ratio. Self-supervision-based point cloud upsampling also showed competitive performance by taking full advantage of self-similarity [24] and local flatness [25] of the point clouds. Zhao *et al*. [26] interpolated arbitrary points on an implicit surface in a self-supervised manner. Akhtar *et al*. [27] were the first to use multiscale 3D sparse convolutional networks for point cloud upsampling. Qiu *et al*. [28] proposed a shifted channel multi-head self-attention-based transformer to upsample the sparse point clouds. Liu *et al*. [29] proposed a progressive point cloud upsampling method (PU-Refiner) that works in a coarse-to-fine fashion. Liu *et al*. [30] proposed a frequency-aware upsampling network (PUFA-GAN), which can not only generate dense point clouds on the underlying surface but also efficiently suppress high-frequency noise. A systematic review of learning-based point cloud upsampling methods can be found in [31].

In summary, current approaches typically consider point cloud upsampling as an "unconstrained generative" problem. They do not impose restrictions on the input sets, resulting in point set perturbation and artifacts. To address this problem, we propose PU-Mask, which treats point cloud upsampling as a local filling problem by explicitly introducing a virtual mask to process the input points locally. This unique feature enables PU-Mask to achieve superior upsampling performance.

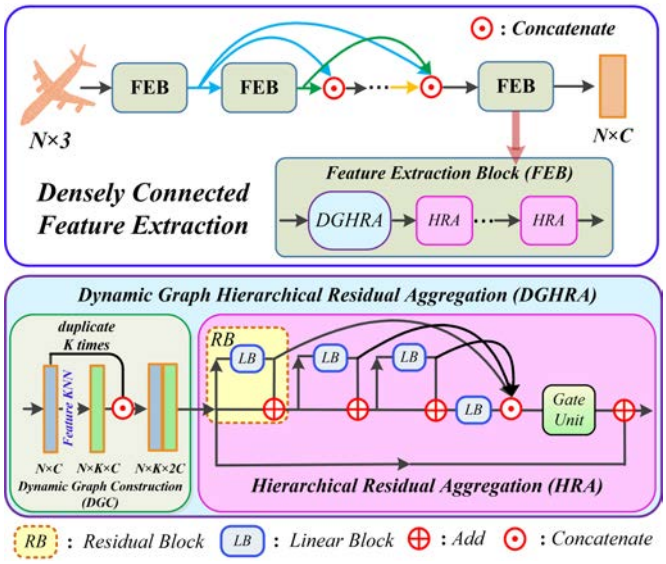


Fig. 4. Densely connected feature extraction module [30].

III. PROPOSED METHOD

A. Overview

A point cloud consists of a set of points, each of which includes geometry (Cartesian coordinates) and corresponding attribute information (e.g., color, reflectance). In this work, we only consider the geometry information. Given a low-resolution (sparse, non-uniform) input point cloud $\boldsymbol{P_{LR}} \in \mathbb{R}^{N\times 3}$ consisting of N points and an upsampling ratio r, the proposed method aims to generate a high-resolution (dense, uniform) point cloud $\boldsymbol{P_{HR}} \in \mathbb{R}^{rN\times 3}$. At the same time, $\boldsymbol{P_{HR}}$ should also be distributed on the underlying surface of $\boldsymbol{P_{LR}}$.

To capture more feature details and address arbitrarily scaled point clouds, we train PU-Mask using patches instead of the entire point cloud. Thus, unless otherwise specified, all the point clouds (e.g., $\boldsymbol{P_{HR}}$, $\boldsymbol{P_{LR}}$) correspond to patches.

Unlike existing upsampling approaches that consider point cloud upsampling as a classical "unconstrained generative" problem, we formulate the point cloud upsampling problem as a "local filling" problem. The proposed PU-Mask allows us to adaptively complete locally empty point clouds and generate naturally dense point clouds. As shown in Fig. 3, the proposed method consists of five modules: feature extraction, virtual mask generation, mask-guided transformer-style asymmetric auto-encoder, point set reconstruction and geometry rectification. In the remainder of this section, we give the details of each module.

B. Feature Extraction

For a given $\boldsymbol{P_{LR}} \in \mathbb{R}^{N\times 3}$, the feature extraction module aims to learn point-wise deep semantic features $\mathbf{F} \in \mathbb{R}^{N\times C}$, where C denotes the number of feature channels. Without loss of generality, we use the densely connected feature extraction module (Fig. 4 top) of our previous work [30]. Specifically, each feature extraction block (FEB) consists of the concatenation of one dynamic graph hierarchical residual aggregation (DGHRA) unit and several hierarchical residual aggregations (HRA) units (Fig. 4 bottom). The DGHRA unit can be regarded as a combination of dynamic graph construction (DGC) and HRA. The DGC can effectively process point clouds through a dynamic neighborhood feature graph for "local" feature collection. The HRA includes four residual blocks (RBs) and one gate unit that captures hierarchical pure residuals from each residual block for fine grained representation. More details can be found in [30].

C. Virtual Mask Generation

The core idea of our method is to assume that each point to be upsampled is occluded by a virtual mask, and aim to predict the potential distribution of the upsampled point set hidden by the mask. However, there is no real mask for a given $\boldsymbol{P_{LR}}$. Therefore, we propose a VMG module to locate and construct the virtual mask for each point in $\boldsymbol{P_{LR}}$.

As shown in Fig. 3, the proposed VMG first identifies the approximate location of each virtual mask. In general, the interpolated points are scattered near the sparse inputs. Based on this assumption, the virtual mask should also cover the neighborhood of each point. Hence, we take each point $P_i \in \mathbb{R}^{1\times 3}$ in $\boldsymbol{P_{LR}}$ as the initial position of the corresponding virtual mask. Subsequently, the shape of the virtual mask requires to be estimated accurately.

As mentioned above, the virtual mask spreads over the neighbors of P_i, suggesting potential relationships between the shape of the virtual mask and its neighborhoods. Taking the local area into consideration, we propose to adaptively infer the shape of the virtual mask by exploring a k nearest neighbor (k-NN)-based geometry prior. Specifically, the edge relation $\tilde{E}_{i,j}$ is first constructed by measuring the geometry similarity,

$$\tilde{E}_{i,j} = P_i - P_{i,j}, \tag{1}$$

where P_i and $P_{i,j}$ are the i-th point and its j-th neighbor, respectively. Then, we form a basic mask $BM_i \in \mathbb{R}^{1\times(3+3j)}$ for P_i by aggregating P_i and its local neighbors (as shown in Fig.3), i.e.,

$$BM_i = [P_i \odot \tilde{E}_{i,1} \odot \tilde{E}_{i,2} \odot ... \odot \tilde{E}_{i,j-1} \odot \tilde{E}_{i,j}], \tag{2}$$

where $\odot$ is the concatenation operator. However, the basic mask lacks precision in terms of location and shape. To obtain a precise mask, we apply a neural network to adaptively learn an implicit virtual mask, which can efficiently adjust the topology of BM_i and fit the local underlying surface. In this way, we build for P_i a learnable virtual mask $\bar{M}_i \in \mathbb{R}^{1\times(C/2)}$ such that

$$\bar{M}_i = \Upsilon(BM_i), \tag{3}$$

where $\Upsilon(\cdot)$ is a mapping consisting of two MLP layers, which integrates neighbor features into $\bar{M}_i$. Next, we use max pooling to enhance the global feature interaction of each $\bar{M}_i$ and obtain a global vector. Finally, each $\bar{M}_i$ is concatenated with the global vector to build the final virtual mask $\mathbf{M} \in \mathbb{R}^{N\times C_1}$, where C_1 denotes the number of features of the virtual mask. Section IV-D shows the benefits of the proposed virtual mask.

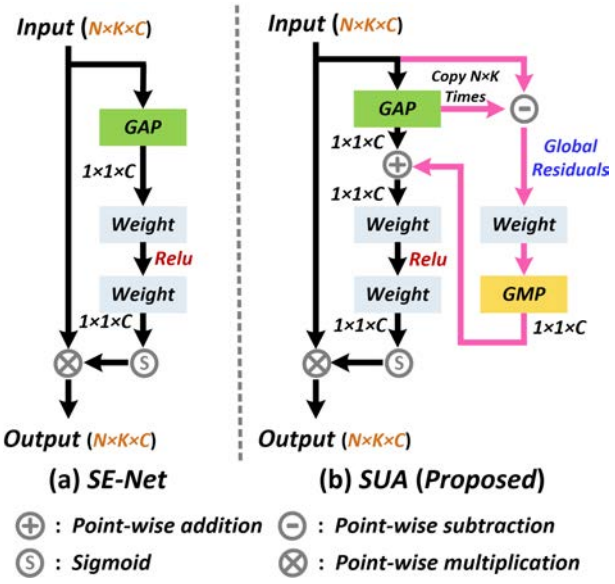


Fig. 5. Architecture of the proposed SUA.

D. Mask-guided Transformer-style Asymmetric Auto-encoder

Given a low-resolution input cloud $\boldsymbol{P_{LR}}$ and corresponding virtual mask $\mathbf{M}$, we can identify the lost points concealed by the mask. Because a patch is the basic processing unit of $\boldsymbol{P_{LR}}$, i.e., the local similarity is self-contained, a strong self-similarity is likely to exist between various point-mask pairs $(P_i,\ M_i)$ in a patch. Transformers have been widely used to effectively capture the self-similarity and enhance the global interaction. Thus, we propose a mask-guided transformer-style asymmetric auto-encoder to better infer the potential distribution of the upsampled point clouds by establishing the long-range dependency among point-mask pairs.

As shown in Fig. 3, the proposed MTAA consists of two components: encoder and decoder. In a preprocessing step, we replace $\boldsymbol{P_{LR}}$ with more expressive features $\mathbf{F}$. We then build the discriminable feature $\hat{\mathbf{F}}$ and mask $\hat{\mathbf{M}}$ by equipping $\mathbf{F}$ and $\mathbf{M}$ with a fixed positional embedded 2D grid. This grid helps the auto-encoder distinguish between them accurately.

Encoder: The encoder maps $\hat{\mathbf{F}}$, which corresponds to a visible $\boldsymbol{P_{LR}}$ to a latent representation $\tilde{\mathbf{F}}$ and provides a global guideline for the decoder. Classical ViT [35] first divides an image into multiple sequential patches, reshapes each patch into a learnable feature vector (i.e., token), and finally uses a standard transformer to obtain the deep semantic features for distinct patches. Inspired by ViT, our encoder views the point-wise feature $\hat{F}_i$ in $\hat{\mathbf{F}}$ as a patch of ViT. Therefore, $\hat{\mathbf{F}}$ can be fed directly into the transformer without any additional actions.

Although the standard transformer can build a long-range feature interaction, it ignores the connection between the feature channels, which may limit its performance. To tackle this problem, we introduce a lightweight channel attention mechanism, namely CA-Former, to the standard transformer. It not only preserves the global receptive field of the transformer, but also can adaptively promote the association between different channels. As shown in Fig 3, we keep the original query (Q) vector and key (K) vector to reduce the extra overhead of the transformer and attach a lightweight channel attention module, namely the proposed second-order unfolding attention (SUA), to the pivotal value (V) vector of the transformer.

Our channel attention module is inspired by SE-Net [36]. Fig. 5(a) shows how SE-Net can adaptively calibrate feature channels by using global average pooling (GAP) to capture their mean responses. However, we found that by viewing SE-Net from the perspective of signal coding [37], [38], we can gain valuable insights. Signal coding usually first extracts the DC coefficient (a one-order response corresponding to the mean response) of each coding unit, then uses the appropriate basis functions to transform the remainder residuals into another orthogonal space and generate many AC coefficients (i.e., high-order responses). Next, the larger AC coefficients are selected by the specified quantization step. Finally, the DC and the selected AC coefficients are sent to the entropy encoder for signal compression. In SE-Net, the mean responses of the feature channels are similar to the DC coefficient of a signal compression technique. Therefore, SE-Net can be seen as a signal compression technique that extracts only the DC coefficient and reconstructs the signal from the DC coefficient only.

To alleviate the above problem, we propose a second-order unfolding attention module, which extends SE-Net to the second-order form. As shown in Fig. 5(b), we first acquire the global residuals by explicitly subtracting the global mean response from the original input. Then, an MLP is used to emulate the transform basis in signal coding to map the global residuals into another implicit space and obtain the corresponding "AC coefficients". Next, we use global max pooling (GMP) for all "AC coefficients" to get the maximum AC response of each channel, i.e., the two-order responses. Finally, the global mean response (i.e., the one-order response) and the two-order responses are added to get a more expressive feature response, and two MLPs are applied to regress the variable weights to calibrate different feature channels. Our SUA not only enriches the original mean response but also reduces the burden of the network due to its lightweight characteristics. Section IV-D shows the benefits of SUA over SE-Net.

Because the encoder is only responsible for getting the latent global representation $\tilde{\mathbf{F}}$, we design a lightweight encoder, i.e., only a CA-Former with one head, which allows us to train a larger decoder with an asymmetric style.

Decoder: The decoder predicts the upsampled features from the latent global representation $\tilde{\mathbf{F}}$ and invisible virtual mask $\hat{\mathbf{M}}$. We first form the mask-guided latent feature $\hat{\mathbf{M}}\tilde{\mathbf{F}}$ by concatenating $\tilde{\mathbf{F}}$ and $\hat{\mathbf{M}}$. Then, three CA-Formers with two heads are used to estimate the potential distribution of the upsampled point sets from $\hat{\mathbf{M}}\tilde{\mathbf{F}}$. Finally, an MLP and reshape operation are used to acquire the desired upsampled features $\boldsymbol{F_{up}} \in \mathbb{R}^{rN\times C'}$ (r is the upsampling ratio and C' is the number of upsampled features). With this asymmetrical auto-encoder structure, the network can form detailed upsampled features while decreasing the computational complexity and memory consumption.

E. Point Set Reconstruction

The point set reconstruction module generates the upsampled point clouds $\boldsymbol{P_{up}} \in \mathbb{R}^{rN\times 3}$ from feature $\boldsymbol{F_{up}} \in \mathbb{R}^{rN\times C'}$. Common strategies use several fully connected (FC) layers to predict the interpolated coordinates. However, such strategies do not consider the feature interaction for the upsampled point sets in local regions, i.e., each upsampled point is generated independently. To address this limitation, we propose a simple yet effective virtual mask-based local max pooling strategy to better predict the interpolated points by actively sharing local features.

As shown in Fig. 3, we first construct N mask-guided local upsampled feature sets $\boldsymbol{F_{set}} \in \mathbb{R}^{r\times C'}$ from $\boldsymbol{F_{up}} \in \mathbb{R}^{rN\times C'}$, which corresponds to each virtual mask (i.e., each $\boldsymbol{F_{set}}$ is interpolated by the unique P_i with a virtual mask and specified r). Then, we use max pooling for each $\boldsymbol{F_{set}}$ to obtain the response vector $\boldsymbol{F_{rv}} \in \mathbb{R}^{1\times C'}$. Next, we generate the reinforced upsampled features with a local "global" receptive field by concatenating the duplicated $\boldsymbol{F_{rv}}$ and $\boldsymbol{F_{up}}$. Finally, we use two FC layers to get a coarse upsampled point cloud $\boldsymbol{P_{HR\_c}} \in \mathbb{R}^{rN\times 3}$. Section IV-D (PU-Mask w/o VMP) demonstrates the effectiveness of the proposed strategy.

F. Geometry Rectification

The geometry rectification module refines $\boldsymbol{P_{HR\_c}} \in \mathbb{R}^{rN\times 3}$ to a more realistic upsampled point cloud $\boldsymbol{P_{HR}} \in \mathbb{R}^{rN\times 3}$. Filter-based approaches are widely used for signal quality enhancement. The most popular filtering technique is linear filtering, which replaces the current point P_i by a linear combination of its nearest neighbors

$$P_i' = \sum\nolimits_{j\in\mathcal{N}(P_i)} w_{i,j}P_{i,j}, \tag{4}$$

where P_i' is the rectified version of P_i, $\mathcal{N}(P_i)$ is the neighbor set of P_i, $P_{i,j}$ is j-th nearest neighbor of P_i and $w_{i,j}$ is a weight. Such plain linear filtering usually filters out some high frequency sub-bands of the signal, making it too smooth or even blurred. To address this issue, we equalize each $w_{i,j}$ such that $\sum_{j\in\mathcal{N}(P_i)} w_{i,j}$ =1, and change Eq.(4) to

$$P_i' = P_i + \sum\nolimits_{j\in\mathcal{N}(P_i)} w_{i,j}(P_{i,j} - P_i), \tag{5}$$

Note that $\sum_{j\in\mathcal{N}(P_i)} w_{i,j}(P_{i,j}-P_i)$ is a typical discrete Laplacian operator [39], [40] that effectively captures the variations between P_i and its neighbors. However, modeling the complex relation between points with a simple linear combination as in Eq.(5) is not always possible. For this reason, we propose to use a learnable pseudo Laplacian operator to improve the calibration capability, that is, we compute

$$\Delta P_i = \mathcal{A}(\{\mathcal{M}(P_i, P_{i,j}) - \mathcal{T}(P_i)|P_{i,j}\in\mathcal{N}(P_i)\}), \tag{6}$$

where ΔP_i is a pseudo Laplacian operator for P_i, and $\mathcal{M}(\cdot)$ and $\mathcal{T}(\cdot)$ are two mapping functions (e.g., MLP or residual block), $\mathcal{T}(P_i)$ and $\mathcal{M}(P_i, P_{i,j})$ implicitly explore P_i and its local neighbors correlation, respectively, $\mathcal{M}(P_i, P_{i,j})-\mathcal{T}(P_i)$ evaluates the influence of $P_{i,j}$ on P_i, $\mathcal{A}(\cdot)$ denotes an aggregation function which collects the sum of local variations to P_i. In this way, our pseudo Laplacian operator not only preserves the same physical meaning as a regular Laplacian operator but is also nonlinear to better refine the point clouds by adaptively perceiving local details. Finally, each rectified point P_i' can be written as

$$P_i' = P_i + \Delta P_i, \tag{7}$$

and we can generate the final upsampled point cloud $\boldsymbol{P_{HR}} = \{P_i' \in \mathbb{R}^3\}_{i=1}^{rN}$.

To acquire a more uniformly distributed upsampled point cloud, we follow the strategy in [17]. Specifically, we first interpolate more upsampled features with upsampling ratio $(r+2)$, i.e., $\boldsymbol{F_{up}} \in \mathbb{R}^{(r+2)N\times C'}$. Then, we get the refined temporary upsampled point cloud $\boldsymbol{P_{HR_t}} \in \mathbb{R}^{(r+2)N\times 3}$ via the point set reconstruction module and geometry rectification module. Finally, we use farthest point sampling (FPS) [41] to uniformly sample $\boldsymbol{P_{HR_t}}$ and generate $\boldsymbol{P_{HR}} \in \mathbb{R}^{rN\times 3}$ for the given upsampling ratio r.

G. Loss Functions

To train PU-Mask, we use a compound loss that combines a reconstruction loss and a uniform loss.

Reconstruction Loss L_{rec} uses a symmetric point-to-point-based Chamfer distance (CD) [42] to rate the reconstruction quality:

$$L_{rec} = L_{cd}(\boldsymbol{P_{HR}}, \boldsymbol{P_{GT}}) = \frac{1}{N}(\sum\nolimits_{p\in\boldsymbol{P_{HR}}} \min_{q\in\boldsymbol{P_{GT}}} \|p-q\|_2^2 + \sum\nolimits_{q\in\boldsymbol{P_{GT}}} \min_{p\in\boldsymbol{P_{HR}}} \|q-p\|_2^2), \tag{8}$$

where $\boldsymbol{P_{GT}}$ is the ground truth upsampled point cloud.

Uniform Loss. We use the loss given by $L_u = \sum_{i=1}^{T} U_g(S_i)\cdot U_l(S_i)$ [17] to ensure that the upsampled points are uniformly distributed on the underlying surface of $\boldsymbol{P_{HR}}$. Here $U_g(\cdot)$ and $U_l(\cdot)$ denote the global and local uniformity, respectively, S_i is a ball query-based point set, and T is the number of seed points obtained with FPS. We have

$$U_g(S_i) = \frac{(|S_i| - \hat{n})^2}{\hat{n}}, \tag{9}$$

where $|S_i|$ and $\hat{n}$ [17] are the actual and desired number of points in S_i. $U_g(S_i)$ aims to make the number of points in each S_i as close as possible. On the other hand, $U_l(S_i)$ is given by

$$U_l(S_i) = \sum\nolimits_{j}^{|S_i|} \frac{(d_{j,i} - \hat{d})^2}{\hat{d}}, \tag{10}$$

where $d_{j,i}$ is the distance from the j-th point in S_i to its nearest point, $\hat{d}$ [17] is the expected distribution distance of each point in S_i. $U_l(S_i)$ aims to make the distances between each point in S_i and its nearest neighbor as close as possible.

Compound loss. Our compound loss function L_{total} is the weighted sum

$$L_{total} = w_{rec}L_{rec} + w_u L_u, \tag{11}$$

where w_{rec} and w_u are weights.

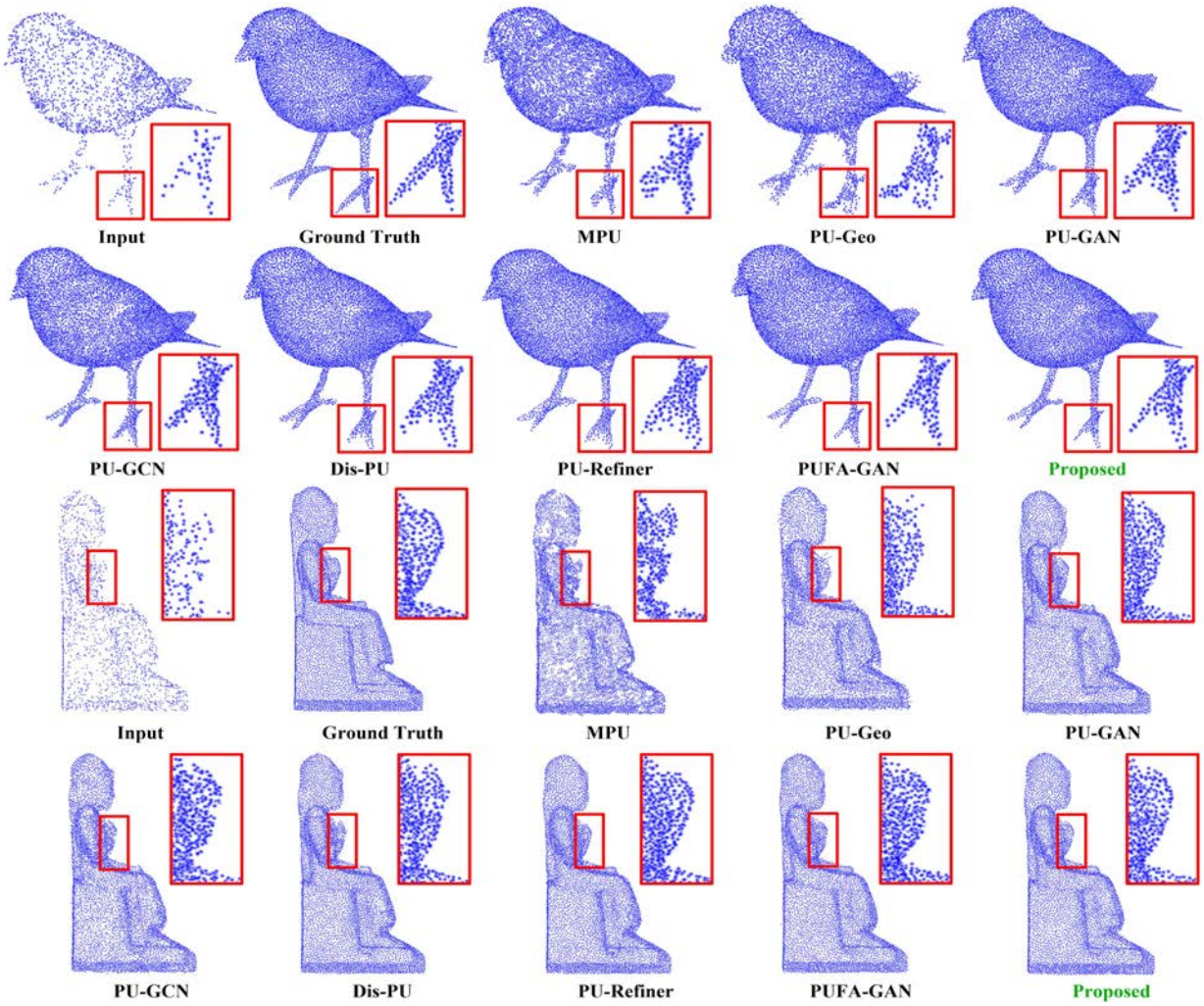


Fig. 6. Visual upsampling results for all methods.

IV. EXPERIMENTS

A. Experimental Setup

Datasets. To evaluate the effectiveness of PU-Mask, we tested it on PU-147 [17], PU1K [20], ModelNet40 [43] and KITTI [44]. The PU-147 dataset contains 147 3D models with various shapes. We randomly selected 120 models for training and used the remaining ones for testing. We randomly cropped 200 patches from each training point cloud during the training. Thus, a total of 24000 patches were used for training. To verify the generalization ability of the proposed method, we tested the model trained on PU-147 on the unseen PU1K, ModelNet40 and KITTI datasets.

Network details. We used the feature extraction module from our previous work [30]. However, to achieve a lightweight design, we implemented only one hierarchical residual aggregation unit in the dynamic graph hierarchical residual aggregation. For the proposed MTAA, the number of CA-Formers in the encoder and decoder was 1 and 3, respectively. This design allows us to train a larger decoder and generate more valid upsampling features. In the geometry rectification module, the mapping functions $\mathcal{M}(\cdot)$ and $\mathcal{T}(\cdot)$ use two MLP layers with a residual block and two MLP layers, respectively. In this way, $\mathcal{M}(\cdot)$ can efficiently explore the details of neighbor features using a residual design and $\mathcal{T}(\cdot)$ independently mines the features of each point through two MLP layers. The aggregation function $\mathcal{A}(\cdot)$ is implemented as a max-pooling layer. Moreover, we empirically set the weights w_{rec} and w_u in the compound loss to 8500 and 1, respectively.

Training details. To improve the robustness of PU-Mask, we used data augmentation techniques for each training patch, including random rotation, scaling, and the addition of Gaussian noise. To train PU-Mask, we used the Adam algorithm [45] with 120 epochs and a batch size of 28. The learning rate of the proposed method was initialized to 0.001. The upsampling ratio r was set to 4. We trained the PU-Mask on a single NVIDIA Tesla V100 GPU in the Tensorflow platform.

Test details. We adopted the common patch-fusion method [16] to obtain the final upsampled point cloud from the multiple upsampled patches. In particular, FPS was first applied to collect a specified number of seed points uniformly distributed on the underlying surface of the point cloud. Then, the local patch with 256 points was obtained by k-NN search for each seed point. Next, each patch was fed into PU-Mask to produce the corresponding upsampled form. Finally, all the upsampled patches were fused together and FPS was used

TABLE I
AVERAGE UPSAMPLING ACCURACY FOR THE TEST DATASET (27 POINT CLOUDS FROM [17]).

METHODS	CD (10^{-3})	HD (10^{-3})	P2F (10^{-3})	HF_CD (10^{-3})	HF_HD (10^{-3})
MPU (2018) [16]	0.438	5.322	3.066	2.998	24.308
PU-GAN (2019) [17]	0.280	4.493	2.514	2.342	22.337
PU-Geo (2020) [18]	0.405	5.338	3.432	3.111	24.778
PU-GCN (2021) [20]	0.323	4.172	2.926	2.978	25.152
Dis-PU (2021) [21]	0.289	3.734	2.289	2.634	22.859
PU-Refiner (2022) [29]	0.270	3,872	2.481	2.177	21.887
PUFA-GAN (2022) [30]	0.258	3.571	2.392	2.081	**19.744**
PU-Mask	**0.252**	**3.462**	**2.167**	**2.059**	**20.825**

*The evaluation value represents the error between the generated upsampled point cloud and the ground truth, i.e., the lower the evaluation value, the better the performance. The bold red and blue values represent best and second best results, respectively.

TABLE II
FLOATING POINT OPERATIONS, TRAINING AND INFERENCE TIME COMPARISON.

Time	MPU	PU-Geo	PU-GAN	PU-GCN	Dis-PU	PU-Refiner	PUFA-GAN	PU-Mask
FLOPs (G)	2.9	31.1	1.0	0.4	3.3	22.3	20.4	11.9
training (h)	6	5	18	2	23	26	28	22
Inference (s)	1.06	0.70	0.46	0.26	0.58	0.67	0.94	0.88

*We used 27 test point clouds from the test dataset to get the average inference time for each method, i.e., each point cloud contained 2048 input points, ultimately generating an upsampled point cloud with 8192 points.

again to generate the final upsampled point cloud with the specified upsampling ratio.

The ground truth point cloud $\boldsymbol{P_{GT}}$ was obtained by generating 8192 points using Poisson disk sampling on the test point cloud. Unless otherwise mentioned, the input point cloud $\boldsymbol{P_{LR}}$ was obtained by generating 2048 points through Monte Carlo random sampling applied to the same test point cloud. For all methods, the goal was to generate 8192 points from $\boldsymbol{P_{LR}}$.

Evaluation metrics. We evaluated the upsampling performance with five widely used evaluation metrics: (i) Chamfer distance (CD) [42], (ii) Hausdorff distance (HD) [46], (iii) point-to-surface (P2F) distance [47], (iv) CD in high frequency regions (HF_CD) [30] and (v) HD in high frequency regions (HF_HD) [30]. CD assesses the point-to-point similarity of $\boldsymbol{P_{HR}}$ and $\boldsymbol{P_{GT}}$. HD measures the point-wise maximum Euclidean distance deviation between $\boldsymbol{P_{HR}}$ and $\boldsymbol{P_{GT}}$. P2F evaluates the deviation of $\boldsymbol{P_{HR}}$ from the surface of $\boldsymbol{P_{GT}}$. HF_CD and HF_HD evaluate the regularity of the point set in high frequency regions (e.g., edges, corners). Note that, as in several previous works (e.g., [16], [18], [20], [21], [29]), CD was used in both training and evaluation.

B. Comparison Study

We compared PU-Mask to seven point cloud upsampling methods: MPU [16], PU-GAN [17], PU-Geo [18], PU-GCN [20], Dis-PU [21], PU-Refiner [29] and PUFA-GAN [30]. For a fair comparison, we re-trained all methods on the training dataset.

Quantitative comparison. Table I gives the upsampling accuracy on the testing dataset for all methods. It can be noticed that PU-Mask achieved the best upsampling performance in terms of **CD** (0.252), **HD** (3.462), **P2F** (2.167) and **HF_CD** (2.059). For **HF_HD**, PU-Mask achieved the second best result (20.825). The results show the superiority of the proposed method.

Qualitative comparison. Fig. 6 shows visual upsampling results of all methods for the Bird and Sculpture point clouds. While PU-Mask generated high-resolution point clouds with a uniform distribution over the underlying surface and without local noise, all other methods failed to preserve fine-grained structures and suffered from noise or local adhesions above the surface, especially in complex regions such as the bird's feet or the sculpture's elbows. Consequently, the point clouds reconstructed by PU-Mask appear more natural and are more similar to the ground truth. More visual results can be found in the **supplementary materials**.

Complexity comparison. Table II shows the number of floating point operations (FLOPs), the training and the inference time. PU-GCN and PU-Geo had the smallest (0.4 G) and largest (31.1 G) number of FLOPs, respectively. The number of FLOPs of our method was in the middle range (11.9 G). The approaches that required the shortest and longest training times were PU-GCN (2 h) and PUFA-GAN (28 h), respectively. The training time of the proposed PU-Mask (22h) was moderate. We also report the inference time. PU-GCN and MPU had the shortest (0.26 s) and longest (1.06 s) inference time, respectively. The inference time of PU-Mask (0.88 s) was comparable to that of the other methods.

TABLE III
UPSAMPLING RESULTS ON PU1K (127 POINT CLOUDS FROM [20]).

METHODS	CD (10^{-3})	HD (10^{-3})	P2F (10^{-3})	HF_CD (10^{-3})	HF_HD (10^{-3})
MPU (2018) [16]	0.618	7.239	1.985	3.562	69.704
PU-GAN (2019) [17]	0.488	7.072	1.964	2.314	60.896
PU-Geo (2020) [18]	0.573	8.712	2.293	3.112	73.156
PU-GCN (2021) [20]	0.495	6.649	1.953	2.695	62.386
Dis-PU (2021) [21]	0.463	7.124	1.770	2.145	57.527
PU-Refiner (2022) [29]	0.440	6.711	1.774	2.158	58.448
PUFA-GAN (2022) [30]	0.439	6.516	1.623	1.939	**56.404**
PU-Mask	**0.425**	**6.476**	**1.542**	**1.813**	59.084

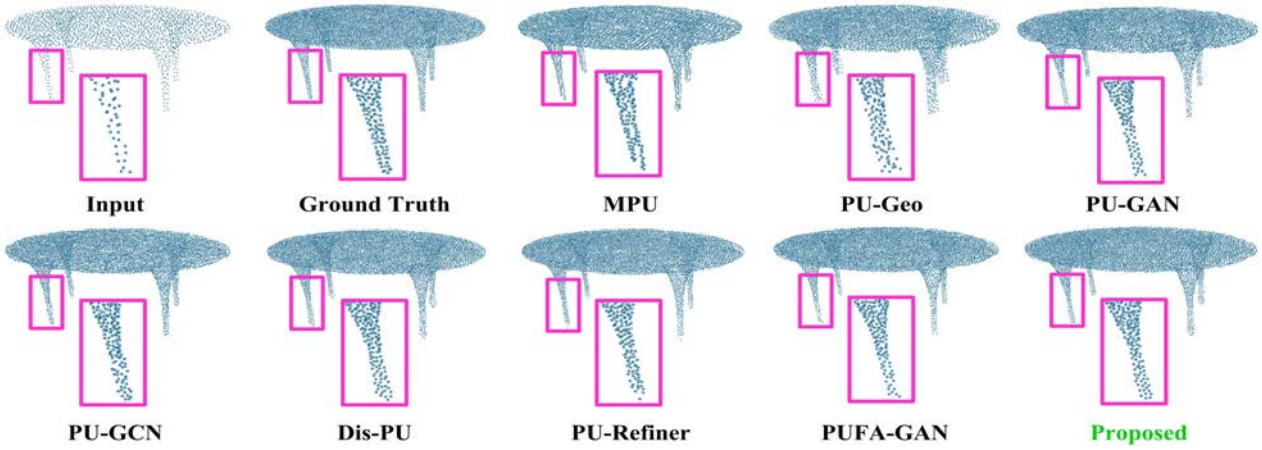


Fig. 7. Upsampling a point cloud from the unseen PU1K dataset. The input point cloud (*Desk*) consisted of 2048 points.

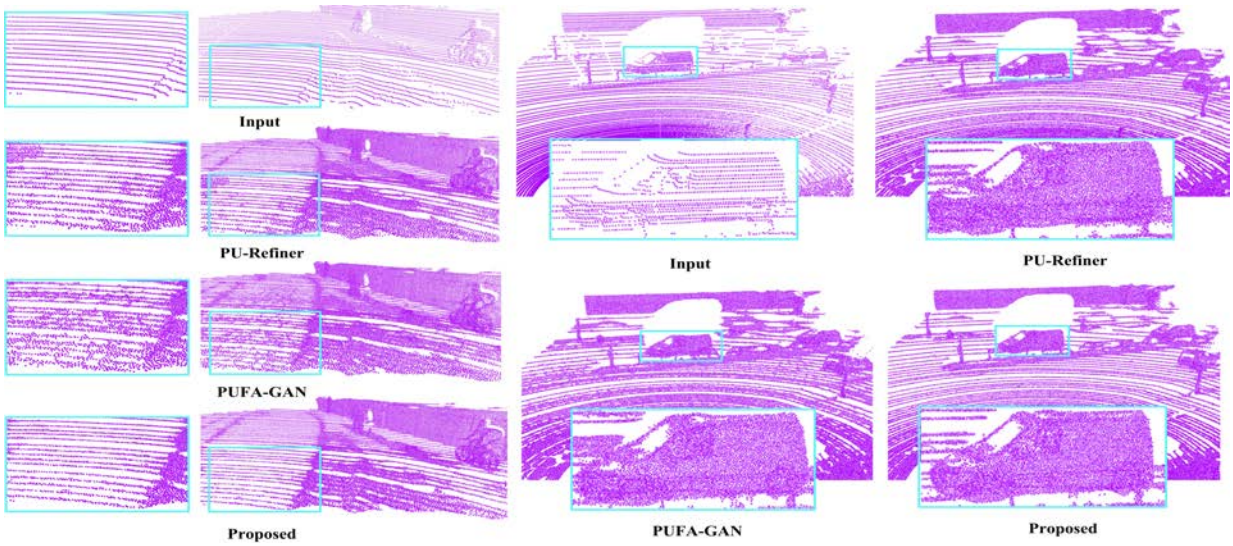


Fig. 8. Upsampling ($\times 4$) real-world LiDAR point clouds *Car* (93768 points) and *Bicyclists* (93417 points) from the KITTI dataset.

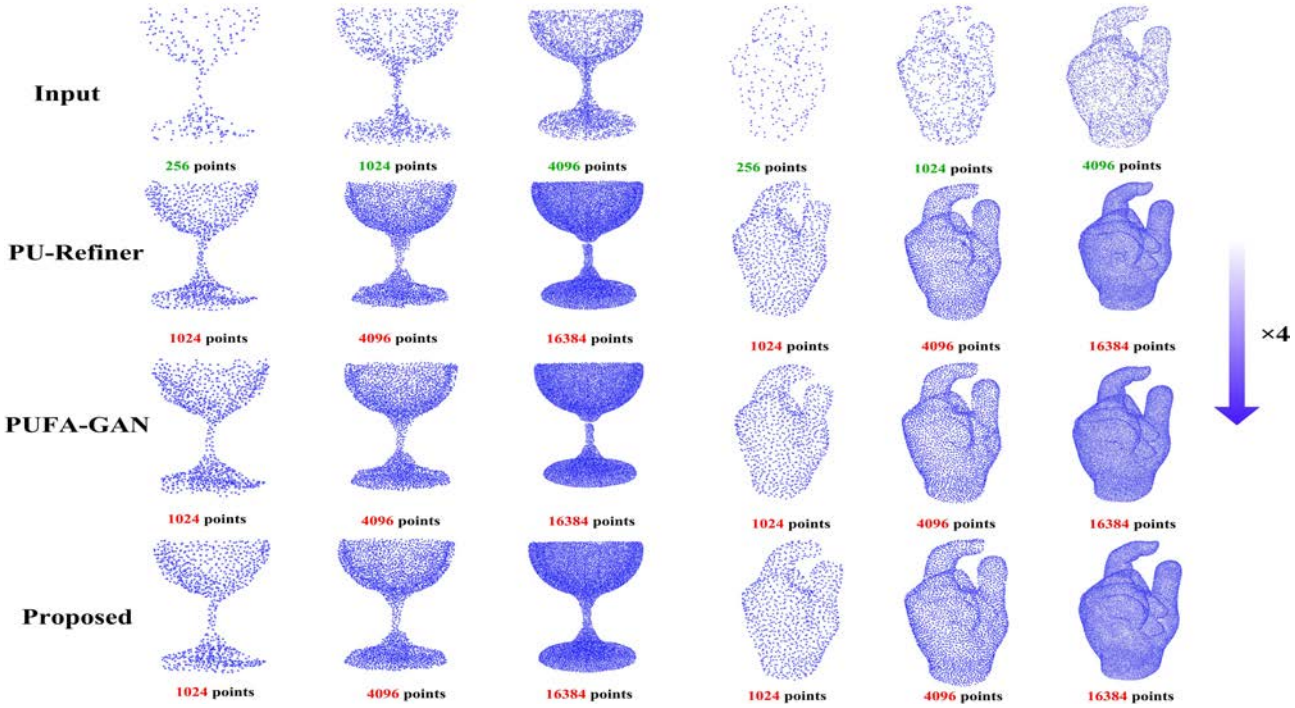


Fig. 9. Upsampling ($\times 4$) the test point cloud *Finger* (left) and the unseen *Cup* (right) from ModelNet40 with 256, 1024 and 4096 input points.

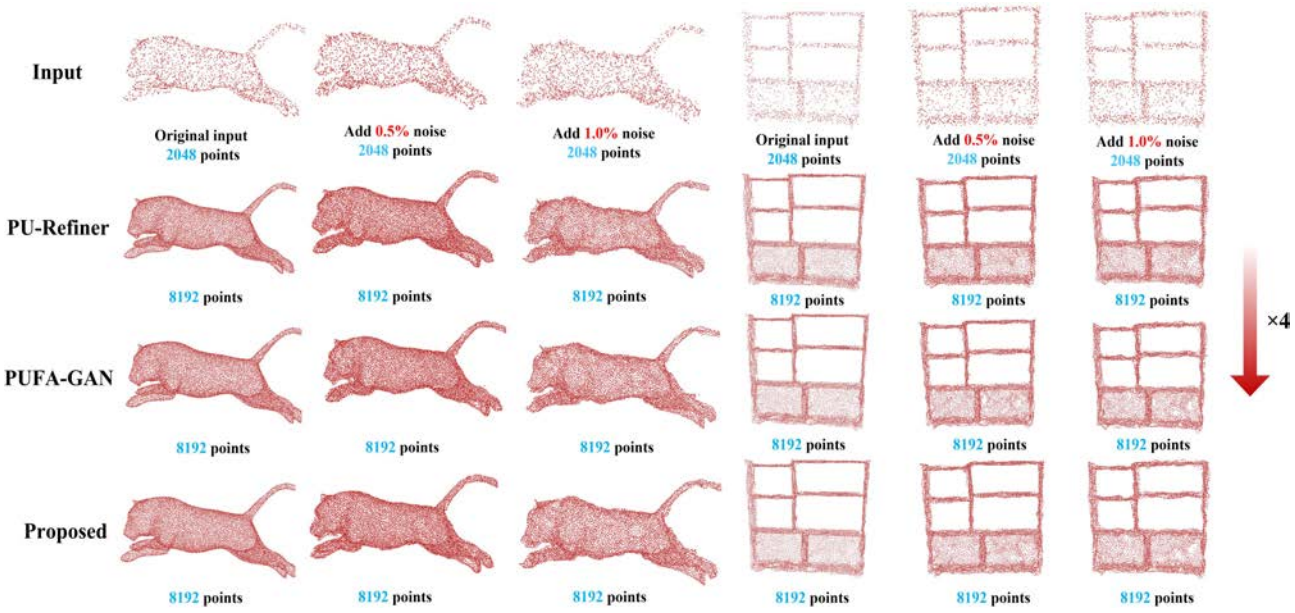


Fig. 10. Upsampling noisy point clouds (input point clouds with Gaussian noise levels: 0, 0.5%, and 1% from left to right), *Tiger* (left) is selected from our test dataset, and *Bookshelf* (right) belongs to the unseen ModelNet40 dataset.

C. Robustness Analysis

To further verify the robustness of PU-Mask, we conducted extended experiments covering the following six aspects: upsampling unseen point clouds (testing directly on a new dataset using the trained model), upsampling real-world LiDAR point clouds, upsampling noisy point clouds, upsampling point clouds with various input sizes, comparison with PU-Dense and surface reconstruction.

Upsampling Unseen Point Clouds. To verify the generalization ability of the proposed method, we studied the model trained on the training dataset on the larger unseen PU1K [20] dataset. Without loss of generality, we used all test samples (i.e., 127 test point clouds) in the PU1K dataset. Table III shows the upsampling results for all methods. PU-Mask achieved the best performance for four evaluation metrics, i.e., **CD** (0.425), **HD** (6.476), **P2F** (1.542), **HF_CD** (1.813). For **HF_HD**, it ranked third (59.084), which may be explained by the fact that the predictive capability of the mask was harmed by the complex geometric distribution in high frequency regions, causing several outliers. A visual comparison

is shown in Fig. 7. Our method produced uniform, noise-free upsampled point clouds with continuous surfaces, and displayed a better visual impression. Therefore, the proposed PU-Mask demonstrated a stronger generalization ability.

Upsampling Real-world LiDAR Point Clouds. We tested the performance of PU-Mask on real-world LiDAR point clouds from the KITTI [44] dataset. LiDAR point clouds are generally sparse, contain occlusions and are noisy; therefore, upsampling them accurately is a challenging task. Fig. 8 shows the visual results. PU-Mask noticeably improved the details of the original LiDAR point clouds, validating the reliability of the proposed method.

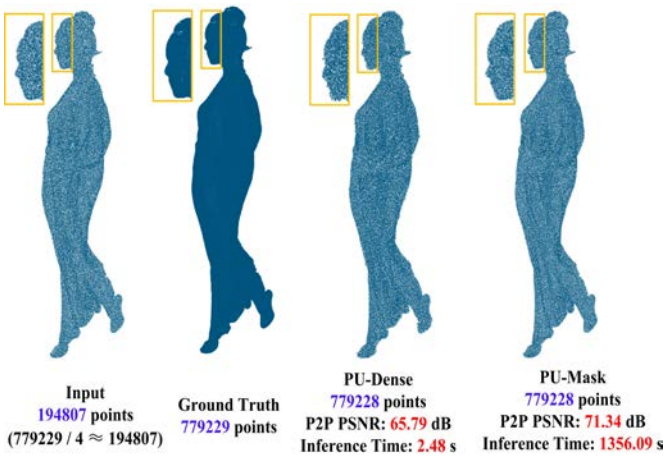


Fig. 11. Upsampling ($\times 4$) results of *Longdress* from the 8iVFB dataset.

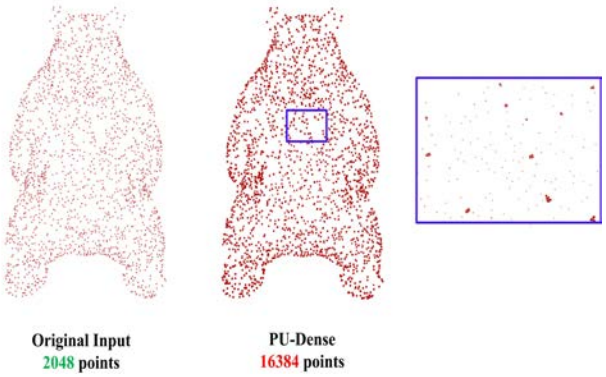


Fig. 12. Visualization results for PU-Dense with upsampling ratio 8.

Upsampling Point Clouds with various Sizes. The number of input points in a point cloud can be arbitrary. Hence, it is necessary to examine the performance of our method for input point clouds of various sizes. We conducted experiments on both the test dataset and the unseen ModelNet40 [43]. As shown in Fig. 9, for various input sparse point clouds, our method generated a high-quality upsampled point cloud, even when there were only 256 points in the input set.

Upsampling Point Clouds with Different Noise Levels. The point clouds collected in real environments often contain noise. To assess the ability of our method to handle noise, we tested it on two randomly selected point clouds from the test dataset and the unseen ModelNet40 dataset, respectively. As shown in Fig. 10, PU-Mask was able to generate realistic upsampled point clouds for noisy input point clouds.

Comparison with PU-Dense. In this section, we compare our method to PU-Dense [27], a state-of-the-art 3D sparse convolution-based upsampling method for dense point clouds. Fig. 11 shows the results for the dense point cloud "Longdress" from the 8iVFB dataset [48] and upsampling rate 4. In the experiment, we retrained PU-Dense according to the settings provided by the authors in [27]. For PU-Mask, we used the model trained on PU-147. The point-to-point peak signal-to-noise ratio [49] (P2P PSNR) to the ground truth was 71.34 dB for PU-Mask and only 65.79 dB for PU-Dense. Visually, PU-Mask generated a uniform and noise-free upsampled point cloud, while the upsampled point cloud generated by PU-Dense was affected by noise. On the other hand, the inference time was 1356.09s for PU-Mask and only 2.48s for PU-Dense.

In a second experiment, we compared our method to PU-Dense on a sparse point cloud. When we trained PU-Dense with PU-147 (the training set used for our method), the upsampled points were grouped together in the neighborhood of the input points (Fig.12). Using the trained model provided by the authors of the PU-Dense paper did not resolve the problem. This suggests that PU-Dense is not suitable for sparse point clouds.

Surface Reconstruction. Point cloud upsampling usually serves downstream tasks, such as surface reconstruction. To demonstrate how our method can assist in downstream tasks, we tested it on a surface reconstruction task. Concretely, we randomly selected a test sample from the unseen PU1K dataset and generated the corrresponding upsampled point cloud for all methods. Next, we used the ball-pivoting surface reconstruction algorithm [50] to reconstruct the surfaces. As shown in Fig. 13, our method successfully deduced the point at the armrest of the chair and effectively contributed to the subsequent surface reconstruction. In contrast, all other methods yielded incomplete reconstruction results. More surface reconstruction results can be found in the **supplementary materials**.

D. Ablation Study

To verify the importance of each proposed component, we compared the following methods,

(1) **PU-Mask w/o CA-Former**: we replaced CA-Former with the same number of MLPs.

(2) **PU-Mask w/o pseudo Laplacian operator (PLO)**: we removed the proposed PLO and generated the final upsampled point cloud from the point set reconstruction module directly.

(3) PU-Mask w/o virtual mask (VM): we removed VM from our PU-Mask and used feature $\boldsymbol{F}$ (from the feature extraction module) as unique input to MTAA.

(4) **PU-Mask w/o SUA**: we replaced the proposed SUA with SE-Net.

(5) **PU-Mask w/o virtual mask-based pooling (VMP)**: we removed VMP from the point set reconstruction module.

As shown in Table IV and Fig. 14, removing any of the proposed components degraded the performance of PU-Mask, illustrating their effectiveness. Because the proposed MTAA directly determines the quality of the upsampled point

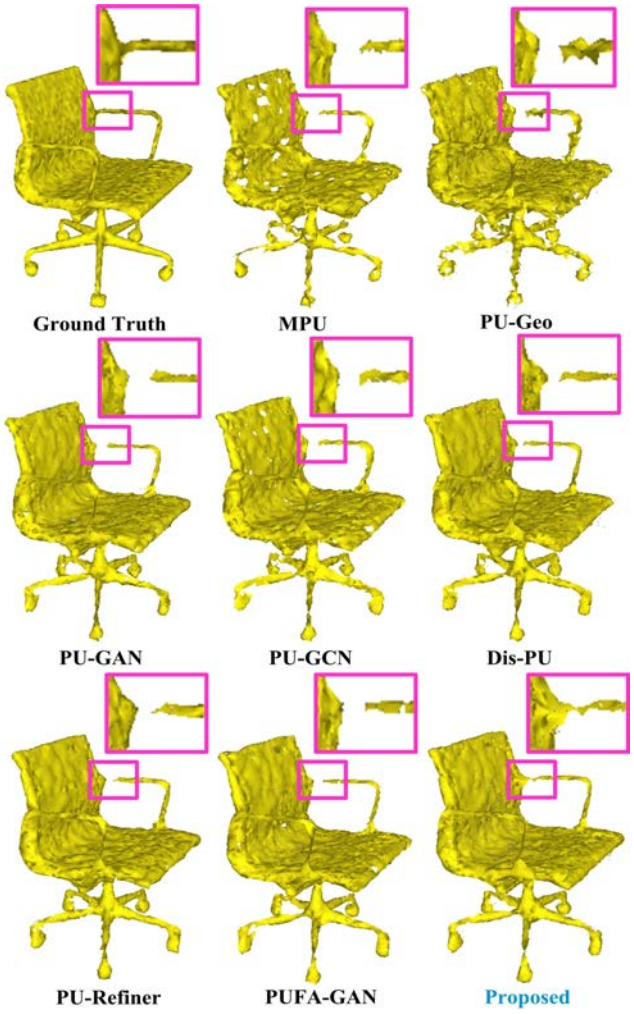


Fig. 13. Visualization of surface reconstruction.

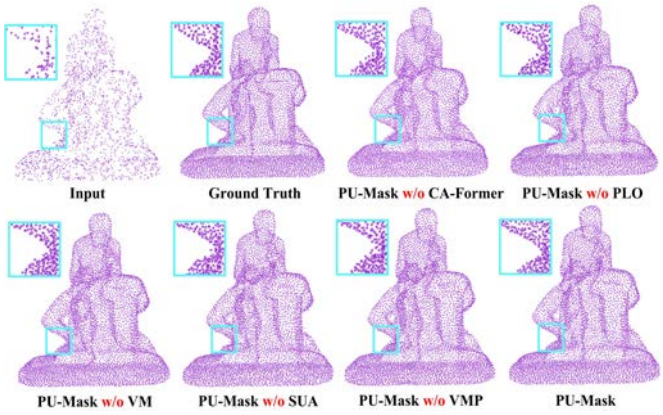


Fig. 14. Visual results for the ablation study.

TABLE IV
ABLATION STUDY ON THE TEST DATASET.

METHODS	CD (10^{-3})	HD (10^{-3})
PU-Mask w/o CA-Former	0.284	4.078
PU-Mask w/o PLO	0.276	3.651
PU-Mask w/o VM	0.267	3.675
PU-Mask w/o SUA	0.265	3.731
PU-Mask w/o VMP	0.261	3.809
PU-Mask	**0.252**	**3.469**

cloud features, removing CA-Former (i,e,. PU-Mask w/o CA-Former) had the most detrimental effect on the proposed method. On the other hand, since the pooling module only mines locally representative features from the upsampled features of each virtual mask, it had the weakest impact on the overall performance of our method.

V. CONCLUSION

We proposed PU-Mask, a virtual mask-guided auto-encoder for point cloud upsampling. Unlike previous approaches, PU-Mask considers the point cloud upsampling problem as a "local filling" problem. It creates a virtual mask around each point of the input point cloud, fills these masks using an auto-encoder that predicts the local distribution of the points, and rectifies the generated points with a pseudo-Laplacian operator. Extensive experiments show that PU-Mask outperformed state-of-the-art point cloud upsampling methods on all objective metrics. Visually, PU-Mask generated high-resolution point clouds that display a uniform distribution across the underlying surface and were free from local noise. In contrast, all other methods faced problems with noise or showed local geometric irregularities. This characteristic proved beneficial for a fine-grained downstream task, specifically surface reconstruction. In terms of time complexity, PU-Mask fell within the mid-range compared to the other methods.

In the future, we will conduct research in the following four areas: 1) investigating self-supervised learning to alleviate the dependence on paired datasets. 2) studying point cloud upsampling with arbitrary upsampling ratios. 3) upsampling point clouds with attributes (e.g., color, normal vector). 4) upsampling dynamic point clouds by exploiting their spatial-temporal correlations.

REFERENCES

[1] L. Xie, G. Xu, D. Cai and X. He, "X-View: non-egocentric multi-view 3d object detector," *IEEE Transactions on Image Processing*, vol. 32, pp. 1488–1497, 2023.
[2] X. Jia, S. Yang, Y. Wang, J. Zhang, Y. Peng and S. Chen, "Dual-View 3d reconstruction via learning correspondence and dependency of point cloud regions," *IEEE Transactions on Image Processing*, vol. 31, pp. 6831–6846, 2022.
[3] W. Wu and C. Zhang, "Immersive 3d communication," in *Proceedings of the 22nd ACM international conference on Multimedia*, Nov. 2014, pp. 1229–1230.
[4] M. Sabbadin, G. Palma, F. Banterle, T. Boubekeur and P. Cignoni, "High dynamic range point clouds for real-time relighting," *Computer Graphics Forum*, vol. 38, pp. 513–525, Nov. 2019.
[5] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 3–15, Jan. 2003.

[6] Y. Lipman, D. Cohen–Or, D. Levin, H. Tal–Ezer, "Parameterization-free projection for geometry reconstruction," *ACM SIGGRAPH*, Jul. 2007.

[7] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, M. Wimmer, "Continuous projection for fast l1 reconstruction," *ACM Transactions on Graphics*, vol. 33, no. 4, Jul. 2014.

[8] 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, Jan. 2013.

[9] S. Wu, H. Huang, M. Gong, M. Zwicker, and D. Cohen-Or, "Deep points consolidation," *ACM Transactions on Graphics*, vol. 34, no. 6, pp. 1–13, Oct. 2015.

[10] C. Dinesh, G. Cheung, I. V. Bajić, "3D point cloud super-resolution via graph total variation on surface normals," *IEEE International Conference on Image Processing (ICIP)*, Sept. 2019, pp. 4390–4394.

[11] C. Dinesh, G. Cheung, I. V. Bajić, "Super-resolution of 3d color point clouds via fast graph total variation," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May. 2020, pp. 1983–1987.

[12] V. Heimann, A. Spruck and A. Kaup, "Frequency-selective mesh-to-mesh resampling for color upsampling of point clouds," *IEEE 23rd International Workshop on Multimedia Signal Processing*, Oct. 2021, pp. 1–6,

[13] T. M. Borges, D. C. Garcia and R. L. de Queiroz, "Fractional super-resolution of voxelized point clouds," *IEEE Transactions on Image Processing*, vol. 31. pp. 1380–1390, Jan. 2022.

[14] L. Yu, X. Li, C. Fu, D. Cohen-Or, and P. Heng, "PU-Net: point cloud upsampling network," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Dec. 2018, pp. 2790–2799.

[15] L. Yu, X. Li, C. Fu, D. Cohen-Or, and P. Heng, "EC-Net: an edge-aware point set consolidation network," in *European Conf. on Computer Vision*, Sept. 2018, pp. 386–402.

[16] Y. Wang, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3d point set upsampling," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2019, pp. 5958–5967.

[17] R. Li, X. Li, C. Fu, D. Cohen-Or and P. Heng, "PU-GAN: a point cloud upsampling adversarial network," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Nov. 2019, pp. 7202–7211.

[18] Y. Qian, J. Hou, K. Kwong, Y. He, "PUGeo-Net: a geometry-centric network for 3d point cloud upsampling," in *European Conf. on Computer Vision*, Nov. 2020, pp. 752–769.

[19] Y. Qian, J. Hou, K. Kwong, Y. He, "Deep magnification-flexible upsampling over 3d point clouds," *IEEE Transactions on Image Processing*, vol.30, pp. 8354–8367, Sept. 2021.

[20] G. Qian, A. Abualshour, G. Li, A. Thabet, B. Ghanem, "PU-GCN: point cloud upsampling using graph convolutional networks,"in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2021.

[21] R. Li, X. Li, P. Heng, C. Fu, "Point cloud upsampling via disentangled refinement," *IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2021.

[22] W. Feng, J. Li, H. Cai, X. Luo and J. Zhang, "Neural points: point cloud representation with neural fields for arbitrary upsampling," *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2022, pp. 18612–18621.

[23] A. DellÉva, M. Orsingher and M. Bertozzi, "Arbitrary point cloud upsampling with spherical mixture of gaussians," *International Conference on 3D Vision*, Sept. 2022, pp. 465–474.

[24] G. Metzer, R. Hanocka, R. Giryes, D. Cohen-Or, "Self-sampling for neural point cloud consolidation," *ACM Transactions on Graphics*, vol. 40, no. 5, pp. 1–14, Sept. 2021.

[25] X. Liu, X. Liu, Y. -S. Liu and Z. Han, "SPU-Net: self-supervised point cloud upsampling by coarse-to-fine reconstruction with self-projection optimization," *IEEE Transactions on Image Processing*, vol. 31, pp. 4213–4226, 2022.

[26] W. Zhao, X. Liu, Z. Zhong, J. Jiang, W. Gao, G. Li and X. Ji, "Self-supervised arbitrary-scale point clouds upsampling via implicit neural representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2022, pp. 1999–2007.

[27] A. Akhtar, Z. Li, G. V. d. Auwera, L. Li and J. Chen, "PU-Dense: sparse tensor-based point cloud geometry upsampling," *IEEE Transactions on Image Processing*, vol. 31, pp. 4133–4148, Jun. 2022.

[28] S. Qiu, S. Anwar and N. Barnes, "PU-Transformer: point cloud upsampling transformer," in *Proceedings of the Asian Conference on Computer Vision*, Dec. 2022, pp. 2475–2493.

[29] H. Liu, H. Yuan, R. Hamzaoui, W. Gao and S. Li, "PU-Refiner: a geometry refiner with adversarial learning for point cloud upsampling," *IEEE International Conference on Acoustics, Speech and Signal Processing*, May. 2022, pp. 2270–2274.

[30] H. Liu, H. Yuan, J. Hou, R. Hamzaoui and W. Gao, "PUFA-GAN: a frequency-aware generative adversarial network for 3d point cloud upsampling," *IEEE Transactions on Image Processing*, vol. 31, pp. 7389–7402, Dec. 2022.

[31] Y. Zhang, W. Zhao, B. Sun, Y. Zhang and W. Wen, "Point cloud upsampling algorithm: a systematic review," *Algorithms*, vol.15, no.4, pp. 124–141, Apr. 2022.

[32] W. Yuan, T. Khot, D. Held, C. Mertz and M. Hebert, "PCN: point completion network," *2018 International Conference on 3D Vision*, Sept. 2018, pp. 728–737.

[33] K. He, X. Chen, S. Xie, Y. Li, P. Dollàr and R. Girshick, "Masked autoencoders are scalable vision learners," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition*,Jun. 2022, pp. 15979–15988.

[34] R. Q. Charles, L. Yi, H. Su, 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.

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

[36] J. Hu, L. Shen, S. Albanie, G. Sun and E. Wu, "Squeeze-and-excitation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020.

[37] H. Liu, H. Yuan, Q. Liu, J. Hou, H. Zeng and S. Kwong, "A hybrid compression framework for color attributes of static 3d point clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 3, pp. 1564–1577, March. 2022.

[38] H. Liu, H. Yuan, Q. Liu, J. Hou and J. Liu, "A comprehensive study and comparison of core technologies for mpeg 3-d point cloud compression," *IEEE Transactions on Broadcasting* vol. 66, no. 3, pp. 701–717, Sept. 2020.

[39] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, P. Vandergheynst, "Graph signal processing: overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, May. 2018.

[40] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: frequency analysis," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3042–3054, Jun. 2014.

[41] Y. Eldar, M. Lindenbaum, M. Porat and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," *IEEE Transactions on Image Processing*, vol. 6, no. 9, pp. 1305–1315, Sept. 1997.

[42] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jul. 2017, pp. 605–613.

[43] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D shapenets: a deep representation for volumetric shapes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2015, pp. 1912–1920.

[44] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, "Vision meets robotics: the kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 111, 2013.

[45] D. P. Kingma, J. L. Ba, "Adam: a method for stochastic optimization," *International Conference on Learning Representations*, 2015.

[46] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, "A benchmark for surface reconstruction," *ACM Transactions on Graphics*, vol. 32, no. 2, pp. 1–17, Apr. 2013.

[47] K. Low, "Linear least-squares optimization for point-to-plane icp surface registration," *Univ. North Carolina Chapel Hill, Chapel Hill, Tech. Rep. TR04-004*, Feb. 2004, vol. 4, no. 10, pp. 1¨C3.

[48] E. d'Eon, B. Harrison, T. Myers, and P. A. Chou, "8i voxelized full bodies - a voxelized point cloud dataset," *ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) WG11M40059/WG1M74006*, 2017.

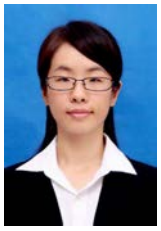
[49] MPEG Requirements Subgroup, "Evaluation criteria for point cloud compression," *ISO/IEC JTC1/SC29/WG11 MPEG*, *N16332*, Geneva, Jun. 2016.

[50] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 4, pp. 349–359, Oct. 1999.

Hao Liu received the B.E. degree in telecommunication engineering from Shandong Agricultural University, Taian, China, in 2017. He received the Ph.D. degree in information science and engineering from Shandong University, Qingdao, China, in 2022. He has been a lecturer with School of Computer Science and Control Engineering, Yantai University, since July 2022. His research interests include 3D point clouds compression and processing.

Hui Yuan (S'08–M'12–SM'17) received the B.E. and Ph.D. degrees in telecommunication engineering from Xidian University, Xi'an, China, in 2006 and 2011, respectively. In April 2011, he joined Shandong University, Ji'nan, China, as a Lecturer (April 2011–December 2014), an Associate Professor (January 2015–October 2016), and a Professor (September 2016). From January 2013–December 2014, and November 2017–February 2018, he also worked as a Postdoctoral Fellow (Granted by the Hong Kong Scholar Project) and a Research Fellow, respectively, with the Department of Computer Science, City University of Hong Kong, Hong Kong. From November 2020 to November 2021, he also worked as a Marie Curie Fellow (Granted by the Marie Skłodowska-Curie Individual Fellowships of European Commission) with the Faculty of Computing, Engineering and Media, De Montfort University, United Kingdom. From October 2021 to November 2021, he also worked as a visiting researcher (secondment of the Marie Skłodowska–Curie Individual Fellowships) with the Computer Vision and Graphics group, Fraunhofer Heinrich-Hertz–Institut (HHI), Germany. His current research interests include video/image/immersive media processing, coding, and adaptive streaming, etc. He served as an Area Chair of IEEE VCIP2020, IEEE ICME2020, ICME2021, and ICME2022.

Raouf Hamzaoui (S'02–SM'07) received the M.Sc. degree in mathematics from University of Montreal, Montral, QC, Canada, in 1993; the Dr.rer.nat. degree from University of Freiburg, Freiburg im Breisgau, Germany, in 1997; and the Habilitation degree in computer science from University of Konstanz, Konstanz, Germany, in 2004. He was an Assistant Professor with the Department of Computer Science, University of Leipzig, Leipzig, Germany, and with the Department of Computer and Information Science, University of Konstanz. He joined De Montfort University, Leicester, U.K., in 2006, where he is currently a Professor in Media Technology. His research interests include image and video coding, multimedia communication systems, error control systems, and machine learning.



Qi Liu received the M.S. degree from Xidian University, Xi'an, China in 2014 and the Ph.D. degree from Shandong University, Qingdao, China in 2021. From 2021.12 to now, she works as an Assistant Professor with the School of Electronic Information, Qingdao University, Qingdao, China. From 2018.09–2019.08, she also worked as an international visiting graduate student with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her research interests include point cloud compression and point cloud quality assessment.

Shuai Li is currently with the School of Control Science and Engineering, Shandong University (SDU), China, as a Professor and QiLu Young Scholar. He was with the School of Information and Communication Engineering, University of Electronic Science and Technology of China (UESTC), China, as an Associate Professor from 2018-2020. He received his Ph.D. degree from the University of Wollongong, Australia, in 2018. His research interests include image/video coding, 3D video processing and computer vision. He was a co-recipient of two best paper awards at the IEEE BMSB 2014 and IIH–MSP 2013, respectively.