PU-REFINER: A GEOMETRY REFINER WITH ADVERSARIAL LEARNING FOR POINT CLOUD UPSAMPLING

Hao Liu[1,2], Hui Yuan (corresponding author)[2], Raouf Hamzaoui[3], Wei Gao[4], Shuai Li[2]

1.School of Information Science and Engineering, Shandong University, Qingdao, China,
2.School of Control Science and Engineering, Shandong University, Jinan, China,
3.School of Engineering and Sustainable Development, De Montfort University, Leicester, UK,
4.School of Electronic and Computer Engineering, Peking University, Shenzhen, China,
{liuhaoxb, yuanhui0325}@gmail.com, rhamzaoui@dmu.ac.uk, gaowei262@pku.edu.cn, shuaili@sdu.edu.cn

ABSTRACT

We present PU-Refiner, a generative adversarial network for point cloud upsampling. The generator of our network includes a coarse feature expansion module to create coarse upsampled features, a geometry generation module to regress a coarse point cloud from the coarse upsampled features, and a progressive geometry refinement module to restore the dense point cloud in a coarse-to-fine fashion based on the coarse upsampled point cloud. The discriminator of our network helps the generator produce point clouds closer to the target distribution. It makes full use of multi-level features to improve its classification performance. Extensive experimental results show that PU-Refiner is superior to five state-of-the-art point cloud upsampling methods. Code: https://github.com/liuhaoyun/PU-Refiner.

***Index Terms*—** Point cloud upsampling, deep learning, generative adversarial networks

1. INTRODUCTION

With the advancement of 3D acquisition technology (e.g., depth cameras and LiDAR), point clouds have become popular for 3D applications, such as autonomous navigation [1], immersive telepresence [2], etc. However, raw point clouds are usually uneven, sparse and noisy, which negatively affects subsequent rendering and analysis. To address this problem many point cloud upsampling methods [3-12] have been proposed. Ideally, the upsampling method should take a sparse, non-uniform and noisy low-resolution (LR) point cloud and produce a dense, uniform and noise-free high-resolution (HR) point cloud.

The existing point cloud upsampling methods can be divided into two main categories: optimization-based methods [3-7] and deep learning-based methods [8-12]. A pioneering optimization-based method was proposed by Alexa *et al.* [3] who built a Voronoi diagram on the surface and inserted points at the vertices of the diagram. Lipman *et al.* [4] proposed an effective locally optimal projection (LOP) operation to resample points. Huang *et al.* [5] proposed an edge-aware (EAR) upsampling method, which first interpolates the points away from the edge then moves points gradually towards the edge singularity. Dinesh *et al.* [6] proposed a model-based point could super-resolution method via graph total variation on surface normal. In [7], the same authors also proposed a fast graph total variation method for color point cloud super-resolution. Recently, deep learning has shown excellent performance for point cloud upsampling. PU-Net [8] introduced the idea of using multiple independent multilayer perceptrons (MLPs) to upsample points. Yu *et al.* [9] exploited the point-to-edge relationship for an edge-aware upsampling task. Wang *et al.* [10] proposed a patch-based progressive upsampling network, which learns different levels of detail in successive steps. Li *et al.* [11] proposed an efficient generative adversarial network (GAN) called PU-GAN to generate the upsampled point clouds. Qian *et al.* [12] used a local 2D parametric surface and corresponding normal vectors to adaptively upsample the point clouds.

Most current deep learning-based upsampling methods use a simple duplication feature-based upsampling strategy to generate the upsampled points directly. However, such a naive technique may prevent the network from exploring feature details. In this paper, we propose a novel and effective GAN for point cloud upsampling, namely PU-Refiner. The generator first generates a coarse upsampled point cloud with a coarse feature expansion module and a geometry generation module. Then, a progressive geometry refinement module is used to “carve” the coarse point cloud into a realistic dense point cloud with an elaborate geometry texture in a coarse-to-fine fashion. The discriminator uses multi-level features and point-wise confidence values to strengthen its recognition accuracy.

2. PROPOSED METHOD

2.1. Overview

Given a sparse, non-uniform LR point cloud $P_{LR} \in \mathbb{R}^{N\times 3}$, where N is the number of points and 3 corresponds to the

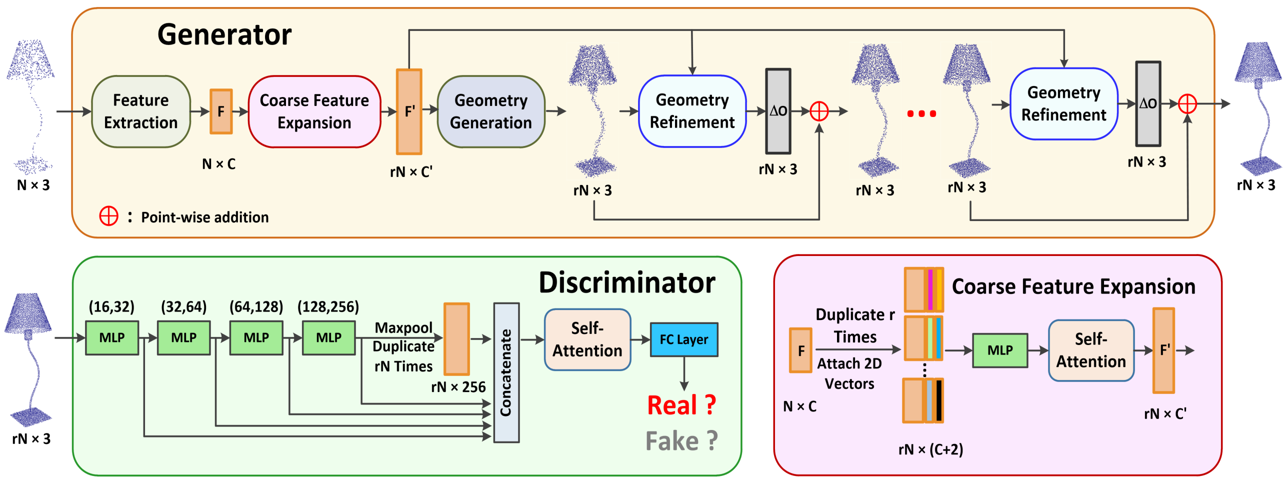

Fig. 1. Architecture of PU-Refiner. Proposed generator (top) and discriminator (bottom).

dimension of the geometry coordinates, PU-Refiner aims to generate a dense, uniform HR point cloud $P_{HR} \in \mathbb{R}^{rN\times 3}$, where r is the upsampling ratio. PU-Refiner is structured as follows.

2.2. Generator

The generator (Fig.1 (top)) aims to generate an HR point cloud. It consists of four modules: feature extraction, coarse feature expansion, geometry generation and geometry refinement.

Feature extraction learns point-wise expressive features $F \in \mathbb{R}^{N\times C}$ (C is the number of features) from P_{LR}. We adopt the feature extraction module of [10] because of its high efficiency.

Coarse feature expansion expands F to coarse upsampled features $F' \in \mathbb{R}^{rN\times C'}$, where C' is the number of upsampled features. As shown in Fig.1, we first copy F r times to form the plain upsampled features. To enhance the variety of features, we attach a 2D vector (it uniformly distributes in the interval [-0.2, 0.2]) [13] to the upsampled feature. Finally, F' is obtained after applying a MLP with a self-attention unit [14].

Geometry generation regresses a coarse upsamped point cloud $P_{HR\_c} \in \mathbb{R}^{rN\times 3}$ from F' via three fully connected (FC) layers.

Geometry refinement (GR). The existing upsampling methods generally produce the upsampled features by a simple feature replication technique in the feature extension module. However, such a simple strategy ignores the details of high-level semantics of the upsampled features. Therefore, we propose a progressive GR to alleviate the drawback of feature replication. As shown in Fig.1 (top), the coarse upsampled point cloud P_{HR_c} and coarse upsampled features F' are fed into the proposed GR module to predict the point-wise coordinate offsets $\Delta O \in \mathbb{R}^{rN\times 3}$. By using ΔO, we can get a refined dense point cloud. In this way, a realistic upsampled point cloud can be progressively approximated via several cascaded GR modules.

As shown in Fig.2, the proposed GR first takes an arbitrary coarse point cloud and F' as inputs. Then, we construct a local relationship via a k-nearest neighbor (k-NN) search for each point, i.e.,

$$\hat{f}_{ij}^{xyz} = [f_i^{xyz} \odot (f_i^{xyz} - f_j^{xyz})], \quad (1)$$

$$\hat{f}_{ij}^{F'} = [f_i^{xyz} \odot (f_i^{xyz} - f_j^{xyz}) \odot f_i^{F'} \odot (f_i^{F'} - f_j^{F'})], \quad (2)$$

where $\hat{f}_{ij}^{xyz}$ and $\hat{f}_{ij}^{F'}$ are the geometry and feature gradient of the i-th point with its j-th neighbor, respectively, f_i^{xyz} and f_j^{xyz} are the i-th point of the coarse point cloud and its j-th neighbor in Euclidean space, respectively, $f_i^{F'}$ and $f_j^{F'}$ are the i-th feature of F' and its j-th feature neighbor in feature space, respectively, and $\odot$ denotes the concatenation operation. Note that the same k-NN indexes of the coarse point cloud are used to group feature neighbors of F'. Next, MLPs are employed to embed geometry feature $\hat{f}^{xyz}$ and enhanced feature $\hat{f}^{F'}$ into the descriptive features $\tilde{f}^{xyz}$ and $\tilde{f}^{F'}$, respectively. Inspired by [15], we propose a multi-head point offset transformer (MPOT) to accurately predict coordinate offset features. Unlike the self-attention unit in [14], for each head, we take $\tilde{f}^{F'}$ as both the query vector (Q) and the key vector (K) and $\tilde{f}^{xyz}$ as the value vector (V) [15]. In this way, it is easier to regress coordinate offsets from geometry space (i.e., $\tilde{f}^{xyz}$). Finally, we apply several residual blocks [16] followed by FC layers to obtain ΔO.

2.3. Discriminator

As a supervisor, the discriminator regulates the generator to produce a refined dense point cloud. We propose an efficient multi-level discriminator, as shown in Fig. 1 (bottom). In particular, the HR point cloud first passes through four MLP layers. Then, we use max pooling for the output of the last MLP layer to get a global response vector, and the duplicated global response vector is concatenated with the output of the

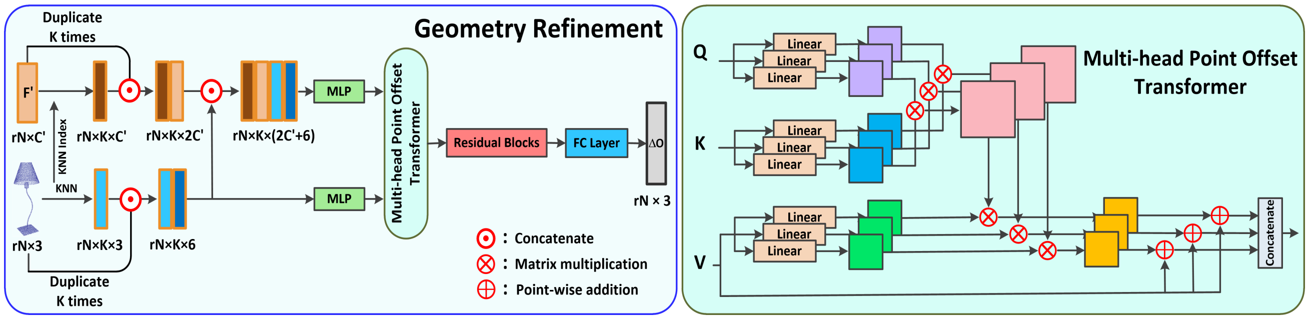

Fig. 2. Proposed geometry refinement (GR) module and multi-head point offset transformer unit.

previous four MLPs for feature enhancement. As such, the representative multi-level features (MF) are fused together, which is helpful to improve the discriminative results. Next, a self-attention unit is used to boost feature interactions. Finally, we obtain point-wise confidence values from the discriminator with several FC layers. Point-wise confidence values are more robust and allow more accurate prediction than a single confidence value obtained from the whole HR point cloud.

2.4. Loss functions

To train PU-Refiner, we used a compound loss function.

Adversarial loss. A least-squares adversarial loss [17] is used to train our generator G and discriminator D,

$$L_g(P_{HR}) = \frac{1}{2}(\uplus(\boldsymbol{D(P_{HR})-1}))^2, \tag{3}$$

$$L_d(P_{HR}, P_T) = \frac{1}{2}(\uplus(\boldsymbol{D(P_{HR})}))^2 + (\uplus(\boldsymbol{D(P_T)-1})^2), \tag{4}$$

where P_{HR} is the final HR output of generator G, $\mathbf{1}$ is a vector of ones, and P_T is the ground truth HR point cloud which has the same number of points as P_{HR}. $\boldsymbol{D(P_{HR})} \in \mathbb{R}^{rN\times 1}$ and $\boldsymbol{D(P_T)} \in \mathbb{R}^{rN\times 1}$ are the confidence value vectors obtained by feeding P_{HR} and P_T into D, respectively, and $\uplus$ denotes the average operator.

Reconstruction loss. We adopt the Chamfer distance (CD) [8] to evaluate the reconstruction error between P_{HR} and P_T.

$$L_{rec} = L_{cd}(P_{HR}, P_T) = \frac{1}{N}(\sum_{p\in P_{HR}} \min_{q\in P_T} \|p-q\|_2^2 + \sum_{q\in P_T} \min_{p\in P_{HR}} \|q-p\|_2^2), \tag{5}$$

where p and q are arbitrary points in P_{HR} and P_T, respectively. Moreover, we also adopt CD to force all the coarse upsampled point clouds P_{HR_c} to be closer to the underlying surface of P_T,

$$L_{coarse} = \sum_{i=1}^{M} w_i L_{cd}(P^i_{HR\_c}, P_T), \tag{6}$$

where $P^i_{HR\_c}$ is the i-th coarse upsampled point cloud, w_i is a weight, and M is the number of coarse upsampled point clouds.

Uniform loss. The uniform loss [11] is also adopted to ensure the uniformity of the generated HR point cloud,

$$L_{uni} = \sum_{i=1}^{T} U_{imblance}(S_i)\cdot U_{cluster}(S_i), \tag{7}$$

where S_i is a ball queried point cloud subset with T selected seed points from P_{HR} by farthest point sampling [18], $U_{imblance}(S_i)$ penalizes the consistency of the number of points of each S_i and $U_{cluster}(S_i)$ ensures that each point in S_i has a similar distance to its nearest neighbor.

Compound loss. To summarise, a compound loss is used to train PU-Refiner, i.e., a generator loss L_G and a discriminator loss L_D, where

$$L_G = w_g L_g(P_{HR}) + w_{rec}L_{rec} + L_{coarse} + w_{uni}L_{uni}, \tag{8}$$

$$L_D = w_d L_d(P_{HR}, P_T), \tag{9}$$

and w_g, w_{rec}, w_{uni} and w_d are weights.

3. EXPERIMENTAL RESULTS

3.1. Experimental setup

Dataset. We used the dataset in [11], which includes 147 3D models with various shapes. As in [11], we randomly picked 120 point cloud models for training and the rest for testing. By randomly cropping 200 patches from each training point cloud, we obtained 24000 patches for training.

Network details. For feature extraction, we set N and C to 256 and 480, respectively. The number of coarse upsampled features C' was 230. The head number of MPOT was set to three and the number of residual blocks (each one contains three linear layers with ReLU) was set to 5 in the GR module. Note that three GR modules were applied to generate the final P_{HR}. M was set to three and the weights w_1, w_2 and w_3 in Eq.(6) were equal to 700, 1300, 2600, respectively. Moreover, we empirically set the weights w_g, w_{rec}, w_{uni} and w_d in the compound loss to 1, 3900, 10, 1, respectively.

Training and Testing. A two time-scale update rule (TTUR) [19] was used to train PU-Refiner for 150 epochs with batch size 28 in the Tensorflow platform. The initial learning rate of the generator was 0.002 and that of the discriminator was 0.0002. During the test, we followed the

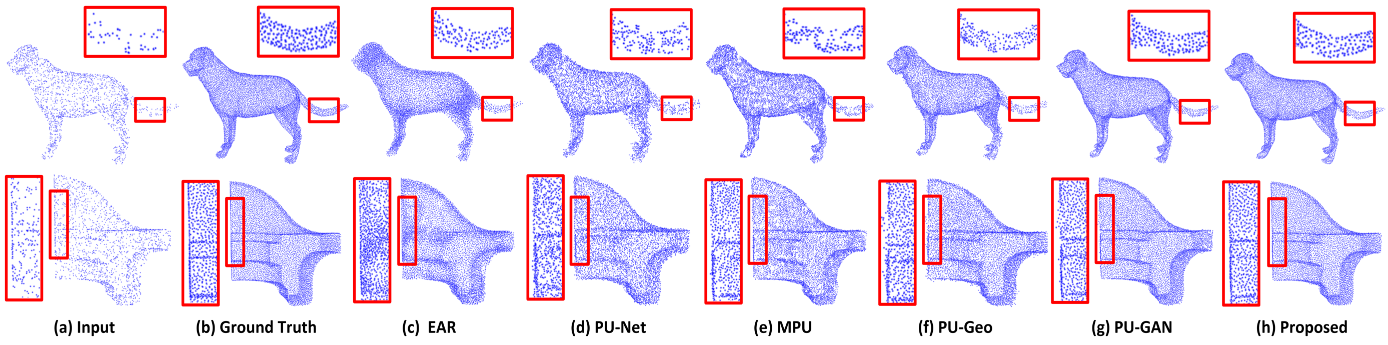

Fig. 3. Visual upsampling ($\times 4$) results of all methods for 2048 input points of the Dog (top) and Handicraft (bottom).

Table 1. Quantitative results for the test dataset.

Method	CD (10^{-3})	HD (10^{-3})	P2F (10^{-3})	Uniformity (10^{-3})
EAR [5]	0.87	10.33	7.79	28.62
PU-Net [8]	0.53	5.95	13.23	974.19
MPU [10]	0.44	5.32	3.07	13.15
PUGeo [12]	0.41	5.34	3.43	9.26
PU-GAN [11]	0.29	4.42	2.94	3.98
PU-Refiner	**0.27**	**3.87**	**2.48**	**3.17**

Table 2. Quantitative results for the unseen ModelNet40.

Method	CD (10^{-3})	HD (10^{-3})	P2F (10^{-3})	Uniformity (10^{-2})
EAR [5]	1.33	12.28	6.90	65.01
PU-Net [8]	1.19	11.92	7.10	33.58
MPU [10]	0.91	8.60	2.80	36.43
PUGeo [12]	0.58	7.09	2.83	32.93
PU-GAN [11]	**0.40**	6.66	2.40	22.86
PU-Refiner	0.44	**6.48**	**2.40**	**21.51**

patch fusion-based strategy [10] to acquire the final HR point clouds.

Evaluation metrics. We used four common evaluation metrics to quantitatively assess the upsampling performance: CD [8], Hausdorff distance (HD) [20], point-to-surface (P2F) distance [21] and Uniformity [11]. To compute these metrics, Monte Carlo random sampling was used to choose 2048 points from each test point cloud as LR point cloud P_{LR}. The ground truth P_T was obtained by sampling 8192 points from each test point cloud using Poisson disk sampling.

3.2. Comparison with other methods

We compared PU-Refiner with five state-of-the-art upsampling methods: one optimization-based EAR [5] and four learning-based methods (PU-Net [8], MPU [10], PU-GAN [11] and PUGeo [12]). The upsampling ratio r was set to 4.

Quantitative results. Table 1 lists the average upsampling accuracy for the test dataset. We can see that PU-Refiner achieved the best performance for all the evaluation metrics.

Qualitative results. Fig. 3 shows visual comparison results. We can observe that PU-Refiner restored the uniform, dense point clouds with natural geometry texture, while the other methods failed. It is worth noting that even when the input point cloud was very sparse (partially incomplete), PU-Refiner was able to infer a reasonable local representation (see the ***dog***'s tail and ***handicraft***'s edge in Fig. 3).

Robustness analysis. To prove the generalization ability of PU-Refiner, we used our trained model on randomly selected 20 point clouds from the unseen ModelNet40 [22] dataset. The objective results, which are reported in Table 2, show that PU-Refiner performed best except for the CD metric for which it was second best.

3.3. Ablation study

To further evaluate the contribution of each component of our network, we include an ablation study (Table 3).

Table 3. Ablation study.

Method	CD (10^{-3})	HD (10^{-3})	P2F (10^{-3})	Uniformity (10^{-3})
w/o GR	0.30	5.30	2.77	5.64
w/o MPOT	0.29	4.37	2.50	4.09
w/o MF	0.28	4.62	2.57	3.28
PU-Refiner	**0.27**	**3.87**	**2.48**	**3.17**

4. CONCLUSION

We proposed a novel point cloud upsampling network with adversarial learning. For the generator, a coarse feature expansion and a GR module are proposed to progressively generate a natural HR point cloud. The discriminator uses multi-level features to guide the generator to produce a faithful HR point cloud. Extensive experimental results demonstrated the superiority of PU-Refiner compared with five state-of-the-art methods. In the future, we will explore how to upsample other attributes of the point cloud, such as color.

5. ACKNOWLEDGMENT

This work has received funding from the National Natural Science Foundation of China under Grant 62172259 and 61871342, and the OPPO Research Fund.

6. REFERENCES

[1] X. Chen, H. Ma, J. Wan, B. Li and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Jul. 2017, pp. 6526–6534.

[2] W. Wu and C. Zhang, "Immersive 3d communication," in *Proceedings of the 22nd ACM international conference on Multimedia*, Nov. 2014, pp. 1229–1230.

[3] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 3–15, Jan. 2003.

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

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

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

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

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

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

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

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

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

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

[14] H. Zhang, I. J. Goodfellow, D. N. Metaxas, and A. Odena. "Self-attention generative adversarial networks," in *Int. Conf. on Machine Learning*, 2019, pp. 7354–7363.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[17] X. Mao, Q. Li, H. Xie, R. Y.K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *IEEE Int. Conf. on Computer Vision*, 2017, pp. 2794–2802.

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

[19] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.

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

[21] K. Low, "Linear least-squares optimization for point-to-plane icp surface registration," *Technical Report TR04-004*, University of North Carolina, 2004.

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