PCTNet:3D Point Cloud and Transformer Network for Monocular Depth Estimation

Yusheng Hong
Department of Software Engineering
University of Science and Technology of China
Hefei, China
yshong@mail.ustc.edu.cn

Xiaolong Liu
Department of Software Engineering
University of Science and Technology of China
Hefei, China
masonl@mail.ustc.edu.cn

Hang Dai
Department of Software Engineering
University of Science and Technology of China
Hefei, China
wanggan0910@mail.ustc.edu.cn

Wenqi Tao
Department of Software Engineering
University of Science and Technology of China
Hefei, China
wink@mail.ustc.edu.cn

2022 10th International Conference on Information and Education Technology (ICIET) | 978-1-6654-8048-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICIET55102.2022.9779008

***Abstract*—Estimating dense depth map from one image is a challenging task for computer vision. Because the same image can correspond to the infinite variety of 3D spaces. Neural networks have gradually achieved reasonable results on this task with the continuous development of deep learning. But the depth estimation method based on monocular cameras still has a gap in accuracy compared with multi-view or sensor-based methods. Thus, this paper proposes to supplement a limited number of sparse 3D point clouds combined with transformer processing to increase the accuracy of the monocular depth estimation model. The sparse 3D point clouds are used as supplementary geometric information and the 3D point clouds are input into the network with the RGB image. After five times integration, the multi-scale features are extracted, and then the swin transformer block is used to process the output feature map of the main network, further improving the accuracy. Experiments demonstrate that our network achieves better results than the best method on the current most commonly used dataset for monocular depth estimation, NYU Depth V2. However, the qualitative results are also better than the best method.**

Keywords—3D point cloud, swin transformer, monocular depth estimation

I. Introduction

The resulting map of the depth estimation is very useful. It can be applied to various fields, such as autonomous driving, and scene understanding. At present, it has been explored in academia for a long time. From the beginning, geometric-based methods such as structure from motion (SFM) [1], then sensor-based methods, and now based on deep learning methods, monocular depth estimation has made great progress. But its accuracy is still unsatisfactory, and the ideal accuracy gaps still exist. The reason for the low accuracy is generally considered to be the lack of sufficient geometric information, such as parallax. Therefore, some scholars used the deep learning method based on multi-view stereo [2] to increase the accuracy. However, the computational complexity also increased.

Therefore, in view of the above problems, we propose a novel network architecture for monocular depth estimation. More specifically, we input the sparse point clouds into the network together with RGB images, to supplement geometric information. It is fused with the original RGB image 5 times, and finally, multi-scale features are extracted. CNN network has been leading the field of computer vision in the past, but the most important convolutional layer only performs global processing when the image is downsampled to a very low resolution. For depth estimation, the global processing of image features should not only be limited to low resolution but also be processed at a large resolution, so we refer to the adabins [3]. We use transformer [4] to process the feature maps output by the previous network. Because vision transformer [5] occupies too much memory, it is difficult to train. Therefore, we use the swin transformer block in swin transformer [6] to replace the encoder module of vision transformer in the original paper. Note that we only use two successive swin transformer blocks here, and do not use the patch merging. Because we have previously extracted multi-scale features, we only use the blocks to reduce network complexity. In this regard, we use the proposed architecture to conduct sufficient experiments on the most commonly used depth estimation dataset, NYU Depth V2 [7]. We achieve higher accuracy than the current best method, and the qualitative results are also better than the current best method, which proves the validity of our ideas. In this paper, our main contributions are as follows:

1) We propose a novel architecture, which supplements certain geometric information lacking in 2D images and performs global processing of image features to improve various indicators of the model.

2) We reduce the complexity of network training to some extent. We achieve state-of-the-art results on the current most commonly used dataset for monocular depth estimation, NYU Depth V2.

II. Related Work

A. Monocular Depth Estimation

Monocular depth estimation has been studied for a long time. It refers to predicting its corresponding dense depth map from one image. As far as we know, the first supervised neural network based on deep learning was developed by Eigen et al. [8]. The network architecture mainly includes two branches, the first branch roughly predicts the whole information, the second branch refines the local information of the predicted image. Therefore, this is a global and local strategy. Then eigen et al. [9] improved it and proposed a multi-scale network framework, adding a third branch to further refine the details of the image.

978-1-6654-8048-2/22/$31.00 ©2022 IEEE

Meanwhile, this network framework can be applied to three tasks. In 2016, some researchers proposed a fully convolutional residual network [10] for depth estimation, which is different from the previous pre-training network that uses a fixed fully connected layer size to obtain image-to-image conversion. This paper directly removes the fully connected layer, returns the advanced features to the same size as the original image, and treats the entire network as an encoder-decoder process. In 2017, the researchers proposed a fast-training two-stream convolutional neural network [11] that predicts depth gradient and depth, then fuses them to get the final depth map. Fu et al. [12] treat it as a classification task, and the number of categories is to divide the farthest actual distance into multiple parts. Some researchers proposed learning relative depth [13] or calibration mode to improve generalization performance. Some researchers [14] proposed a method that uses the ubiquitous planar structure in indoor environments as prior knowledge to guide depth estimation. They added a DAV module between encoder and decoder. Obviously, this model is not very suitable for outdoor scenes. AdaBins [3] divided the depth value into bins, and the depth value is formed by the central values of these bins. And they used the encoder part of the transformer to globally process the feature map, which is the current best method.

B. 3D Point Cloud

The 3D point cloud is a dataset of points. It contains a lot of information, such as classification value, three-dimensional coordinates X, Y, Z, intensity value, color, and so on. The point cloud is divided into two types in terms of composition characteristics. One is ordered point cloud, and the other is unordered point cloud. The ordered point cloud is generally restored by the depth map. The ordered point cloud is very convenient for some processing. But in many cases, it is impossible to obtain the ordered point cloud. The unordered point cloud is easy to understand. There is no order between the points, and the exchange of the order of the points has no effect. It is a relatively common form of the point cloud. Chen et al. [15] proposed to fuse 3D point cloud for depth completion, and Lam et al. [16] also introduced sparse 3D point cloud data for depth estimation and achieved good results. Therefore, this paper also introduces sparse 3D point cloud data to fuse image features for depth estimation. We simplify the network structure proposed by Lam et al [16] and reduce the complexity of the model.

C. Transformer

Transformer was first applied to NLP tasks. It was widely applied to computer vision tasks starting from Vision Transformer [5] in 2020. This paper is the first time to use a complete transformer structure for image classification tasks. It cuts images into patches and inputs them into the network. Later, scholars began to widely apply transformer in the image classification task. Due to the large memory usage of vit and the straight-tube structure, which is not conducive to extracting multi-scale features. PVT [17] proposed SRA to cut the patch on the feature map again, so that the size of the feature map can be changed. However, in this way, the receptive field of Q and K is different. Swin Transformer [6] proposed the patch merging method to decrease the size of the feature map. And they used W-MSA to make Q only calculated with a fixed number of K. Thus, the receptive field is identical. But obviously, this will lose global information, limit the ability of the model. Therefore, the author added SW-MSA to enhance the interaction of each window. Of course, the transformer can be applied not only to the image classification task, but also to many downstream tasks, such as DETR [18] for object detection, maskformer [19] for semantic segmentation, and so on. However, at present, there is not much research on transformer network architecture applied to monocular depth estimation, so this paper conducts a more in-depth study on it.

III. Method

In this part, we will talk about the proposed model architecture PCTNetwork in detail, as shown in Fig. 1. It mainly includes three parts. The first part is the input processing. Then it will pass through PCN(Point Cloud Network), extracting multi-scale features. The PCN is the second part. Later, it will pass through SAM(Simple AdaBins Module) that global processes feature maps at large resolutions. The SAM is the third part. After it, the dense depth map is obtained.

A. Input Processing

The input of our model is RGB image and sparse 3D point clouds (128 3D points), as shown in Fig. 1. We need to process the input RGB image and sparse 3D point clouds. First, the 3D point cloud is projected to a two-dimensional plane to form a sparse depth map (point cloud data is projected to a 2D depth map using the camera's internal reference matrix). And then divided into two branches, one branch combines the sparse depth map with the RGB image to form RGBD image and then goes through two convolutions layers. The other branch directly passes through two 3*3 convolutional layers. In the end, the feature maps are obtained by concat the two branches and input into the Point Cloud Network. The input processing of data can be understood as extracting some low-level features. Of course, the point cloud not only participates in input processing but also in the subsequent Point Cloud Network. Only in this way can the geometric information of the point cloud be fully utilized. Input processing is the first part of our network, next, we will introduce the second part.

B. Point Cloud Network

After input processing, it will go through 3D Point Cloud Network for 5 times, and after each PCN, downsampling will be carried out. Therefore, each PCN network processes feature map's resolution keeps getting smaller, and the final output of the main network is multi-scale feature maps. The PCN mainly includes two parts, as shown in Fig. 2, the first part is the encoder, the second part is the decoder.

Our input 3D point clouds are rare and the geometric information they represent is not like image data, it is not enough to integrate the information directly by concat as before. Thus, another integration is performed in the encoding stage of each PCN. The encoder mainly includes two large branches, the upper part is the 3D branch and the lower part is the 2D branch. The 2D branch includes 3 sub-branches. The first branch has a convolution with a stride of 1, the size of the convolution kernel is 3*3. And the second branch also has two 3*3 convolutions and one upsampling. The first convolution layer is used for downsampling with stride 2, finally, upsample back to the original size. The two branches will eventually add, which is equivalent to learning multi-scale features. The third branch is

to prevent the vanishing gradient. The 3D branch mainly learns the structural features of 3D point cloud. Obviously, 3D point cloud cannot be processed by ordinary 2D convolution. Here, a point cloud convolution method is used, feature kernel alignment convolution[20]. 3D branch mainly goes through two feature kernel alignment convolutions, and finally, projects on the two-dimensional plane form a C*H*W feature map. In order to effectively integrate their features, the feature maps obtained from the 2D branch and 3D branch are added, and then a 3*3 convolution is performed to further integrate the features. The decoder is relatively simple, mainly includes three 3*3 convolutional layers and a residual connection. To avoid the network being too complex, we haven't changed the encoder and decoder for the five fusions.

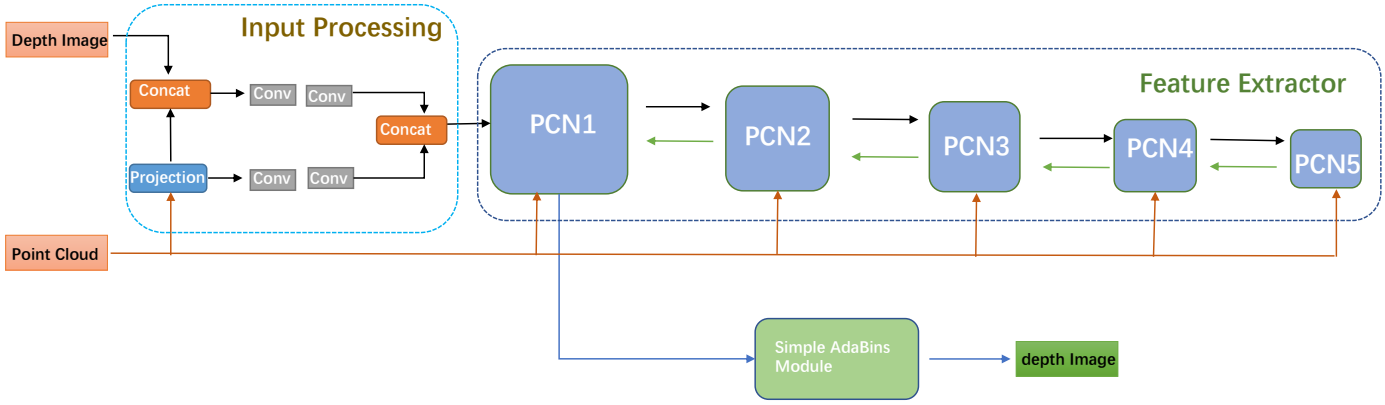

Fig. 1. Overview architecture of the 3D point cloud and transformer network.

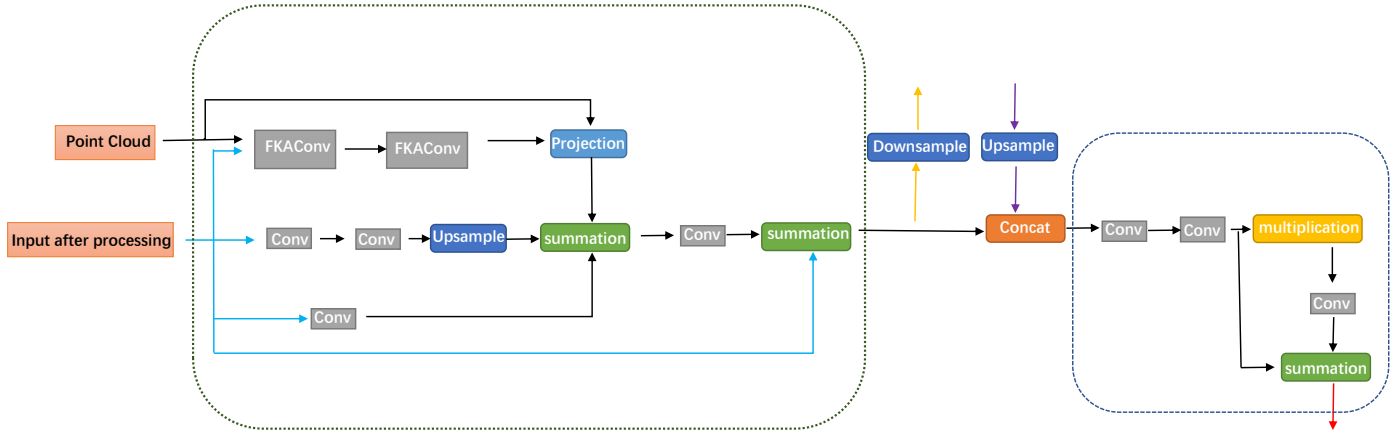

Fig. 2. The details of the PCN. Cyan represents the output of the previous PCN or the input after processing. Orange represents the output to the next PCN. Purple represents the input from the latter PCN. Red represents the output to the previous PCN.

C. *Simple Adabins Module*

A disadvantage of the convolution layer is that global information can be processed only when the feature map has a very small resolution. But global processing is also important for large resolution. Thus, we further processed the feature map output by PCN to improve the accuracy, as shown in Fig. 3.

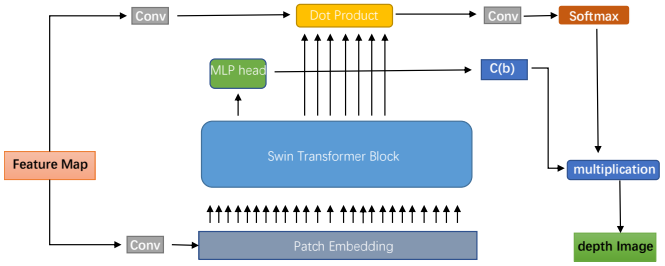

Fig. 3. The details of the simpe adabins module. The feature map is the output after five times PCN. The convolution of the lower branch is p*p.

The output feature map needs to go through two branches. The upper branch goes through an ordinary convolution layer, and the lower branch goes through a p*p convolution. Because the input of the transformer requires a sequence of fixed-size vectors. Assuming that the original feature map's size is H*W*C, then after the convolution of p*p, it's size becomes to $\frac{H}{p} * \frac{W}{p}$*C. Later, it is flattened and changed to a tensor of $\frac{H*W}{p^2}$ * C. There are $\frac{H*W}{p^2}$ pacths in total, and the length of each patch is the number of channels. The first output encoding vector passes through the MLP head to generate an N-dimensional vector, which is normalized to form an N-dimensional b vector. Here, the depth value is also divided into N parts according to the farthest distance, and then uses the formula proposed by Bhat et al.[3] calculates each central value. The other encoding vectors output from the swin transformer block are used as 1*1 convolution kernels and perform element-by-element dot product with the feature map obtained by the above branch. This is equivalent to integrating the global information encoded by

the Transformer into the feature map output from the point cloud network. After that, the number of channels is changed to N through one convolution, then softmax is used to predict the probability of N central depth values of each pixel. Finally, it will multiply and add the corresponding central value to obtain each pixel's depth value.

D. Loss Function

Our loss function consists of two parts, where the first part is proposed by Eigen et al. [8]:

$$L_p = \frac{1}{n}\sum_i g_i{}^2 - \frac{\alpha}{n^2}\left(\sum_i g_i\right)^2 \quad (1)$$

where $g_i = log\widehat{d}_i$-logd_i and n represents the number of pixels, d_i is the ground truth depth. The second part:

$$L_t = chamfer(G, C) + chamfer(C, G) \quad (2)$$

G represents the set of all ground truth depth values of the image, C represents the set of all bin center values, and chamfer represents the bi-directional Chamfer Loss[24]. The final loss is obtained by adding the two parts together.

$$L_s = L_p + \beta L_t \quad (3)$$

IV. Experiments

A. Datasets and Evaluation Metrics

NYU Depth V2 is an indoor scene dataset with a resolution of 640*480, including 464 scenes. We use the official train and test dataset division strategy for training and testing. The farthest depth value is set to 10 meters. We use the most important indicators for evaluating the feasibility of monocular depth estimation. These indicators include REL, RMSE, Threshold. The specific definitions are as follows:

Mean absolute relative error (REL)=$\frac{1}{n}\sum_{i=1}^{n}\frac{|d_i-\widehat{d}_i|}{d}$

Root mean square error (RMSE)=$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i-\widehat{d}_i)^2}$

Thresholded accuracy (δ): Max $\left(\frac{\widehat{d}_i}{d_i}, \frac{d_i}{\widehat{d}_i}\right) = \delta^i < 1.25^i, i = 1, 2, 3$

n represents the total number of valid pixels, d_i represents the ground truth depth of pixel i, and $\widehat{d}_i$represents the predicted depth of pixel i.

B. Implementation Details

We use the most commonly used framework, Pytorch [21], to implement the proposed network architecture. And we use the AdamW [22] optimizer, the initial learning rate is $1.2*10^{-4}$, starting from the 10th epoch, decreasing by 6% every 5 epochs. The parameters α and β in the loss function are set to 0.5, 0.1 respectively. Of course, we also augmented the data in the training stage, such as random rotation and horizontal flip.

C. Experimental Results

As shown in Table I, on the NYU Depth V2 dataset, we outperform all the original methods in every metric, which demonstrates the feasibility and correctness of our network architecture. Fig. 4 shows the qualitative results of our network architecture on the NYU Depth V2 dataset, and it is clear that our network architecture produces a better depth map than the others.

TABLE I. RESULTS ON THE NYU DEPTH V2. THE LOWER THE FIRST TWO INDICATORS, THE BETTER, AND THE HIGHER THE LAST THREE INDICATORS, THE BETTER

Method	REL	RMSE	δ_1	δ_2	δ_3
Eigen et al. [8]	0.158	0.641	0.769	0.950	0.988
Laina et al. [10]	0.127	0.573	0.811	0.953	0.988
Fu et al. [12]	0.115	0.509	0.828	0.965	0.992
Huynh et al. [14]	0.108	0.412	0.882	0.980	0.996
Song et al. [23]	0.105	0.384	0.895	0.983	0.996
Bhat et al. [3]	0.103	0.364	0.903	0.984	0.997
Ours	**0.101**	**0.346**	**0.905**	**0.986**	**0.997**

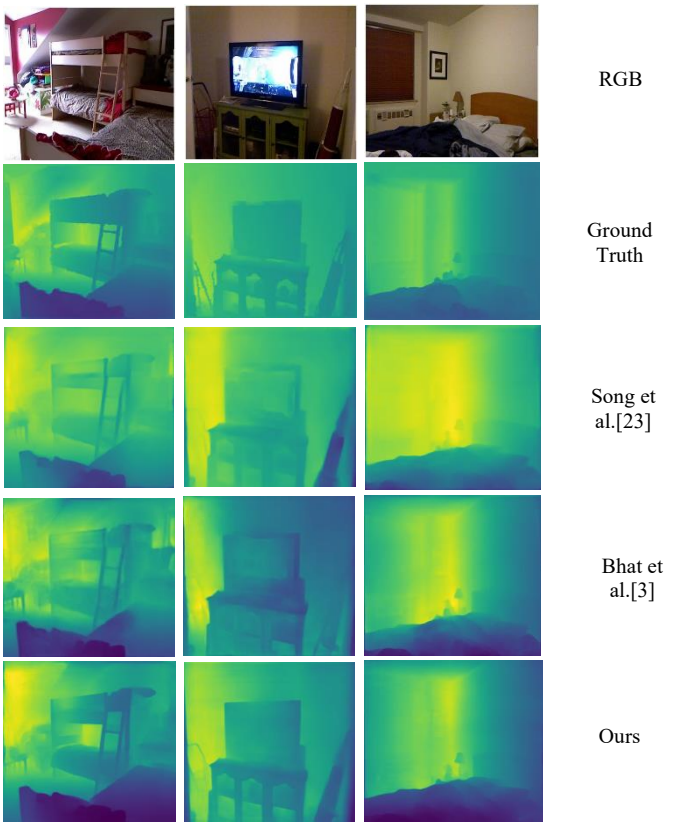

Fig. 4. Qualitative results on the NYU Depth V2 dataset.

V. Conclusion

In this paper, we propose a novel network architecture to solve the problem of estimating depth from one image. We exploit the limited number of sparse 3D point clouds as a complement of geometric information, and transformer structure is used to process images globally, which effectively improves the performance of monocular depth estimation. State-of-the-art is achieved on the most popular datasets. For future work, we think that the network architecture can be optimized to further reduce the complexity and increase the accuracy of the network model when the parameters decrease.

REFERENCES

[1] S. Ullman, “The interpretation of structure from motion,” Proceedings of the Royal Society of London. Series B. Biological Sciences, vol. 203, no. 1153, pp. 405–426, 1979.

[2] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan.Mvsnet: Depth inference for unstructured multi-view stereo.European Conference on Computer Vision (ECCV), 2018. 1,5, 6f

[3] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4009–4018, 2021.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and IlliaPolosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, pages 5998–6008, 2017. 1,2, 4

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov,Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner,Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image isworth 16x16 words: Transformers for image recognition atscale. In International Conference on Learning Representations, 2021. 1, 2, 3, 4, 5, 6, 9

[6] Liu Z, Lin Y, Cao Y, et al. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows[J]. 2021.

[7] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and RobFergus. Indoor segmentation and support inference fromrgbd images. In Computer Vision – ECCV 2012, pages 746–760, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.1, 2, 5, 11

[8] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network.NIPS 2014

[9] Eigen et al, Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture, ICCV 2015

[10] Laina et al, Deeper Depth Prediction with Fully Convolutional Residual Networks, 3DV 2016

[11] Li et al, A Two-Streamed Network for Estimating Fine-Scaled Depth Maps from Single RGB Images, ICCV 2017

[12] Fu et al, Deep Ordinal Regression Network for Monocular Depth Estimation, CVPR 2018

[13] Lee et al, Monocular depth estimation using relative depth maps, CVPR 2019

[14] Lam Huynh, Phong Nguyen-Ha, JiriMatas, Esa Rahtu, and Janne Heikkila. Guiding monocular depth estimation using depth-attention volume. In European Conference on Computer Vision, pages 581–597. Springer, 2020.

[15] Yun Chen, Bin Yang, Ming Liang, and Raquel Urtasun.Learning joint 2d-3d representations for depth completion. In Proceedings of the IEEE International Conference on Computer Vision, pages 10023–10032, 2019. 2, 3, 5

[16] Huynh L, Nguyen P, Matas J, et al. Boosting Monocular Depth Estimation with Lightweight 3D Point Fusion[J]. 2020.

[17] Wang W, Xie E, Li X, et al. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions[J]. 2021.

[18] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In ECCV, 2020.

[19] Cheng B, Schwing A G, Kirillov A . Per-Pixel Classification is Not All You Need for Semantic Segmentation[J]. 2021.

[20] Alexandre Boulch, Gilles Puy, and Renaud Marlet. FKAConv: Feature-Kernel Alignment for Point Cloud Convolution. In 15th Asian Conference on Computer Vision (ACCV2020), 2020. 3

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: Animperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alche-Buc, ´E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32, pages 8026–8037. Curran Associates, Inc., 2019. 6

[22] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In ICLR, 2019. 6

[23] Song M, Lim S, Kim W. Monocular depth estimation using laplacian pyramid-based depth residuals[J]. IEEE transactions on circuits and systems for video technology, 2021, 31(11): 4381-4393.

[24] H. Fan, H. Su, and L. 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), pages 2463–2471, 2017. 5