POINT CLOUD DEFORMATION FOR SINGLE IMAGE 3D RECONSTRUCTION

Seonghwa Choi, Anh-Duc Nguyen, Jinwoo Kim, Sewoong Ahn, and Sanghoon Lee

Yonsei University

ABSTRACT

We propose an approach to reconstruct a precise and dense 3d point cloud from a single image. Previous works employed reconstruction to complexity 3D shape or directly regression location from image. However, while the former requires overhead construction of 3D shape or is inefficient because of high computing cost, the latter does not scale well as the number of trainable parameters depends on the number of output points. In this paper, we explore a method to infer a point cloud representation given an input image. We extract shape information from an input image, and then we embed the two kinds of shape information into the point cloud: point-specific and global shape features. After that, we deform a randomly generated point cloud to the final representation based on the embedded point cloud feature. Our method does not require overhead construction, and is efficient and scalable because the number of trainable parameters is independent of the point cloud size, which is the first work to be able to do so according to our knowledge. Thorough experimental results suggest that our proposed method outperforms with other state-of-the-art methods in dense and precise point cloud generation.

Index Terms— 3D reconstruction, point cloud processing, neural network, deep learning

1. INTRODUCTION

Deducing 3D shapes from static 2D images is a fundamental yet very challenging task in computer vision because going from 2D to 3D is equivalent to reconstructing the kernel space of the projection from singular, which is impossible to perform exactly. Traditionally, the solution is approximated by Structure-from-Motion [1] or Shape-from-Shading [2]. However, while the former requires a sequence of multiple images of the same scene from different perspectives and an excellent image matching algorithm, the latter requires prior knowledge of the light sources as well as albedo maps, which makes it suitable mainly for studio environment. Learning-based approach, which can learn the shape priors from data, has also been considered. A notable work is Saxena *et al.* [3] which constructed a Markov random field to model the relationship between image depth and various visual cues.

This work was supported by Samsung Research Funding Center of Samsung Electronics under Project Number SRFC-IT1702-08

Recently, thanks to the advancement of deep learning, neural networks (NNs) have been applied to many fields [4–11], including 3D reconstruction. These methods can reconstruct an object from a single image by taking advantage of the symmetry of the object and the phenomenal ability to extract statistics from images of NNs. The obtained results are usually much more impressive than traditional single image 3D reconstruction methods. Wu *et al.* [12] employed a conditional deep belief network to model volumetric 3D shapes. Yan *et al.* [13] introduced an encoder-decoder network regularized by a perspective loss to predict 3D volumetric shapes from 2D images. In [14], the authors predicted 3D voxel representations from given images but the voxel representation is not computationally friendly even with the aid of GPUs. For mesh representation, Wang *et al.* [15] gradually deformed a predefined mesh according to the derived features from the input image by using graph convolution, but mesh representation requires overhead construction and may result in computing redundancy as masking is needed. Also, it computes local of points, so a point-to-point distance loss is not suitable. The closest works to ours are perhaps those in [4,16]. Fan *et al.* [4] proposed a straight-forward network to directly map an input image to its point cloud representation using two losses to directly relate the ground truth point clouds and estimations. Insafutdino *et al.* [16] similarly predicted a 3D point cloud representation for a given image, but the loss function is designed for the 2D projections. A disadvantage of these methods is that the number of trainable parameters are proportional to the number of points in the output cloud, thus there is always an upper bound for the point cloud size.

In this paper, we propose Pix2Pc, a method which taking a single image as input and outputs a 3D point cloud representation of the object. Inspired by [15], we first distill shape features from an input image by a convolutional neural network (CNN), and then utilize the extracted shape information to deform a randomly initialized point cloud into the shape of the given object. To embed the shape information from an image into point cloud, we consider two types of feature: one is based on projection as done in [15] and the other is inspired by an image style transfer algorithm [5]. The shape information is obtained from various layers, thus the prediction can be based on coarse-to-fine features. Unlike [15], our network simply consists of simple 2D convolution and dot product,

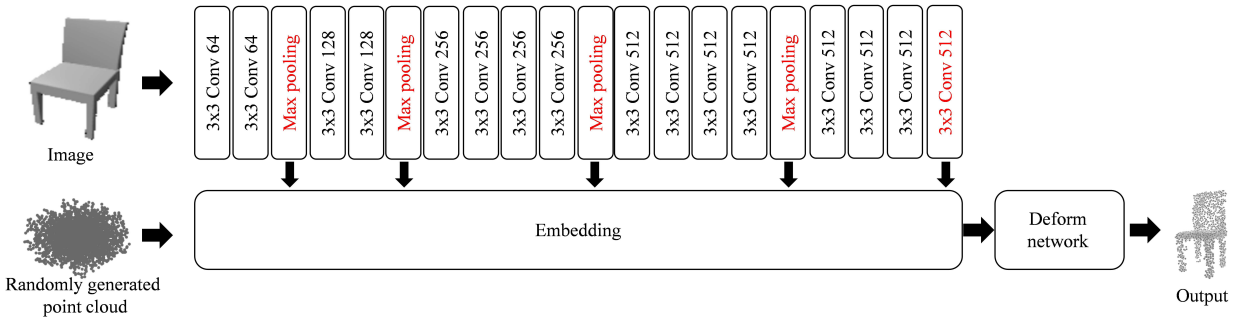

Fig. 1. Overview of the proposed framework.

which are efficient and optimized in most contemporary deep learning libraries. Also, as the number of trainable parameters is independent of the number of points, our method obviously can scale much better compared to prior arts. In fact, with a simple hack, we can produce as many points in the output cloud as we desire. Thorough experiments demonstrate that our method can infer precise shapes from images and outperforms with previous works in 3D point cloud generation.

2. PIX2PC

2.1. Overview

Our framework is described in Fig. 1. we randomly generate a point cloud, use a CNN to derive 2D feature from a given image, and then transfer the feature maps information to feature vector for every point. Prediction of the precise and dense point cloud is carried out by processing the extracted point cloud features independently for each point via a multilayer perceptron (MLP).

2.2. Shape information extraction

To obtain a point cloud for a given image, we first need to exact features from the 2D input image. We use a VGG-19 [17], which consists of a series of convolutions and downsamplings. We use this architecture to distill the object's shape information. After that, we use the shape information to calculate the feature vector for each point. We propose two ways for extracting point features, which are described in 2.3.

2.3. Shape feature embedding

In this section, we consider two types of feature: one is processed by projection and the other is processed by the adaptive instance normalization (AdaIN) [5] method. Each feature represents two types of feature The procedures to extract the features are described in 2.3.1 and 2.3.2.

2.3.1. Projection

Similar to [15], to embed multi-resolution information into the feature vectors, we utilize the 2D feature maps preceding to the pooling layers which project the point cloud onto 2D features. Let X_i be the feature map from the i-th max-pooling layer as shown in Fig. 1, and p is a point from the initial cloud. The feature vector $\hat{y}_i$ is defined as

$$\hat{y}_i^{proj} = project(p, X_i) \tag{1}$$

where $project(x, y)$ projects x onto y. To obtain shape features at various resolutions, we concatenated a total of five feature vectors by projecting the point cloud onto the five feature maps from max-pooling.

2.3.2. Adaptive instance normalization

In addition to projecting every point onto the feature maps and obtain a feature vector for each point, we can consider feature extraction from a more global point of view. Concretely, we can encode the shape statistics of the object and then transfer it to the point cloud so that the point cloud can be deformed into the expected shape. We first obtain point cloud features by applying an MLP to the initial point cloud. The MLP consists of five blocks, each of which is composed of two fully connected layers having the same number of nodes. Inspired by the way AdaIN works, we propose to perform the same normalization to the point cloud features using the extracted image features. Let y_i be the output feature vector of the point cloud from block i-th of the MLP. The AdaIN feature is defined as

$$\hat{y}_i^{adain} = \sigma_{X_i} \frac{y_i - \mu_{y_i}}{\sigma_{y_i}} + \sigma_{X_i} \tag{2}$$

where μ_{X_i} and σ_{X_i} are the mean and standard deviation of X_i taken over the spatial locations, and μ_{y_i} and σ_{y_i} are the mean and standard deviation of y_i. Here, we ask the network to encode the shape statistics in the feature maps, and through

AdaIN we can transfer the statistics to the point cloud features, which will guide the deformation network how to deform the point cloud. We concatenate five projection features and AdaIN features with coordinate of point

$$\hat{y}_p = \left[[\hat{y}_i^{proj}]_{i=1}^5, [\hat{y}_i^{adain}]_{i=1}^5, p \right] \quad (3)$$

where $[\cdot]_{i=1}^5$ is the concatenation of feature maps extracted at 5 resolutions. After concatenation, we use it as input to a deformation network.

2.4. Deformation network

After obtaining a feature vector for every point, we use a deform network to obtain the final point cloud. Our proposed network takes feature vector of point cloud as an input and deforms each points using MLP. To measure the discrepancy between the estimated point cloud and ground truth, we use Chamfer distance (CD), which measures the sum of the distances between each point in a cloud to the nearest point in the other and vice versa, to regress the coordinates of points. Since our network is designed to process each point in the cloud independently, Chamfer distance is a natural choice as it concerns only the distance between two points and disregard their neighbors. This is unlike [15] in which Chamfer distance is not really a suitable choice given the fact that mesh representation should be deformed locally. The Chamfer distance compute the distance of each point to the other set. The Chamfer distance loss $L_{chamfer}$ is defined as:

$$L_{chamfer} = \sum_p \min_q ||p-q||_2^2 + \sum_q \min_p ||p-q||_2^2 \quad (4)$$

where p is the point of estimated point cloud and q is the point of the ground truth point cloud.

2.5. Implementation details

When we generate a random point cloud naively, its projection does not densely cover all the area of the image. In order to solve this problem, we generate a point cloud in two steps. First, we randomly generate 2D points to be uniform distribution or normal distribution and 3D Z coordinate to cover the entire image. Second, the points on the 3D space are calculated. Also, we find that uniform distribution works better than normal, and so we stick with that choice for all the experiments. Also, we generate a random point cloud for every iteration. When we obtain point cloud feature by MLP, the numbers of nodes of the blocks are 64, 128, 256, 512, and 512. Also, our deformation network consists of MLP of 4 layers. The first three layers have dimensions of 512, 256, and 128 and the last layer produces the deformed point cloud. For the AdaIN and full models, due to the long training time, we actually bootstrap the network in the first 15 epochs by using only projection and then for the last 5 epochs we continue training by adding or replacing with the AdaIN branch. We randomly generated 5k points, and our proposed model takes the generated point cloud and 224x224 grayscale image as input. The batch size is 1. We trained for a total of 20 epochs. The learning rate was 1e-4 and dropped to 3e-5 after 15 epochs. We use the ADAM optimizer with weight decay 1e-5. The model is implemented in Tensorflow [18].

3. EXPERIMENTAL RESULT

3.1. Database Setup

We used the dataset created in [19]. The dataset is based on ShapeNet [20]. The images in this database were rendered using various viewpoints from the model. We chose all categorys in ShapeNet. We split training and test sets by 8:2 on the object basis.

3.2. Comparison to state of the art

To compare with state-of-the-art and our proposed method, we chose PSG [4], 3D-R2N2 [19] and GAL [21]. PSG and GAL are a method for reconstructing a point cloud from a single image. 3D-R2N2 is a method for estimating a voxel from a multi view image. We evaluate quantity reconstruction result with IoU. To compute IoU of two point clouds, we convert our approximate point cloud to a 32^3 volumetric using the voxelization method in [16]. In Table 1, we demonstrate the IoU of our network compared to 3D-R2N2, PSG and GAL. Even though Px2Pc is not directly trained by IoU, or does not learn how to voxelize like the competing methods, our IoU is higher than those of other methods. Also, in Table 1, we present the CD of our network compared to 3D-R2N2 and PSG. As can be seen from the Table, our CD is smaller than those of other methods.

3.3. Qualitative result

Fig. 2 shows the results of the proposed method versus those of PSG. As shown in the figure, our proposed model generates point clouds of more precise shapes than PSG. Our method takes an image and a randomly generated point cloud to reconstruction the target point cloud, so it generates an arbitrary-sized point cloud by using generated point cloud differently. However, PSG takes only one image to generate a point cloud. Therefore, it generates a fixed point cloud, and it cannot concatenate to make a scalable point cloud. To show off the scalability of our method, Fig. 3 presents more results of scalable point cloud. As can be seen from the figure, the number of points varies from 5k, 15k and 50k, and the point clouds are voxelized having 256 resolution. These results show that our proposed model is arbitrarily scalable point cloud according to user's desire.

Table 1. Comparison to other methods. The best, second best and third best results are indicated by **bold**. "↑" indicates higher is better. "↓" indicates the opposition.

Category		table	car	chair	plane	couch	firearm	lamp	watercraft	bench	speaker	cabinet	monitor	cellphone	mean
CD ↓	3D-R2N2	1.116	0.845	1.432	0.895	1.135	0.993	4.009	1.215	1.891	1.507	0.735	1.707	1.137	1.445
	PSG	0.517	0.333	0.645	0.430	0.549	0.423	1.196	0.633	0.629	0.756	0.439	0.722	0.438	0.593
	Pix2Pc	**0.314**	**0.220**	**0.333**	**0.129**	**0.289**	**0.128**	**0.560**	**0.300**	**0.211**	**0.471**	**0.310**	**0.275**	**0.181**	**0.286**
IOU ↑	3D-R2N2	0.580	**0.836**	0.550	0.561	0.706	0.600	0.421	0.610	0.527	0.717	0.772	0.565	0.754	0.631
	PSG	0.606	0.831	0.544	0.601	0.708	0.604	0.462	0.611	0.550	**0.737**	0.771	0.552	0.746	0.640
	GAL	**0.714**	0.737	**0.700**	0.685	0.739	0.715	**0.670**	0.675	0.709	0.698	**0.772**	**0.804**	0.773	0.712
	Pix2Pc	0.676	0.820	0.693	**0.779**	**0.784**	**0.757**	0.552	**0.769**	**0.739**	0.713	0.769	0.764	**0.846**	**0.743**

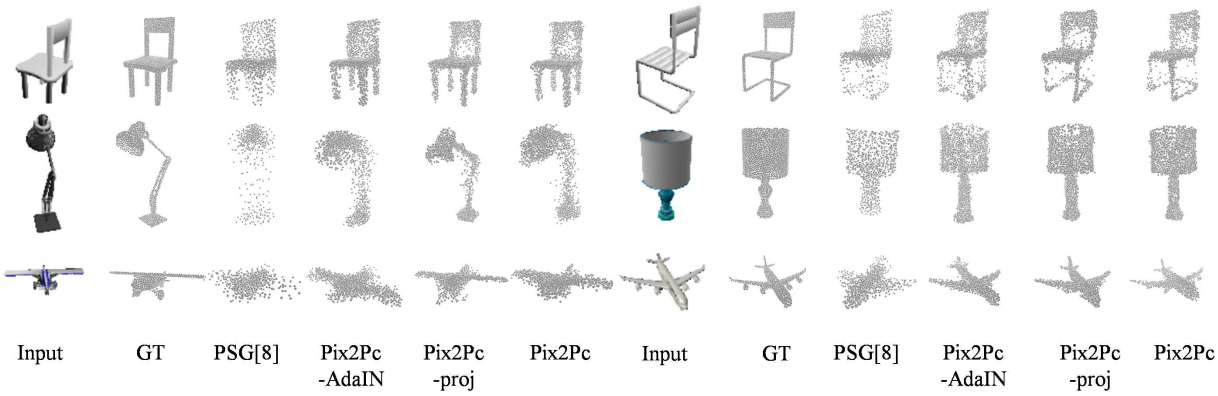

Fig. 2. Qualitative results of different methods.

Table 2. Ablation study results

Category	chair	plane	lamp	mean
Pix2Pc-proj	0.683	0.680	0.536	0.633
Pix2px-AdaIN	0.640	0.666	0.483	0.596
Pix2Px	0.693	0.779	0.552	0.675

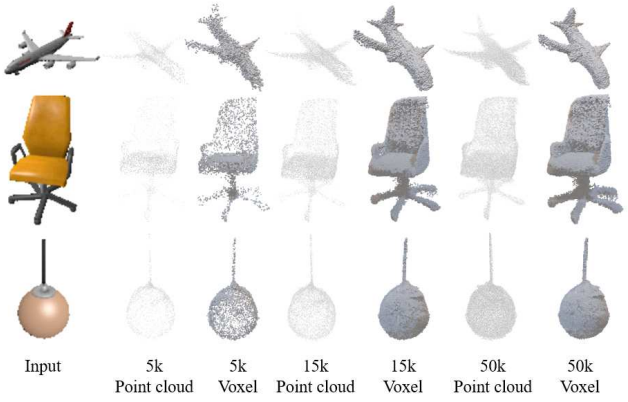

Fig. 3. Scalability of the proposed method.

3.4. Ablation study

We performed an ablation study to analyze the role of each component. Table 2 demonstrates the IoU of each model. As shown in Table 2, the IoU of Pix2Pc without AdaIN or projection declines. The main reason is the lack of features that the projection method and the AdaIN method provide. As can be seen in Fig. 2, Pix2Pc-proj produces better local shapes. e.g. the leg of the chair. Conversely, Pix2Pc-AdaIN produces a global shape well, e.g. the body of the plane. Therefore, the result of Pix2Pc combining the two methods shows the point cloud generated considering the global and local shapes

4. CONCLUSION

We have proposed a network that extracts shape information from a single image and deforms the randomly generated point clouds according to the distilled information. To transfer the shape information from the image feature maps to the initial point cloud, we process two types of feature: one is based on projection to squeeze a feature vector for each point, and another is inspired by AdaIN in image style transfer. After that, the obtained features are concatenated and fed to a deformation network which is a straight-forward MLP. Experimental results show that the model estimates a precise shape from randomly generated point cloud with the outperform quality for a given image.

5. REFERENCES

[1] Johannes L Schonberger and Jan-Michael Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4104–4113. 1

[2] Emmanuel Prados and Olivier Faugeras, *Shape from shading*, pp. 375–388, Springer, 2006. 1

[3] Ashutosh Saxena, Min Sun, and Andrew Y Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 824–840, 2009. 1

[4] Haoqiang Fan, Hao Su, and Leonidas Guibas, "A point set generation network for 3d object reconstruction from a single image," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 2463–2471, IEEE. 1, 3

[5] Xun Huang and Serge Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," *CoRR, abs/1703.06868*, vol. 2, pp. 3, 2017. 1, 2

[6] Jongyoo Kim, Anh-Duc Nguyen, and Sanghoon Lee, "Deep cnn-based blind image quality predictor," *IEEE transactions on neural networks and learning systems*, , no. 99, pp. 1–14, 2018. 1

[7] Woojae Kim, Jongyoo Kim, Sewoong Ahn, Jinwoo Kim, and Sanghoon Lee, "Deep video quality assessor: From spatio-temporal visual sensitivity to a convolutional neural aggregation network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 219–234. 1

[8] Heeseok Oh, Sewoong Ahn, Sanghoon Lee, and Alan Conrad Bovik, "Deep visual discomfort predictor for stereoscopic 3d images," *IEEE Transactions on Image Processing*, vol. 27, no. 11, pp. 5420–5432, 2018. 1

[9] Kyoungoh Lee, Inwoong Lee, and Sanghoon Lee, "Propagating lstm: 3d pose estimation based on joint interdependency," pp. 119–135. 1

[10] Hyewon Song, Jiwoo Kang, and Sanghoon Lee, "Concatnet: A deep architecture of concatenation-assisted network for dense facial landmark alignment," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. pp. 2371–2375, IEEE. 1

[11] Sewoong Ahn and Sanghoon Lee, "Deep blind video quality assessment based on temporal human perception," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. pp. 619–623, IEEE. 1

[12] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920. 1

[13] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee, "Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision," in *Advances in Neural Information Processing Systems*, pp. 1696–1704. 1

[14] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum, "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling," in *Advances in Neural Information Processing Systems*, pp. 82–90. 1

[15] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," *arXiv preprint arXiv:1804.01654*, 2018. 1, 2, 3

[16] Eldar Insafutdinov and Alexey Dosovitskiy, "Unsupervised learning of shape and pose with differentiable point clouds," in *Advances in Neural Information Processing Systems*, pp. 2807–2817. 1, 3

[17] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 2

[18] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, and Michael Isard, "Tensorflow: a system for large-scale machine learning," in *OSDI*, vol. 16, pp. 265–283. 3

[19] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," in *European conference on computer vision*. pp. 628–644, Springer. 3

[20] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, and Hao Su, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015. 3

[21] Li Jiang, Shaoshuai Shi, Xiaojuan Qi, and Jiaya Jia, "Gal: Geometric adversarial loss for single-view 3d-object reconstruction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 802–816. 3