PlaneTR: Structure-Guided Transformers for 3D Plane Recovery

Bin Tan[*1]   Nan Xue[*1]   Song Bai[2]   Tianfu Wu[3]   Gui-Song Xia[†1]

[1] School of Computer Science, Wuhan University
[2] ByteDance AI Lab   [3] Department of ECE, NC State University

 https://git.io/PlaneTR

Abstract

This paper presents a neural network built upon Transformers, namely PlaneTR, to simultaneously detect and reconstruct planes from a single image. Different from previous methods, PlaneTR jointly leverages the context information and the geometric structures in a sequence-to-sequence way to holistically detect plane instances in one forward pass. Specifically, we represent the geometric structures as line segments and conduct the network with three main components: (i) context and line segments encoders, (ii) a structure-guided plane decoder, (iii) a pixel-wise plane embedding decoder. Given an image and its detected line segments, PlaneTR generates the context and line segment sequences via two specially designed encoders and then feeds them into a Transformers-based decoder to directly predict a sequence of plane instances by simultaneously considering the context and global structure cues. Finally, the pixel-wise embeddings are computed to assign each pixel to one predicted plane instance which is nearest to it in embedding space. Comprehensive experiments demonstrate that PlaneTR achieves state-of-the-art performance on the ScanNet and NYUv2 datasets.

1. Introduction

Recovering 3D planar structures from a single RGB image is a fundamental problem in 3D vision and is challenging due to its ill-posed nature. The goal of this problem is to detect regions of plane instances and estimate their 3D planar parameters (e.g. surface normal and offset) in an image. As a fundamental representation of 3D scenes, the reconstructed planes have a wide range of applications in downstream tasks, such as augmented reality [4], visual SLAM [30, 44, 20] and indoor scene understanding [31, 16].

Some early methods [6, 2, 21, 15, 25] tend to utilize geometric elements such as line segments, junctions, and van-

---
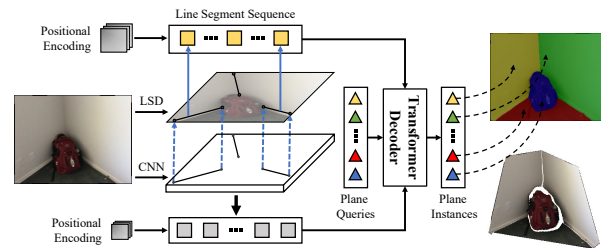[*]Equal Contribution.
[†]Correspondence Author.

Figure 1. Illustration of the proposed PlaneTR. Our network holistically leverages context features and line segments represented by two tokenized sequences to predict a set of plane instances in an image.

ishing points to tackle this problem in a bottom-up manner. These geometric elements are usually first divided into various groups and then carefully analyzed under a series of strict presuppositions (e.g. Manhattan World) and rules to recover 3D planes. Although these structure-based methods have achieved successes to some extent, they are suffering from the issues of missing or incorrect detection of geometric primitives, complex technique processes, and limited scenes which affect their performance and applications.

Recently, Convolutional Neural Networks (CNNs) addressed this problem [19, 43, 18, 45, 11, 24]. Some methods [19, 43, 18] directly predict plane instance masks with corresponding 3D planar parameters from the input image in a top-down manner. By contrast, PlaneAE [45] takes a bottom-up manner and achieves plane instances by clustering pixels that are similar in an embedding space. These methods relax constraints in structure-based methods and have achieved promising performance. However, they mainly leverage the context information from CNNs and ignore structure cues in the image which are useful for 3D plane recovery.

In this paper, we are interested in exploiting geometric structures for the problem of 3D plane recovery of the indoor scene under a learning-based framework. Although there are various low-level geometric primitives, we find that line segments are usually used to construct 3D planes [16, 25] and contain more holistic 3D information of the scene when comparing with other geometric primitives,

such as feature points, edges, and vanishing points. Besides, benefiting from recent state-of-the-art works in line segment detection [42, 40, 41, 47], it is convenient for us to achieve line segments from an image. Thus, in this paper, we use line segments as the geometric structures for plane recovery.

In some recent works [35, 12], geometric structures have already been used for depth and normal estimation. In these methods, structures are represented as dense maps (e.g. line segment map) to meet the representation of CNNs. However, such dense representation of structures is hard for the network to leverage global structure cues with two drawbacks: (i) limited receptive field of CNNs and (ii) sparse distribution of structure pixels in the dense map. Although CNNs-based attention mechanisms [36, 38] can alleviate the first drawback, the second one is still hard to be well tackled. Therefore, this paper is going toward answering the question:

If it is possible to holistically exploit sparse line segments for the learning of plane recovery?

Most recently, the sequence-to-sequence model of Transformers [32] has been successfully used in vision tasks [3, 39]. In these works, the input features and output targets are represented as visual tokens and globally interacted with each other via the attention mechanism of Transformers. Motivated by the tokenized representation in vision Transformers, we address the above question by proposing our PlaneTR, a Transformer model that leverages the informative context features and meaningful geometric structures for plane recovery.

For a given input image and its detected line segments, our PlaneTR encodes the line segments and context features into two sets of tokenized sequences, respectively. Then, a set of learnable plane queries are used to holistically interacted with the context and line segment sequences via a structure-guided plane decoder which outputs a set of tokenized plane instances. As a final step, we design a simple instance-to-pixel segmentation strategy motivated by the associative embedding [22] and PlaneAE [45], which yields the pixel-wise plane segmentation results by assigning each pixel to its nearest plane instance in the embedding space.

In summary, the main contributions of this paper are as follows:

- We leverage line segments as tokenized sequences instead of dense maps to guide the learning of 3D plane recovery with geometric structures.
- We develop a novel Transformer, PlaneTR, to simultaneously detect and reconstruct plane structures from a single image in a sequence-to-sequence manner.
- Our method obtains state-of-the-art performance on the ScanNet [5] and NYUv2 [28] datasets, verifying the effectiveness of our method.

2. Related Work

2.1. Single Image 3D Plane Recovery

Geometric structures such as line segments and vanishing points are widely used in traditional methods [6, 2, 21, 15, 25, 14, 46, 9] for piece-wise 3D plane recovery from a single image. These methods usually tend to leverage structures via a grouping and optimization (or fitting) strategy to solve this problem. For example, Lee *et al.* [15] propose to reconstruct the planar 3D scene from a single image by first proposing a number of structure hypotheses and then finding a model which best fits the collected line segments. Qian *et al.* [25] design a Line-Segment-to-3D algorithm which first groups line segments into a set of minimal spanning Manhattan trees and then lifts these trees to 3D under the Manhattan constraints. Li *et al.* [16] assume a box prior to the scene and propose to find the best representation of the image via a structure-guided search algorithm. To effectively conduct their algorithms, these methods are generally built upon a series of strict assumptions and rules which limited their application and performance.

The recent learning-based methods [43, 19, 45, 18, 24] relax constraints in traditional structure-based methods by directly detecting and reconstructing 3D planes from single-view images. PlaneNet [19] generates a large planar depth maps dataset from the ScanNet dataset [5] and proposes an end-to-end deep neural network to predict a set of plane segmentation masks and their corresponding 3D planar parameters. PlaneRecover [43] designs a plane structure-induced loss and trains a plane reconstruction network directly from the RGB-D data without 3D plane annotations. PlaneR-CNN [18] addresses the plane recovery problem by applying a detection-based framework (e.g. Mask R-CNN [10]) and proposes a segmentation refinement network and a warping loss to improve performance. Different from previous top-down methods, PlaneAE [45] solves this problem in a bottom-up manner. It obtains plane segmentation masks by first learning an embedding for each pixel and then grouping them to various plane instances via an efficient mean shift algorithm. Despite the promising performance achieved by these learning-based methods, they only consider the context information in deep features and ignore structure cues which are important for 3D plane recovery. By contrast, we combine the idea from both traditional and learning-based methods and propose a method to jointly utilize the context information and geometric structures for 3D plane recovery.

2.2. Structure Guided Learning

Structures have been successfully used in some learning-based methods [35, 12, 27, 29, 26]. Wang *et al.* [35] propose to learn a Manhattan Label Map from the input RGB image and its corresponding Manhattan line map for normal esti-
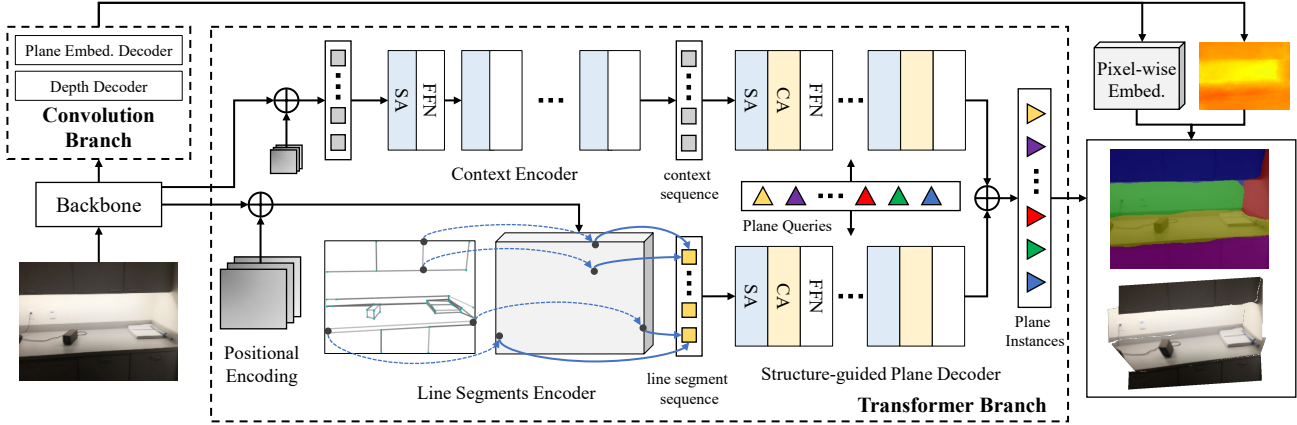
Figure 2. Overview of our proposed PlaneTR. It mainly consists of two branches: (i) a Transformer branch which models holistic relations between plane instances and context (line segment) sequence; and (ii) a convolution branch which predicts pixel-wise embeddings used for plane segmentation and depths of non-plane regions.

mation. Jin *et al*. [12] leverage geometric structure as a prior and a regularizer to facilitate the learning of depth maps. In their method, they first estimate geometric structures (e.g. corner map, boundary map, and plane map) from the input image as a prior and then estimate the same structures from the output depth map as a regularizer. Song *et al*. [29] propose to embed learned edge features into the deep context features to guide the estimation of disparities in image details and boundaries. These methods tend to serve structures as local guidance and represent them as pixel-level dense maps which are difficult for the network to utilize global structure cues. Different from these methods, in this paper, we take a novel tokenized representation of line segments to introduce global structure cues into a learning-based framework.

2.3. Vision Transformers

Transformers [32] are proposed for sequence-to-sequence machine translation in NLP. Recently, some researchers attempt to introduce Transformers into vision tasks, such as objection detection [3, 48], image segmentation [33, 37, 17], and image classification [7]. DETR [3] proposes a new simple paradigm for object detection built upon a Transformer encoder-decoder architecture which is free from many hand-designed components such as anchor generation and non-maximum suppression. DETR formulates the problem of object detection into a sequence-to-sequence prediction problem and directly predicts a set of objects from the learned object queries which are interacted with the context feature sequence. Liang *et al*. [17] exploit a self-attention network in Transformers to model the intricate dependencies in polygon vertexes and learn to predict a set of offsets for the input vertexes to deform the initial polygon to object boundaries. Inspired by DETR, Xu *et al*. [39] propose a network to predict a set of line segments

directly from tokenized image features which simplifies the line segment detection process and achieves state-of-the-art performance. On one hand, these works show the promising application of Transformers in vision tasks. On the other hand, such sequence-to-sequence architecture based on the tokenized representation gives us a suitable manner to jointly leverage the instance-level structures and context features for plane recovery. Thus, in this paper, we propose a novel structure-guided network built upon Transformers for the 3D plane detection and reconstruction, namely PlaneTR.

3. The PlaneTR Model

As illustrated in Figure 2, the proposed PlaneTR is made up of a convolutional backbone followed by a Transformer branch for plane instances prediction (Sec. 3.1) and a convolution branch for pixel-wise plane embedding and non-plane region depth estimation via two decoders. The estimated pixel-wise embeddings are then used to plane instance segmentation (Sec. 3.2). We use the modified HRNet-w32 [34] as our backbone which takes an image $I \in \mathbb{R}^{H \times W \times 3}$ as input and outputs feature maps with four scales, denoted by $\mathbb{F} = \{F_i \in \mathbb{R}^{H_i \times W_i \times C_i}\}_{i=1}^4$ ($H_i = H/2^i$, $W_i = W/2^i$). According to [45], we define the 3D parameter of a plane as $n \doteq \tilde{n}/d \in \mathbb{R}^3$, where $\tilde{n} \in \mathbb{R}^3$ is the surface normal and d indicates the distance from plane to camera center.

3.1. Plane Instance Prediction with Transformers

Context Encoder. We first encode context features from the backbone feature map F_4 into a tokenized feature sequence with a standard Transformer encoder which consists of six encoder layers as used in DETR [3]. Specifically, we feed a flattened feature sequence $f_c \in \mathbb{R}^{d \times (H_4 W_4)}$

generated from F_4 followed by a 1×1 convolution and its sine/cosine positional encoding $E_c \in \mathbb{R}^{d \times (H_4 W_4)}$ into the encoder. In each encoder layer, the input feature tokens interact with each other via a self-attention (SA) operation, and then a fully connected feed forward network (FFN) is used to get the output. The final output context sequence of the last encoder layer is defined as $S_c \in \mathbb{R}^{d \times (H_4 W_4)}$. Here, d is set to 256 in this paper.

Line Segments Encoder. To achieve the tokenized line segment sequence, we first detect n line segments $\mathbb{L} = \{l_i = (\mathbf{x}_i^1, \mathbf{x}_i^2)\}_{i=1}^n$ using the state-of-the-art line segment detection algorithm HAWP [42] with their pretrained model. Here, $\mathbf{x}_i^1$ and $\mathbf{x}_i^2$ are two endpoints of l_i. Then, we build the line segment sequence based on a high-resolution backbone feature F_2 in order to distinguish adjacent line segments.

Denoted by $F_r \in \mathbb{R}^{H_2 \times W_2 \times d}$, a 1×1 convolution layer is applied on the top of the backbone feature map F_2. After computing the positional encoding map E_r of F_r, we sample feature vectors and the positional encoding vectors for each line segment l at its two endpoints using the bi-linear interpolation, denoted by (f_1, f_2) and (e_1, e_2), respectively. In the next step, an MLP layer is applied to yield the feature of line segment l and its corresponding positional encoding as:

$$f_l = \text{MLP}(f_1 \otimes f_2), e_l = (e_1 + e_2)/2, \qquad (1)$$

where $\otimes$ indicates a concatenation operation. Finally, the features and the positional encodings of all line segments are concatenated into $S_l \in \mathbb{R}^{d \times n}$, $E_l \in \mathbb{R}^{d \times n}$ as the line segment sequence.

Structure-guided Plane Decoder. Our Structure-guided Plane Decoder has two paralleled branches as shown in Figure 2. Each branch is built upon a standard Transformer decoder which consists of six encoder layers as used in DETR [3]. The two branches take the context sequence and the line segment sequence as inputs respectively, and share the same learnable plane queries $E_p \in \mathbb{R}^{d \times K}$. Here, K is a fixed number of predicted plane instances ($K = 20$ in this paper). By interacting with the context and line segment sequence simultaneously with a cross-attention (CA) operation of the Transformer, the plane queries are able to perceive plane instances by holistically considering the context and structure cues in an image. We define the output sequences of the two branches as O_c and O_l, respectively. Then, the final decoded plane instance sequence of the Structure-guided Plane Decoder can be calculated as $S_p = O_c + O_l \in \mathbb{R}^{d \times K}$.

Plane Instance Prediction Layer. For each plane instance in S_p, we apply a multi-head linear layer to predict a set of parameters, including the 3D planar parameters $n \in \mathbb{R}^3$, the plane and non-plane probability p, and the plane instance embedding vector $\mathcal{E} \in \mathbb{R}^\epsilon$ ($\epsilon = 8$ in this

paper), respectively. In the inference stage, we select the predicted plane instances as outputs if their plane probabilities are larger than 0.5.

3.2. Instance-to-Pixel Plane Segmentation

After achieving plane instances from the Transformer branch, it is required to obtain the segmentation mask for each plane. Inspired by PlaneAE [45], we apply the idea of associative embedding and calculate pixel-wise embedding vectors via a Plane Embedding Decoder as used in PlaneAE in the convolution branch. However, instead of achieving plane instance segmentation masks via the clustering algorithm based on pixel embedding, we take a simple instance-to-pixel method by directly comparing the distance between pixel embedding and plane instance embeddings. Then, we assign a pixel to one plane instance which is nearest to it in the embedding space if their distance is lower than a threshold T which is set to 1.0 in this paper.

3.3. Network Training

Auxiliary Plane Center Prediction. We find that predicting auxiliary plane centers in the training stage is beneficial (we will discuss this point in the experimental ablation study).The ground truth plane center of each plane is defined as the normalized average image coordinate of its corresponding plane pixels. In practice, we predict plane centers for both plane instances and pixels. The instance-wise plane centers $c \in \mathbb{R}^{2 \times K}$ are predicted from the outputs of the Structure-guided Plane Decoder followed by a linear layer. The pixel-wise plane centers $\mathcal{C} \in \mathbb{R}^{H \times W \times 2}$ are predicted via a convolution decoder which has the same architecture as the Plane Embedding Decoder.

Bipartite Matching. In our network, we predict a fixed number of K plane instances $\{s_p^i = (p_i, n_i, c_i, \mathcal{E}_i)\}_{i=1}^K$, which are generally more than the ground truth plane instances $\{\hat{s}_p^j = (\hat{p}_j, \hat{n}_j, \hat{c}_j)\}_{j=1}^M$ in the image ($K \geq M$). Here, $\hat{p}_j \in \{0, 1\}$ (0: non-plane, 1: plane). Therefore, to effectively train the network, we first pad the ground truth plane instances to the number K with non-plane instances. Then, we find a bipartite matching between the predict plane instances and the ground truth plane instances via searching for a permutation $\hat{\sigma}$ which has a minimization matching cost as:

$$\hat{\sigma} = \arg\min_\sigma \sum_{i=1}^K D\left(\hat{s}_p^i, s_p^{\sigma(i)}\right), \qquad (2)$$

where $\sigma(i)$ indicates the matched index of the predict plane instance to the ground truth plane instance $\hat{s}_p^i$ and D is a function that measures the matching cost between two plane

instances. The formulation of D is defined as:

$$D = -p_{\sigma(i)}\left(\hat{p}_i\right) + \mathbb{1}_{\{\hat{p}_i=1\}} L_1\left(\hat{n}_i, n_{\sigma(i)}\right)$$
$$+ \mathbb{1}_{\{\hat{p}_i=1\}} \omega L_2\left(\hat{c}_i, c_{\sigma(i)}\right), \quad (3)$$

where $\mathbb{1}_{\mathbf{a}}$ is an indicator function taking 1 if $\mathbf{a}$ is true and 0 otherwise. ω is a weight to balance the magnitude for cost terms and is set to 2 in this paper.

Loss Functions. Our network is trained via losses based on the bipartite matching results and consists of five parts: a classification loss, a plane parameter loss, a plane center loss, an embedding loss, and a pixel-wise depth loss. The classification loss is defined as:

$$\mathcal{L}_{\text{cls}}^{(i)} = -\log p_{\hat{\sigma}(i)}\left(\hat{p}_i\right). \quad (4)$$

The plane parameter loss is defined as:

$$\mathcal{L}_{\text{parm}}^{(i)} = \mathbb{1}_{\{\hat{p}_i=1\}} L_1\left(\hat{n}_i, n_{\hat{\sigma}(i)}\right) +$$
$$\mathbb{1}_{\{\hat{p}_i=1\}} \beta_1\left(1 - \cos\left(\hat{n}_i, n_{\hat{\sigma}(i)}\right)\right) +$$
$$\mathbb{1}_{\{\hat{p}_i=1\}} \beta_2 \sum_{q \in Q_i} \|n_{\hat{\sigma}(i)}^T q - 1\|, \quad (5)$$

where Q_i is the set of 3D points calculated from pixels belonging to ground truth plane instance $\hat{s}_p^i$ via the ground truth depth map. $\beta_1 = 5$ and $\beta_2 = 2$ in this paper. The embedding loss is defined according to PlaneAE [45] which consists of two terms, called pull loss and push loss. The formulation is as follows:

$$\mathcal{L}_{\text{emb}}^{(i)} = \mathcal{L}_{\text{pull}}^{(i)} + \mathcal{L}_{\text{push}}^{(i)}, \quad (6)$$

where

$$\mathcal{L}_{\text{pull}}^{(i)} = \mathbb{1}_{\{\hat{p}_i=1\}} \sum_{g \in G_i} \max\left(\|\mathcal{E}_{\hat{\sigma}(i)} - g\| - \delta_1, 0\right), \quad (7)$$

$$\mathcal{L}_{\text{push}}^{(i)} = \mathbb{1}_{\{\hat{p}_i=1\}} \sum_{j=1, j \neq i}^{M} \max\left(\delta_2 - \|\mathcal{E}_{\hat{\sigma}(i)} - \mathcal{E}_{\hat{\sigma}(j)}\|, 0\right) \quad (8)$$

Here, G_i is a set of embedding vectors of pixels belonging to the ground truth plane instance $\hat{s}_p^i$. $\mathcal{E}_{\hat{\sigma}(i)}$ and $\mathcal{E}_{\hat{\sigma}(j)}$ are the embeddings of predicted plane instances. δ_1 and δ_2 are the margin for pull and push losses and are set to 0.5 and 1.5 in this paper. The depth loss is defined as:

$$\mathcal{L}_{\mathcal{D}} = L_1\left(d, \hat{d}\right), \quad (9)$$

where d indicates the predicted depth map from Depth Decoder and $\hat{d}$ is the ground truth depth map. The plane center losses of plane instances and pixels can be defined as:

$$\mathcal{L}_{\text{c}}^{(i)} = \mathbb{1}_{\{\hat{p}_i=1\}} L_2\left(\hat{c}_i, c_{\hat{\sigma}(i)}\right), \quad (10)$$

$$\mathcal{L}_{\mathcal{C}} = L_2\left(\mathcal{C}, \hat{\mathcal{C}}\right), \quad (11)$$

where $\hat{c}_i$ and $\hat{\mathcal{C}}$ indicate instance-wise and pixel-wise ground truth plane centers.

The final loss is the sum of the above losses and is defined as:

$$\mathcal{L} = \sum_{i=1}^{K}\left(\mathcal{L}_{\text{cls}}^{(i)} + \mathcal{L}_{\text{parm}}^{(i)} + \mathcal{L}_{\text{c}}^{(i)} + \lambda \mathcal{L}_{\text{emb}}^{(i)}\right) + \mathcal{L}_{\mathcal{D}} + \mathcal{L}_{\mathcal{C}},$$
$$(12)$$

where λ is a weight to balance the magnitude for loss terms and is set to 5 in this paper. Besides, according to DETR [3], we also predict plane instances from O_l in the Structure-guided Plane Decoder and serve it as the intermediate supervision. The weights of the intermediate supervision is set to 0.2.

4. Experiments

In this section, we conduct experiments on the public ScanNet dataset [5] and NYUv2 dataset [28] to evaluate and analyze the performance of the proposed PlaneTR.

4.1. Dataset and Metrics

ScanNet. The ScanNet dataset [5] is a large indoor RGB-D video dataset. To train and test our network, we use the plane data processed by PlaneNet [19] from the original ScanNet dataset. The processed dataset contains 50,000 training and 760 testing images with an image size of 256×192.

NYUv2 and its Variant. The NYUv2 dataset [28] is an indoor RGB-D dataset. The official NYUv2 dataset contains 795 training and 654 testing images with ground truth depth maps. The image size is 640×480. To evaluate the plane recovery performance, we use a variant of the NYUv2 dataset, namely NYUv2-Plane, which is generated by PlaneAE [45] with an image size of 256×192.

Evaluation Metrics. Following PlaneNet [19], we apply plane and pixel recalls to evaluate the plane detection performance. The plane (pixel) recalls are defined as the percentage of the correctly predicted ground truth plane instances (pixels). A plane is considered as a correctly predicted plane if its Intersection over Union (IOU) is larger than 0.5 and the depth (surface normal) error is less than a threshold. Besides, to further evaluate the segmentation performance, we apply three popular metrics used in segmentation [1, 43] called rand index (RI), variation of information (VI) and segmentation covering (SC). On the NYUv2 dataset, we also evaluate the accuracy of depths inferred from reconstructed planes. Following [8], we apply these popular depth metrics: mean absolution relative error (Rel), mean $\log_{10}$ error ($\log_{10}$), root mean square error (RMSE) and the threshold accuracy ($\delta_i < 1.25^i$, $i = 1, 2, 3$).

4.2. Implementation Detail

Our network is implemented with Pytorch [23] and trained with the Adam optimizer [13]. We train our network on the ScanNet training set with a total of 60 epochs on 3 GPUs. The initial learning rate is set to 1×10^{-4} and reduced to half after every 15 epochs. The batch size is set to 24 for each GPU and the weight decay is set to 1×10^{-5}.

4.3. Results on the ScanNet Dataset

Qualitative Results. Figure 4 shows the plane detection and reconstruction results of our method. PlaneTR can effectively detect and reasonably reconstruct the planar structures in scenes. In the first 4 rows of Figure 5, we visualize the plane segmentation results of our PlaneTR and the state-of-the-art CNN-based approaches (including PlaneNet [19], PlaneAE [45] and PlaneRCNN [18]) on the ScanNet dataset [5]. PlaneNet and PlaneAE usually incorrectly merge different planes. PlaneRCNN tends to divide a large plane into several plane instances such as the wall and floor shown in the third row of Figure 5. Besides, the plane segmentation masks detected by these methods are always incomplete. In contrast to those methods, our PlaneTR can correctly detect holistic plane structures with complete segmentation masks.

Quantitative Results. We then compare the quantitative performance of our method with PlaneNet [19], PlaneAE [45], and PlaneRCNN [18]. Note that the PlaneRCNN is trained on a new benchmark constructed from the ScanNetv2 dataset [5] which is different from other methods. Thus, we only treat PlaneRCNN as a reference when evaluating on the ScanNet dataset and mainly compare our PlaneTR with the remaining methods. In Table 1, we show the results of plane segmentation performance. Our method significantly outperforms all other methods. In Figure 3, we show the plane and pixel recalls of various methods and the depth error threshold varies from 0m to 0.6m with an increment of 0.05m. Our method significantly outperforms PlaneNet with all thresholds. When compared with PlaneAE, our method performs better in pixel recalls which are benefited from our complete predicted plane segmentation masks as shown in Figure 5. For the plane recalls, our method performs slightly lower than PlaneAE. It is mainly because that our method tends to detect holistic plane structures which may lead to a missing of some planes such as the white board on the table as shown in the fourth row of Figure 5.

4.4. Results on the NYUv2 Dataset

To verify the generalization of our method, we first evaluate the plane segmentation performance of our method on the NYUv2-Plane dataset. In Table 1, we show the comparison results against PlaneNet [19], PlaneAE [45], and Plan-
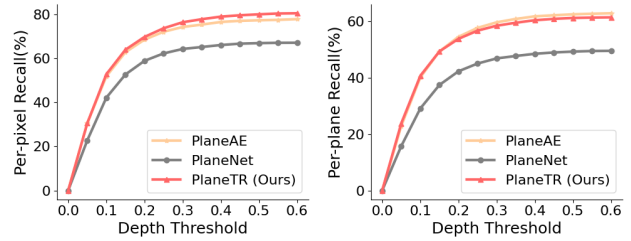

Figure 3. Plane and pixel recalls on the ScanNet dataset.

Method	ScanNet			NYUv2-Plane		
	VI $\downarrow$	RI $\uparrow$	SC $\uparrow$	VI $\downarrow$	RI $\uparrow$	SC $\uparrow$
PlaneNet [19]	1.259	0.858	0.716	1.813	0.753	0.558
PlaneRCNN [18]	1.337	0.845	0.690	1.596	0.839	0.612
PlaneAE [45]	1.025	0.907	0.791	1.393	0.887	0.681
Ours	**0.767**	**0.925**	**0.838**	**1.110**	**0.898**	**0.726**

Table 1. Comparison of plane instance segmentation results on the ScanNet dataset and NYUv2-Plane dataset.

Method	PlaneNet	PlaneAE	PlaneRCNN	Ours
Rel$\downarrow$	0.236	0.205	**0.183**	0.195
$\log_{10} \downarrow$	0.124	0.097	**0.076**	0.095
RMSE$\downarrow$	0.913	0.820	**0.619**	0.803
$\delta_1 \uparrow$	53.0	61.3	**71.8**	63.3
$\delta_2 \uparrow$	78.3	87.2	**93.1**	88.2
$\delta_3 \uparrow$	90.4	95.8	**98.3**	96.1

Table 2. Depth accuracy comparison on the NYUv2 dataset.

Settings		Per-plane recall (depth & normal)				Plane Segmentation		
Line Segments	Plane Center	@0.10 m	@0.60 m	@5°	@30°	VI $\downarrow$	RI $\uparrow$	SC $\uparrow$
		35.71	57.03	38.77	56.40	0.885	0.913	0.809
	✓	38.07	60.33	40.85	59.81	0.778	0.923	0.833
✓		38.44	58.82	41.72	58.36	0.831	0.919	0.823
✓	✓	**40.74**	**61.49**	**43.14**	**60.68**	**0.767**	**0.925**	**0.838**

Table 3. Ablation studies of proposed PlaneTR on the ScanNet dataset.

eRCNN [18]. Our method outperforms all other methods with a large margin in all metrics. Then, we further evaluate the pixel-wise depth accuracy on the NYUv2 dataset. In this experiment, we achieve the final depth map by first calculating the depth values of plane regions via the predicted 3D planar parameters and then filling out the non-plane region with predicted pixel-wise depth from the Depth Decoder. As shown in Table 2, our method outperforms PlaneNet and PlaneAE while PlaneRCNN achieves the best depth performance. Note that PlaneRCNN is trained and tested with the image size of 640×640 which is larger than the image size used in other methods (256×192). Such a large image size is beneficial to PlaneRCNN on the depth performance. The comparison of qualitative plane segmentation results on the NYUv2-Plane dataset is shown in the last two rows of Figure 5. Our method can detect complete and reasonable planes in the scene which are better than other methods.
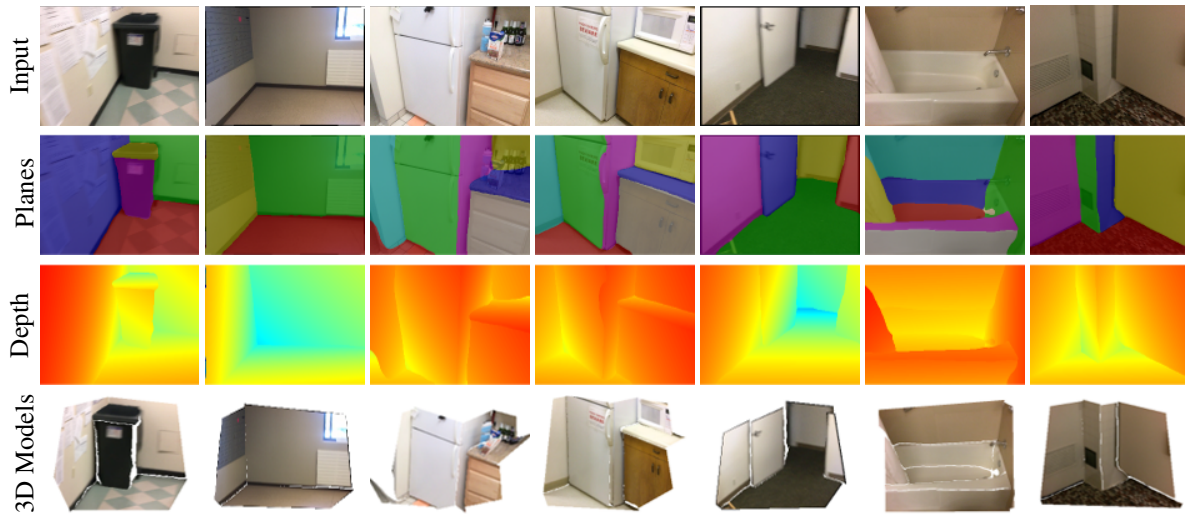
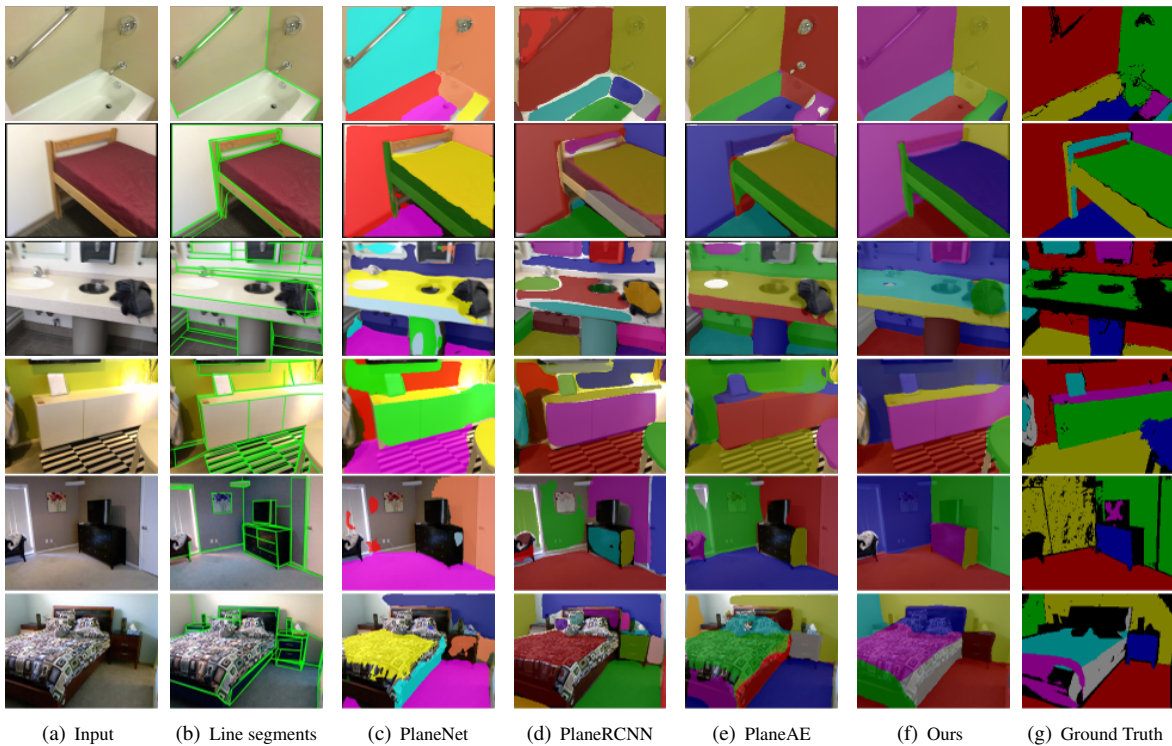Figure 4. 3D plane reconstruction results of PlaneTR on the ScanNet dataset.

(a) Input (b) Line segments (c) PlaneNet (d) PlaneRCNN (e) PlaneAE (f) Ours (g) Ground Truth

Figure 5. Comparison of plane instance segmentation results on the ScanNet dataset (rows: 1-4) and NYUv2-Plane dataset (rows: 5-6).

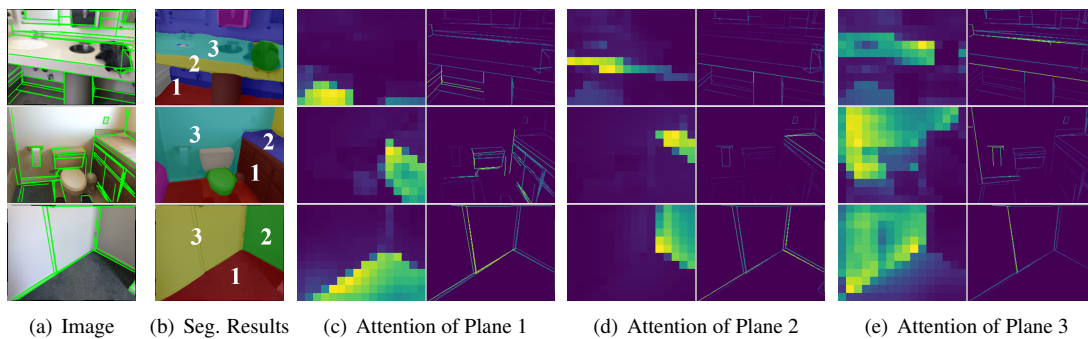(a) Image (b) Seg. Results (c) Attention of Plane 1 (d) Attention of Plane 2 (e) Attention of Plane 3

Figure 6. Context and line segments attention maps in the Structure-guided Plane Decoder.

| (a) Image | (b) w/o line | (c) w/ line 3 + 4 | (d) w/ line 1 + 2 |

| (e) Image | (f) w/o line | (g) w/ line 1 | (h) w/ line 2 |

Figure 7. Illustration of the plane detection guided by line segments in PlaneTR.

4.5. Ablation Studies

In this section, we conduct experiments on the ScanNet dataset to validate various components in PlaneTR. We evaluate the performance by using plane recalls based on depth and normal thresholds, respectively. Specifically, the depth thresholds are set to 0.1m and 0.6m. The normal thresholds are set to $5°$ and $30°$. Besides, we also use VI, RI, and SC to evaluate the plane segmentation performance.

Auxiliary Plane Center Prediction. We first validate the effectiveness of auxiliary plane center prediction which we used to assist the learning of our method. As shown in Table 3, learning extra plane centers during training is beneficial for our network to achieve better results. One main reason is that plane centers provide important location information of planes which can help the plane queries to learn where the planes are in an image.

Line Segments. Then we further validate the effectiveness of line segments used in our Structure-guided Plane Decoder. As shown in Table 3, models without line segments are trained by only taking the context sequence as input to predict plane instances. We can observe that both plane recalls and plane segmentation results are improved by applying line segments into the network. In Figure 7, we list two examples to further show how the input line segments guide the detection of planes in our full model. Here, the setting 'w/o line' specifically means that we input an empty line sequence into the network. By inputting relative line segments (e.g. line 1 and line 2 in the first example) into the network, the missed plane can be successfully detected. It demonstrates that our Structure-guided Plane Decoder has learned the relationship between planes and line segments.

Plane Query Number. We also experiment the influence of plane query numbers and the results are shown in Table 4. We can observe that as the number of plane queries increases from 20 to 40, the plane recalls have been degraded to some extent. One reasonable explanation is that the plane number of each image in the ScanNet dataset is lower than 20 and a large number of plane queries will lead to an im-

Number	Per-plane recall			
	@0.10 m	@0.60 m	@5°	@30°
20	**40.74**	**61.49**	43.14	**60.68**
30	40.58	61.08	**43.36**	60.35
40	39.25	60.11	42.51	59.41

Table 4. Plane recalls of PlaneTR with different numbers of plane queries on the ScanNet dataset.

balance between positive and negative samples which may degrade the network performance.

Attention Visualization. In Figure 6, we show attention maps between the predicted plane instances and the input context sequence (line segment sequence) in the Structure-guided Plane Decoder. As we can see, for the plane instance such as the Plane 1 in the second row of Figure 6, its attention with the context sequence mainly focuses on those pixels which are spatially located on the plane region. By contrast, its attention with the input line segments shows more complex relationships and global structure information of the scene.

5. Conclusion

In this paper, we present a novel model of Transformers, PlaneTR, that simultaneously leverages the context information and global structure cues to recover 3D planes from a single image with state-of-the-art performance obtained. Our PlaneTR represents context features and line segments with tokenized sequences instead of dense maps to address the plane recovery problem in a sequence-to-sequence manner. Different from the existing CNNs-based plane recovery or structure-guided learning methods, such a tokenized representation of line segments enables our network to explicitly exploit the holistic structure cues in the end task. In our future work, we plan to study the utilizing of geometric structures in other geometric vision tasks.

Acknowledgement

References

[1] Pablo Arbelaez, Michael Maire, Charless C. Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 5

[2] Olga Barinova, Vadim Konushin, Anton Yakubenko, KeeChang Lee, Hwasup Lim, and Anton Konushin. Fast automatic single-view 3-d reconstruction of urban scenes. In *European Conference on Computer Vision, ECCV*, volume 5303, pages 100–113, 2008. 1, 2

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision, ECCV*, volume 12346, pages 213–229, 2020. 2, 3, 4, 5

[4] Denis Chekhlov, Andrew P. Gee, Andrew Calway, and Walterio W. Mayol-Cuevas. Ninja on a plane: Automatic discovery of physical planes for augmented reality using visual SLAM. In *IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR*, pages 153–156, 2007. 1

[5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2432–2443, 2017. 2, 5, 6

[6] Erick Delage, Honglak Lee, and Andrew Y. Ng. Automatic single-image 3d reconstructions of indoor manhattan world scenes. In *Robotics Research: Results of the 12th International Symposium, ISRR*, volume 28, pages 305–321, 2005. 1, 2

[7] 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 is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 3

[8] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *IEEE International Conference on Computer Vision, ICCV*, pages 2650–2658, 2015. 5

[9] David Ford Fouhey, Abhinav Gupta, and Martial Hebert. Unfolding an indoor origami world. In *European Conference on Computer Vision, ECCV*, volume 8694, pages 687–702, 2014. 2

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV*, pages 2980–2988, 2017. 2

[11] Ziyu Jiang, Buyu Liu, Samuel Schulter, Zhangyang Wang, and Manmohan Chandraker. Peek-a-boo: Occlusion reasoning in indoor scenes with plane representations. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 110–118, 2020. 1

[12] Lei Jin, Yanyu Xu, Jia Zheng, Junfei Zhang, Rui Tang, Shugong Xu, Jingyi Yu, and Shenghua Gao. Geometric structure based and regularized depth estimation from 360 indoor imagery. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 886–895, 2020. 2, 3

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations, ICLR*, 2015. 6

[14] Pierre-Alain Langlois, Alexandre Boulch, and Renaud Marlet. Surface reconstruction from 3d line segments. In *International Conference on 3D Vision, 3DV*, pages 553–563, 2019. 2

[15] David C. Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2136–2143, 2009. 1, 2

[16] Yikai Li, Jiayuan Mao, Xiuming Zhang, Bill Freeman, Josh Tenenbaum, Noah Snavely, and Jiajun Wu. Multi-plane program induction with 3d box priors. In *Neural Information Processing Systems, NeurIPS*, 2020. 1, 2

[17] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 9128–9137. IEEE, 2020. 3

[18] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4450–4459, 2019. 1, 2, 6

[19] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. Planenet: Piece-wise planar reconstruction from a single RGB image. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2579–2588, 2018. 1, 2, 5, 6

[20] Lingni Ma, Christian Kerl, Jörg Stückler, and Daniel Cremers. CPA-SLAM: consistent plane-model alignment for direct RGB-D SLAM. In *IEEE International Conference on Robotics and Automation, ICRA*, pages 1285–1291, 2016. 1

[21] Branislav Micusík, Horst Wildenauer, and Markus Vincze. Towards detection of orthogonal planes in monocular images of indoor environments. In *IEEE International Conference on Robotics and Automation, ICRA*, pages 999–1004, 2008. 1, 2

[22] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Neural Information Processing Systems, NeurIPS*, pages 2277–2287, 2017. 2

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019. 6

[24] Yiming Qian and Yasutaka Furukawa. Learning pairwise inter-plane relations for piecewise planar reconstruction. In *European Conference on Computer Vision, ECCV*, volume 12352, pages 330–345, 2020. 1, 2

[25] Yiming Qian, Srikumar Ramalingam, and James H. Elder. LS3D: single-view gestalt 3d surface reconstruction from manhattan line segments. In *Asian Conference on Computer Vision, ACCV*, volume 11364, pages 399–416, 2018. 1, 2

[26] Xuchong Qiu, Yang Xiao, Chaohui Wang, and Renaud Marlet. Pixel-pair occlusion relationship map (P2ORM): formulation, inference and application. In *European Conference on Computer Vision, ECCV*, volume 12349, pages 690–708, 2020. 2

[27] Michaël Ramamonjisoa and Vincent Lepetit. Sharpnet: Fast and accurate recovery of occluding contours in monocular depth estimation. In *IEEE International Conference on Computer Vision Workshops, ICCV Workshops*, pages 2109–2118, 2019. 2

[28] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision, ECCV*, volume 7576, pages 746–760, 2012. 2, 5

[29] Xiao Song, Xu Zhao, Liangji Fang, Hanwen Hu, and Yizhou Yu. Edgestereo: An effective multi-task learning network for stereo matching and edge detection. *International Journal of Computer Vision*, 128(4):910–930, 2020. 2, 3

[30] Yuichi Taguchi, Yong-Dian Jian, Srikumar Ramalingam, and Chen Feng. Point-plane SLAM for hand-held 3d sensors. In *IEEE International Conference on Robotics and Automation, ICRA*, pages 5182–5189, 2013. 1

[31] Grace Tsai, Changhai Xu, Jingen Liu, and Benjamin Kuipers. Real-time indoor scene understanding using bayesian filtering with motion cues. In *IEEE International Conference on Computer Vision, ICCV*, pages 121–128, 2011. 1

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems, NeurIPS*, pages 5998–6008, 2017. 2, 3

[33] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan L. Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. *CoRR*, abs/2012.00759, 2020. 3

[34] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *CoRR*, abs/1908.07919, 2019. 3

[35] Rui Wang, David Geraghty, Kevin Matzen, Richard Szeliski, and Jan-Michael Frahm. Vplnet: Deep single view normal estimation with vanishing points and lines. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 686–695, 2020. 2

[36] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7794–7803, 2018. 2

[37] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *CoRR*, abs/2011.14503, 2020. 3

[38] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: convolutional block attention module. In *European conference on computer vision, ECCV*, volume 11211, pages 3–19, 2018. 2

[39] Yifan Xu, Weijian Xu, David Cheung, and Zhuowen Tu. Line segment detection using transformers without edges. *CoRR*, abs/2101.01909, 2021. 2, 3

[40] Nan Xue, Song Bai, Fudong Wang, Gui-Song Xia, Tianfu Wu, and Liangpei Zhang. Learning attraction field representation for robust line segment detection. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1595–1603, 2019. 2

[41] Nan Xue, Song Bai, Fudong Wang, Gui-Song Xia, Tianfu Wu, and Liangpei Zhang. Learning regional attraction for line segment detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 43(6):1998–2013, 2021. 2

[42] Nan Xue, Tianfu Wu, Song Bai, Fudong Wang, Gui-Song Xia, Liangpei Zhang, and Philip H. S. Torr. Holistically-attracted wireframe parsing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2785–2794, 2020. 2, 4

[43] Fengting Yang and Zihan Zhou. Recovering 3d planes from a single image via convolutional neural networks. In *European Conference on Computer Vision, ECCV*, volume 11214, pages 87–103, 2018. 1, 2, 5

[44] Shichao Yang and Sebastian A. Scherer. Monocular object and plane SLAM in structured environments. *IEEE Robotics & Automation Letters*, 4(4):3145–3152, 2019. 1

[45] Zehao Yu, Jia Zheng, Dongze Lian, Zihan Zhou, and Shenghua Gao. Single-image piece-wise planar 3d reconstruction via associative embedding. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1029–1037, 2019. 1, 2, 3, 4, 5, 6

[46] Aamer Zaheer, Maheen Rashid, Muhammad Ahmed Riaz, and Sohaib Khan. Single-view reconstruction using orthogonal line-pairs. *Computer Vision and Image Understanding.*, 172:107–123, 2018. 2

[47] Yichao Zhou, Haozhi Qi, and Yi Ma. End-to-end wireframe parsing. In *IEEE International Conference on Computer Vision, ICCV*, pages 962–971. IEEE, 2019. 2

[48] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. *CoRR*, abs/2010.04159, 2020. 3