ARBITRARY POINT CLOUD UPSAMPLING VIA DUAL BACK-PROJECTION NETWORK

Zhi-Song Liu, Zijia Wang, Zhen Jia

Dell Research

ABSTRACT

Point clouds acquired from 3D sensors are usually sparse and noisy. Point cloud upsampling is an approach to increase the density of the point cloud so that detailed geometric information can be restored. In this paper, we propose a Dual Back-Projection network for point cloud upsampling (DBPnet). A Dual Back-Projection is formulated in an up-down-up manner for point cloud upsampling. It not only back projects feature residues but also coordinates residues so that the network better captures the point correlations in the feature and space domains, achieving lower reconstruction errors on both uniform and non-uniform sparse point clouds. Our proposed method is also generalizable for arbitrary upsampling tasks (e.g. $4\times$, $5.5\times$). Experimental results show that the proposed method achieves the lowest point set matching losses with respect to the benchmark. In addition, the success of our approach demonstrates that generative networks are not necessarily needed for non-uniform point clouds.

Index Terms— Back projection, point cloud processing, upsampling

1. INTRODUCTION

A point cloud is one of the popular ways to represent 3D surfaces because they capture high-frequency geometric information without requiring much memory. Recently, the study on point-cloud-based 3D analysis is getting more attention. A fundamental problem is, however, that the 3D sensors used in robotics or autonomous cars can only produce sparse, incomplete, or noisy point cloud data. For small objects or objects far from the camera, the collected point cloud cannot be directly used for analysis. The objective of this work is, given a sparse and noisy point cloud, to upsample it as dense ones by learning or non-learning approaches. Recently, learning based point cloud upsampling methods [1, 2, 3, 4] show that learning priors from data is a very promising direction. Although these deep-learning-based methods already achieved good progress on this task, there are still many issues to be solved: 1) developing a universal up-sampling model applicable to arbitrary upsampling tasks would be highly desirable, and 2) point cloud from real life is usually non-uniformly distributed and noisy. An ideal point cloud upsampling model



Fig. 1: $16 \times$ upsampling from noisy non-uniform points.

should be robust against these different types and levels of artifacts.

To address these two challenges, we introduce a novel, Dual Back-Projection network (DBPnet), specially designed for point cloud upsampling. Its originality is to tackle both coordinate and feature refinement using back projection. As shown in Figure 1, in contrast with existing methods [2, 3], our new method is able to upsample noisy non-uniform distributed point clouds into dense, uniformly distributed ones. More importantly, we train our proposed network for a large upsampling factor, e.g., $16\times$, and downsample it to achieve arbitrary upsampling. We claim the following original contributions 1) we introduce a novel, dual back-projection process to learn point correlations for upsampling tasks. Information from both feature and coordinate spaces are updated using an up-down-up process. 2) Position-aware attention is embedded into the back-projection process to learn non-local point correlations. With the help of this positional embedding, the attention module can incorporate both position and feature correlations to capture global representation. 3) A resampling stage is introduced as a second-stage point cloud refinement where dense points are downsampled to sparse point sets. Hence, each generated point gets a chance to attend to information from the neighborhoods of the original sparse point cloud. This stage acts as a self-supervision where local and global information are combined for coordinate estimation.

2. RELATED WORK

Point cloud Upsampling Early work in the Computer Graphics field used optimization methods to solve point cloud upsampling problems [5, 6, 7, 8]. For example, [5] assumed that the new inserted points should lay on a smooth surface

corresponding email: zhisong.liu@dell.com



Fig. 2: Overview of our proposed DBPnet model. It includes two stages of $\alpha \times$ upsampling processes. The upsampling block contains feature-based back-projection for feature expansion. The initial upsampled points are resampled by kNN to obtain α subsets of estimated points. Along with the input points, the second stage of upsampling process works as a coordinate-based back-projection to update the residual distance between input points and neighborhood points.

so that they can be optimized by the moving least squares. To ensure sharp reconstruction, Huang et al. [8] proposed a progressive method called EAR for edge-aware resampling of points. Deep learning techniques recently spread in a variety of fields, including 3D processing. The first work on point cloud upsampling was PUNet [1]. The authors proposed a convolutional neural network to directly extract features from point sets. To upsample the points, the learned features were expanded and separately mapped to a subset of the point cloud. To avoid point clustering, repulsion loss was used to balance the point distances. To explicitly upsample the point cloud, Wang et al. [2] proposed to use 1d code to attach features to different locations. By stacking multiple $2\times$ upsampling blocks, their method can progressively upsample the point cloud to the desired number of points. As mentioned in [1], upsampling points is equivalent to upsampling features. Li et al. [3] resolve it by using a generative adversarial network to learn point distribution from the latent space. Most recently, SSPU [9] utilizes multi-view rendering to supervise the dense point cloud generation. SAPCU [10] seeks to find rich points from a surface learned from neural implicit functions. PU-GCN [11] designs a graphic convolution network to better encode coordinate information for upsampling.

3. METHOD

Given a sparse non-uniform 3D point cloud $P = p_i_{i=1}^N$, our network aims to generate dense point cloud $Q = q_i_{i=1}^{\alpha N}$, where α is the upsampling factor. The generated point cloud should follow the same underlying surface as the target object and be uniformly distributed. As shown in Figure 2 and Figure 3, our proposed Dual Back-Projection network (DBPnet) consists of three components: 1) Dense feature extraction, 2) Back-Projection upsampling, and 3) Coordinate estimation to achieve a two-step upsampling process. The overall process is a coordinate-based back-projection. The initial upsampled point sets are downsampled again to match with the input point sets, and the residual distances computed using the k nearest neighbors are back-projected for refinement. The refined and the initial point sets are combined together for the final coordinate estimation. Within the upsampling block, the feature-based back-projection iteratively updates point features for estimation. Combining coordinate and feature-based back-projection, we achieve significant improvements over previous works.

Motivation The overall upsampling network is built upon the coordinate-based back-projection, which can be described using the following equation:

$$Q^* = f\left(\mathcal{S}\{g[f(P)] \setminus P\}\right) \cap f(P) \tag{1}$$

where $f(\cdot)$ is the upsampling operation, $g(\cdot)$ the resampling operation, \setminus is the set-difference operation, S is the dimension switch operation, $P \in \mathbb{R}^{N \times 3}$ is the input sparse points and $Q^* \in \mathbb{R}^{\alpha N \times 3}$ is the upsampled points. We omitted feature extraction and coordinate reconstruction for simplicity.

Following the process of back projection, we first upsample the input points and then resample them to the same dimension as the input. The resampling process is different from the downsampling process in the sense that we do not sample only one sparse point subset but rather gather the points into a set of subgroups. For example, we can find Npoints as cluster centroids and then use k Nearest Neighborhood (kNN) to group neighboring points into these clusters. As shown in Figure 2, the upsampled points P can be resampled into α sparse point subsets. Together with the input point set, we form a pool of sparse point sets that can be used for neighborhood feature extraction. Hence we learn the relative complement of the set P with respect to the set Q. As mentioned in PUNet [1], features and points are interchangeable, so feature expansion is equivalent to expanding the number of points. After resampling, we thus expand features and then switch the dimensions of features and points. For $\alpha \times$ upsampling, we use $\alpha + 1$ features (one point from the input set plus α resampled points for reconstruction. There are three merits of using coordinate-based back-projection: 1) it forms a global loop of self-supervision where the target points should locate around the input points, achieving kNN interpolation, 2) it avoids the expensive computation of kNN searches and feature extraction in every layer, and 3) by using resampling, it ensures each point collects both local and global information for estimation.

Back-Projection Upsampling. The key component of the upsampling block is the use of back-projection. We call it feature-based back-projection because it is done in the feature domain. As shown in Figure 4, it includes three components, namely upsampling, downsampling, and self-attention. The basic procedure is shown in Figure 4A. The input sparse point features are firstly upsampled to the dense features, and then the dense features are downsampled and subtracted from the



Fig. 3: The proposed DBPnet model, including Deep feature extraction, Back-Projection up-sampling and Coordinate estimation.



Fig. 4: Back-Projection upsampling. It includes up-sampling, down-sampling and self-attention blocks. It iteratively updown-up upsamples the point features.

input features to calculate residues. The residual information is further upsampled for updating the dense point features.

Position-aware self-attention. We strive for a better feature representation for point cloud upsampling. To achieve this goal, attention is a good choice because of its nonlocal learning ability. It has been widely used in many fields [12, 13, 14, 15], including 3D point cloud [16, 17, 18, 19]. What distinguishes our position-aware self-attention from other works is that we introduce the position embedding technique for attention computation. As introduced in [20], adding position codes in the attention computation can incorporate the order information into the model. Though the point cloud is unordered data that is invariant to permutation, the relative position defines the structure of the underlying 3D surface. It is necessary to keep the position information, that is, coordinate data for feature extraction. To this end, we add positional en*coding* to the point feature to retain positional information. It can also be understood as a Laplacian theory L=D-A, where L is the Laplacian matrix, D is the degree matrix and A is the adjacency matrix. Position-aware attention is equivalent to replacing D by positional encoding and replacing A by point features. Their combination forms the matrix representation of a graph. By using deep learning, both degree and adjacency matrices can be learned end-to-end. Mathematically, we can describe the process as follows,

$$\mathbf{y} = \omega \{ softmax \left(\theta(\mathbf{z}) \, \phi(\mathbf{z}^T) \right) g(\mathbf{z}) \} + x \tag{2}$$

where $z = x + E_{pos}$ is the position-aware encoding. It is the sum of point feature x and *positional encoding* E_{pos} . For attention computation, the input data are separately transformed by MLPs and multiplied for autocorrelation. We also add a skip-connection to further improve the learning ability, where ω is the weighting process to finetune the amount of attention values.



Fig. 5: $4 \times$ upsampling results from 2048 input points (uniformly and non-uniformly) and reconstructed mesh.

Loss functions To keep data fidelity, the full loss function is defined as $\mathcal{L} = \mathcal{L}_{CD} + \lambda \mathcal{L}_{uni}$, where λ is the weighting factor used to balance the Chamfer loss (\mathcal{L}_{CD}) [2, 3] with respect to the uniform loss (\mathcal{L}_{uni}) [1].

4. EXPERIMENTS

Training dataset and Implementation details. For training data, we used the same database as PUNet [1], consisting of 60 different 3D models from the Visionary repository [21]. For each 3D model, we used farthest point sampling to select seeds. Then we employed the Poisson disk sampling [22] to generate target dense patches Q. Different from [4, 2], we used random sampling to generate input sparse patches *P* from *Q*. The number of input and target patches is *N* and αN with N=256 and α =16. Therefore, our training task is to map the non-uniformly distributed sparse P to the uniformly distributed dense points Q. Note that we only trained $16 \times$ upsampling model. The reason is that if a $16 \times$ result can represent the underlying surface well, a simple uniform downsampling process can faithfully represent the same 3D structure. Hence, it is not necessary to train the same model for smaller upsampling scenarios. We thus used the furthest sampling approach to downsample the $16 \times$ results, in order to achieve $2\times$, $4\times$ and $8\times$ results, respectively. For testing data, we used the same data as PUGAN [3], consisting of 27 different 3D models with a wide variety of simple and complex objects. From each model, 2048 random points were extracted to form the input sparse point cloud. To better demonstrate the robustness of our method, we tested four different scenarios: clean uniformly sampled points, clean non-uniformly sampled points, noisy non-uniformly sampled points and real points collected from LIDAR. Our comparison includes EAR [8], PUNet [1], MPU [2] and PUGAN [3].

Table 1: Quantitative evaluation of state-of-the-art point clouds upsampling approaches for scales $2\times$, $4\times$, $8\times$ and $16\times$. Red indicates the best.

No. o	of sparse point clouds	2048							
Sampling		Uniform				Non-uniform			
Eval. (10 ⁻³)		CD	HD	P2F	Uniformity	CD	HD	P2F	Uniformity
2x	EAR [8]	0.471	2.912	2.896	1.749	0.705	7.027	2.481	3.748
	PUNet [1]	0.822	4.089	4.974	1.579	1.009	7.331	9.776	2.296
	PUGAN [3]	0.433	4.133	1.550	1.189	0.499	4.420	3.801	2.001
	MPU [2]	0.446	2.096	1.546	1.937	0.712	6.234	1.968	3.234
	PU-GCN [11]	0.355	1.967	1.533	1.870	0.660	5.742	1.955	3.077
	Ours	0.409	2.904	1.531	1.154	0.426	3.762	1.966	1.166
4x	EAR [8]	0.372	4.027	5.370	1.160	0.577	8.356	5.430	2.017
	PUNet [1]	0.525	4.601	5.900	1.123	0.544	6.072	6.841	0.773
	PUGAN [3]	0.224	3.973	1.907	1.014	0.243	4.394	2.341	0.892
	MPU [2]	0.251	1.624	1.669	1.086	0.528	6.243	2.112	1.685
	PU-GCN [11]	0.220	1.611	1.581	0.966	0.333	4.303	2.115	0.829
	Ours	0.120	1.606	1.562	0.807	0.238	3.511	2.009	0.760
8x	EAR [8]	-	-	-	-	-	-	-	-
	PUNet [1]	2.527	7.078	4.261	0.958	2.438	6.971	4.441	1.039
	PUGAN [3]	-	-	-	-	-	-	-	-
	MPU [2]	0.160	3.821	1.665	0.732	1.676	6.084	2.727	0.859
	PU-GCN [11]	0.162	3.833	1.650	0.741	1.700	6.079	2.790	0.891
	Ours	0.081	2.338	1.414	0.539	0.142	3.727	2.447	0.528
16x	EAR [8]	-	-	-	-	-	-	-	-
	PUNet [1]	0.460	4.561	5.891	0.658	0.480	6.143	4.112	0.804
	PUGAN [3]	0.093	4.552	2.578	0.491	0.107	4.594	3.030	0.430
	MPU [2]	0.138	1.914	1.804	0.489	0.294	6.182	2.472	0.519
	PU-GCN [11]	0.106	1.845	1.776	0.479	0.288	5.887	2.456	0.495
	Ours	0.047	1.692	1.578	0.440	0.097	3.519	2.170	0.421

Table 2: Quantitative comparison of the network using with or without proposed key components. *Baseline* is the vanilla simple network. *Full pipeline* is the proposed network.

$Metric(10^{-3})$	CD	HD	P2F	Uniformity
Feature BP	0.256	6.113	3.234	0.667
Coordinate BP	0.137	4.122	3.001	0.486
Position Embeddings	0.238	5.879	3.112	0.640
Baseline	0.322	6.547	3.556	0.714
Full pipeline	0.097	3.519	2.170	0.427

For evaluation, we use Chamfer distance (CD), Hausdorff Distance (HD), Point-to-Surface distance (P2F) mean value and uniformity value.

Comparison with State-of-the-art Methods. Table 1 summarizes the quantitative comparison results. We have two sets of input data: uniform (input points are evenly distributed) and non-uniform (input points are randomly distributed). Our DBPnet achieves the best performance with the lowest values consistently for different evaluation metrics. Specifically, it can be observed that our new method achieves the lowest P2F and Uniformity values on every task, indicating that it achieves the generation of uniform, dense point cloud that remain close to the underlying surface. For visual comparison (Figure 5), we applied surface reconstruction to the upsampled point sets by using PCA normal estimation (number of neighborhood=20) [23] and screened Poisson reconstruction (depth=9) [24]. Results show that our method is able to fill holes and robustly generate uniformly-sampled point sets, while former methods, such as EAR, PUNet and MPU, adequately handle uniform point sets but fail on non-uniformly sampled regions.



Fig. 6: $16 \times$ upsampling results on real 3D points. (a) KITTI data [25] of the driving screen and (b) Paris-Lille-3D data [26] of the city view of Paris.

Ablation studies. As introduced in Section 2, the coordinatebased back-projection works as a second-stage refinement. To make a comparison, we define the *baseline* as our network without the coordinate-based back-projection and featurebased back-projection blocks. Next, we define *Feature BP* as the baseline network using feature-based back-projection for upsampling; *Coordinate BP* as the baseline network using coordinate-based back-projection for upsampling; *Position Embedding* as the baseline network using position-aware attention and *full pipeline* as the complete proposed network. Table 2 shows the evaluation results. The full pipeline performs the best. Removing any key components can reduce the performance, hence it indicates the contributions of each component.

Real World Point cloud. To evaluate our model on real scans, we tested it on the KITTI [25] dataset and the Paris-Lille-3D [26] dataset. Figure 6 It can be found that using our method can upsample the point sets to better represent objects, like cars and pedestrians.

5. CONCLUSION

In this work, we introduced a Dual Back-Projection network (DBPnet), specially designed for point cloud upsampling. Combining both Coordinate and Feature-based backprojections, this model is able to reveal detailed geometric structures from sparse and noisy input point clouds. We also proposed a position-aware attention mechanism, to complement the non-local representation with positional information. Overall, we formed a network enabling both global coordinate refinement and local feature refinement. The such an adaptive patch-based network learns point distributions and is able to handle both uniformly and non-uniformly distributed point sets. Extensive experiments and studies demonstrated that the proposed solution outperforms state-of-the-art methods in both quantitative and qualitative comparisons.

6. REFERENCES

- L. Yu, X. Li, C. Fu, D. Cohen-Or, and P. Heng, "Pu-net: Point cloud upsampling network," *Proc. CVPR*, pp. 2790–2799, jun 2018.
- [2] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3d point set upsampling," *Proc. CVPR*, pp. 5951–5960, jun 2019.
- [3] R. Li, X. Li, C. Fu, D. Cohen-Or, and P. Heng, "Pu-gan: A point cloud upsampling adversarial network," *Proc. ICCV*, pp. 7202–7211, nov 2019.
- [4] Yue Qian, J. Hou, S. Kwong, and Y. He, "Pugeo-net: A geometry-centric network for 3d point cloud upsampling," *ArXiv*, vol. abs/2002.10277, 2020.
- [5] G. Singh, R. H. Lau, and Y. L. Chrysanthou, "Guest editors' introduction: Special section on acm vrst 2005," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 01, pp. 3–4, jan 2007.
- [6] Yaron L., Daniel Cohen-Or, David Levin, and Hillel Tal-Ezer, "Parameterization-free projection for geometry reconstruction," in ACM SIGGRAPH 2007 Papers, New York, NY, USA, 2007, SIGGRAPH '07, p. 22–es.
- [7] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–7, 2009.
- [8] Hui Huang, Shihao Wu, Minglun Gong, Daniel Cohen-Or, Uri Ascher, and Hao (Richard) Zhang, "Edge-aware point set resampling," ACM Trans. Graph., vol. 32, no. 1, 2013.
- [9] Yifan Zhao, Le Hui, and Jin Xie, SSPU-Net: Self-Supervised Point Cloud Upsampling via Differentiable Rendering, p. 2214–2223, 2021.
- [10] Zhao Wenbo, Liu Xianming, Zhong Zhiwei, Jian Junjun, Gao Wei, Li Ge, and Ji Xiangyang, "Self-supervised arbitrary-scale point clouds upsampling via implicit neural representation," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [11] Guocheng Qian, Abdulellah Abualshour, Guohao Li, Ali Thabet, and Bernard Ghanem, "Pu-gcn: Point cloud upsampling using graph convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 11683–11692.
- [12] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), June 2018.
- [13] Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen, "Axial-deeplab: Stand-alone axial-attention for panoptic segmentation," in *European Conference on Computer Vision (ECCV)*, 2020.

- [14] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," in *NeurIPS*, Cambridge, MA, USA, 2015, p. 577–585, MIT Press.
- [15] Yonghui Wu, Mike Schuster, and et al. Zhifeng Chen, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016.
- [16] Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu, "Attentional shapecontextnet for point cloud recognition," in *Proc. CVPR*, June 2018.
- [17] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua E Siegel, and Sanjay E Sarma, "Pointgrow: Autoregressively learned point cloud generation with self-attention," in *Winter Conference on Applications of Computer Vision*, 2020.
- [18] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian, "Modeling point clouds with self-attention and gumbel subset sampling," in *Proc. CVPR*, 2019, pp. 3323–3332.
- [19] Wenxiao Zhang and Chunxia Xiao, "Pcan: 3d attention map learning using contextual information for point cloud based retrieval," in *Proc. CVPR*, June 2019.
- [20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, vol. 30, pp. 5998–6008.
- [21] "Visionair," http://visionair.ge.imati.cnr.it/ ontologies/shapes/link.jsp, Accessed: 14-November-2017.
- [22] M. Corsini, P. Cignoni, and R. Scopigno, "Efficient and flexible sampling with blue noise properties of triangular meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 914–924, 2012.
- [23] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle, "Surface reconstruction from unorganized points," in *Proceedings of the 19th Annual Conference* on Computer Graphics and Interactive Techniques, New York, NY, USA, 1992, SIGGRAPH '92, p. 71–78.
- [24] Michael Kazhdan and Hugues Hoppe, "Screened poisson surface reconstruction," ACM Trans. Graph., vol. 32, no. 3, Jul 2013.
- [25] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.
- [26] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette, "Paris-lille-3d: A large and high-quality groundtruth urban point cloud dataset for automatic segmentation and classification," *The International Journal of Robotics Research*, vol. 37, no. 6, pp. 545–557, 2018.