TriDepth: Triangular Patch-based Deep Depth Prediction

Masaya Kaneko[1,2], Ken Sakurada[2], Kiyoharu Aizawa[1]
[1]The University of Tokyo, [2]National Institute of Advanced Industrial Science and Technology (AIST)
[1]{kaneko, aizawa}@hal.t.u-tokyo.ac.jp, [2]k.sakurada@aist.go.jp

Abstract

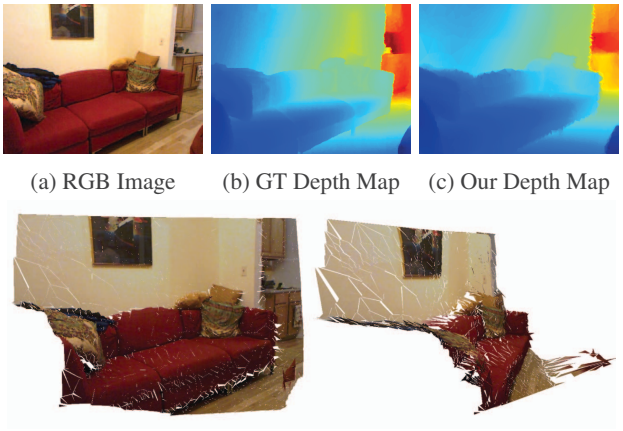
We propose a novel and efficient representation for single-view depth estimation using Convolutional Neural Networks (CNNs). Point-cloud is generally used for CNN-based 3D scene reconstruction; however it has some drawbacks: (1) it is redundant as a representation for planar surfaces, and (2) no spatial relationships between points are available (e.g, texture and surface). As a more efficient representation, we introduce a triangular-patch-cloud, which represents the surface of the 3D structure using a set of triangular patches, and propose a CNN framework for its 3D structure estimation. In our framework, we create it by separating all the faces in a 2D mesh, which are determined adaptively from the input image, and estimate depths and normals of all the faces. Using a common RGBD-dataset, we show that our representation has a better or comparable performance than the existing point-cloud-based methods, although it has much less parameters.

1. Introduction

Image-based 3D reconstruction and modeling are important problems for a variety of applications such as robotics, autonomous vehicles, and augmented reality. The representative techniques include Structure from Motion (SfM), Multi-View Stereo (MVS), and Simultaneous Localization and Mapping (SLAM).

Recently, there have been many studies that used Convolutional Neural Networks (CNNs) for 3D reconstruction. CNN-based single-view dense depth map prediction is a successful example [3, 4, 7]. In those works, the point-cloud is used for general representation since it is easy to use in CNNs, but it has some drawbacks: (1) the parameter size is too large, and (2) the spatial relationships between the points are not described. On the other hand, a mesh is one representation that can solve these issues and represent 3D structures more efficiently, because it can simplify surfaces (e.g., room wall) and maintain the texture and surface information of the object. However, due to the incompatibility of meshes with CNNs, conventional CNN-based approaches [5, 11] cannot be used for general 3D scene reconstruction. They are only suitable for the representation of simple 3DCG models [1].

This work is partially supported by KAKENHI 18K18071 and JST CREST JPMJCR19F4.



(a) RGB Image (b) GT Depth Map (c) Our Depth Map

(d) Predicted 3D structure from two views

Figure 1: We present a single-view depth prediction method using triangular-patch-cloud. Our representation has better performance than that of the point-cloud methods, despite having much less parameters.

To solve these problems, we introduce a novel intermediate representation, namely triangular-patch-cloud, between the point-cloud and mesh, and create a novel CNN architecture for the 3D structure estimation (shown in Fig. 2). The representation is a set of triangular patches created by separating all the faces of a 2D mesh, which is determined adaptively to the input image (in Fig. 3). Since it is derived from a mesh, it has the same properties as that of the mesh representation, which means a more efficient representation than point-cloud, while still being a CNN-friendly representation. In our framework, we estimate the depths and normals of all the faces of the representation using CNNs, and finally obtain the 3D structure. We evaluated the performance of our method on NYU Depth v2 [10]. Our method achieved better or comparable performance to the existing pixel-wise-based dense depth map estimation methods. It should be noted that our representation has much less parameters than the existing methods.

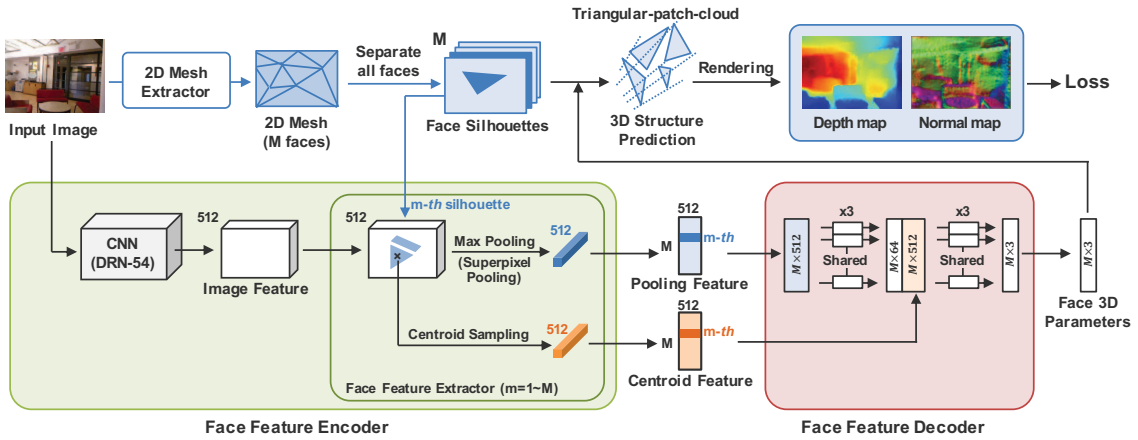


Figure 2: **Illustration of our framework.** After determining a 2D mesh adaptively to the input image, the CNNs estimate the 3D position of each face of the 2D mesh. We train the CNNs by back-propagating the loss between the rendered results.

2. Proposed Method

2.1. Formulation of Triangular-patch-cloud

We introduce a novel representation, the triangular-patch-cloud, as an intermediate representation between the point-cloud and mesh, which means that it has the best of both worlds. It is created by ignoring the adjacency connection between the faces in a 2D mesh, and the faces are treated as independent triangular patches. Each triangular patch represents a partial surface of the 3D structure. In our framework, we first determine the base 2D mesh adaptively to the input image, and estimate the 3D positions of all the patches using CNNs, as illustrated in Fig. 3. The detailed procedure of our approach is summarized as follows:

1. **2D mesh extraction.** We extract an appropriate 2D mesh for an input image, as shown in Fig. 4. We construct partially connected vertices from the simplification of the Canny edge and obtain the final mesh by applying Constrained Delaunay Triangulation (CDT) [2] to these vertices. The 2D mesh has an adaptive number of vertices and faces to the input image.

2. **3D structure prediction.** The faces of the obtained 2D mesh are treated independently as a triangular-patch-cloud, and the CNNs estimate the 3D positions (depths and normals) of the faces.

2.2. Network Architecture

The architecture of the proposed CNN framework for 3D structure estimation is illustrated in Fig. 2. The framework is roughly composed of two parts:

1. **Face Feature Encoder.** We first extract the global feature map of the input image using DRN-54 [13] and convert it to the face features of the prepared 2D mesh (whose number of faces M is adaptive for each input image). For the face feature extraction, we reduce the information from one feature vector per pixel to one feature vector per face region (silhouette). We adopt two methods: (1) Superpixel Pooling [6], which extract the max values in the region, and (2) Face Centroid Sampling, which extracts the value on the centroid position of the region.

2. **Face Feature Decoder.** By using the features — face pooling feature and face centroid feature —, we finally estimate the parameters representing the 3D position of each face. A patch of the triangular-patch-cloud has similar properties as that of a point in a point-cloud because both of them are unordered and interact with each other. Therefore, we created a CNN composed of a shared multi-layer perceptron (MLP) network, which is similar to PointNet [9].

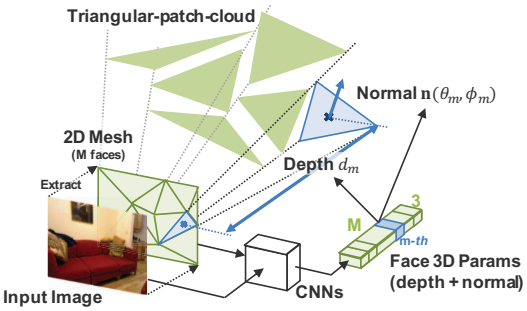
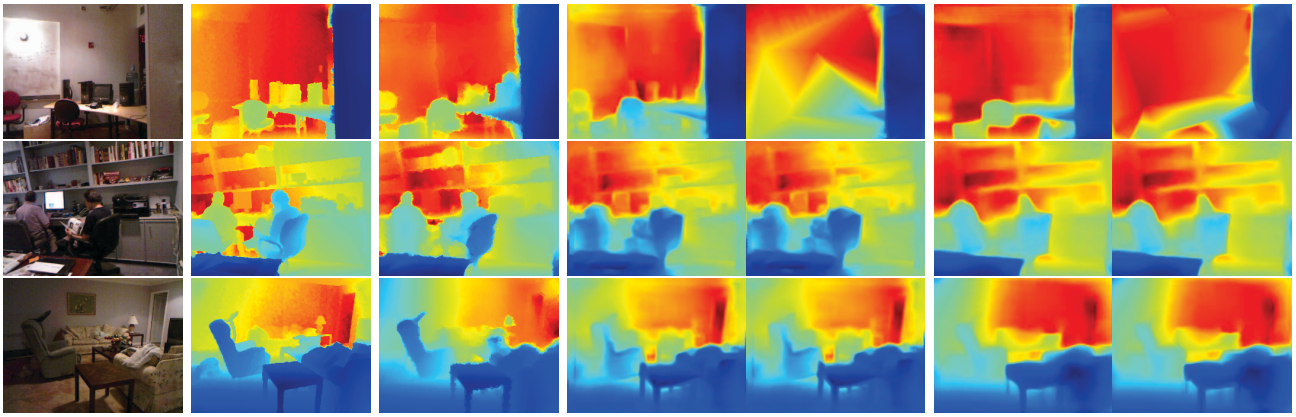


Figure 3: **Triangular-patch-cloud.** For each face, three parameters are used to determine its 3D position.

(a) Scene Image (b) Canny Edge (c) Final 2D mesh

Figure 4: **2D Mesh Extraction.** We create a 2D mesh based on (b) the Canny edge of (a) the input image.



(a) Input (b) GT (c) **Ours** (d) Eigen *et al.* [3] (original, naive) (e) Laina *et al.* [7] (original, naive)

Figure 5: **Depth Map Predictions.** Qualitative results showing our depth map results, the results of the pixel-wise-based methods [3, 7] (left) and their mesh-based naive methods (right). Each depth map is scaled for better visualization.

2.3. Loss Function

Here, we explain the method to optimize the proposed CNN framework. We use general RGBD datasets for the training of our framework and define the loss (L_{depth}) between the depth map D rendered by Neural Mesh Renderer [5] and its ground truth (GT) depth map D^*. In addition, we include the normal loss (L_{normal}) between the corresponding normal map N and the GT normal map N^* calculated by [12] in our loss function, as follows.

$$\begin{aligned} L_{sum} &= L_{depth} + \lambda_n L_{normal} \\ &= \frac{1}{n}\sum_i (D_i - D_i^*) + \lambda_n(-\frac{1}{n}\sum_i (N_i \cdot N_i^*)) \end{aligned} \quad (1)$$

where i is the valid pixel id, n is the total number of valid depth pixels, and λ_n is a balancing factor (we use $\lambda_n = 0.5$ as the best value).

3. Experimental Results

For the evaluation of our proposed approach, we trained our framework using NYU depth v2 [10], which is one of the largest RGBD datasets for indoor scene reconstruction. This dataset is composed of pairs of an RGB image and the depth image of 464 scenes captured by Microsoft Kinect. We followed the official splitting, i.e., 249 scenes for training and 215 scenes for testing. For this evaluation, we used approximately 48K pairs, which were sampled spatially uniformly from the scenes in the raw training dataset for training, and used 654 labelled images for evaluating the final performance. The input image to the network was resized to 228×304 following previous works [3, 7, 8]. We augmented them with some random transformations (small rotations, scaling, color jitter, color normalization, and flips with 0.5 chance) [8].

We trained our method for approximately 50 epochs with a batch size of 4 on a single NVIDIA Tesla P40 with 24GB of GPU memory. We used 1% of the training dataset separately as a validation dataset for the hyperparameter search.

Table 1: **Comparison with point-cloud-based methods.** It is advantageous to have low error metrics (REL, RMSE, $\log_{10}$) and high accuracy metrics ($\delta_1 \sim \delta_3$).

Method	rel	rms	$\log_{10}$	δ_1	δ_2	δ_3	#param.
Eigen and Fergus [3]	0.158	0.641	-	0.769	0.950	**0.988**	921K
Laina [7]	**0.127**	0.573	**0.055**	**0.811**	0.953	**0.988**	921K
Ours	0.146	**0.530**	0.062	0.803	**0.954**	**0.988**	**32K**

Table 2: **Comparison with naive mesh-based methods.** For each method, we use the results provided by the authors.

Method	rel	rms	$\log_{10}$	δ_1	δ_2	δ_3
ver. Eigen and Fergus [3]	0.163	0.559	0.069	0.762	0.948	0.987
ver. Laina [7]	0.154	0.535	0.064	0.793	0.949	0.987
Ours	**0.146**	**0.530**	**0.062**	**0.803**	**0.954**	**0.988**

The final score of the test dataset was evaluated using the trained model with the highest validation score. As the quantitative evaluation metrics, we used the general error metrics (RMSE, REL, $\log_{10}$, $\delta_1 \sim \delta_3$) [3, 7] for performance comparison.

3.1. Comparison with Point-cloud-based methods.

We compare our method with the baseline methods of dense depth map prediction (point-cloud-based methods). To perform an evaluation similar to the point-cloud-based methods, we use the depth map rendered from the estimated 3D mesh (see Fig. 6). The results are provided in Table 1. The performance of our method was better or comparable to that of the existing methods, despite having much less parameters. The parameter size implies the size of the 3D points registered in the 3D map in SfM and visual SLAM.

3.2. Comparison with Naive mesh-based methods.

Next, we compare the performance of our method with those of the naive mesh-based methods, since there are no CNN-based 3D scene mesh reconstruction methods. Here, a naive method involves the following procedure: (1) A dense mesh is constructed by connecting the adjacent vertices of the dense depth map obtained by a point-cloud-based methods [3, 7]. (2) The dense mesh is simplified until

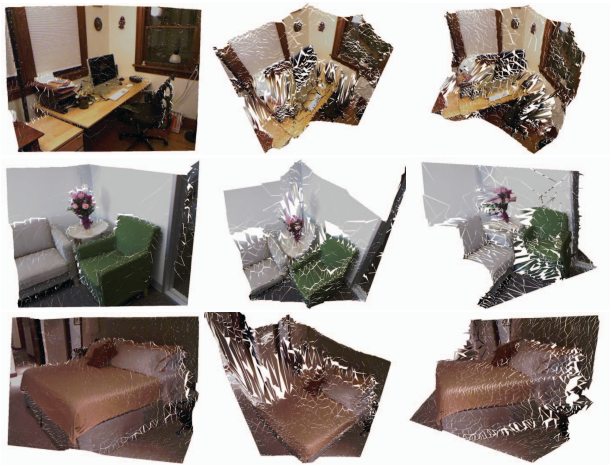


Figure 6: **3D Predictions of Triangular-patch-cloud.** Visualization results of our predictions from multi views.

it has the same number of faces as that in our method. This process gives the results rendered by a mesh with a similar parameter size.

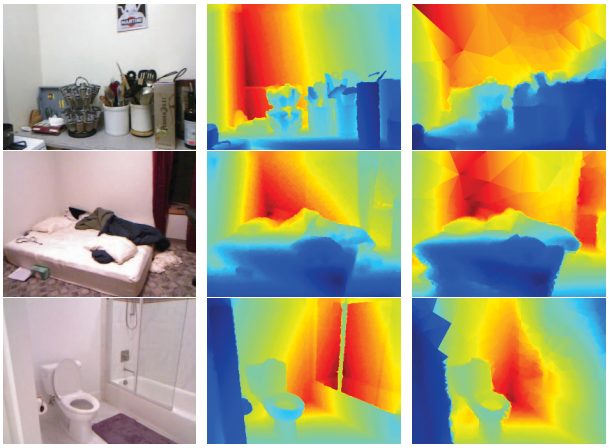
The results are shown in Table 2. Our method achieved better accuracy than both methods for all the evaluation metrics. Furthermore, according to the qualitative results displayed in Fig. 5, the naive methods could only estimate rough and poor depth maps (see the top line), but our method could estimate the depth maps that reflected the object boundaries clearly. This results demonstrate that the depth map prediction based on our representation is effective for complex 3D scenes.

4. Conclusions

We presented a novel approach for CNN-based 3D scene reconstruction of complex indoor scenes, using intermediate representation between the mesh and point-cloud. Our representation is CNN-friendly and more efficient than point-cloud. We showed that our framework could predict a visually clean 3D structure, and the results indicated equal or better performance than that of the existing methods even though our representation had much less parameters.

As a future work, we plan to partially connect the faces that are completely separated in this work. The triangular-patch-cloud is created by separating all the faces of a 2D mesh to represent complex shapes (especially occlusions); however, there are many faces that should be connected. By connecting the faces partially, we will create a mesh, a so-called partially-disconnected-mesh, whose appearance would surely become clearer while the number of parameters are reduced further.

In addition, we will solve the limitation of our method by improving the 2D mesh extraction process. Currently the process is created based on the Canny edge, which means that it does not work well in the scenes where it is difficult to recognize the geometric boundary from the RGB information (shown in Fig. 7). We plan to make 2D mesh extraction trainable from a dataset in order to improve their robustness.



(a) Input (b) GT (c) Predictions

Figure 7: **Failure Cases.** The current approach cannot estimate the 3D structure correctly in the scenes where the geometric edges are hard to detect from RGB information.

References

[1] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q.-X. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. *arXiv:1512.03012*, 2015. 1

[2] L. P. Chew. Constrained delaunay triangulations. In *SCG*, 1987. 2

[3] D. Eigen and R. Fergus. Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture. In *ICCV*, 2015. 1, 3

[4] D. Eigen, C. Puhrsch, and R. Fergus. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In *NIPS*, 2014. 1

[5] H. Kato, Y. Ushiku, and T. Harada. Neural 3D Mesh Renderer. In *CVPR*, 2018. 1, 3

[6] S. Kwak, S. Hong, and B. Han. Weakly Supervised Semantic Segmentation Using Superpixel Pooling Network. In *AAAI*, 2017. 2

[7] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper Depth Prediction with Fully Convolutional Residual Networks. In *3DV*, 2016. 1, 3

[8] F. Ma and S. Karaman. Sparse-to-Dense: Depth Prediction from Sparse Depth Samples and a Single Image. 2018. 3

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *CVPR*, 2017. 2

[10] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*, 2012. 1, 3

[11] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y. Jiang. Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images. In *ECCV*, 2018. 1

[12] Z. Yang, P. Wang, W. Xu, L. Zhao, and R. Nevatia. Unsupervised Learning of Geometry From Videos With Edge-Aware Depth-Normal Consistency. In *AAAI*, 2018. 3

[13] F. Yu, V. Koltun, and T. Funkhouser. Dilated Residual Networks. In *CVPR*, 2017. 2