

# Progressive Point Cloud Upsampling via Differentiable Rendering

Pingping Zhang, Xu Wang<sup>1</sup>, Member, IEEE, Lin Ma<sup>2</sup>, Member, IEEE, Shiqi Wang<sup>3</sup>, Member, IEEE, Sam Kwong<sup>4</sup>, Fellow, IEEE, and Jianmin Jiang<sup>5</sup>

**Abstract**—In this paper, we propose one novel progressive point cloud upsampling framework to tackle the non-uniform distribution issue during the point cloud upsampling process. Specifically, we design an Up-UNet feature expansion module which is capable of learning the local and global point features via a down-feature operator and an up-feature operator, respectively, to alleviate the non-uniform distribution issue and remove the outliers. Moreover, we design a hybrid loss function considering both the multi-scale reconstruction loss and the rendering loss. The multi-scale reconstruction loss enables each upsampling module to generate a denser point cloud, while the rendering loss via point-based differentiable rendering ensures that the proposed model preserves the point cloud structures. Extensive experimental results demonstrate that our proposed model achieves state-of-the-art performance in terms of both qualitative and quantitative evaluations.

**Index Terms**—Point cloud upsampling, point-based differential rendering, feature expansion unit.

## I. INTRODUCTION

RECENT years have witnessed dramatically increased interest in point cloud based applications. However, due to the hardware and computational constraints, 3D sensors such as LiDAR often produce sparse, noisy and non-uniformly distributed raw point cloud data [1]–[4]. Since the performance of 3D vision tasks is highly influenced by the integrity of the input point cloud data, improvement of the raw data quality via

point cloud upsampling becomes an essential step to enable the downstream applications, such as 3D object classification [5], semantic segmentation [6], and shape reconstruction [7]–[11].

Compared with the traditional 2D image super-resolution task, the point cloud upsampling task is more challenging. First, unlike the 2D image with a regular sampling grid [12], the raw point cloud is irregular, sparse, noisy and non-uniform. Thus, the point cloud upsampling task aims to not only produce a denser version, but also remove noise, protect the structure and thereby generate a dense and uniformly distributed point cloud. Second, the objectives of the point cloud upsampling task are supposed to be application dependent [5], [13]. As a point cloud set is only an intermediate representation of the 3D scene, the generated points should be informative and uniform to assist other applications, such as surface reconstruction and view synthesis.

Numerous efforts [9], [10], [14]–[18] have been devoted to investigating point cloud upsampling techniques to ensure the consistency and integrity of point cloud data. The earlier works focused on reconstructing a piece-wise smooth representation of the original shape, and specialized priors have been incorporated to address the challenges from data imperfections [19]–[21]. However, these priors may not always be appropriate, and in practice, certain shapes may fail to adhere to these priors. More recently, inspired by the success of point-based deep learning [22], deep learning-based point cloud upsampling techniques [9], [10] have attracted growing attention. Many learning-based works attempt to reconstruct uniformly distributed points, which are located close to underlying surfaces. For example, Yu *et al.* [9] proposed a repulsion loss in the PU-Net to make points more uniformly. Li *et al.* [10] proposed a uniform loss to generate point clouds with uniform distribution. However, these models largely ignore the local quality of the reconstructed surface. Since point clouds are unordered, existing loss functions can only measure the global quality of dense point cloud data but cannot preserve the local quality of reconstructed surfaces. Moreover, most models were designed for a fixed upsampling ratio. To upsample with varying scales, they need to train multiple models with pre-defined ratios. Some models need to call numerous times because they break upsampling into multiple small upsampling. For example, a  $4\times$ -upsampling operation needs to be broken into two  $2\times$  steps, so this model needs to be called twice. These methods increase the model complexity, as well as the training and testing time.

Manuscript received December 24, 2020; revised April 13, 2021; accepted July 18, 2021. Date of publication July 26, 2021; date of current version December 6, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61871270 and Grant 61620106008, in part by the Shenzhen Natural Science Foundation under Grant JCYJ20200109110410133 and Grant 20200812110350001, and in part by the National Engineering Laboratory for Big Data System Computing Technology of China. This article was recommended by Associate Editor F. Dufaux. (Corresponding author: Xu Wang.)

Pingping Zhang is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China, and also with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: ppingyes@gmail.com).

Xu Wang and Jianmin Jiang are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China, and also with the Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University, Shenzhen 518060, China (e-mail: wangxu@szu.edu.cn; jianmin.jiang@szu.edu.cn).

Lin Ma is with Meituan, Beijing 518057, China (e-mail: forest.linma@gmail.com).

Shiqi Wang and Sam Kwong are with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: shiqiwan@cityu.edu.hk; cssamk@cityu.edu.hk).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TCSVT.2021.3100134>.

Digital Object Identifier 10.1109/TCSVT.2021.3100134

To resolve the challenges, we propose a progressive point cloud upsampling framework via the point-based differentiable rendering, which tackles the non-uniform distribution issue of the point cloud. Specifically, the proposed upsampling framework consists of a contextual representation (CR) module and a number of cascaded upsampling modules with shared parameters. To support the progressively upsampling, our model shares different resolution information both inside and between upsampling modules, and each of them produces a denser point cloud to achieve multi-scale upsampling. Inspired by the concept of the point-based differentiable rendering [23], a rendering loss is proposed to enforce the network to learn the underlying geometry of a latent target object via measuring the fidelity of synthesized images under different camera poses. Accordingly, the error back propagates through the differentiable renderer, so the network is capable of learning the structure preserved features. We conduct extensive experiments on a sparse, noisy and incomplete dataset to demonstrate the effectiveness of our approach, aiming to generate high quality dense and complete point clouds. As a result, our approach leads to outstanding performance for the final point cloud upsampling results surpassing the state-of-the-art (SOTA) methods. Moreover, ablation studies and experimental analyses demonstrate the intermediate benefits of our method.

In summary, the main contributions of this paper lie in the following three folds.

- 1) We propose a flexible framework that can upsample point cloud progressively to support multi-scale upsampling. The proposed Up-UNet feature expansion unit enables the model to learn global point features, remove the outliers and alleviate the non-uniform issue.
- 2) We design a hybrid loss function considering both multi-scale reconstruction loss and rendering loss. The multi-scale reconstruction loss enables each upsampling unit to output a denser point cloud, while the rendering loss via the point-based differentiable rendering forces the model to learn the structure preserved features.
- 3) Extensive experiments and analyses verify the effectiveness of our approach on different point cloud tasks. The proposed approach improves the quality of reconstructed point cloud and outperforms the SOTA methods.

## II. RELATED WORK

### A. Point Cloud Processing

There are three typical point cloud processing tasks: point cloud upsampling, point cloud denoising and point cloud completion. From the aspect of optimization-based methods [13], [14], [24], [25], most of them consider various shape priors to constrain the geometry reconstruction. Alexa *et al.* [24] provided a framework to approximate a smooth manifold surface defined by a set of points and resampled the surface to generate an adequate representation of the surface. Lipman *et al.* [25] introduced a Locally Optimal Projection for surface approximation from point cloud data. Huang *et al.* [13] modified and extended the LOP operator to produce a clean and uniformly distributed point set endowed with reliable normals. These traditional point cloud processing

methods mostly consider surface reconstruction. Due to the excellent performance of deep learning, these traditional tasks have a better outcome under this method. To generate uniform dense point clouds, Yu *et al.* [26] presented EC-Net, which is the first edge-aware network for consolidating point clouds. PU-Net [9], PU-GAN [10] and MPU [16] can remove outliers and generate dense point clouds simultaneously. PUGeNet [11] can compute normals for the original and generated point clouds to improve the quality of the surface reconstruction. To improve the surface approximation via point set upsampling, Lin *et al.* [27] proposed CAD-PU to realize the curvature-adaptive feature expansion.

Most of the upsampling models cannot process point clouds with large holes [26], since the point cloud structure was destroyed and these models cannot extract the critical structure effectively. In this context, point cloud upsampling tasks share the same aim as point cloud completion tasks, as they need to achieve point cloud inpainting to guarantee the completion of the whole structure and obtain a uniform and dense point cloud [28]. As such, point cloud completion is also a popular topic in 3D reconstruction. PF-Net [29] generated the target point cloud with both rich semantic profiles and detailed characters while retaining the existing contour. These models can achieve point cloud inpainting to guarantee the completion of the whole structure. Nevertheless, it cannot indicate that their models can remove the outliers and generate a dense point cloud uniformly. PCN [30] generated a dense point cloud in a coarse-to-fine fashion on raw point clouds without voxelization. ECG [31] can complete the point cloud by two stages, which are the coarse and fine stages. Likewise, Wang *et al.* proposed a cascaded refinement network together with a coarse-to-fine strategy to synthesize the detailed object shapes [32]. Their model considered the local details of partial inputs with the global shape information together. Analogously, our model can achieve the point cloud reconstruction progressively via multiscale fine-tuning. Benefited from the special model design and the hybrid loss, our proposed model can complete various point cloud tasks, *e.g.*, upsampling, denoising and completion.

### B. Point-Based Functional Module

Existing point cloud upsampling models [9], [10], [16] design point-based functional modules by employing the common architectures such as ResNet [33], DenseNet [34] and UNet [35] as backbones. Owing to the limitation of model structure, these models can only process noisy point clouds and conduct dense point clouds under the same structure, but they cannot guarantee the good quality of the point cloud upsampling task with serious noise. For example, PU-Net [9] and PU-GAN [10] can process the point cloud with slight noise. Our proposed Up-UNet feature expansion module consists of an up-feature operator and a down-feature operator, which combine the benefits of existing point-based functional modules to process the noisy point clouds well.

A popular and simple up-feature is to duplicate the feature multiple times. However, this strategy could not protect the point structure information, such that PU-GAN and

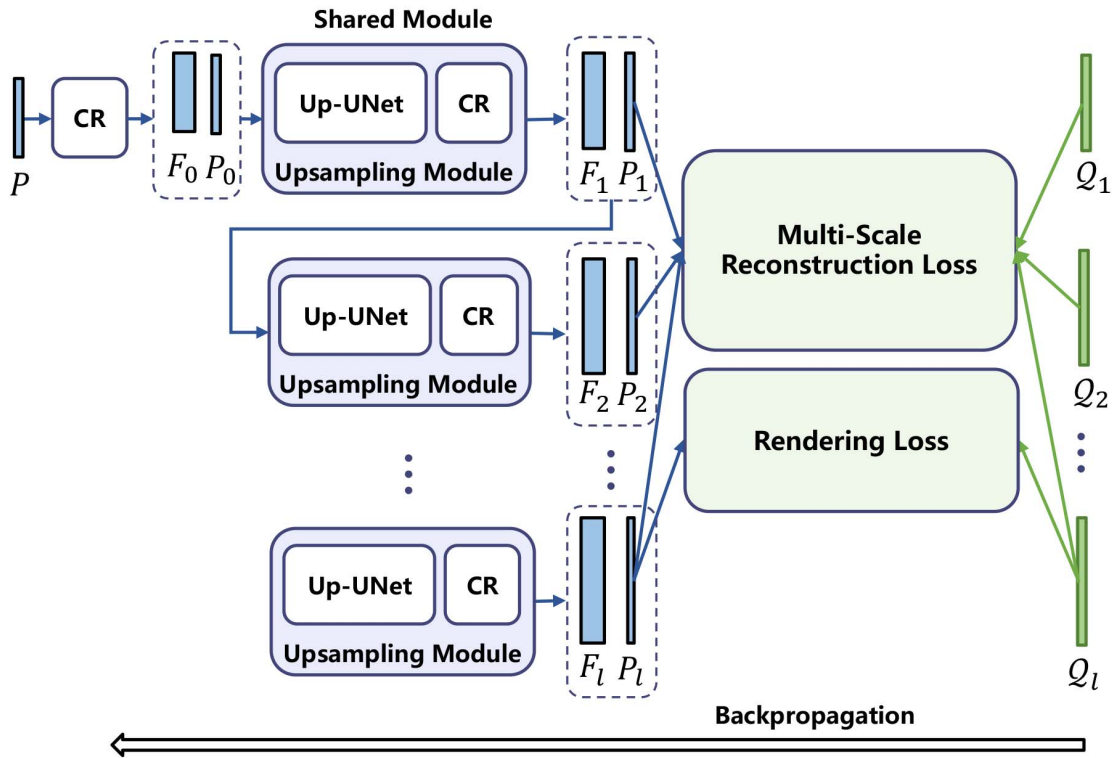


Fig. 1. The architecture of our proposed model. The shared upsampling module consisting of an Up-UNet and a CR module is cycled multiple times. The loss function mainly is composed of the multi-scale reconstruction loss and the rendering loss. Due to the point-based differentiable rendering, the error can be backpropagated.

MPU introduce the grid information to guide point cloud upsampling. The down-sampling layer, the inverse process of upsampling, can extract key points and corresponding features. The traditional methods, such as farthest point sampling and Poisson disk sampling, have been widely used for decades. Inspired by the success of PointNet [36] on classification, many key point extraction methods were studied in recent years. The PointNet++ [22] captures both local geometry contexts via a hierarchical feature learning architecture. These models only deal with a single function and cannot effectively combine. Our proposed Up-UNet, based on the UNet, learns the global and local features via an up-feature operator and a down-feature operator, respectively, to effectively alleviate the non-uniform distribution issue and remove the outliers.

### C. Differentiable Rendering

Renderers are traditionally designed to solve the forward process of image synthesis. Recently, deep learning has been popular in 3D reconstruction tasks [23], [37]–[41]. Therefore, various differentiable rendering techniques were introduced to generate rendering images, which also can help 3D model reconstruction via error backpropagation. Existing differentiable renderers can be classified into four categories according to the geometric representation: point-based [23], [40], [42], [43], voxel-based [44]–[46], mesh-based [37], [47], [48] and implicit neural function based [49], [50]. Voxel-based methods come with high memory requirements even for relatively coarse geometries. Mesh-based methods exploit

the sparseness of the underlying geometry in the 3D space. However, converting into a mesh form is a challenging and error-prone operation. These methods are limited to global and topological changes, and connectivity is not differentiable. More recently, implicit neural functions are popularly applied to represent scenes, as they can achieve a high spatial resolution. However, existing approaches are limited by the low network capacity and inaccurate intersections of camera rays with the scene geometry. Point-based methods directly operate on point samples of the geometry, which is flexible and efficient. In this context, a fast and effective point-based differentiable renderer [23] is adopted to capture rendering images from various camera poses, which contributes to the local geometry reconstruction.

## III. PROPOSED MODEL

Suppose the captured raw point cloud is denoted as  $\mathcal{P} = \{\mathbf{p}_i \in \mathbb{R}^d\}_{i=1}^N$  of  $N$  points, where  $d$  is the dimension of the input point cloud attributes, *i.e.*, coordinates, color and normal. Here, we only consider the 3D coordinates with  $d = 3$ . The point cloud upsampling network aims to generate a denser point cloud  $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_i \in \mathbb{R}^d\}_{i=1}^{rN}$  with upsampling ratio  $r$  from  $\mathcal{P}$ , by minimizing the reconstruction distance between  $\hat{\mathcal{P}}$  and the ground truth (GT) points  $\mathcal{Q} = \{\mathbf{q}_i \in \mathbb{R}^d\}_{i=1}^{rN}$ . As shown in Fig. 1, the pipeline of our proposed model consists of a CR module and  $L$  cascaded upsampling module with shared parameters. The CR module first transforms the raw point cloud into the feature domain and outputs the aggregated point-wise

feature map  $\mathcal{F}_0$  and initially reconstructed point cloud  $\mathcal{P}_0$ . The following upsampling module will progressively process the point cloud  $\mathcal{P}_{l-1}$  with the upsampling ratio  $r_l$  for the  $l$ -th module and output the reconstructed point cloud  $\mathcal{P}_l$ . When  $L$  is set to 1 and the first up-feature operator is withdrawn, it becomes a point cloud completion network. Details of our proposed CR module and Up-UNet feature expansion module are discussed as follows.

### A. Contextual Representation Module

The proposed CR module aims to extract point features from the noisy input and output the reconstructed points for the next upsampling iteration. Specifically, our proposed CR module consists of feature extraction and coordinate reconstruction operators. We adopt the feature extraction module in [16], which can integrate features across different layers through dense connections. In each dense block, it generates a local neighbourhood which is computed dynamically via feature-based K-Nearest Neighbor (KNN). Then, a chain of densely connected Multilayer Perceptron (MLP) layers refine grouped features, and finally, a max-pooling operator is applied to achieve order-invariance. Accordingly, the coordinate reconstruction operator is capable of extracting long-range and non-local information. The CR module is embedded into the primary stage and the end of each Up-UNet upsampling module. The purpose of the first CR module aims to extract the contextual features  $F_0$ , and initially reconstructed points  $P_0$ .

### B. Upsampling Module

As shown in Fig. 1, our proposed upsampling module consists of an Up-UNet feature expansion unit and a CR module. Unlike the official UNet architecture [35], our proposed module firstly upsamples point features via an up-feature operator, which not only extracts the local point features but also adjusts the features according to the neighbouring features via a channel attention operator. Besides, such upsampling module can maintain the order of the input point cloud since the number of features is doubled by duplication. Then, to keep the consistency of the guided point cloud, we split the first  $N$  point features from the upsampled features. The first down-feature operator only conducts the sampling operation without changing the number of points, which extracts neighbouring information and builds the relation of closing points. The second down-feature operator performs the real down-sampling to extract key points and important point features. Subsequently, the continuous upsampling operations, together with the expansive paths, allow the network to propagate context information to reconstruct a dense point cloud.

For progressive upsampling, a CR module is stacked after an Up-UNet feature expansion unit for the coordinate reconstruction of the upsampled point cloud. Since all the Up-UNet and CR units share the same parameters, our proposed upsampling module can handle point clouds with various upsampling ratios. Details of the up-feature operator and the down-feature operator are discussed as follows.

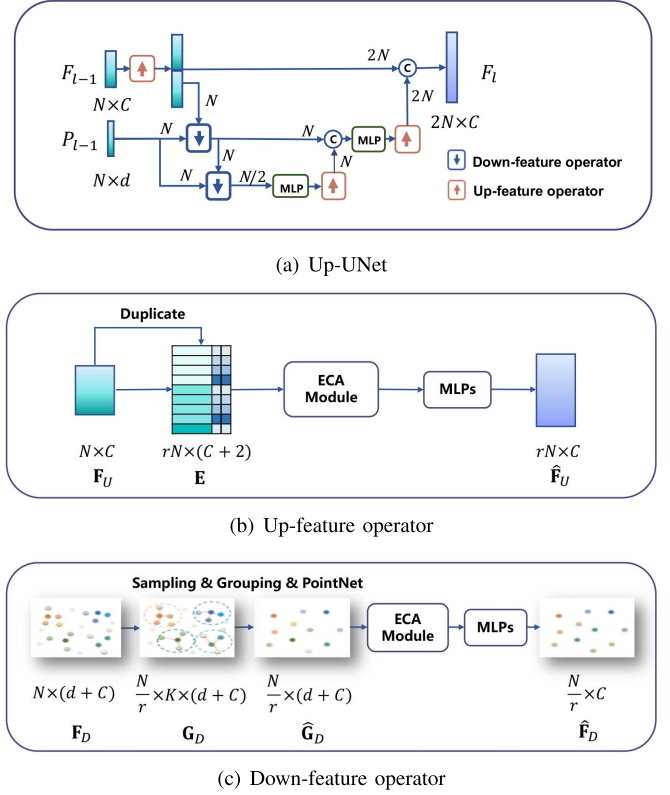


Fig. 2. Figure (a) is the case of  $2\times$  upsampling. Operators in our proposed Up-UNet feature expansion modules: (a) Up-UNet; (b) Up-feature operator; (c) Down-feature operator.

1) *Up-Feature Operator*: The up-sampling operator is indispensable in the point cloud upsampling task. A common and simple way is to duplicate the feature multiple times. However, this way cannot protect the point structure information. Therefore, to expand the input point feature  $\mathbf{F}_U \in \mathbb{R}^{N \times C}$  for  $r$  times, our proposed up-feature operator unfolds the point features based on the 2D grid. As shown in Fig. 2, the 2D grid mechanism in FoldingNet [51] is adopted to generate a unique 2D vector via per feature-map copy. Specifically, the grid position matrix is denoted as  $\mathbf{G}_U \in \mathbb{R}^{r \times 2}$ . Meanwhile, the expanded feature  $\mathbf{E} \in \mathbb{R}^{rN \times (C+2)}$  is generated by duplicating and concatenating input point features and a grid position vector. The  $j$ -th node of expanded feature  $\mathbf{E}$  is defined as  $\mathbf{E}_j = (\mathbf{G}_{j \% r}, F_{j/r})$ . Based on the folding method, the up-feature operator can increase the variation of point features in the distribution density.

To improve both local channel interaction of point features, we adopt the ECA unit [52] to readjust features via feature aggregation of the nearest neighbouring features. Compared with other attention modules, such as the spatial attention module [53], [54] and the convolutional block attention module [55], the ECA unit can appropriate cross-channel interaction to preserve features and benefits the point feature reconstruction. At the end of the module, a set of MLPs is applied to refine and produce more consistent geometric details for the final expanded point features.

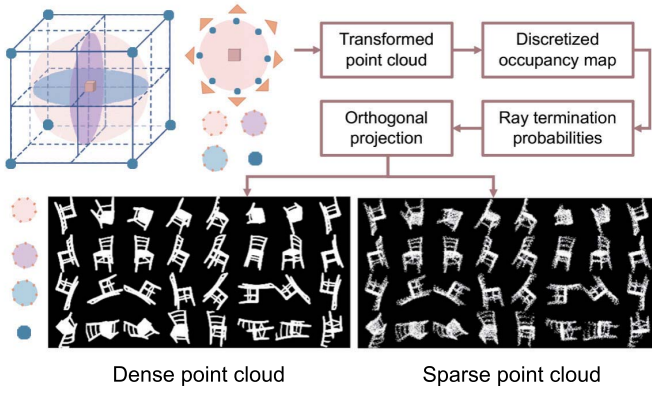


Fig. 3. The first row shows the pipeline of the differentiable point cloud renderer [23]. Points in the cube represent the camera poses. The second row is the rendering results of dense point clouds and its sparse version with 32 camera poses.

2) *Down-Feature Operator*: The down-feature operator, the inverse process of upsampling, can extract key points and corresponding features. The traditional methods, such as farthest point sampling and Poisson disk sampling, are widely used for decades. Inspired by the success of PointNet++ [22], it can capture local geometry contexts via a hierarchical feature learning architecture. Thus, our down-feature operator integrates a similar downsampling policy to extract important point features.

Accordingly, our proposed down-feature operator aims to hierarchically group input point features and progressively abstracts key points features. To extract both local and global salient point features from noisy input point features, we apply the local learning mechanism on the input point features  $\mathbf{F}_D$  with the guidance of the input point cloud  $\mathcal{P}_D$ . Specifically, the proposed local feature extractor comprises three key layers: a sampling layer, a grouping layer and a PointNet layer. The sampling layer initially selects the centroids of local regions following iterative farthest point sampling (FPS) to choose a subset with  $\frac{N}{T}$  points and the corresponding point features. Then, the grouping layer constructs a group of local point features  $\mathbf{G}_D \in \mathbb{R}^{\frac{N}{T} \times K \times (d+C)}$  by finding  $K$  neighbouring points around the centroids. Finally, each local region is abstracted by its centroid and local features via a mini-PointNet, which encodes local region patterns  $\mathbf{G}_D$  into feature vectors  $\hat{\mathbf{G}}_D \in \mathbb{R}^{\frac{N}{T} \times (d+C)}$ . The module can capture point-to-point relations in the local region by using relative coordinates together with point features. Same as the up-feature operator, an ECA module and a set of MLPs are applied to obtain point features  $\hat{\mathbf{F}}_D \in \mathbb{R}^{\frac{N}{T} \times C}$ .

### C. Loss Function

We propose a novel hybrid loss function to train the network for point cloud upsampling. This hybrid loss contains a reconstruction loss and a rendering loss, which can encourage the model to reconstruct detailed point clouds.

1) *Multi-Scale Reconstruction Loss*: The multi-scale reconstruction loss measures the difference between the output points and the GT points with different resolutions. Point clouds are disordered, such that we employ the Chamfer

Distance (CD) [56] as the reconstruction loss, which is invariant to permutations of the points. Specifically, we use the following symmetric CD function, which calculates the average closest distance between the output point cloud  $\mathcal{P}_l$  of the  $l$ -th upsampling unit and the corresponding GT points  $\mathcal{Q}_l$ , the downsampled version of the GT points  $\mathcal{Q}$ . In this context, our model can produce upsampling point clouds with multiple ratios and ensuring the quality of the upsampled point cloud.

$$\mathcal{L}_{CD}(\mathcal{P}_l, \mathcal{Q}_l) = \frac{1}{|\mathcal{P}_l|} \sum_{\mathbf{p} \in \mathcal{P}_l} \min_{\mathbf{q} \in \mathcal{Q}_l} \|\mathbf{p} - \mathbf{q}\|_2 + \frac{1}{|\mathcal{Q}_l|} \sum_{\mathbf{q} \in \mathcal{Q}_l} \min_{\mathbf{p} \in \mathcal{P}_l} \|\mathbf{q} - \mathbf{p}\|_2. \quad (1)$$

The first term in Eq. (1) focuses on minimizing the distance of output points and the closest GT point, whereas the second term ensures that the GT point cloud is covered by the output point cloud. Finally, we measure the overall reconstruction loss  $\mathcal{L}_R$  by merging the reconstruction loss of all the upsampling units,

$$\mathcal{L}_R = \sum_{l=1}^L \lambda_l \mathcal{L}_{CD}(\mathcal{P}_l, \mathcal{Q}_l), \quad (2)$$

where  $\lambda_l$  represents the weight of the  $l$ -th upsampling unit.

2) *Rendering Loss*: Inspired by the model from Insafutdinov *et al.* [23], we introduce the rendering loss to learn high-fidelity shape models solely from their projections. Instead of learning camera poses, we fix the multiple cameras poses to estimate the rendering views. In this way, our model can reconstruct point clouds by preserving the details.

To ensure the fidelity of the synthesized images rendered from the reconstructed point cloud, we employ the point-based differentiable renderer  $\pi$  as in [23], which projects 3D point cloud data into 2D view images according to the camera pose settings. The pipeline of the point cloud renderer is illustrated in Fig. 3. First, it transforms the 3D coordinate of the raw point cloud into the standard coordinate frame by the projective transformation corresponding to the camera pose. Second, to guarantee the back-propagation of a gradient in the training stage, the discretized point is represented as scaled Gaussian densities to obtain the occupancy map. By introducing the differentiable ray tracing operator, the occupancies are converted into ray termination probabilities. Final, the projected image is obtained by projecting the volume to the plane.

Specifically, given the  $s$ -th pose  $c_s$ , we obtain raw projected view images  $\mathbf{I}_s = \pi(\mathcal{Q}, c_s)$  and reconstructed projected view images  $\hat{\mathbf{I}}_s = \pi(\hat{\mathcal{P}}, c_s)$  from the groundtruth  $\mathcal{Q}$  and final output  $\hat{\mathcal{P}}$ , respectively. The rendering loss  $\mathcal{L}_v$  is defined as the mean absolute difference of  $\hat{\mathbf{I}}_s$  and  $\mathbf{I}_s$  for all the camera poses. That is,

$$\mathcal{L}_v = \frac{1}{SWH} \sum_{s=1}^S \sum_{x=1}^W \sum_{y=1}^H |\mathbf{I}_s(x, y) - \hat{\mathbf{I}}_s(x, y)|, \quad (3)$$

where  $S$  is the total number of camera poses. In this work, we take 8 camera poses evenly on the projection plane of each rotation axis  $(x, y, z)$ , and there are 8 camera positions

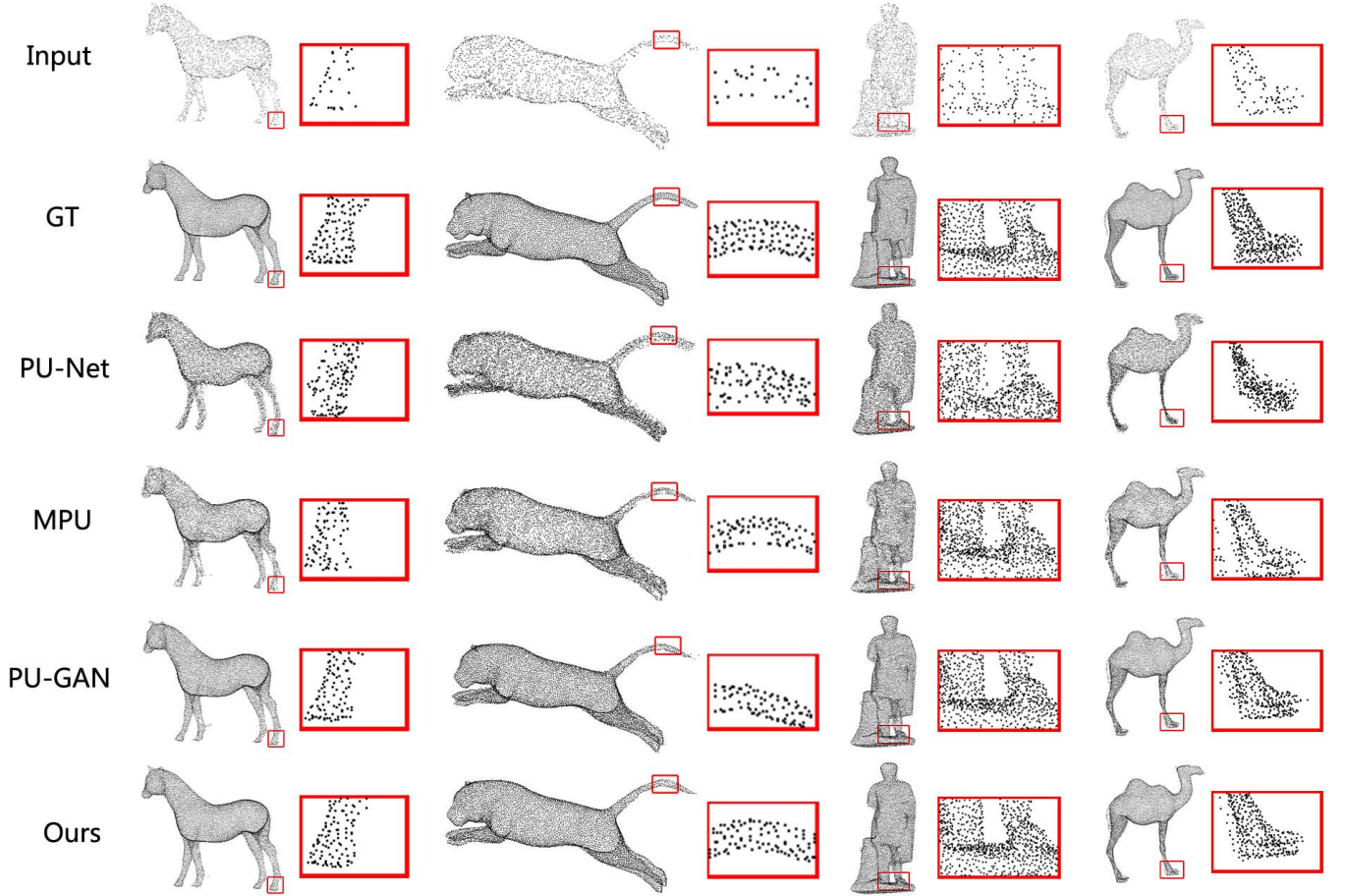


Fig. 4. Visual comparisons of upsampled point clouds ( $\times 4$ ) by PU-Net [9], MPU [16], PU-GAN [10] and ours.

in each diagonal position with a total of 32 camera poses, as shown in Fig. 3. Specifically, three colour planes in Fig. 3 represent different projection planes. The 8 points in the colour planes denote 8 camera positions of rendering images in each row. Compared with the rendering images of the sparse point clouds, the rendering images from the dense point clouds are more clear and have high resolution. The rendering loss combined with multiple local rendering images will enforce the network to reconstruct local features of point clouds.

3) *Hybrid Loss*: Overall, we train our model by minimizing  $\mathcal{L}$ :

$$\mathcal{L} = \mathcal{L}_R + \alpha \mathcal{L}_v + \beta \|\theta\|^2, \quad (4)$$

where  $\alpha$  is the weight of  $\mathcal{L}_v$ ,  $\theta$  indicates the parameters in our network and  $\beta$  denotes the multiplier of the weight decay.

#### IV. EXPERIMENTAL RESULTS

In this section, we conduct extensive experiments to evaluate the performance of our proposed model. In the following subsections, we first describe the experimental settings such as datasets, evaluation metrics and implementation details. Details on the performance comparison between the proposed model and the SOTA models are described as follows.

##### A. Experimental Settings

1) *Datasets*: To objectively compare the performance of the model, we train and test our model under the dataset provided

by PU-GAN [10], which collected 147 3D models from the released datasets of PU-Net [9] and MPU [16], as well as from the Visionair repository [57]. For each point cloud, we randomly crop 200 patches and collect 24,000 patches in total. By default, we set the input number  $N$  as 256, the upsampling ratio  $r$  as 4. Moreover, we add different degrees  $\theta \in \{0.5\%, 1\%, 1.5\%, 2.5\%\}$  of Gaussian noise to evaluate the model's ability of noisy removal. For the point completion task, we also train and test our model from a subset of the Shapenet dataset [58] derived from the dataset in Yuan *et al.* [30]. In our experiments, both input and GT point clouds are sampled 2048 points uniformly. To avoid overfitting in training, we augment the network inputs by random rotation and scaling.

2) *Implementation Details*: We trained the network with 100 epochs via the Adam algorithm [59]. We set the learning rates as 0.001, and it gradually reduces with the increase of iterations. The batch size is 28, and  $\{\lambda_1, \lambda_2\} = \{1.0, 0.3\}$  and both  $\alpha$  and  $\beta$  are empirically set as 1.0. Different from MPU, our training only needs one stage. We implemented our network using TensorFlow and trained it on the Tesla V100 GPU.

Our model can adapt to multi-resolution upsampling. Take the  $4\times$  model as an example that PU-GAN only achieves the  $N \times 4$  upsampling. If PU-GAN requires to realize  $2\times$  upsampling, it needs to samples the results of  $4\times$  upsampling.

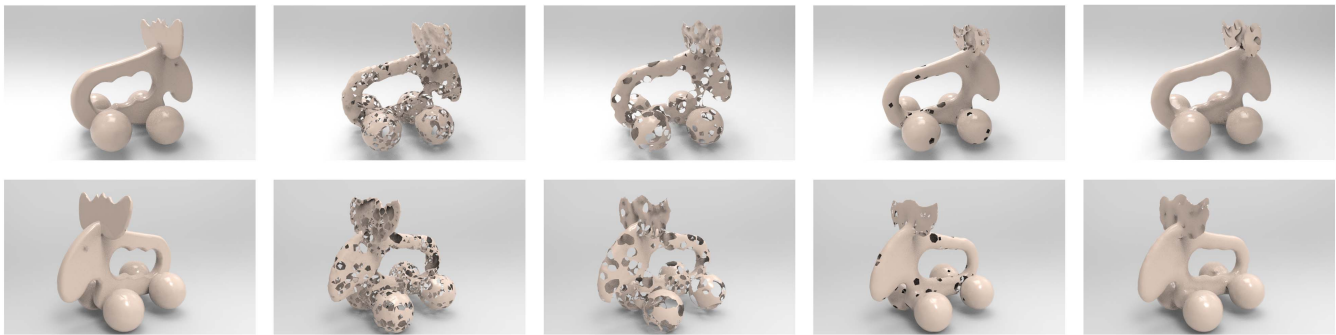


Fig. 5. Examples of synthesized images rendered by the point cloud renderer [23] from the “Elk” model, where each row corresponds to a fixed camera pose. From left to right: they are the rendered images for the GT points, upsampled points of methods PU-Net [9], MPU [16], PU-GAN [10] and Ours, respectively.

Unlike PU-GAN, our model can directly obtain the output of the first Up-UNet module as the result of  $2\times$  upsampling, and the second Up-UNet module, as a result of  $4\times$  upsampling. If we want to achieve a higher ratio upsampling, we need to carry on the cycle operation on the basis of  $4\times$  upsampling because the transformation of  $4\times$  feature space is considered in the model training. If our model upsamples points with  $16\times$  or higher times, we can get the middle  $2\times$ ,  $4\times$ ,  $8\times$  results directly. For other upsampling ratios, we can sample the given number of points from the upsampled points.

3) *Evaluation Metrics*: For quantitative evaluation, we employ the point-to-surface (P2F) distance, CD, and Hausdorff distance (HD) [60] to evaluate our proposed model against the SOTA methods on point cloud upsampling and denoising. Each ground truth 3D model contains 8192 points, and then we randomly select 2048 points as the testing input. Similarly, we follow the patch-based strategies in PU-GAN, MPU, EC-Net and PU-GAN to extract a local patch with 256 points per seed. Then, after processing, these sub-patches are combined as the final output. For the point completion task, we evaluate our model across 8 classes from the Shapenet dataset [58]. For each class, the CD is employed to evaluate the reconstructed models.

### B. Comparisons With the State-of-the-Art Methods

In this subsection, our proposed method is compared with the SOTA methods on different tasks, including point cloud upsampling, denoising and completion.

1) *Point Cloud Upsampling*: We qualitatively and quantitatively compare our proposed method with three SOTA point cloud upsampling methods, including PU-Net [9], MPU [16], and PU-GAN [10]. We use their public code and retrain their networks on our training dataset. Table I shows the quantitative comparison results, which are the average results in the testing data. Our proposed method achieves the lowest values for all the evaluation metrics. Besides quantitative results, some examples of upsampling results for all the methods are provided in Fig. 4 for four different 3D models (“Horse”, “Tiger”, “Status” and “Camel”). The first row (Input) shows the input point clouds, and the second row (GT) is the corresponding dense points. We amplify the local point cloud and find that our proposed method has a good performance to deal with complex areas and produces more fine-grained details.

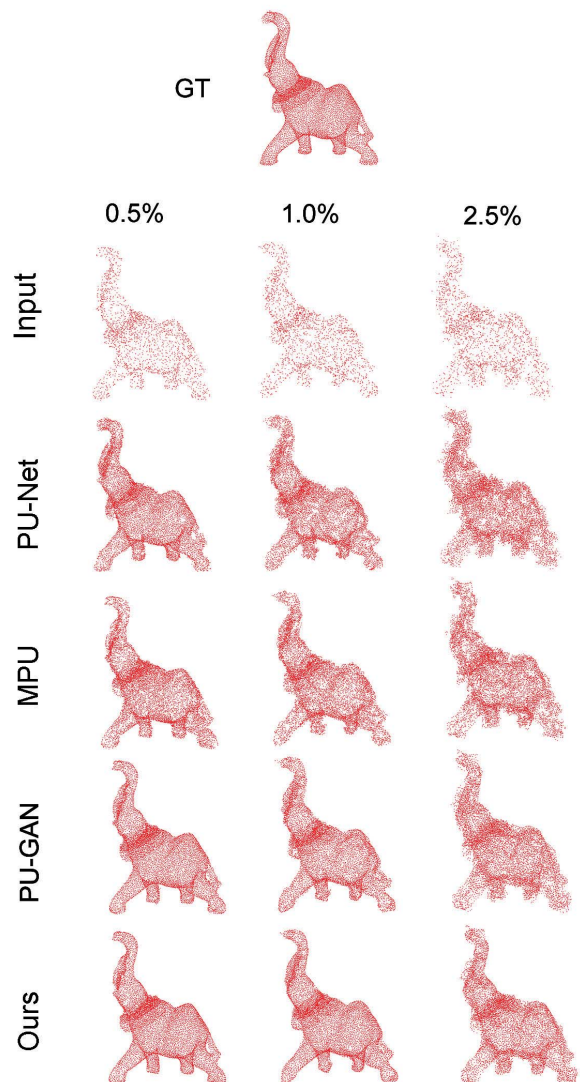


Fig. 6. The comparison of results under different noisy inputs (“Tiger”). From top to bottom: they are the GT points, input points, upsampled points of PU-Net [9], MPU [16], PU-GAN [10] and Ours. From left to right: they are the results of different models according to inputs with noisy levels of 0.5%, 1.0% and 2.5%.

To analyze the influence of the reconstructed point cloud on the synthesized images, we further provide examples of synthesized images for the upsampled 3D point clouds in Fig. 5. It is observed that there are many holes in the synthesized

TABLE I

QUANTITATIVE COMPARISON WITH THE SOTA METHODS FOR  $4\times$ -UPSAMPLING RESULTS ON THE POINT CLOUD UPSAMPLING TASK WITHOUT NOISE AND WITH 0.5%, 1.0%, 1.5%, AND 2.5% NOISE. (BOLD DENOTES THE BEST PERFORMANCE)

Methods	level=0			level=0.5%			level=1.0%			level=1.5%			level=2.5%			#Network size	Runtime
	P2F	CD	HD	P2F	CD	HD	P2F	CD	HD	P2F	CD	HD	P2F	CD	HD		
	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$	$10^{-3}$		
PU-Net	8.230	0.584	6.656	5.513	0.455	5.373	9.164	0.620	7.970	13.205	0.986	10.830	21.375	1.737	21.218	10.10	<b>0.178</b>
MPU	4.288	0.480	6.033	5.233	0.613	7.060	7.958	0.695	8.329	11.371	0.922	11.240	18.676	1.420	16.555	92.50	1.807
PU-GAN	2.708	0.262	4.178	3.753	<b>0.311</b>	5.454	7.244	<b>0.439</b>	7.656	11.608	0.751	13.174	20.392	1.526	21.745	<b>9.57</b>	0.484
Ours	<b>2.533</b>	<b>0.259</b>	<b>3.911</b>	<b>3.613</b>	<b>0.311</b>	<b>4.955</b>	<b>4.219</b>	0.561	<b>5.481</b>	<b>8.465</b>	<b>0.574</b>	<b>8.638</b>	<b>16.568</b>	<b>1.134</b>	<b>16.056</b>	34.31	0.921

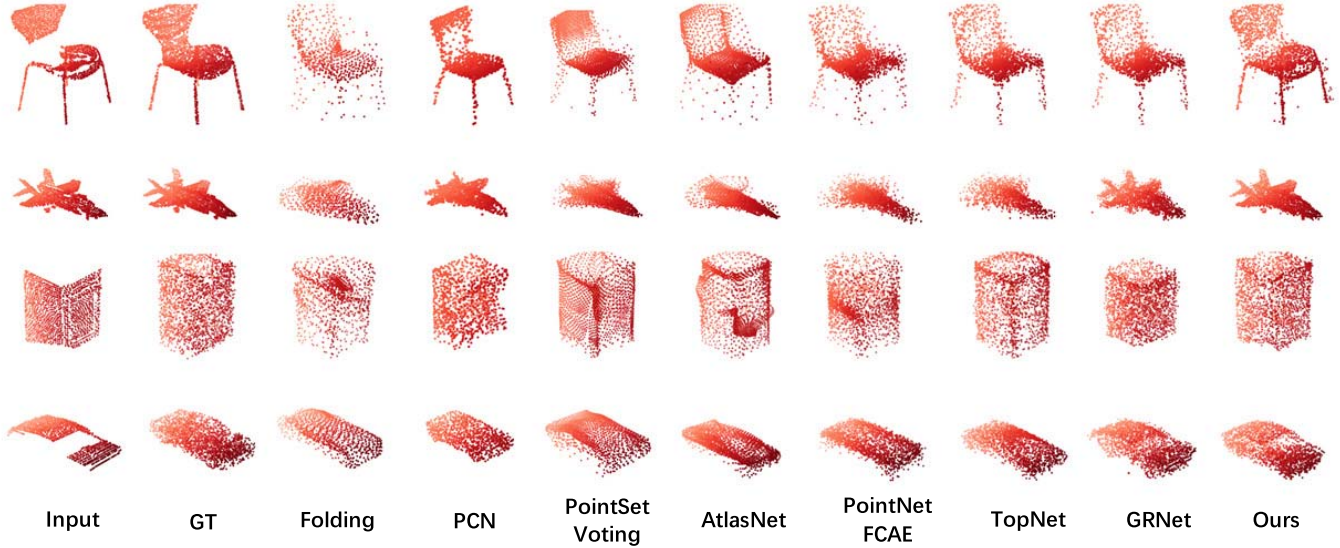


Fig. 7. Visual comparison of the results of Folding [51], PCN [30], PointSetVoting [61], AtlasNet [62], PointNetFCAE [base.], TopNet [63], GRNet [64] and our proposed method on the point cloud completion task. From top to bottom, there are the results of “Chair”, “Airplane”, “Cabinet” and “Car” models.

images for methods PU-Net, MPU and PU-GAN, which do not consider the surface reconstruction in their architecture design. The above models cannot deal with the complex 3D model with occluded objects or surfaces since the input sparse point clouds do not have complete structures. Our proposed method can learn the structure prior from the training dataset via the constraint of rendering loss.

2) *Point Cloud Denoising*: To demonstrate the robustness of our proposed model on noise tolerance, we evaluate all pre-trained models on a synthetic dataset with different noisy levels, including 0.5%, 1.0%, 1.5%, and 2.5%, respectively. All the methods are implemented to  $4\times$ -upsampling the noisy point clouds. Table I shows the comparison results of point cloud upsampling with different noisy levels. It is observed that the performance of all the methods decreases as the noise level increases. As illustrated in Fig. 6, for point clouds with noise level 0.5%, most models can get rid of noise and keep a good shape. However, for point clouds with noise level 2.5%, the results of PU-Net, MPU and PU-GAN contain apparent outliers, *e.g.*, camel’s legs. Thus, these methods can only process point clouds with a slight noise level and not process serious noise. Compared with the SOTA methods, our proposed model is relatively robust to the noise level and can generate uniform point clouds, *e.g.*, cow’s body.

3) *Point Cloud Completion*: For performance evaluation on point cloud completion task, we only use one Up-UNet feature expansion module and remove the first up-feature operator in our proposed method. To protect the completion of point cloud structure, the raw incomplete point clouds is fed into the network as a whole. The training data for this task has 2048 points per 3D scene. For fast training, the number of camera poses in the point cloud render is reduced from 32 to 6. Both EMD and CD as the reconstruction loss better result than most existing models, but EMD is better than the CD from our experiment. Therefore, we employ EMD instead of CD as the reconstruction loss in the point cloud completion task. In the testing stage, we evaluate our model across 8 category from the Shapenet dataset, including “Airplane”, “Cabinet”, “Car”, “Chair”, “Lamp”, “Sofa”, “Table” and “Watercraft”. Table II summarizes the comparison results on the leaderboard<sup>1</sup> provided by the official testing platform. Notably, the table shows the average result of each category, not a single 3D model. Our proposed method is better than most SOTA models for point cloud completion, but not over GRNet [64]. As shown in Fig. 7, our proposed method can protect the whole structure, even if the input points have serious distortion, *e.g.*, the car

<sup>1</sup><https://completion3d.stanford.edu/results>



TABLE II

PERFORMANCE COMPARISON WITH THE SOTA APPROACHES FOR POINT CLOUD COMPLETION IN THE CD. (BOLD AND UNDERLINING DENOTE THE BEST AND SECOND PERFORMANCE, RESPECTIVELY)

Methods	Airplane	Cabinet	Car	Chair	Lamp	Sofa	Table	Watercraft	CD ( $10^{-4}$ )
Folding	12.83	23.01	14.88	25.69	21.79	21.31	20.71	11.51	19.07
PCN	9.79	22.70	12.43	25.14	22.72	20.26	20.27	11.73	18.22
PointSetVoting	6.88	21.18	15.78	22.54	18.78	28.39	19.96	11.16	18.18
AtlasNet	10.36	23.40	13.40	24.16	20.24	20.82	17.52	11.62	17.77
PointNetFCAE	10.30	19.06	11.82	24.68	20.30	20.09	17.57	10.50	16.88
TopNet	7.32	<u>18.77</u>	12.88	19.82	14.60	<u>16.29</u>	<u>14.89</u>	8.82	14.25
GRNet	<u>6.13</u>	<b>16.90</b>	<b>8.27</b>	<b>12.23</b>	<b>10.22</b>	<b>14.93</b>	<b>10.08</b>	<b>5.86</b>	<b>10.64</b>
Ours	<b>4.45</b>	20.50	<u>9.99</u>	<u>17.24</u>	<u>13.38</u>	20.70	15.27	<u>6.52</u>	<u>13.60</u>

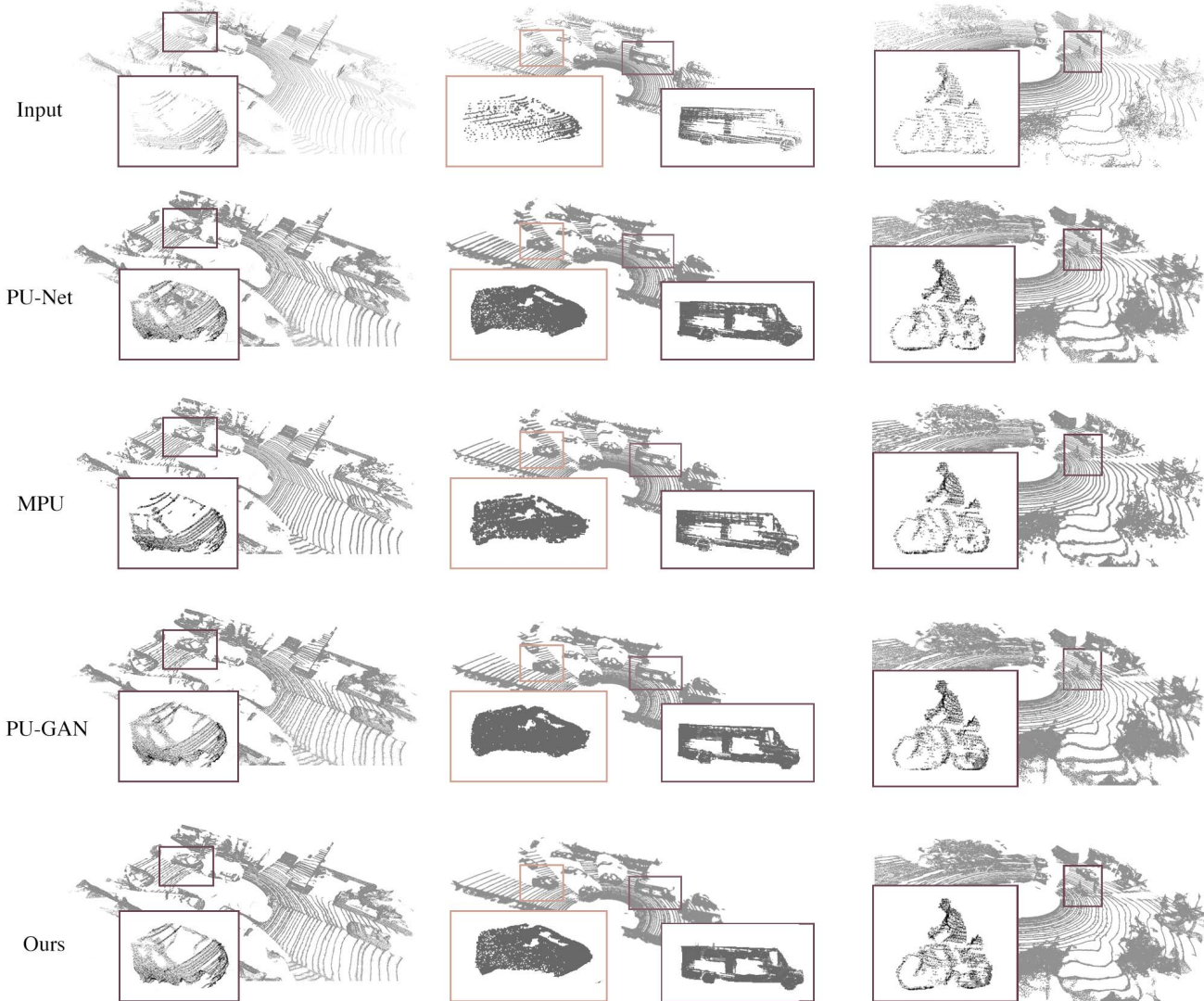


Fig. 8. Upsampling results of our proposed method on real-scanned LiDAR point clouds. We magnify some cases (*e.g.*, the vehicles, the pedestrian, and the cyclist), and our model can make point clouds denser and increase more geometrical details.

in the last row. Although our model would generate outliers in the “Chair” case, like the legs of the chair, it has a clear skeleton. The “Chair” model completed by PCN has a clear structure, but this model cannot keep the original shape. For the “Cabinet” model, even though our model has low CD values, it can get a similar structure of the GT points and good visualization. For the “Airplane” model, our model can

gain more detailed information, such as the wing and head of the plane.

C. Upsampling Real-Scanned Data

Our proposed method is also evaluated on real-world scanned data downloaded from KITTI dataset [1], which is captured by LiDAR for autonomous driving. As shown

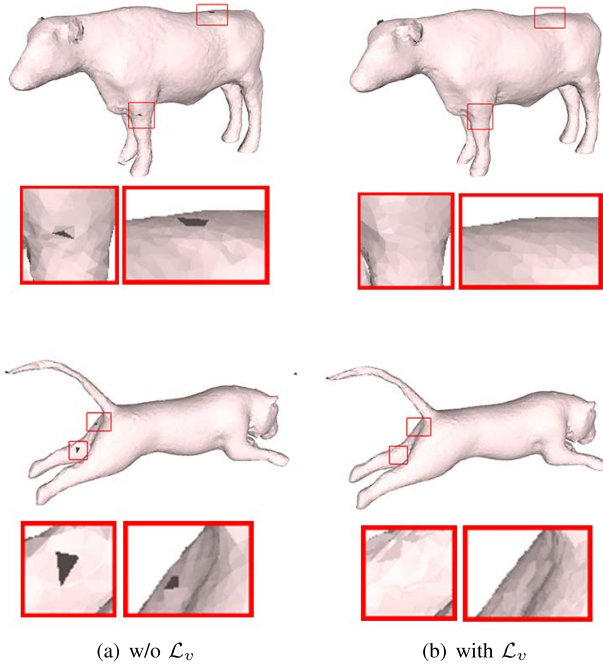


Fig. 9. The surface reconstruction of models trained without and with the rendering loss. (a) the result of the model without  $\mathcal{L}_v$ , and (b) the result of the model with  $\mathcal{L}_v$ .

in Fig. 8, due to the limitation of hardware, the original point cloud suffers from outliers, sparsity and non-uniform distribution issues, *e.g.*, vehicles, pedestrians, and cyclists, are represented with only a few points. Due to a large amount of point cloud data from the KITTI dataset, we divide every point cloud into many patches with 256 points, upsample them 4 times separately, and finally merge them. In this manner, the point cloud becomes denser and increases more geometrical details. The dense point clouds with a good geometrical structure may benefit the other tasks, *e.g.*, point cloud segmentation and classification.

#### D. Ablation Study

1) *Up-UNet Feature Expansion Module*: To quantitatively evaluate the contribution of our proposed Up-UNet module, two deformations, including UNet-Up and Up-Down-Up, are employed to replace the Up-UNet module. Specifically, the UNet-Up module only changes the position of the first up-feature operator. The Up-Down-Up module is from PU-GAN, and it replaces the Up-UNet. The results are shown in Table III. For the UNet-Up structure, part of the point information will lose via the first two down-feature layers. Implementing an up-feature operator first can help keep the raw information. For the Up-Down-Up structure, their proposed down-feature operator does not consider the local relationship. Overall, our proposed Up-UNet module has a good performance on neighbouring feature extraction and point cloud reconstruction.

Moreover, an ablation study quantitatively evaluates the contribution of each of our proposed components, *e.g.*, the model **Ours(Up)** using the Up-feature operator instead of Up-UNet, the model **Ours(w/o CR)** without the CR module,

TABLE III

THE QUANTITATIVE COMPARISONS OF UP-UNET FEATURE EXPANSION MODULES AND REMOVING EACH SPECIFIC COMPONENT FROM HYBRID LOSSES

Setting	P2F ( $10^{-3}$ )	CD ( $10^{-3}$ )	HD ( $10^{-3}$ )
Ours(UNet-Up)	2.869	0.267	4.375
Ours(Up-Down-Up)	2.904	0.279	4.029
Ours(Up)	2.947	0.295	4.949
Ours(w/o CR)	4.740	0.421	6.401
Ours(w/o ECA)	3.696	0.342	5.953
#Ours(Up)	18.246	1.299	19.834
#Ours(w/o CR)	17.804	1.236	20.558
#Ours(w/o ECA)	17.424	1.222	18.560
#Ours	16.056	1.134	16.568
Ours (w/o $\mathcal{L}_v$ )	2.552	0.260	4.298
Ours (w/o $\mathcal{L}_R$ )	<b>2.457</b>	<b>0.253</b>	4.128
Ours	2.533	0.259	<b>3.911</b>

and the model **Ours(w/o ECA)** without ECA in the upsampling layer. The model names with the signal of “#” (*e.g.*, **#Ours(Up)**, **#Ours(w/o CR)** and **#Ours(w/o ECA)**) represent that these models work for noisy point clouds. These results are from point clouds with a noise level of 2.5%. Our full pipeline performs well, and removing any component reduces the overall performance, meaning that each component contributes. In particular, removing the CR module significantly increased the difficulty of the task. Quantitative results shown in **Ours(Up)** and **#Ours(Up)** demonstrate that the Up-UNet structure plays an important role in extracting the key points and reduces noise. The model without the ECA module cannot readjust features, which leads to poor performance on the final result.

2) *Hybrid Loss*: To evaluate the contributions of hybrid loss on our proposed model, we remove each component of hybrid loss and evaluate the effect of it. As shown in Table III, if we do not adopt the rendering loss (denoted as **Ours (w/o  $\mathcal{L}_v$ )**) during the training stage, the upsampling result on the evaluation of the HD value decreases from 3.911 to 4.128. Besides quantitative results, we visualize the upsampling and Poisson surface reconstruction results in Fig. 9. The setting **Ours (w/o  $\mathcal{L}_v$ )** tends to produce noisy and non-uniform point sets, thus leads to more holes in the reconstructed surfaces. By contrast, our proposed method can produce more fine-grained details in the upsampled results and smoother surface, *e.g.*, tiger’s leg and cow’s leg.

Besides, we evaluate the setting **Ours (w/o  $\mathcal{L}_R$ )**, which disables the loss calculation of the first upsampling module. There is no denying that the setting **Ours (w/o  $\mathcal{L}_R$ )** has a better result in P2F and CD, but it cannot consider the output of every iteration that introduce more artifacts reflected in Fig. 10. Since each iteration parameters are shared, this training way tries to obtain a better final output and does not consider the middle outputs. Hence, the points from the middle stages may contain some noises. To improve the quality of reconstruction after every iteration, we constrict the outputs of every iteration to have more generalization ability to upsample point clouds with various sizes, as shown in Fig. 10. Comparing with

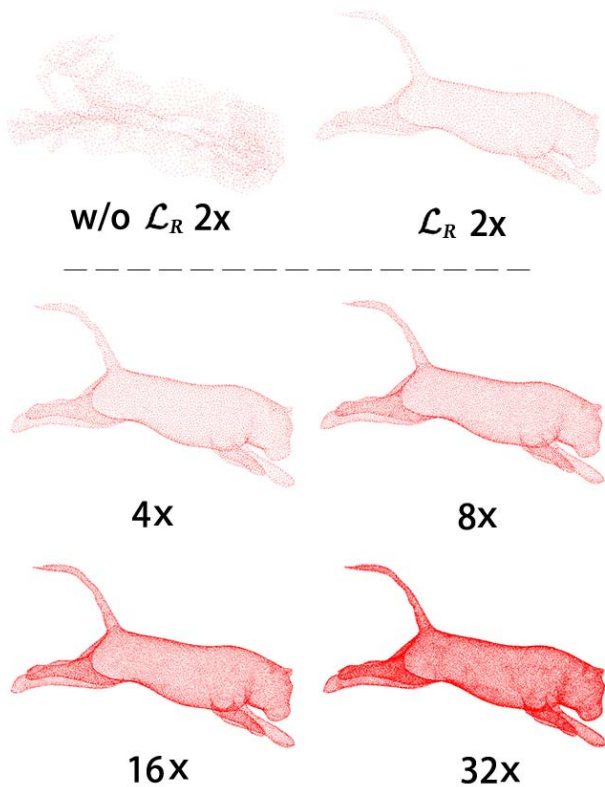


Fig. 10. Visualization of point clouds with different upsampling ratios ( $2\times$ ,  $4\times$ ,  $8\times$ ,  $16\times$  and  $32\times$ ). The first point cloud is generated by the model without the middle reconstruction loss, so compared with the second point cloud, it does not have a clear structure. The rest point clouds are from the same input with different upsampling ratios. It is thus clear that our model can achieve progressive upsampling.

the result from the setting **Ours** (**w/o**  $\mathcal{L}_R$ ), the output point cloud of our proposed method has fewer artifacts. Due to the constraint of outputs of every stage, the model has a chance to correct the mistakes introduced in earlier stages. Besides, progressive point cloud upsampling is necessary to capture more local details because our model would adjust the scope of receptive fields according to the spatial span of the input.

## V. CONCLUSION

In this paper, we have proposed a differentiable rendering based point cloud upsampling framework, which exploits local and global structure for progressive point cloud upsampling. Specifically, the down-feature operator of our proposed Up-UNet feature expansion module can extract key point features by removing the outliers, while the up-feature operator can expand the point features uniformly via the grid-based folding approach. With the constraint of the hybrid loss function, our proposed method can improve reconstructed point cloud data quality. Finally, we demonstrated the effectiveness of our proposed model on different tasks via extensive experiments.

## REFERENCES

- [1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.
- [2] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 567–576.
- [3] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5828–5839.
- [4] R. Mekuria, K. Blom, and P. Cesar, "Design, implementation, and evaluation of a point cloud codec for tele-immersive video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 4, pp. 828–842, Apr. 2017.
- [5] H. Zhou, K. Chen, W. Zhang, H. Fang, W. Zhou, and N. Yu, "DUP-Net: Denoiser and upsampler network for 3D adversarial point clouds defense," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1961–1970.
- [6] X. Wang, J. He, and L. Ma, "Exploiting local and global structure for point cloud semantic segmentation with contextual point representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 4573–4583.
- [7] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 40–49.
- [8] X. Huang, J. Zhang, Q. Wu, L. Fan, and C. Yuan, "A coarse-to-fine algorithm for matching and registration in 3D cross-source point clouds," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 2965–2977, Oct. 2018.
- [9] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-Net: Point cloud upsampling network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2790–2799.
- [10] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-GAN: A point cloud upsampling adversarial network," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7203–7212.
- [11] Y. Qian, J. Hou, S. Kwong, and Y. He, "PUGeo-Net: A geometry-centric network for 3D point cloud upsampling," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 752–769.
- [12] Y. Zhang, X. Tuo, Y. Huang, and J. Yang, "A TV forward-looking super-resolution imaging method based on TSVD strategy for scanning radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 7, pp. 4517–4528, Jul. 2020.
- [13] H. Hui, L. Dan, Z. Hao, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–7, 2009.
- [14] M. Berger *et al.*, "A survey of surface reconstruction from point clouds," *Comput. Graph. Forum*, vol. 36, no. 1, pp. 301–329, 2017.
- [15] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proc. 19th Annu. Conf. Comput. Graph. Interact. Techn.*, 1992, pp. 71–78.
- [16] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3D point set upsampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5958–5967.
- [17] G. Qian, A. Abualshour, G. Li, A. Thabet, and B. Ghanem, "PU-GCN: Point cloud upsampling using graph convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 11683–11692.
- [18] S. Ye, D. Chen, S. Han, Z. Wan, and J. Liao, "Meta-PU: An arbitrary-scale upsampling network for point cloud," *IEEE Trans. Vis. Comput. Graphics*, early access, Feb. 9, 2021, doi: [10.1109/TVCG.2021.3058311](https://doi.org/10.1109/TVCG.2021.3058311).
- [19] R. A. Newcombe *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. 10th IEEE Int. Symp. Mixed Augmented Reality*, Oct. 2011, pp. 127–136.
- [20] A. Tagliasacchi, H. Zhang, and D. Cohen-Or, "Curve skeleton extraction from incomplete point cloud," in *Proc. ACM SIGGRAPH Papers*, 2009, pp. 1–9.
- [21] R. Schnabel, R. Wahl, and R. Klein, "Efficient ransac for point-cloud shape detection," *Comput. Graph. Forum*, vol. 26, no. 2, pp. 214–226, 2007.
- [22] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.
- [23] E. Insafutdinov and A. Dosovitskiy, "Unsupervised learning of shape and pose with differentiable point clouds," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2802–2812.
- [24] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Trans. Vis. Comput. Graphics*, vol. 9, no. 1, pp. 3–15, Jan./Mar. 2003.
- [25] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer, "Parameterization-free projection for geometry reconstruction," *ACM Trans. Graph.*, vol. 26, no. 3, p. 22, Jul. 2007.

- [26] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "EC-Net: An edge-aware point set consolidation network," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 386–402.
- [27] J. Lin, X. Shi, Y. Gao, K. Chen, and K. Jia, "CAD-PU: A curvature-adaptive deep learning solution for point set upsampling," 2020, *arXiv:2009.04660*. [Online]. Available: <http://arxiv.org/abs/2009.04660>
- [28] H. Son and Y. M. Kim, "SAUM: Symmetry-aware upsampling module for consistent point cloud completion," in *Proc. ACCV*, in Lecture Notes in Computer Science, vol. 12622, 2021, pp. 158–174.
- [29] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, "PF-Net: Point fractal network for 3D point cloud completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 7662–7670.
- [30] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "PCN: Point completion network," in *Proc. Int. Conf. 3D Vis. (3DV)*, Sep. 2018, pp. 728–737.
- [31] L. Pan, "ECG: Edge-aware point cloud completion with graph convolution," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4392–4398, Jul. 2020.
- [32] X. Wang, M. H. Ang, and G. H. Lee, "Cascaded refinement network for point cloud completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 790–799.
- [33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
- [34] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
- [35] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.*, 2015, pp. 234–241.
- [36] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2017, pp. 652–660.
- [37] M. M. Loper and M. J. Black, "OpenDR: An approximate differentiable renderer," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 154–169.
- [38] H. Kato, Y. Ushiku, and T. Harada, "Neural 3D mesh renderer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3907–3916.
- [39] T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen, "Differentiable Monte Carlo ray tracing through edge sampling," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–11, Jan. 2019.
- [40] W. Yifan, F. Serena, S. Wu, C. Öztireli, and O. Sorkine-Hornung, "Differentiable surface splatting for point-based geometry processing," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–14, Nov. 2019.
- [41] S. Liu, W. Chen, T. Li, and H. Li, "Soft rasterizer: A differentiable renderer for image-based 3D reasoning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7708–7717.
- [42] C.-H. Lin, C. Kong, and S. Lucey, "Learning efficient point cloud generation for dense 3D object reconstruction," 2017, *arXiv:1706.07036*. [Online]. Available: <http://arxiv.org/abs/1706.07036>
- [43] R. Roveri, A. C. Öztireli, I. Pandle, and M. Gross, "Pointprnets: Consolidation of point clouds with convolutional neural networks," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 87–99, 2018.
- [44] G. Liu, D. Ceylan, E. Yumer, J. Yang, and J.-M. Lien, "Material editing using a physically based rendering network," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2261–2269.
- [45] T. H. Nguyen-Phuoc, C. Li, S. Balaban, and Y. Yang, "RenderNet: A deep convolutional network for differentiable rendering from 3D shapes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7891–7901.
- [46] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik, "Multi-view supervision for single-view reconstruction via differentiable ray consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2626–2634.
- [47] P. Hermosilla, T. Ritschel, P.-P. Vázquez, À. Vinacua, and T. Ropinski, "Monte Carlo convolution for learning on non-uniformly sampled point clouds," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–12, Jan. 2019.
- [48] H.-T. D. Liu, M. Tao, and A. Jacobson, "Paparazzi: Surface editing by way of multi-view image processing," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–221, 2018.
- [49] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3D-structure-aware neural scene representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1121–1132.
- [50] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 405–421.
- [51] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 206–215.
- [52] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-Net: Efficient channel attention for deep convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11534–11542.
- [53] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7794–7803.
- [54] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.
- [55] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.
- [56] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 605–613.
- [57] Visionair. Accessed: Nov. 14, 2017. [Online]. Available: <http://www.infra-visionair.eu/>
- [58] A. X. Chang *et al.*, "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*. [Online]. Available: <http://arxiv.org/abs/1512.03012>
- [59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015.
- [60] M. Berger, J. A. Levine, L. G. Nonato, G. Taubin, and C. T. Silva, "A benchmark for surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 2, pp. 1–17, Apr. 2013.
- [61] J. Zhang, W. Chen, Y. Wang, R. Vasudevan, and M. Johnson-Roberson, "Point set voting for partial point cloud analysis," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 596–603, Apr. 2021.
- [62] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mache approach to learning 3D surface generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 216–224.
- [63] L. P. Tchappmi, V. Kosaraju, H. Rezaatoughi, I. Reid, and S. Savarese, "TopNet: Structural point cloud decoder," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 383–392.
- [64] H. Xie, H. Yao, S. Zhou, J. Mao, S. Zhang, and W. Sun, "GRNet: Gridding residual network for dense point cloud completion," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 365–381.



**Pingping Zhang** received the M.S. degree from the College of Computer Science and Software Engineering, Shenzhen University, China, in 2020. She is currently pursuing the Ph.D. degree with the Department of Computer Science, City University of Hong Kong, Hong Kong. Her research interests include point cloud upsampling and image/video compression.



**Xu Wang** (Member, IEEE) received the B.S. degree from South China Normal University, Guangzhou, China, in 2007, the M.S. degree from Ningbo University, Ningbo, China, in 2010, and the Ph.D. degree from the Department of Computer Science, City University of Hong Kong in 2014. In 2015, he joined the College of Computer Science and Software Engineering, Shenzhen University, where he is currently an Associate Professor. His research interests include video coding and 3D vision.



**Lin Ma** (Member, IEEE) received the B.E. and M.E. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 2006 and 2008, respectively, and the Ph.D. degree from the Department of Electronic Engineering, The Chinese University of Hong Kong, in 2013.

He was a Researcher with Huawei Noah's Ark Laboratory, Hong Kong, from 2013 to 2016. He was a Principal Researcher with Tencent AI Laboratory, Shenzhen, China, from 2016 to 2020. He is currently a Researcher with Meituan, Beijing, China. His current research interests lie in the areas of computer vision, multimodal deep learning, specifically for image and language, image/video understanding, and quality assessment. He received the Best Paper Award from the Pacific-Rim Conference on Multimedia in 2008. He was a recipient of the Microsoft Research Asia Fellowship in 2011. He was a finalist of the HKIS Young Scientist Award in engineering science in 2012.



**Shiqi Wang** (Member, IEEE) received the B.S. degree in computer science from the Harbin Institute of Technology in 2008 and the Ph.D. degree in computer application technology from Peking University in 2014. From 2014 to 2016, he was a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. From 2016 to 2017, he was a Research Fellow with the Rapid-Rich Object Search Laboratory, Nanyang Technological University, Singapore. He is currently an Assistant

Professor with the Department of Computer Science, City University of Hong Kong. He has authored over 40 technical proposals in ISO/MPEG, ITU-T, and AVS standards. His research interests include image/video compression, analysis, and quality assessment.



**Sam Kwong** (Fellow, IEEE) received the B.S. degree in electrical engineering from the State University of New York at Buffalo in 1983, the M.S. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1985, and the Ph.D. degree from University of Hagen, Germany, in 1996. From 1985 to 1987, he was a Diagnostic Engineer with Control Data Canada. He joined Bell Northern Research Canada as a Member of Scientific Staff. In 1990, he became a Lecturer at the Department of Electronic Engineering, City University of Hong Kong, where he is currently a Chair Professor with the Department of Computer Science. His research interests include video and image coding and evolutionary algorithms.



**Jianmin Jiang** received the Ph.D. degree from the University of Nottingham, Nottingham, U.K., in 1994. From 1997 to 2001, he was a Full Professor of computing with the University of Glamorgan, Pontypridd, U.K. In 2002, he joined the University of Bradford, Bradford, U.K., as a Chair Professor of Digital Media, and the Director of the Digital Media and Systems Research Institute. He was a Full Professor with the University of Surrey, Guildford, U.K., from 2010 to 2014, and a Distinguished Chair Professor (1000-Plan) with Tianjin University, Tianjin, China, from 2010 to 2013. He is currently a Distinguished Chair Professor and the Director of the Research Institute for Future Media Computing, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. He has published over 400 refereed research articles. His current research interests include image/video processing in compressed domain, digital video coding, medical imaging, computer graphics, machine learning and AI applications in digital media processing, and retrieval and analysis. He was a Chartered Engineer, a fellow of IEE and RSA, a member of EPSRC College in the U.K., and an EU FP-6/7 Evaluator.

He is currently a Distinguished Chair Professor and the Director of the Research Institute for Future Media Computing, College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. He has published over 400 refereed research articles. His current research interests include image/video processing in compressed domain, digital video coding, medical imaging, computer graphics, machine learning and AI applications in digital media processing, and retrieval and analysis. He was a Chartered Engineer, a fellow of IEE and RSA, a member of EPSRC College in the U.K., and an EU FP-6/7 Evaluator.