SPU-PMD: Self-Supervised Point Cloud Upsampling via Progressive Mesh Deformation

Yanzhe Liu[1], Rong Chen *[1], Yushi Li *[2], Yixi Li[1] and Xuehou Tan[3]

[1]Dalian Maritime University
[2]Xi'an Jiaotong-Liverpool University
[3]Tokai University
{liuyanzhe,rchen,superlyxi}@dlmu.edu.cn
yushi.li@xjtlu.edu.cn xtan@tsc.u-tokai.ac.jp

Abstract

Despite the success of recent upsampling approaches, generating high-resolution point sets with uniform distribution and meticulous structures is still challenging. Unlike existing methods that only take spatial information of the raw data into account, we regard point cloud upsampling as generating dense point clouds from deformable topology. Motivated by this, we present SPU-PMD, a self-supervised topological mesh deformation network, for 3D densification. As a cascaded framework, our architecture is formulated by a series of coarse mesh interpolator and mesh deformers. At each stage, the mesh interpolator first produces the initial dense point clouds via mesh interpolation, which allows the model to perceive the primitive topology better. Meanwhile, the deformer infers the morphing by estimating the movements of mesh nodes and reconstructs the descriptive topology structure. By associating mesh deformation with feature expansion, this module progressively refines point clouds' surface uniformity and structural details. To demonstrate the effectiveness of the proposed method, extensive quantitative and qualitative experiments are conducted on synthetic and real-scanned 3D data. Also, we compare it with state-of-the-art techniques to further illustrate the superiority of our network. The project page is: https://github.com/lyz21/SPU-PMD.

1. Introduction

As unordered sets of discrete elements, point clouds provide high flexibility for 3D data representation [7, 8]. Due to these properties, point clouds have been applied in a wide variety of fields. However, the point clouds captured by 3D scanning (*e.g.*, LiDAR or RGB-D cameras) are often sparse and fragmented, which affects downstream tasks such as shape classification, object detection, and semantic segmentation [2, 9, 30, 42]. Hence, point cloud upsampling is a fundamental and crucial issue in 3D vision.

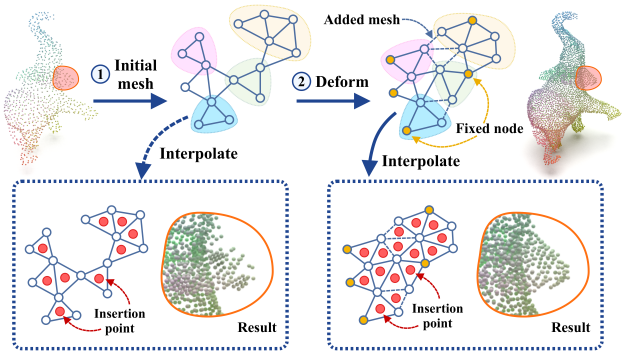

Figure 1. Illustration of point cloud upsampling based on mesh deformation. Although the meshes constructed from the point cloud topology coarsely locate the insertion points, the unconnected meshes often result in unexpected vacancies and detail missing (as shown in the lower left part of the figure). The proposed upsampling method that combines mesh interpolation with deformation successfully compensates for potential holes and maintains descriptive structures.

With the development of data-driven models, some learning-based models [4, 12, 13, 17, 19–21, 27, 29, 40, 41, 44] have been proposed for point cloud upsampling. The typical frameworks consist of three components: feature extraction (FE), feature expansion (FX), and coordinate reconstruction (CR). Early, this upsampling problem

*Corresponding authors. This work is supported by the National Natural Science Foundation of China (No.62002039, No. 61672122), and the Fundamental Research Funds for the Central Universities (No.36330603).

was simplified as a variant of image super-resolution. Thus, the pioneering works employ various extractors including Multi-Layer-Perceptron (MLP) [13, 40, 41], Convolution Neural Network (CNN) [5, 28], and Graph Convolutional Network (GCN) [19, 27] to learn representative features for estimating potential distribution. In feature expansion, most existing methods [4, 12, 13, 17, 40, 44] adopt direct feature duplication relying on the assumption that latent code amplification brings about spatial expansion of the corresponding points. To better expand the feature, [19] utilizes hierarchical folding to propagate features from sparse to dense. Finally, the reconstruction module is applied to regress the coordinates or offsets in the Cartesian system.

Although these approaches based on the FE-FX-CR architecture can learn structures from raw data, they only take spatial information into account and neglect the underlying topology maintained, which often results in unexpected distortion and surface outliers. In other words, they have limited ability in high-fidelity expression and generation. Furthermore, most of these approaches require dense point sets as the ground truth in supervised training, making them unavailable for the 3D data scanned in the real world.

To address these problems, we propose a self-supervised upsampling network, called SPU-PMD, that treats this task as topology-based point propagation. Instead of the FE-FX-CR, we devise a new pipeline of deformers based on mesh interpolation and deformation, which can associate local topology with spatial information and regulate the upsampling. Specifically, we first propose an interpolation algorithm that estimates the local centroids based on meshes and takes them as the insertion points to achieve primary point growth. Although this algorithm helps us construct the initial dense point cloud, the result produced by this interpolation is coarse and non-uniform since the meshes are created from sparse and irregular points. Therefore, we further introduce a mesh deformer that predicts the optimal topology modification to progressively generate the point clouds with meticulous structures and build uniform distributions. As shown in Fig.1, the mesh deformer architecture significantly improves uniformity and structure preservation.

In mesh deformation, we devise a Recurrent Feature Aggregation (RFA) module to infer the feature variation of each mesh node in latent space. RFA counts on a feature memory mechanism to utilize the information of different deformation for guiding future modification. With the spatiotemporal information of multi-step revision, this unit allows the model to prevent redundant deformation and better preserve descriptive geometries. We further design a motion estimation module that associates the latent features with mesh deformation in Cartesian space. In this motion component, a unit, called Gate-based Coordinate Reconstruction (GCR) is applied to determine the movable points and recover their spatial locations. Finally, the cascaded mesh deformation network is constructed by serially connecting a sequence of the deformers.

To validate the proposed model, we conduct extensive experiments on both the synthetic data and the point clouds captured in the real world. For synthetic data, we compare our framework with state-of-the-art upsampling approaches on public datasets PU1K [27] and PU-GAN [12]. For assessing the effectiveness of SPU-PMD in processing real data, the KITTI [6] dataset is employed. All the quantitative and qualitative experiments demonstrate that our method is competitive in the point cloud upsampling task. The comprehensive ablation study is also presented to exploit the effects of different components on the overall architecture.

2. Related Work

2.1. Point Cloud Learning

As a way to represent discrete data, point clouds are unpermuted and irregular, making their processing difficult. Early methods [16, 23, 24, 24, 32, 34, 37] often map the 3D shape into multi-views or voxel grid. However, these conversions are time-consuming and cannot preserve complete information. To overcome these problems, PointNet [25] directly extract the raw point clouds features by MLPs. Inspired by this, variant MLP-based networks [22, 26, 35, 39, 45, 46] are proposed to improve global feature aggregation.

Since the graph is a natural representation of point cloud structure, graph-based learning has been introduced in this domain. As a pioneering work, Simonovsky *et al*. [31] proposed Edge-Conditioned Convolution (ECC) that conditions the filter weights on edge labels. Similarly, several studies [2, 36, 38] highlight aggregated local information based on graphs. Different from spatial convolution, spectral convolution associates the Laplacian method with graph signals. RGCNN [33] defines the spectral convolution on a graph by Chebyshev polynomial approximation and adapts it to point cloud learning.

With the advancement of Attention methods, Li *et al*. [14, 15] combined spectral GCN with attention mechanism in unsupervised point cloud learning. As another attention-based model, GAPNet [1] integrates graph attention mechanism into MLPs to learn local geometric representation. In addition, some studies represent point clouds as neural fields [43], limiting their applicable scenarios.

2.2. Supervied Point Cloud Upsampling Methods

Upsampling point clouds in a supervised manner requires paired dense point sets as the ground truth for training. As an early work, PU-Net [41] first establishes the FE-EX-CR structure, which lays the foundation for upsampling frameworks. PU-GCN [27] introduces graph convolution to encode local features from the neighborhood and incorporates it into the upsampling framework. PU-Transformer [29] is

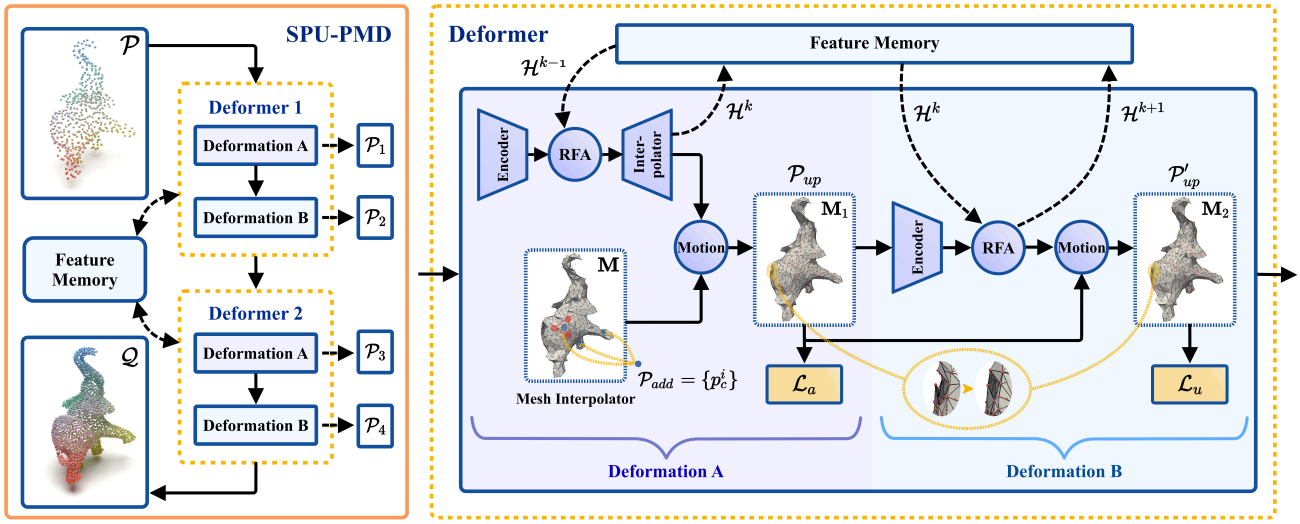

Figure 2. Illustration of the proposed SPU-PMD. The left part presents the overall architecture of SPU-PMD, and the right part indicates the internal details of our deformer consisting of several essential modules.

the first model that applies Transformer in point cloud upsampling, which relies on a new multi-head self-attention approach to enhance both point-wise and channel-wise relations of the features. PC^2-PU [20] considers the point and patch correlations. By expanding the perception range, this model makes the generated points closer to the underlying surface. Based on FE-EX-CR architecture, some models [4, 13, 40] use cascade structure to improve performance. Different from these networks that have PU-Net-like architectures, [12, 17] take GAN as the basic pipeline. To achieve flexible upsampling, [10] interpolates the input points and refines their positions in an iterative process.

2.3. Unsupervied Point Cloud Upsampling Methods

Because obtaining a large number of paired sparse and dense point sets is expensive and tedious, some unsupervised learning models have been proposed to avoid this barrier. L2G-AE [18] builds an autoencoder to learn the local and global features of point clouds through local-to-global reconstruction and finally fuse the reconstruction results to produce upsampled point clouds. This model is not specifically designed for point cloud upsampling, so its capability in this task is limited. SAPCU [47] transforms point cloud upsampling into finding the nearest projected seed points on an implicit surface. However, the unexpected noise created by this model makes it unavailable in practice. Afterward, Liu *et al*. [19] proposed a self-supervised point cloud upsampling model, SPU-Net. This model uses sparse input as the supervision and overlaps multiple reconstruction results to obtain the final upsampled set like L2G-AE.

3. Methodology

In this work, we model the point cloud upsampling as inferring dense sets from deformable meshes. Given a sparse point cloud $\mathcal{P} = \{p_i\}_{i=1}^{N}, p_i \in \mathbb{R}^3$ as input, the proposed model first generates a mesh $\mathbf{M}$ to present the topology of the original shape and produces the initial upsampled set through mesh interpolation. Then, it uses the deformers to progressively revise $\mathbf{M}$ and densify the point cloud based on the well-modified mesh. The pipeline of the SPU-PMD model is shown in Fig.2, consisting of two deformers.

To incorporate point upsampling with mesh revision, each deformer includes two stages of deformation. In deformation A, the feature of each point is first extracted by an encoder, followed by an RFA that fuses the information from different deformation stages to guide the movements of mesh nodes. Subsequently, the interpolator expands the features to correspond with the point clouds upsampled $\mathcal{P}_{up}$ by mesh interpolation. After estimating the moving nodes, the motion unit provides the new mesh $\mathbf{M_1}$. While deformation A allows the model to expand features and perform mesh reform, some unexpected node connections lead to small holes and non-uniform point insertions. Hence, deformation B is applied to refine the mesh and upsampled point distribution gives $\mathbf{M_2}$ and $\mathcal{P}'_{up}$. This refinement enables our model to better recover precious details and close off surface holes. In the following, we discuss the details of the mesh interpolation and each module in the deformer.

3.1. Mesh interpolation

The mesh interpolation aims at increasing the point number on the basis of the initial structure. Specifically, we first calculate the average distance d between each point p and

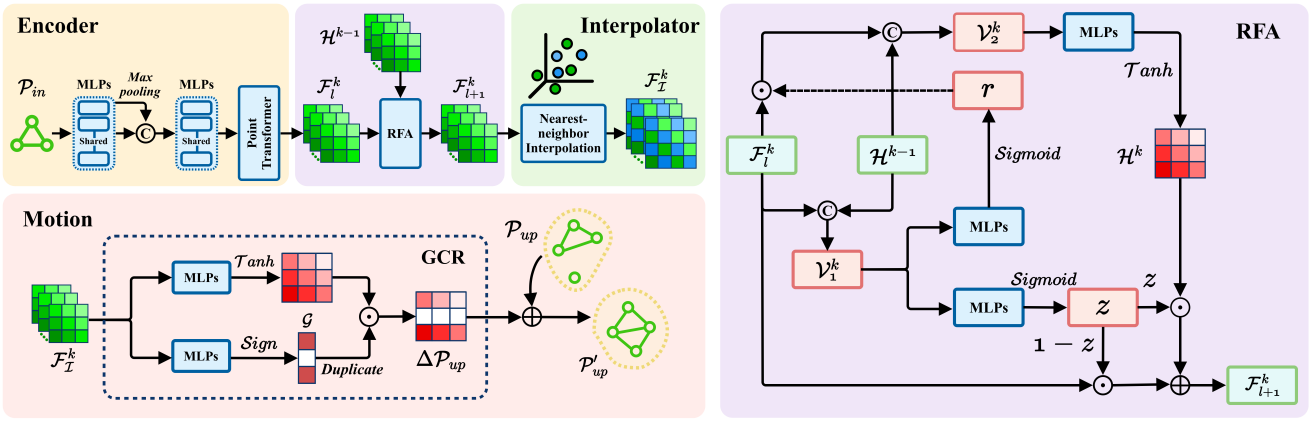

Figure 3. Internal modules of deformer. In this figure, we present the details of different modules in the proposed deformer including the encoder, interpolator, recurrent feature aggregation (RFA), and motion estimation unit.

its nearest neighbor p', and use d as the radius $\mathcal{R}$ of the ball query F to construct meshes $\mathcal{M} \in \mathbb{R}^{K\times 3\times 3}$. After this, we calculate the centroid p_c of each mesh and insert it into the original point cloud $\mathcal{P}_{in}$ for coarse densification. This mesh-based interpolation is repeated until the quantity of points reaches or exceeds the expected number rN, where r is the upsampling ratio. Finally, we carry out the farthest point sampling to downsample the point cloud to $\mathcal{P}_{up}=\{p_i\}_{i=1}^{rN}$, denoted as:

$$\mathcal{P}_{up} = \mathcal{FPS}(\Phi(\mathcal{P}_{in})) \tag{1}$$

where Φ presents the interpolation. The detailed algorithm of our mesh interpolation is presented in Algorithm.1.

Algorithm 1: Mesh Interpolation.

Input: a sparse point set: $\mathcal{P}_{in} \in \mathbb{R}^{N\times 3}$
upsampling rate: $r \in R$
Output: a dense point set: $\mathcal{P}_{up} \in \mathbb{R}^{rN\times 3}$

$\mathcal{P}_{up} \leftarrow \mathcal{P}_{in}$; $N_{up} \leftarrow N$
$\mathcal{P}_{add} \leftarrow \emptyset$
while $N_{up} < rN$ **do**
  $d = \sum_{i=1}^{N} \|p_i - p_i'\|_2 \ / \ N$;
  $\mathcal{R} = [\, d,\ 1.3 \times d \,]$;
  $\mathcal{M} = F\,(\,\mathcal{P}_{in},\ \mathcal{R}\,)$; // $\mathcal{M} \in \mathbb{R}^{K\times 3\times 3}$
  for $i \in \{1, 2, \cdots, K\}$ **do**
    $x = \sum_{j=0}^{2} \mathcal{M}\,[\,i,\ j,\ 0\,] \ / \ 3$;
    $y = \sum_{j=0}^{2} \mathcal{M}\,[\,i,\ j,\ 1\,] \ / \ 3$;
    $z = \sum_{j=0}^{2} \mathcal{M}\,[\,i,\ j,\ 2\,] \ / \ 3$;
    $p_c^i = [\,x,\ y,\ z\,]$;
  end
  Obtain: $\mathcal{P}_{add} = \{\, p_c^0,\ p_c^1,\ \cdots, p_c^K \,\}$
  $\mathcal{P}_{up} = \mathcal{P}_{up} \cup \mathcal{P}_{add}$ // $\mathcal{P}_{up} \in \mathbb{R}^{N_{up}\times 3}$
end
$\mathcal{P}_{up} = \mathcal{FPS}(\,\mathcal{P}_{up}\,)$ // $\mathcal{P}_{up} \in \mathbb{R}^{rN\times 3}$

3.2. Mesh deformation

3.2.1 Encoder

For coding the input point cloud $\mathcal{P}_{in}$ into high-dimensional features, we design a transformer-based encoder. As shown in Fig.3, MLPs and max pooling are mixed to extract individual point and global features. After feature concatenation, an additional MLP block is employed for feature fusion. At the end of the encoder, we adopt Point Transformer [46] to refine the local shape context and provide the feature $\mathcal{F} = \{f_i\}_{i=1}^{N}, f_i \in \mathbb{R}^C$ for the following RFA. This design facilitates the encoder to capture multiple-scale information from the point cloud.

3.2.2 RFA

The main challenge of mesh deformation is effectively merging the spatial-temporal deforming information to guide the mesh change. To address this issue, we propose a Recurrent Feature Aggregation (RFA) module that can reconcile historical and current information to dominate the offset of each mesh node and prevent contradictory motion decisions. The internal architecture of RFA is illustrated in the right part of Fig.3. At first, RFA combines the current information $\mathcal{F}_l^k$ (feature of l-th level at k-th stage) with the historical memory feature $\mathcal{H}^{k-1}$, which is defined as:

$$\mathcal{V}_1^k = Concat(\mathcal{F}_1^k, \mathcal{H}^{k-1}) \tag{2}$$

$$\mathcal{V}_2^k = Concat(r \odot \mathcal{F}_1^k, \mathcal{H}^{k-1}) \tag{3}$$

where $Concat$ denotes concatenation, and r is defined as:

$$r = \sigma(h(\mathcal{V}_1^k, \mathbf{W}_r, \mathbf{b}_r)) \tag{4}$$

In this equation, σ is the $sigmoid$ function and h presents the MLP. Then, we compute the current memory feature $\mathcal{H}^k$ in accordance with the input $\mathcal{F}_1^k$ and previous information $\mathcal{H}^{k-1}$as:

$$\mathcal{H}^k = \varphi(h(\mathcal{V}_2^k, \mathbf{W}, \mathbf{b})) \tag{5}$$

where φ is $tanh$ function. By fusing the current memory feature with $\mathcal{V}_1^k$ updated from F_l^k and H^{k-1}, we get the final output of RFA:

$$\mathcal{F}_{l+1}^k = z \odot \mathcal{H}^k + (1 - z) \odot \mathcal{F}_l^k \tag{6}$$

given that

$$z = \sigma(h(\mathcal{V}_1^k, \mathbf{W}_z, \mathbf{b}_z)) \tag{7}$$

As a variant of Gate Recurrent Unit (GRU), RFA computes the gates z and r to embed the historical information into the current feature. Unlike GRU, RFA utilizes the reset gate r to manage the input $\mathcal{F}_l$ instead of the historical information $\mathcal{H}^{k-1}$ since maintaining the complete historical information in memory feature estimation enables the module to better supervise mesh deformation and prevent contradictory motion.

3.2.3 Interpolator

To incorporate the point clouds upsampled by mesh interpolation with fitting features, we propose an interpolator (as shown in the upper part of Fig.3). Unlike traditional FX which directly expands the features in latent space, our unit associates feature interpolation with the spatial neighborhood of points. In particular, the feature corresponding to the insertion point generated by mesh interpolation is estimated from the nearest neighboring point's feature. Our interpolation can be expressed as:

$$\mathcal{F}_{\mathcal{I}}^k = \{f_{p_i} | \, f_{p_i} \in \mathcal{N}_{p_i}(\mathcal{F}_{l-1}^k), p_i \in \mathcal{P}_{up}\}_{i=1}^{rN} \tag{8}$$

where $\mathcal{F}_{\mathcal{I}}^k$ is the new features produced by the interpolator. Meanwhile, $\mathcal{F}_{l-1}^k = \{f_{p_i}\}_{i=1}^N$. Note that $\mathcal{N}_{p_i}$ presents searching for the nearest neighbor features of p_i in the feature space.

3.2.4 Motion estimation

After corresponding the features with upsampled points, we use a motion estimation unit with a gate mechanism called GCR to realize the mesh deformation by moving the mesh nodes. This module employs two branches to separately process the input features. In the lower branch, we compute a soft gate to decide whether the point needs to be moved as follows:

$$\mathcal{G} = \gamma(h(\mathcal{F}_{\mathcal{I}}^k, \mathbf{W}_g, \mathbf{b}_g)) \tag{9}$$

where γ is $sign$ function. Parallelly, the upper branch regresses the input features into spatial offset:

$$\Delta\mathcal{P}_{up} = \varphi(h(\mathcal{F}_{l+1}^k, \mathbf{W}_o, \mathbf{b}_o)) \tag{10}$$

Relying on the estimated gate and offset a new point cloud is obtained:

$$\mathcal{P}'_{up} = \mathcal{P}_{up} + \tau \cdot (\mathcal{G} \odot \Delta\mathcal{P}_{up}) \tag{11}$$

where τ is the moving radius set for controlling the movement range of mesh nodes. The detailed design of the motion estimation module is indicated in the lower left side of Fig.3.

3.3. Loss

To train the proposed model, we apply the Chamfer Distance (CD) and uniform losses. The CD loss evaluates the difference between point sets by calculating the average shortest distance. It forces the upsampled point clouds to maintain the original geometry and is defined as:

$$\mathcal{L}_{CD}(\mathcal{P}, \mathcal{Q}) = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \min_{q \in \mathcal{Q}} \|p - q\|_2^2 + \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \min_{p \in \mathcal{P}} \|p - q\|_2^2 \tag{12}$$

Where $\mathcal{Q}$ and $\mathcal{P}$ represent the upsampled point cloud and the corresponding ground truth. Due to the mesh-based method, the upsampled points are often located in the neighborhood of initial mesh nodes, which allows us to directly use the initial input as $\mathcal{P}$.

The uniform loss is applied to measure the uniformity of the upsampled result. It calculates the distance between the points in each patch:

$$\mathcal{L}_u(\mathcal{Q}) = \sum_{j=1}^{M} \left[\frac{(|S_j - \hat{n}|)^2}{\hat{n}} \times \sum_{j=1}^{|S_j|} \frac{\left(d_{j,k} - \hat{d}\right)^2}{\hat{d}} \right] \tag{13}$$

where M is the sample number, and S_j is the subset obtained by performing a spherical query with radius r_d on the M sample points. Additionally, $\hat{n} = N \times r_d^2$ is the number of expected points in S_j. $d_{j,k}$ is the distance to the nearest neighbor of the kth point in S_j, and $\hat{d} = \sqrt{\frac{2\pi r_d^2}{|S_j|\sqrt{3}}}$ is the corresponding expected distance. The first part of this formula accounts for the nonlocal uniformity, and the second part measures the local uniformity.

To preserve the original structure, we take both the CD and uniformity evaluations as the loss in deformation A:

$$\mathcal{L}_a = \mathcal{L}_{CD} + \alpha\mathcal{L}_u \tag{14}$$

where α is a hyperparameter used to weight the uniform loss. Different from this, only uniformity loss is applied in deformation B for further constraining the point distribution. Overall, the total training loss function is defined as:

$$\begin{aligned} \mathcal{L}_{total} = \mathcal{L}_a(\mathcal{P}_1, \mathcal{P}) + \mathcal{L}_u(\mathcal{P}_2) \\ + \mathcal{L}_a(\mathcal{P}_3, \mathcal{P}) + \mathcal{L}_u(\mathcal{P}_4) \end{aligned} \tag{15}$$

Among them, $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, and $\mathcal{P}_4$ represent the point clouds updated at different stages.

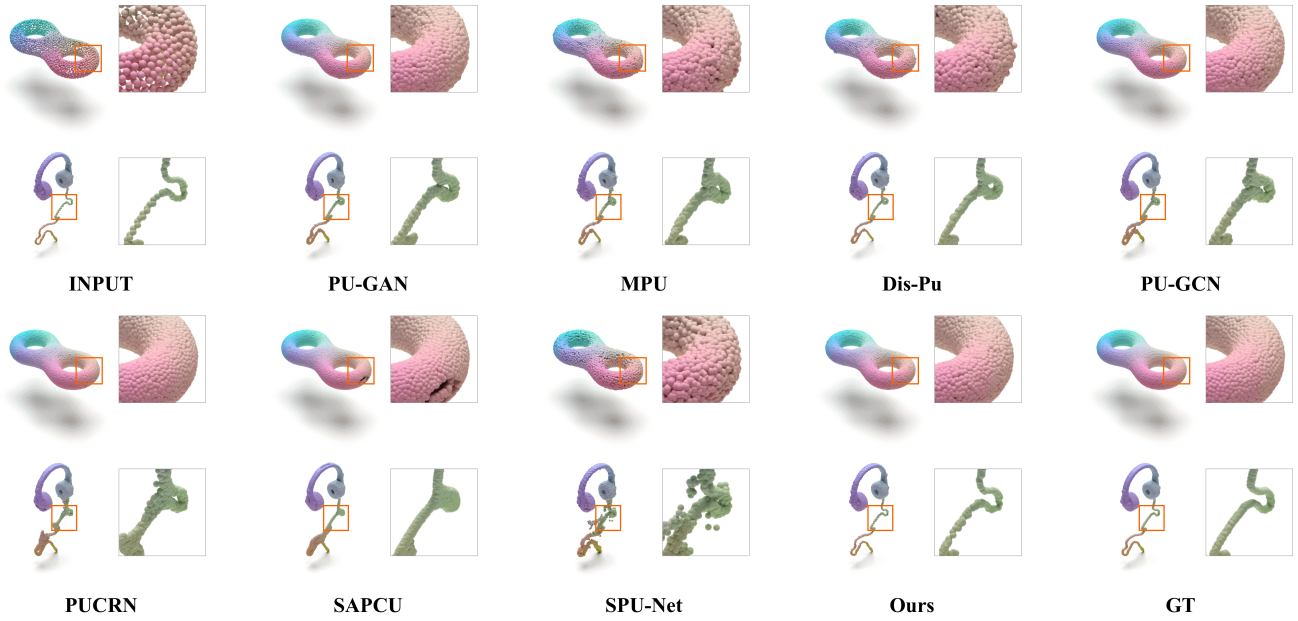

Figure 4. Visualization comparison with other methods. From these results, we can see that our model outperforms other methods in uniform generation and meticulous structure preservation.

4. Experiments

In this section, we compare SPU-PMD with state-of-the-art (SoTA) supervised and unsupervised methods to validate the effectiveness of our method. All quantitative and qualitative experiments are conducted on the synthetic (PU1K [27] and PU-GAN [12]). The real scanned KITTI [6] is applied to analyze the performance of SPU-PMD in practice. We also provide deformation analysis and various ablation studies to exploit the mesh deformation property of SPU-PMD and the contributions of different modules.

Datasets. We evaluate different upsampling techniques on PU1K [27] and PU-GAN [12]. PU1K contains 1147 3D models, of which 1020 are used for training and 127 are for testing. In comparison with PU-GAN, PU1K is more challenging as it has a larger amount of data and more diverse models. The PU-GAN dataset contains 147 objects that 120 are used in training. Besides the synthetic datasets, the LiDAR data from KITTI [6] is applied for further evaluation.

Experimental Setting. We train the proposed network using an Aadm optimizer with a learning rate of 0.00001. The max training epoch of our model is 100 and the batch size we employed is 32. The hyperparameter α in Eq.14 is set as 0.1. For a comprehensive comparison, all models are retrained following their respective paper settings. This enables us to standardize the indicators across both datasets. The GPUs utilized in this work are two NVIDIA 2080TI.

Evaluation Metrics. Following recent point cloud upsampling works [10, 21, 27], we choose Chamfer distance (CD), Hausdorff distance (HD), and point-to-surface distance (P2F) as the evaluation metrics. In the quantitative evaluation tables, a smaller metric means better results. Meanwhile, the bolded values represent the optimal results, and the underlined ones are suboptimal.

4.1. Comparison with SoTA methods

Evaluation on PU1K Dataset. To fairly compare SPU-PMD with SoTA approaches, we follow the setting of PU-CRN [4] and experiment on the PU1K point clouds [27] with different densities: sparse, medium, and dense (As shown in Table.1). These quantitative results demonstrate that our model significantly surpasses both compared supervised (PU-Net [41], MPU [40], PU-GAN [12], Dis-PU [13], PU-GCN [27], and PU-CRN [4]) and unsupervised models (SAPCU [47] and SPU-Net [19]) in HD and P2F metrics. Only PU-CRN slightly outpaces our model in CD results.

In Fig. 4, we present the visual results produced by different upsampling methods to illustrate their performance in uniform generation and detail preservation. To the ring (the upper row), most SoTA methods provide results with non-uniform distributions and surficial fluctuations. By contrast, SPU-PMD generates a much smoother and more uniform point cloud. From the headphone results (the lower row), we can discern that only SPU-PMD maintains a meticulous structure close to the ground truth. More visual comparison is presented in the supplementary material.

Evaluation on PU-GAN Dataset. To assess the gener-

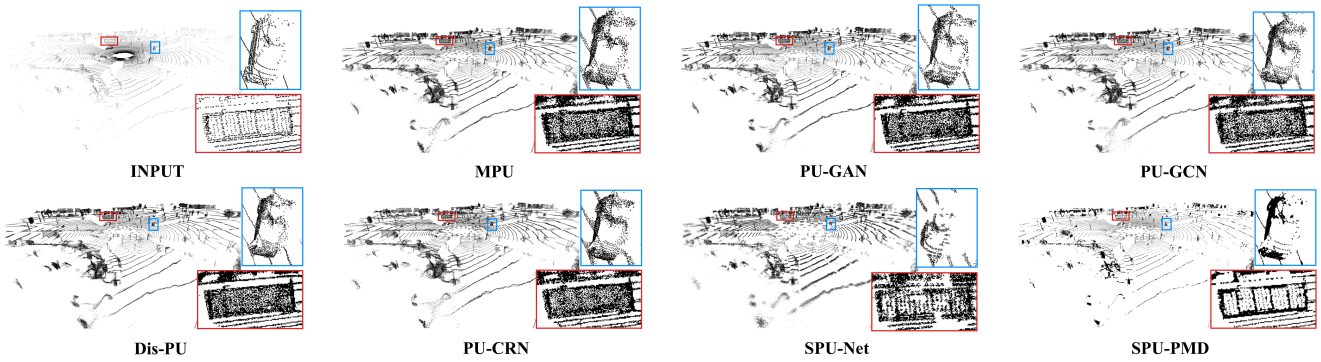

Figure 5. Visualization on real-scanned scene. Our model achieves competitive results especially in preserving the initial structure details.

Table 1. Quantitative results on the PU1K dataset

Methods	GT	Sparse (512) input			Medium (1,024) input			Dense (2,048) input		
		CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}
PU-Net [41]	✔	2.999	36.129	11.077	1.899	24.754	7.321	1.155	15.170	4.847
MPU [40]	✔	2.803	30.843	8.334	1.679	21.119	5.450	0.935	13.327	3.560
PU-GAN [12]	✔	1.991	22.642	6.979	1.132	14.809	4.530	0.707	10.411	2.963
Dis-PU [13]	✔	3.616	37.134	9.911	2.265	24.455	6.120	1.380	16.524	3.880
PU-GCN [27]	✔	1.817	19.153	6.104	1.035	12.032	3.946	0.585	7.577	2.504
PU-CRN [4]	✔	**1.611**	18.835	5.161	**0.861**	12.214	3.246	**0.499**	8.068	2.027
SAPCU [47]	✘	2.973	30.237	9.030	1.754	21.292	4.712	1.130	14.903	2.462
SPU-Net [19]	✘	2.863	63.031	10.262	1.338	37.368	6.444	0.955	21.058	4.083
Ours	✘	1.690	**13.568**	**4.082**	0.892	**8.252**	**2.765**	0.544	**4.926**	**1.861**

alization of the proposed model, we directly evaluate the model trained with PU1K data on the PU-GAN dataset [12]. From Table.2, we can see that SPU-PMD outperforms all the other methods in HD and P2F assessments. The CD results of our model are only worse than PU-CRN which is a supervised network. This additional test demonstrates that our network can be well generalized to unseen data.

Table 2. Quantitative results on the PU-GAN dataset

Methods	GT	Sparse (512) input			Medium (1,024) input			Dense (2,048) input		
		CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}
PU-Net [41]	✔	2.619	33.877	20.075	1.309	19.138	11.970	0.817	11.150	7.838
MPU [40]	✔	2.268	24.651	13.522	1.236	16.116	8.449	0.713	10.614	5.381
PU-GAN [12]	✔	1.478	16.807	10.859	0.768	12.250	6.593	0.469	8.220	4.047
PU-GCN [27]	✔	1.504	18.105	10.414	0.774	9.594	6.197	0.401	5.630	3.650
PU-CRN [4]	✔	**1.167**	19.238	7.635	**0.537**	8.819	4.165	**0.289**	4.175	2.369
SAPCU [47]	✘	2.490	28.215	16.236	1.183	19.986	7.670	0.443	10.397	3.446
SPU-Net [19]	✘	2.799	69.416	17.895	1.104	39.023	10.289	0.509	23.497	6.106
Ours(PUGAN)	✘	1.186	**10.603**	**6.554**	0.602	**5.762**	**4.040**	0.314	**3.320**	**2.441**
Ours(PU1K)	✘	1.226	**10.626**	**6.675**	0.612	**5.724**	**4.124**	0.318	**3.317**	**2.508**

Evaluation on KITTI Dataset. Apart from synthetic data, we compare our unsupervised method with SoTA using the real-world KITTI dataset [6]. In particular, the supervised models are trained on the PU1K dataset since there is no paired ground truth for supervision. The visual results generated are presented in Fig. 5, which exhibits that our model exceeds others in conserving the initial structures such as the fence and car emphasized by the red and blue frames.

Table 3. Analysis of Every Stage.

Stage	CD ↓ (10^{-3})	HD ↓ (10^{-3})	P2F avg ↓ (10^{-3})	P2F std ↓ (10^{-3})
Deformer 1-A	1.586	11.811	1.503	2.903
Deformer 1-B	1.279	9.776	1.672	2.940
Deformer 2-A	0.551	4.950	1.828	3.204
Deformer 2-B	0.544	4.926	1.861	3.239

4.2. Mesh deformation analysis

To exploit the property of mesh deformation, we visualize the mesh changing in different deformation steps in Fig. 6. According to these visual results, the meshes gradually become more compact and uniform, which is consistent with the expectation. With the mesh deforming, the holes and surface fluctuations are disposed of, resulting in much smoother objects with more precise details.

We compare the CD, HD, P2F (avg), and P2F (std) metrics estimated from the point clouds generated in different deformation steps to further evaluate the effect of each deformation. From Table. 3, both CD and HD results are significantly improved by the mesh deformation, meaning this approach effectively rectifies the point distribution. Because the insertion point locations are determined by the approximated surface mesh, P2F metrics increase in the second deformer. Nevertheless, our model provides the best results in P2F in comparison with other methods.

Fig. 7 shows the morphing of the mesh nodes. To clearly show the node movements, we use different colors to represent the moving magnitude of mesh nodes. For instance, purple indicates a small motion range and the lighter colors mean apparent movements. Like the mesh analysis, the de-

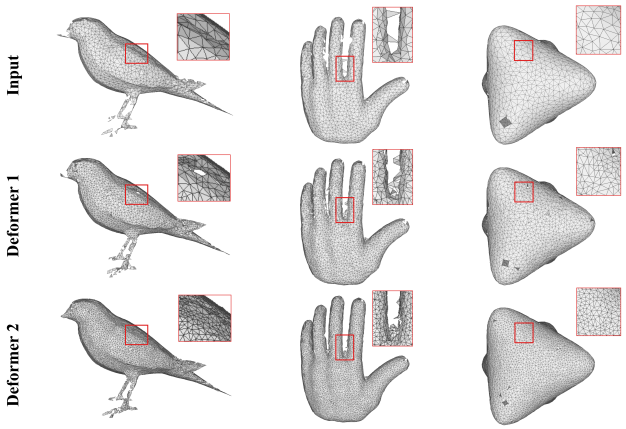

Figure 6. Mesh deformation analysis.

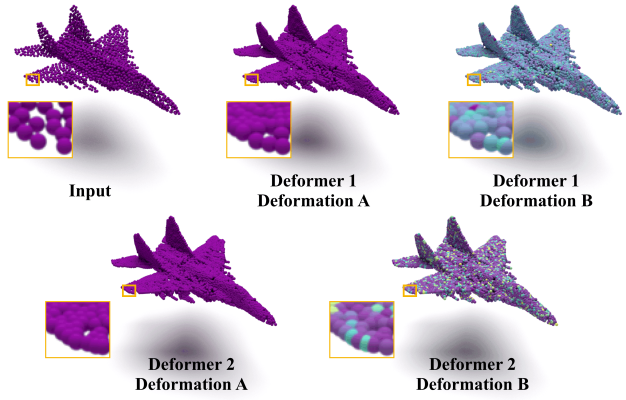

Figure 7. Visualization of node motion.

formers gradually improve the distribution and compactness of the upsampled points by revising the regional topology based on meshes. More experimental results of deformation analysis are discussed in the supplementary material.

4.3. Ablation Study

We perform the ablation study on the PU1K dataset to evaluate the contributions of different components.

Network architecture. In rows 2-4 of Table. 4, we evaluate how the performance varies with different network configurations. At first, RFA is removed from our network. Without this module, all quantitative metrics increase obviously. Then, we replace the RFA with GRU, which provides better results than the network without RFA. To assess the contribution of GCR in the motion estimation unit, we remove it and only retain the coordinate regression branch to predict the node movement. This study indicates that all the proposed modules are beneficial to SPU-PMD.

Interpolation. This study is conducted to explore the pre-interpolation required in our framework. We employ two

Table 4. Ablation Study.

Methods	Sparse (512) input			Medium (1,024) input			Dense (2,048) input		
	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}	CD ↓ 10^{-3}	HD ↓ 10^{-3}	P2F ↓ 10^{-3}
w/o RFA	1.722	13.957	4.179	0.894	8.343	2.844	0.560	5.074	1.909
GRU	1.711	13.687	4.106	0.889	8.288	2.789	0.554	5.024	1.875
w/o GCR	1.716	13.845	4.252	0.887	8.210	2.889	0.553	4.945	1.962
LI	11.228	63.876	0.473	11.616	73.153	0.352	12.994	81.709	0.259
PD	3.162	17.502	0.508	1.903	11.869	0.367	1.401	8.507	0.269
$\mathcal{L}_{cd} + \mathcal{L}_u$	1.965	15.054	3.767	1.060	9.125	2.571	0.657	5.476	1.733
$\mathcal{L}_{cd}$	2.000	15.273	3.734	1.085	9.194	2.558	0.695	5.723	1.723
baseline	1.690	13.568	4.082	0.892	8.252	2.765	0.544	4.926	1.861

interpolation methods: linear interpolation (LI) and point duplication (PD) to coarsely upsample the point cloud. In this experiment, we respectively integrate these two interpolation approaches with the learnable deformers. According to the results in rows 5 and 6 in Table. 4, none of them can work well as mesh interpolation in our framework. Except for P2F, all the other metrics are negatively affected. Under our analysis, LI and PD provide the results with lower P2F values on account that they are performed on the same planes, reducing the distances from individual points to the underlying surface.

Loss functions. We examine the performance of different loss combinations. To compare with the baseline, we conduct two separate comparative experiments: (1) $\mathcal{L}_a$ is used in each stage; (2) only $\mathcal{L}_{cd}$ is applied. It can be observed from rows 7 and 8 of Table. 4 that adding $\mathcal{L}_u$ is useful in optimizing the model. Among the variations, equipping $\mathcal{L}_a$ with deformation A and $\mathcal{L}_u$ with deformation B (the loss applied in the baseline) provides the best results in both metrics. This demonstrates that constraining the uniformity in deformation A is more effective.

5. Conclusion

In this paper, we propose a novel self-supervised point cloud upsampling model, SPU-PMD, which utilizes the topology presented by mesh to guide the point cloud upsampling. Different from traditional FE-FX-CR architecture, this network locates the upsampled points through progressive mesh deformation. Relying on the proposed deformer, this new upsampling mechanism allows SPU-PMD to ameliorate uniformity, and recover rich structural details as well. We present comprehensive experiments to demonstrate the effectiveness of our model in the upsampling task. In comparison with SoTA techniques, SPU-PMD significantly outperforms them in most evaluation metrics. Even though some compared methods are supervised, our model is superior to them in uniform generation and structure preservation.

References

[1] Can Chen, Luca Zanotti Fragonara, and Antonios Tsourdos. Gapointnet: Graph attention based point neural network for exploiting local feature of point cloud. *Neurocomputing*, 438:122–132, 2021. 2

[2] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[3] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 5

[4] Hang Du, Xuejun Yan, Jingjing Wang, Di Xie, and Shiliang Pu. Point cloud upsampling via cascaded refinement network. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 586–601, 2022. 1, 2, 3, 6, 7

[5] Wanquan Feng, Jin Li, Hongrui Cai, Xiaonan Luo, and Juyong Zhang. Neural points: Point cloud representation with neural fields for arbitrary upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18633–18642, 2022. 2

[6] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 2, 6, 7

[7] Yulan Guo, Ferdous Sohel, Mohammed Bennamoun, Min Lu, and Jianwei Wan. Rotational projection statistics for 3d local surface description and object recognition. *International journal of computer vision*, 105:63–86, 2013. 1

[8] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, and Jianwei Wan. 3d object recognition in cluttered scenes with local surface features: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11): 2270–2287, 2014. 1

[9] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep Learning for 3D Point Clouds: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4338–4364, 2021. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. 1

[10] Yun He, Danhang Tang, Yinda Zhang, Xiangyang Xue, and Yanwei Fu. Grad-pu: Arbitrary-scale point cloud upsampling via gradient descent with learned distance functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2023. 3, 6

[11] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3), 2013. 3

[12] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. PU-GAN: A Point Cloud Upsampling Adversarial Network. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7202–7211, 2019. ISSN: 2380-7504. 1, 2, 3, 6, 7

[13] Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu. Point cloud upsampling via disentangled refinement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 344–353, 2021. 1, 2, 3, 6, 7

[14] Yushi Li and George Baciu. Hsgan: Hierarchical graph learning for point cloud generation. *IEEE Transactions on Image Processing*, 30:4540–4554, 2021. 2

[15] Yushi Li and George Baciu. Sg-gan: Adversarial self-attention gcn for point cloud topological parts generation. *IEEE Transactions on Visualization and Computer Graphics*, 28(10):3499–3512, 2021. 2

[16] Yangyan Li, Soeren Pirk, Hao Su, Charles R Qi, and Leonidas J Guibas. Fpnn: Field probing neural networks for 3d data. *Advances in neural information processing systems*, 29, 2016. 2

[17] Hao Liu, Hui Yuan, Junhui Hou, Raouf Hamzaoui, and Wei Gao. Pufa-gan: A frequency-aware generative adversarial network for 3d point cloud upsampling. *IEEE Transactions on Image Processing*, 31:7389–7402, 2022. 1, 2, 3

[18] Xinhai Liu, Zhizhong Han, Xin Wen, Yu-Shen Liu, and Matthias Zwicker. L2G Auto-encoder: Understanding Point Clouds by Local-to-Global Reconstruction with Hierarchical Self-Attention. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 989–997, Nice France, 2019. ACM. 3

[19] Xinhai Liu, Xinchen Liu, Yu-Shen Liu, and Zhizhong Han. Spu-net: Self-supervised point cloud upsampling by coarse-to-fine reconstruction with self-projection optimization. *IEEE Transactions on Image Processing*, 31:4213–4226, 2022. 1, 2, 3, 6, 7

[20] Chen Long, WenXiao Zhang, Ruihui Li, Hao Wang, Zhen Dong, and Bisheng Yang. Pc2-pu: Patch correlation and point correlation for effective point cloud upsampling. In *Proceedings of the 30th ACM International Conference on Multimedia*, page 2191–2201, New York, NY, USA, 2022. Association for Computing Machinery. 3

[21] Luqing Luo, Lulu Tang, Wanyi Zhou, Shizheng Wang, and Zhi-Xin Yang. Pu-eva: An edge-vector based approximation solution for flexible-scale point cloud upsampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16208–16217, 2021. 1, 6

[22] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022. 2

[23] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 922–928. IEEE, 2015. 2

[24] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016. 2

[25] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference*

on computer vision and pattern recognition, pages 652–660, 2017. 2

[26] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2

[27] Guocheng Qian, Abdulellah Abualshour, Guohao Li, Ali Thabet, and Bernard Ghanem. PU-GCN: Point Cloud Upsampling using Graph Convolutional Networks. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11678–11687, Nashville, TN, USA, 2021. IEEE. 1, 2, 6, 7

[28] Yue Qian, Junhui Hou, Sam Kwong, and Ying He. Pugeonet: A geometry-centric network for 3d point cloud upsampling. In *European conference on computer vision*, pages 752–769. Springer, 2020. 2

[29] Shi Qiu, Saeed Anwar, and Nick Barnes. Pu-transformer: Point cloud upsampling transformer. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 2475–2493, 2022. 1, 2

[30] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. Towards 3D Point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008. 1

[31] Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3693–3702, 2017. 2

[32] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 2

[33] Gusi Te, Wei Hu, Amin Zheng, and Zongming Guo. Rgcnn: Regularized graph cnn for point cloud segmentation. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 746–754, 2018. 2

[34] Dominic Zeng Wang and Ingmar Posner. Voting for voting in online point cloud object detection. In *Robotics: science and systems*, pages 10–15. Rome, Italy, 2015. 2

[35] Junliang Wang, Chuqiao Xu, Lu Dai, Jie Zhang, and Ray Zhong. An unequal deep learning approach for 3-d point cloud segmentation. *IEEE Transactions on Industrial Informatics*, 17(12):7913–7922, 2020. 2

[36] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 2

[37] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 2

[38] Qiangeng Xu, Xudong Sun, Cho-Ying Wu, Panqu Wang, and Ulrich Neumann. Grid-gcn for fast and scalable point cloud learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5661–5670, 2020. 2

[39] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5589–5598, 2020. 2

[40] Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung. Patch-based progressive 3d point set upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 3, 6, 7

[41] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-net: Point cloud upsampling network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 6, 7

[42] Yiming Zeng, Yu Hu, Shice Liu, Jing Ye, Yinhe Han, Xiaowei Li, and Ninghui Sun. RT3D: Real-Time 3-D Vehicle Detection in LiDAR Point Cloud for Autonomous Driving. *IEEE Robotics and Automation Letters*, 3(4):3434–3440, 2018. Conference Name: IEEE Robotics and Automation Letters. 1

[43] Biao Zhang, Jiapeng Tang, Matthias Nießner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Trans. Graph.*, 42(4), 2023. 2

[44] Pingping Zhang, Xu Wang, Lin Ma, Shiqi Wang, Sam Kwong, and Jianmin Jiang. Progressive point cloud upsampling via differentiable rendering. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(12):4673–4685, 2021. 1, 2

[45] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5565–5573, 2019. 2

[46] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 2, 4

[47] Wenbo Zhao, Xianming Liu, Zhiwei Zhong, Junjun Jiang, Wei Gao, Ge Li, and Xiangyang Ji. Self-supervised arbitrary-scale point clouds upsampling via implicit neural representation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1989–1997, 2022. 3, 6, 7