IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

ECG: Edge-aware Point Cloud Completion with Graph Convolution

Liang Pan

***Abstract*— Scanned 3D point clouds for real-world scenes often suffer from noise and incompletion. Observing that prior point cloud shape completion networks overlook local geometric features, we propose our ECG - an Edge-aware point cloud Completion network with Graph convolution, which facilitates fine-grained 3D point cloud shape generation with multi-scale edge features. Our ECG consists of two consecutive stages: 1) skeleton generation and 2) details refinement. Each stage is a generation sub-network conditioned on the input incomplete point cloud. The first stage generates coarse skeletons to facilitate capturing useful edge features against noisy measurements. Subsequently, we design a deep hierarchical encoder with graph convolution to propagate multi-scale edge features for local geometric details refinement. To preserve local geometrical details while upsampling, we propose the Edge-aware Feature Expansion (EFE) module to smoothly expand/upsample point features by emphasizing their local edges. Extensive experiments show that our ECG significantly outperforms previous state-of-the-art (SOTA) methods for point cloud completion.**

I. INTRODUCTION

Many robotics applications, such as 3D object recognition [1] and robotics manipulation [2], frequently employ point clouds. However, real-world 3D points captured by LiDAR and/or depth cameras are often sparse, irregular and incomplete due to noise, clutter and occlusion, hence leading to challenges for deep learning. For instance, grasp planning can be challenging based on incomplete information regarding scene geometry, and prior methods usually enable stable robotic grasp planning via shape completion [2], [3], [4], [5]. Despite novel networks are recently proposed for point cloud completion [6], [7], [8], limited studies have been carried out on exploiting multi-scale local geometrical details for high-fidelity point cloud shape reconstruction.

The object shape completion task can be considered as generating a complete 3D shape conditioning on a partial 3D point cloud observation. As shown in Fig. 1, an incomplete point cloud can contain global shape information (such as shape categories, scale and size) and local geometric details (such as decorative patterns and sharp edges). For example, an incomplete chair x (e.g without one leg) can be coarsely reconstructed as a complete chair (e.g with four legs) y'_{coarse} only based on global features, but we often cannot identify y'_{coarse} as the specific chair y due to the absence of many characteristic local features (e.g. sharp decorative structures). Analogous to us human beings, many chairs can have similar skeletons (global), and their different appearances (local) help us recognize and distinguish. Nonetheless, previous frameworks [9], [6], [7] on point cloud completion follow an encoder-decoder architecture to generate a complete shape only based on the embeded global features, which highly limits their shape completion performance.

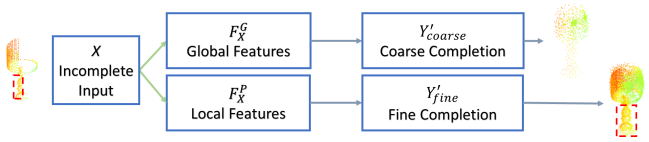

Fig. 1. An incomplete point cloud can describe its coarse skeleton and local fine details, but previous methods overlook local features F_X^P.

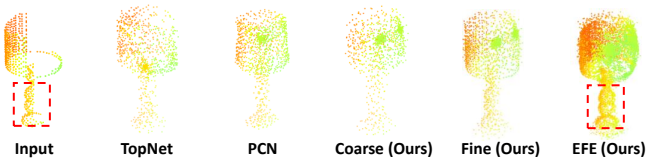

Fig. 2. An example on reconstructing a complete 3D shape by observing an incomplete point cloud. Compared to prior methods [6], [7], our results can better preserve observed edges and hence generating better reconstructions.

In this paper, we propose our ECG - a deep end-to-end permutation-invariant (i.e. all applied operations are symmetric) network to exploit local geometric features for point cloud completion. Because it can be challenging for networks to directly exploit edge features in irregularly distributed incomplete point cloud, we circumvent this problem by designing a two-staged network consisting of 1) skeleton generation; 2) details refinement. Similar to PCN [6], we generate the coarse skeleton in our first stage with the embeded global features. In contrast to PCN, our ECG has a second stage to exploit local geometric features with the help of the generated skeleton. More specifically, we propose a deep hierarchical encoder with graph convolution to exploit multi-scale edge features, from our generated skeleton and the incomplete input point cloud, for fine details reconstruction. Furthermore, we propose an Edge-aware Feature Expansion (EFE) module to upsample/expand point features with graph convolution, which can better preserve local geometric details than naively duplicating point features for upsampling ("Folding" [10]). Experimental results show that our ECG provides better point cloud completion results than previous SOTA methods because of better preserved edge features (an example shown in Fig. 2).

Our **Key** contributions can be summarized as:

- We propose a novel two-staged network (dubbed ECG) that exploits global features and local features for generating fine-grained complete point clouds.
- We apply novel modules with graph convolutions (e.g. EFE) for edge-aware feature learning and/or preserving.

Liang PAN is with Advanced Robotics Centre, National University of Singapore pan.liang@u.nus.edu

- Extensive experiments validate the effectiveness of our ECG, which significantly outperforms previous SOTA for point cloud completion.

II. RELATED WORK

a) Graph Convolution for Point Cloud.: Graph convolution can conveniently probe non-Euclidean data structures by collectively aggregate information from graph structures [11]. Accordingly, graph convolution can be used for unordered and unorganized point cloud, which can flexibly learn to represent nodes, edges or subgraphs in low-dimensional vectors [12], [13]. As for point cloud analysis, graph convolution based on neighborhood graphs, which are constructed in either metric space or feature space, is frequently applied for capturing local geometric features. PointNet++ [14] adaptively combines multi-scale local features by sampling neighboring points using ball query methods. In a similar spirit, many follow-up works [15], [16], [17], [18], [19], [20] propose novel kernels for graph convolutions to learn local geometric features. The learned edge features from local neighborhood graphs effectively improve its deep learning capability on point cloud.

b) Point Cloud Upsampling.: Different from shape completion that generates entire object shape from partial input, point cloud upsampling improve point distribution uniformity for complete but low-resolution input points. Yu et al. propose PU-Net [21] to upsample point set based on PointNet++ [14] architectures, which expands a point set by mixing-and-blending point features in the feature space. Later, they propose EC-Net [22] that is an edge-aware point cloud consolidation network with a novel edge-aware joint loss for minimizing point-to-edge distances. Wang et al. [23] formulate the upsampling task as a multi-step process, which progressively upsamples point cloud with various scale local patches in a coarse-to-fine style. To improve the point distribution uniformity generated from the latent space, PU-GAN [24] applies the adversarial learning strategy of the GAN framework, which combines completion and uniformity together with upsampling.

c) Point Cloud Shape Completion.: Most previous learning-based shape completion methods [25], [26], [27], [28] represent shapes using voxels, which can be inefficient in terms of memory and time, especially for fine-grained shape completion. PCN [6] firstly recovers a complete coarse reconstruction based on the embeded global features from the partial observation, and then upsamples the coarse prediction by using point features with folding operations. TopNet [7] proposes a novel decoder that follows a hierarchical rooted tree structure, which can learn arbitrary grouping of points including any topology on the point set. Both approaches [6], [7] regard the point cloud shape completion as a point cloud generation problem based on the feature embedding of the incomplete shape, which is obtained by a single global max-pooling operation. Consequently, local characteristic patterns of the incomplete input points are entirely missing. To unravel this problem, we aim to generate complete 3D point clouds by constructing global structures and their fine-grained geometric details. Very recently, Liu et. al. [29] tended to generate evenly distributed high fidelity point clouds by morphing and sampling, which did not explicitly consider local features of the input incomplete points.

III. OUR METHOD

We define the point set X as an incomplete shape for a 3D object, which is sampled from the "observed" surfaces of the object. Correspondingly, we define a point set Y as an complete shape for the 3D object, which is sampled from the whole surfaces of the object. Since X and Y are obtained by two separate sampling processes, X is not necessary a subset of Y. Therefore, the point cloud completion problem can be formulated as predicting Y' given X, which is a conditional generation problem. The incomplete point cloud X can preserve both global structural information (F_X^G) and local pattern features (F_X^P) for the observation. Bearing this in mind, we reformulate the point cloud shape completion problem with two sub-tasks: skeleton generation and details refinement (shown in Equ. (1), where E denotes an encoder and D denotes a decoder). Therefore, our final reconstructed 3D point cloud Y'_{fine} can benefit from both F_X^G and F_X^P.

$$\begin{cases} F_X^G, F_X^P = E(X) \\ Y'_{coarse} = D_{coarse}(F_X^G) \\ Y'_{fine} = D_{fine}(F_X^P, Y'_{coarse}) \end{cases} \tag{1}$$

However, it is difficult to exploit useful edge features in the noisy, incomplete and irregularly distributed point cloud X. Specifically, those "fake edges" (e.g. caused by missing observation) can impose strong negative impact on edge-aware feature learning. Therefore, we alleviate the problem by using a multi-stage training strategy. The overview of our method is shown in Fig. 3, which consists of two encoder-decoder sub-networks to complete 3D shapes Y' in a coarse-to-fine fashion. We firstly reconstruct a complete but low-resolution shape Y'_{coarse} based on the embeded global features F_X^G, which serves as the skeleton of the structure. With the help of Y'_{coarse}, we can better distinguish "real edges" and "missing observation". Following, we recover local details for the reconstructed skeleton conditioned on the learned dense point features F_X^P.

A. Skeleton Generation

Similar to prior works [6], [7], we generate a coarse complete shape Y'_{coarse} by decoding the global embedding F_X^G (shown in equ. (2)). Different from PCN which upsamples the coarse point cloud by directly duplicating coarse predicted points as fine predictions, we apply the coarse shape as the skeleton to facilitate the dense point feature F_X^P learning from the incomplete observed point cloud. Specifically, various encoder architectures, including our hierarchical model with graph convolution, are evaluated and discussed in the experiments section.

$$\begin{cases} F_X^G = E_{coarse}(X) \\ Y'_{coarse} = D_{coarse}(F_X^G) \end{cases} \tag{2}$$

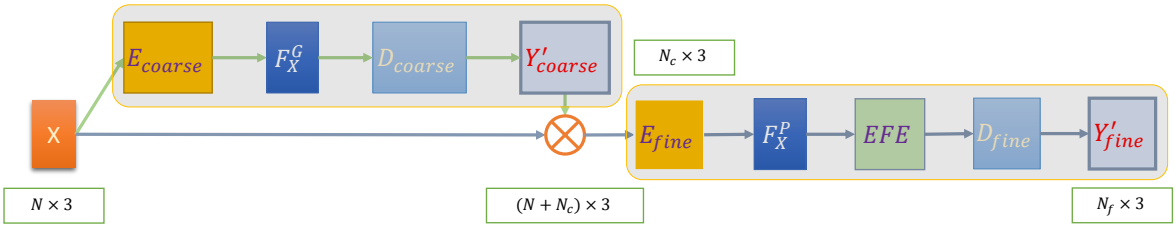

Fig. 3. System overview of our ECG. Our ECG consists of two sub-networks that both follow the encoder-decoder architecture. The first sub-network aims to generate the coarse shape Y'_{coarse}, which serves as the skeleton to exploit local features in the partial observation X. The second sub-network exploits multi-scale edge-aware features for high-grained shape completion. Note that we take the EFE module out of the decoder D_{fine} for better illustration.

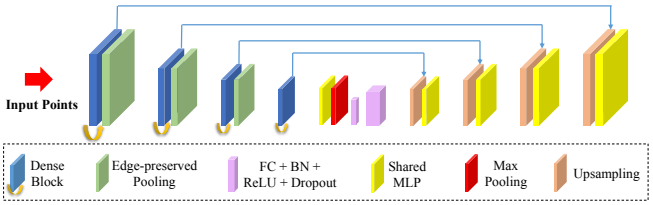

Fig. 4. Our hierarchical encoder architecture for learning multi-scale edge feature with graph convolutions.

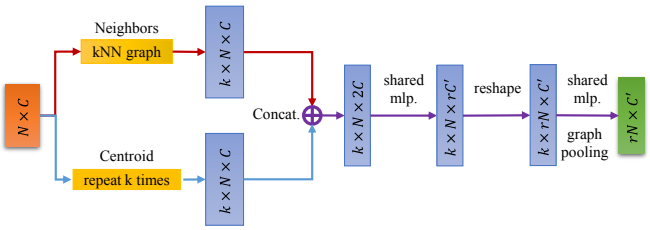

Fig. 5. Illustration for our edge-aware feature expansion module.

B. Details Refinement

Our details refinement stage also follows an encoder-decoder architecture, which aims to generate high-quality 3D complete shape based on local geometric details of the observed incomplete 3D shape. However, it is very challenging to effectively extract critical edge features in incomplete and non-uniformly distributed point cloud, especially for those sparse and isolated 3D points. We circumvent this problem by exploiting local geometric details in both coarse complete point set Y'_{coarse} and X (shown in Eq. (3)). In such way, the shape skeleton can be regarded as adaptive 3D anchor points and benefits in capturing useful local edge features from X.

$$\begin{cases} F_X^P = E_{fine}(X, Y'_{coarse}) \\ Y'_{fine} = D_{fine}(F_X^P) \end{cases} \tag{3}$$

1) Edge Feature Learning with Graph Convolution: Previous methods [14], [18] exploit local edge features by searching neighboring points with two main strategies: 1) ball query; 2) k-nearest-neighbors (kNN). The ball query algorithm applied by PointNet++ [14] always selects the first #K points in multiple specified search balls with predefined radii. Consequently, it cannot guarantee that the closest #K points can always be selected [30], [20]. Hence, we select neighboring point features by selecting k nearest neighbors. We apply the EdgeConv [18] operation, which constructs neighborhood graphs by selecting #K most neighboring points in the feature space. The encoded edge feature f'_p can be represented as:

$$f'_p = g\left[H_\Theta\left(f_p \oplus (f_p - f_q^1)\right), ..., H_\Theta\left(f_p \oplus (f_p - f_q^k)\right)\right], \tag{4}$$

where f_p is the input feature of the centroid point p, f_q^k is the feature of its k^{th} nearest neighbor, $\oplus$ denotes feature concatenation, $H_\Theta(\cdot)$ denotes *shared* multilayer perceptrons (MLPs) and $g[\cdot]$ is a max-pooling operation to generate order-invariant edge features over the neighborhood graph. Inspired by MPU [23], we add skip connections (dense block) within each network hierarchy. In contrast to MPU, we propose a deep hierarchical encoder structure (shown in Fig. 4) for explicitly learning multi-scale edge features, which improves our network performance and efficiency.

2) An Efficient Hierarchical Network: With a fixed density of input points, the receptive fields of convolution filters are dependent on the size of kNN for constructing neighborhood graphs. Large neighborhood graphs can enlarge the scope of convolution filters, while highly increases the network complexity, especially when dealing with dense high-dimensional features. In order to efficiently learn multi-scale edge-aware features from 3D points, we apply Edge-preserved Pooling (EP) operations [20], which propagate both the selected centroid point features and their local neighboring point features during subsampling input points. Accordingly, the receptive field becomes wider as the point set becomes sparser. Different from PointNet++ that only propagates local neighboring features with a max-pooling operation, our EP operations can propagate less duplicate and less correlated feature to the next hierarchy, which is defined as:

$$f'_p = f_p \oplus g\left[f_q^1, ..., f_q^k\right]. \tag{5}$$

Correspondingly, we progressively recover spatial information in the decoder by considering low-level features and interpolated features from the previous hierarchy.

$$f'_p = f_p^e \oplus w\left[f_q^1, ..., f_q^k\right], \tag{6}$$

where f_p^e is the corresponding feature propagated from the encoder by a skip connection directly, $w[\cdot]$ is the inverse distance weighted average operation and f_q^k is feature of the k^{th} nearest neighbor of point p in its previous hierarchy. Note that we select neighboring points in the metric space rather than the feature space to gather spatial information.

3) Edge-aware Feature Expansion: Because incomplete point sets are usually with low-resolution, we can upsample to recover fine-grained geometric details. PU-Net [21] uses multiple independent MLPs to expand their exploited dense point features, which is equivalent to duplicating the exploited point features for a single "large" MLPs. MPU [23] and PU-GAN [24] directly copy point features, where each duplicated feature is appended with a unique 2D vector based on the 2D grid mechanism proposed by FoldingNet [10]. However, those duplicated point features highly limit the variations among the expanded point features. Wang et al. [31] propose the dense upsampling convolution (DUC) layer to upsample high-dimensional image features with learnable convolution operations. The DUC module first propagates the input feature map of dimension $(B\times H/d\times W/d\times C)$ to the output feature map of dimension $(B \times H/d \times W/d \times (d^2 \times L))$, which is then reshaped to the prediction map of dimension $(B \times H \times W \times L)$. On the other hand, new expended point features can be intuitively generated/interpolated via their neighboring point features assuming smooth and consistent surfaces of the complete reconstructed 3D shape. In view of this, we propose our Edge-aware Feature Expansion (EFE) module (illustrated in Fig. 5) to expand and upsample point features with graph convolution operations, as follows:

$$f'_p = g\left[\Gamma\left(H_\Theta(f_p \oplus f_q^1), ..., H_\Theta(f_p \oplus f_q^k)\right)\right], \quad (7)$$

where $\Gamma(\cdot)$ denotes a reshape operation, that is to reshape an input tensor with the size $(B \times K \times N \times rC)$ to an output tensor with the size $(B \times K \times rN \times C)$. In this way, we upsample the point features r times by upsampling many local neighborhood graphs first. Since the constructed neighborhood graph with each point is unique, the expanded points are merely duplicated. Particularly, we smoothly expand point features by considering both the centroid features f_p and its neighboring point feature f_q^i. Therefore, our EFE module can effectively preserve edge features for new point features generation.

C. End-to-end Multistage Training

Our network consists of two stages for generating complete point cloud from coarse to fine, and each stage follows an encoder-decoder architecture. The first stage generates coarse predictions Y'_{coarse} based on exploited global features, which can be regarded as 3D anchor points to facilitate local geometric details exploitation from X. In the second stage, we leverage multi-scale edge-aware features for high-grained 3D model reconstruction Y'_{fine}. Therefore, we apply a joint loss function for both Y'_{coarse} and Y'_{fine} during training.

a) Reconstruction Loss.: Two types of reconstruction loss functions are widely applied, Chamfer Distance (CD) and Earth Mover's Distance (EMD). In line with TopNet [7], we select the symmetric CD loss, which calculates the average closest point distance between two point sets, Y' and Y.

$$\begin{aligned}\mathcal{L}_{rec}(Y', Y) = d_{CD}(Y', Y) = \frac{1}{|Y'|}\sum_{p_x \in Y'} \min_{p_y \in Y} \|p_x - p_y\|^2 \\ + \frac{1}{|Y|}\sum_{p_y \in Y} \min_{p_x \in Y'} \|p_x - p_y\|^2,\end{aligned} \quad (8)$$

where p_x is a point in our predicted point set Y', and p_y is a point in the ground truth point set Y.

b) Uniform Loss.: Inspired by PU-GAN [24], we also use a uniform loss during training to generate uniformly distributed point cloud:

$$\begin{cases} U_{imbalance} = \dfrac{\left(|S_j| - \hat{n}\right)^2}{\hat{n}}, \text{ where } \hat{n} = (rN \cdot r_d^2) \\ U_{clutter} = \displaystyle\sum_{k=1}^{|S_j|} \frac{(d_{j,k} - \hat{d})^2}{\hat{d}}, \text{ where } \hat{d} = \sqrt{\dfrac{2\pi r_d^2}{|S_j|\sqrt{3}}}, \\ \mathcal{L}_{uni}(Y') = \displaystyle\sum_{j=1}^{M} U_{imbalance}(S_j) \cdot U_{clutter}(S_j) \end{cases} \quad (9)$$

where S_j $(j = 1, ..., M)$ is a point subset sampled from our prediction Y' by a ball query of radius r_d, rN is the number of points after upsampling, r_d^2 is the expected point density in S_j. Consequently, $U_{imbalance}$ measures the deviation of $|S_j|$ from $\hat{n}$ that is the expected number of points in S_j. $U_{clutter}$ penalizes the deviation of point-to-neighbor distance $d_{j,k}$ from the expected distance $\hat{d}$.

c) Compound Loss.: Based on the introduced loss functions, our compound loss function is defined as:

$$\begin{aligned}\mathcal{L} = \omega_c \left[\lambda_{rec}\mathcal{L}_{rec}(Y'_{coarse}, Y) + \lambda_{uni}\mathcal{L}_{uni}(Y'_{coarse})\right] \\ + \omega_f \left[\lambda_{rec}\mathcal{L}_{rec}(Y'_{fine}, Y) + \lambda_{uni}\mathcal{L}_{uni}(Y'_{fine})\right],\end{aligned} \quad (10)$$

where λ_{rec}, λ_{uni}, ω_c and ω_f are weights for reconstruction loss term, uniform loss term, coarse prediction and fine prediction, respectively.

TABLE I
POINT CLOUD AUTOENCODER ON THE SHAPENETPART.

Encoder	PointNet [32]	PCN [6]	Ours
CD Loss (10^{-4})	22.64	20.79	**19.67**
Training time (h)	2.95	**2.75**	4.48

IV. EXPERIMENTS

In this section, we firstly evaluate the effectiveness on encoding point features of our hierarchical encoder with graph convolution on the point cloud autoencoder task. Subsequently, we provide qualitative and quantitative comparisons between our method and previous methods on the point cloud completion task. For both applications (point

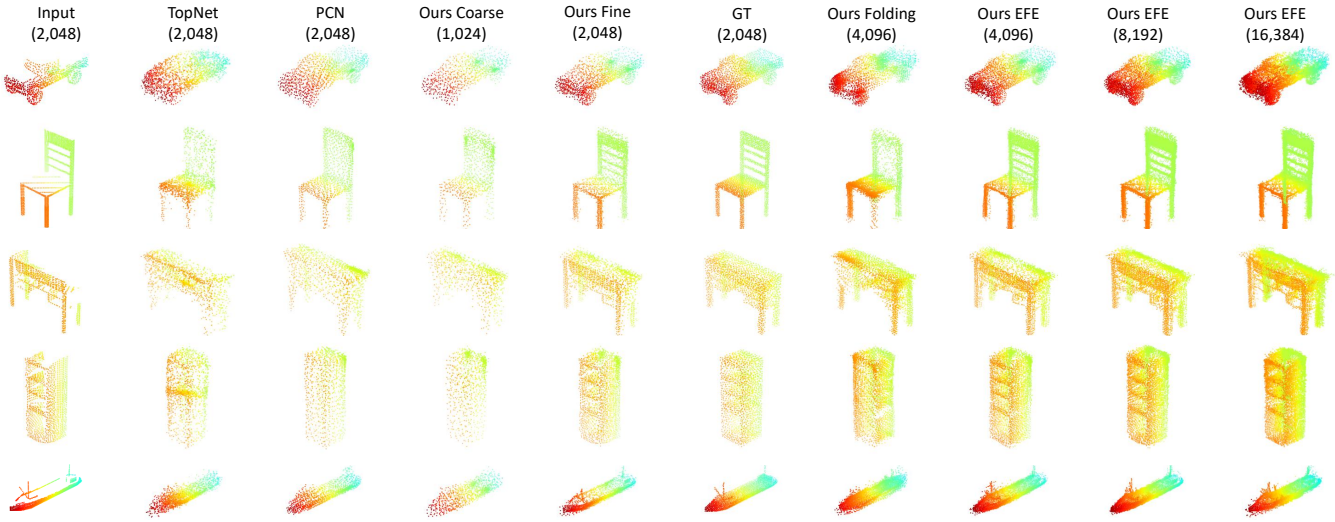

Fig. 6. Qualitative results for generated complete 3D shapes with different resolutions (2,048, 4,096, 8,192, and 16,384 points).

TABLE II

PER-CATEGORY SHAPE COMPLETION RESULTS WITH THE RESOLUTION ($N \approx 2,048$). THE CD LOSS IS MULTIPLIED BY 10^4.

	Plane	Cabinet	Car	Chair	Lamp	Couch	Table	Watercraft	Average
AtlasNet [9]	10.37	23.40	13.41	24.16	20.24	20.82	17.52	11.62	17.69
PointNetFCAE [7]	10.31	19.07	11.83	24.69	20.31	20.09	17.57	10.50	16.80
Folding [10]	11.18	20.15	13.25	21.48	18.19	19.09	17.80	10.69	16.48
PCN [6]	5.44	19.88	9.22	17.86	16.29	16.62	13.72	7.95	13.37
TopNet [7]	9.29	18.79	11.57	18.44	14.69	18.63	13.45	8.65	14.19
Ours	**4.99**	**15.09**	**8.95**	**12.86**	**10.65**	**12.90**	**10.03**	**6.08**	**10.19**

TABLE III

POINT CLOUD COMPLETION RESULTS WITH VARIOUS RESOLUTIONS. THE CD LOSS IS MULTIPLIED BY 10^4.

# points	2,048	4,096	8,192	16,384
AtlasNet [9]	17.69	16.12	15.32	14.85
PointNetFCAE [7]	16.80	14.79	12.57	10.61
Folding [10]	16.48	13.19	12.95	12.26
PCN [6]	13.37	11.39	10.07	8.88
TopNet [7]	14.19	12.65	10.60	8,77
Ours	**10.19**	**8.79**	**7.82**	**7.41**

cloud autoencoder and point cloud completion), our proposed networks provide better results than previous SOTA methods. Our code and models will be publicly available on the project website[1].

A. Point Cloud Autoencoder

Autoencoders are frequently used for feature embedding applications, such as dimension reduction, which aims to learn an efficient representation for the complicated input data. A typical autoencoder consists of an encoder and a decoder. The encoder is trained to learn critical low-dimensional features that can prevent noise and tedious signals. Along with the feature reduction, the decoder aims to reconstruct the original input. Ideally, if most characteristic features of the original input are embeded, the autoencoder can reconstruct very similar signals as the input. Hence, we evaluate our network encoder with novel operations, including graph convolutions and EP modules, on this point cloud autoencoder task. We train and test autoencoders on the ShapeNetPart dataset [33], [34] that consists of 16,881 shapes from 16 categories. Following previous setting [34], we split the dataset into a training set (14,043 shapes) and a test set (2,847 shapes). Each input object has 2,048 3D points, which is then encoded as a feature vector with the dimension 1,024. To evaluate the performance of different encoders, all the evaluated autoencoders have the same decoder architectures and training strategies. As reported in Table I, our hierarchical encoder with graph operations provide the best reconstruction results. However, nearest neighbors search operations make our graph-based encoder less efficient than the other two encoders.

B. Point Cloud Shape Completion

To achieve fair comparisons, we use the same training set and test set with TopNet [7], which are subsets of the dataset created by PCN [6]. Following TopNet [7], we also keep $N = 2048$ for both incomplete input points and groundtruth points. Because TopNet does not provide the groundtruth data of the test set, we generate groundtruth data (the same setting as TopNet [7]) by sampling 2,048 points from the 16,384 points provided by PCN [6].

a) Implementation Details.: We train our models for 100 epochs with batch size equals to 32. We use Adam optimizer with the initial learning rate 1e-4, decay step

[1]https://github.com/paul007pl/ECG

TABLE IV

ABLATION STUDY OF OUR NETWORKS. THE CD LOSS IS MULTIPLIED BY 10^4.

# points	E_{coarse}		E_{fine}					D_{fine}			CD Loss
	PCN [6]	EP	PCN [6]	MPU [23]	EP	Y'_{coarse}	X	Folding [6]	EFE	MLPs	
2,048				✓						✓	21.31
	✓							✓			13.37
		✓						✓			12.83
	✓			✓		✓				✓	13.14
	✓		✓			✓	✓			✓	12.95
	✓			✓		✓	✓			✓	11.29
	✓				✓	✓	✓			✓	10.19
4,096	✓							✓			11.39
	✓				✓	✓	✓	✓		✓	11.81
	✓				✓	✓	✓		✓	✓	8.79

TABLE V

ABLATION STUDY RESULTS FOR THE FEATURE UPSAMPLING MODULE.

# points	Folding [10]	# kNN	Graph Operation Kernel		CD Loss
			$(f_p \oplus f_p - f_q^i)$	$(f_p \oplus f_q^i)$	
4,096	✓	-	-	-	11.81
	-	0	-	-	9.15
	-	4	✓	-	9.59
	-	16	-	✓	8.92
	-	4	-	✓	8.79
8,192	✓	-	-	-	10.02
	-	4	-	✓	7.82

150,000 and decay rate 0.7. No batch normalization or layer normalization operations are used. For previous networks, we keep their default training settings and report their best reconstruction results.

b) Evaluation.: We firstly evaluate the reconstruction performance of 2,048 points, and the per-category completion results are reported in Table II. In comparison with previous SOTA methods, our ECG can provide much better completion results for all the categories. TopNet [7] does not perform well on our dataset. Because the incomplete 3D points are sampled from dense 3D surfaces, we also evaluate our ECG on shape completion with a high-resolution (N=2048, 4096, 8192 and 16384). Following TopNet [7], we keep the ground truth with 2,048 points for computational efficiency. As reported in Table III, our ECG outperform all previous methods for all evaluated resolutions. The qualitative results are provided in Fig. 6. Apparently, our reconstructed model can better preserve the observed geometric details. Furthermore, although we can exploit edge features to generate complete shapes (shown in the 5^{th} column), a simple folding operation with duplicated point features can obviously influence those learnt edges (shown in the 7^{th} column). In contrast, our EFE module effectively preserve edge features while upsampling processes, even for very high resolution (shown in the last column).

c) Ablation Studies.: To evaluate the effectiveness of each module in our network, we provide ablation studies in Table IV. We firstly try to direct exploit local geometric features [23] for point cloud completion, which however cannot generate reasonable complete shapes. We observe that our hierarchical encoder architecture benefits in generating both global features F_X^G and dense point features F_X^P, which is consistent with our experiments on the autoencoder task. However, we prefer using PCN encoder for encoding F_X^G due to its high efficiency. According to our experiments, using both Y'_{coarse} and X as input to our second subnetwork can improve generated shape quality. Using a single stage network cannot generate detailed structures, even with our hierarchical encoder. Another observation is that the folding operation which is widely applied in many point cloud upsampling tasks can easily diminish the edge-aware features. Because it directly duplicates point features, which highly limits the variations of point features. However, it benefits in generating potential surfaces smoothly. Note that we apply a list of MLPs layers as the D_{coarse} for all the experiments, which directly regresses the global feature F_X^G and generates Y'_{coarse}.

In addition, we evaluate different point feature upsampling modules as reported in Table V. In our experiments, we try folding operations [10] for many times and reported the best results. Apparently, folding operations cannot generate complete 3D shape with high-quality, especially for high-resolution shapes. As for our EFE module, alternative structures have been studied. The setting with 0 kNN means that we upsample the point features by reshape $(B \times N \times rC)$ to $(B \times rN \times C)$ without considering local neighborhood graphs. Further, we evaluate the edge kernel $(f_p \oplus f_p - f_q^i)$, but it does not provide better results. In addition, using larger #kNN does not increase our performance but makes the network inefficient.

V. ANALYSIS & FUTURE DIRECTIONS

We reformulate the point cloud shape completion as a conditional generation problem by considering both global embeded features and dense local edge features. Prior methods focus on predicting the whole shape by decoding the encoded global feature of X, hence ignoring local geometric details. Since many point cloud applications [16], [15], [21], [35], such as classification, segmentation, detection and upsampling, benefits from local features, edge features at various scales should improve our understanding on predicting its 3D shape. Due to noisy and wrong observations, the conditional generation problem always hesitates between two choices, ***exploration*** and ***exploitation***. For example, a typical problem in the shape completion task is to differentiate "real" edges and "fake" edges. "real" edges can be important features

for better depicting 3D shapes, but "fake" edges caused by lacking of observations can easily confuse networks. Without semantic understanding associated with each point, it is still an open question to decide whether observed sharp edges belong to smooth surfaces or not.

VI. CONCLUSION

In this work, we propose a novel two-staged network - ECG, which leverages global features and local edge-aware features for point cloud completion. In the first stage, our ECG generates a coarse skeleton based on global features. In the second stage, ECG refines local details based on the coarse skeleton and the incomplete input. Experiments show that our ECG significantly outperforms previous SOTA methods for point cloud completion.

REFERENCES

[1] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[2] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 2442–2447.

[3] J. Lundell, F. Verdoja, and V. Kyrki, "Robust grasp planning over uncertain shape completions," *arXiv preprint arXiv:1903.00645*, 2019.

[4] ——, "Beyond top-grasps through scene completion," *arXiv preprint arXiv:1909.12908*, 2019.

[5] M. Tahoun, C. M. Mateo, J.-A. Corrales-Ramón, O. Tahri, Y. Mezouar, and P. Gil, "Visual completion of 3d object shapes from a single view for robotic tasks," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019, pp. 1777–1782.

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

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

[8] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3d point clouds," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80. PMLR, 10–15 Jul 2018, pp. 40–49.

[9] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.

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

[11] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun, "Graph neural networks: A review of methods and applications," *arXiv preprint arXiv:1812.08434*, 2018.

[12] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 5, pp. 833–852, 2018.

[13] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[15] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 4, 2018.

[16] J. Li, B. M. Chen, and G. H. Lee, "So-net: Self-organizing network for point cloud analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9397–9406.

[17] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102.

[18] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, pp. 1–12, 2019.

[19] L. Pan, P. Wang, and C.-M. Chew, "Pointatrousnet: Point atrous convolution for point cloud analysis," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4035–4041, 2019.

[20] L. Pan, C.-M. Chew, and G. H. Lee, "Pointatrousgraph: Deep hierarchical encoder-decoder with atrous convolution for point clouds," *arXiv preprint arXiv:1907.09798*, 2019.

[21] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-net: Point cloud upsampling network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2790–2799.

[22] ——, "Ec-net: an edge-aware point set consolidation network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 386–402.

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

[24] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "Pu-gan: a point cloud upsampling adversarial network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7203–7212.

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

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

[27] E. J. Smith and D. Meger, "Improved adversarial systems for 3d object generation and reconstruction," in *Conference on Robot Learning*, 2017, pp. 87–96.

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

[29] M. Liu, L. Sheng, S. Yang, J. Shao, and S.-M. Hu, "Morphing and sampling network for dense point cloud completion," *arXiv preprint arXiv:1912.00280*, 2019.

[30] A. Komarichev, Z. Zhong, and J. Hua, "A-cnn: Annularly convolutional neural networks on point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7421–7430.

[31] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding convolution for semantic segmentation," in *2018 IEEE winter conference on applications of computer vision (WACV)*. IEEE, 2018, pp. 1451–1460.

[32] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, vol. 1, no. 2, p. 4, 2017.

[33] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, pp. 1–12, 2016.

[34] C. R. Qi, "Autoencoder for point clouds," https://github.com/charlesq34/pointnet-autoencoder, 2018.

[35] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 918–927.