VPDETR: End-to-End Vanishing Point DEtection TRansformers

Taiyan Chen, Xianghua Ying*, Jinfa Yang, Ruibin Wang, Ruohao Guo, Bowei Xing, Ji Shi

National Key Laboratory of General Artificial Intelligence, School of Intelligence Science and Technology, Peking University
chenty@stu.pku.edu.cn, xhying@pku.edu.cn

Abstract

In the field of vanishing point detection, previous works commonly relied on extracting and clustering straight lines or classifying candidate points as vanishing points. This paper proposes a novel end-to-end framework, called VPDETR (Vanishing Point DEtection TRansformer), that views vanishing point detection as a set prediction problem, applicable to both Manhattan and non-Manhattan world datasets. By using the positional embedding of anchor points as queries in Transformer decoders and dynamically updating them layer by layer, our method is able to directly input images and output their vanishing points without the need for explicit straight line extraction and candidate points sampling. Additionally, we introduce an orthogonal loss and a cross-prediction loss to improve accuracy on the Manhattan world datasets. Experimental results demonstrate that VPDETR achieves competitive performance compared to state-of-the-art methods, without requiring post-processing.

Introduction

Vanishing points are intersection points of parallel lines in the 3D world projected onto the 2D image under the pinhole camera model. Vanishing point detection is a fundamental problem in 3D vision. An accurate vanishing point detection algorithm can benefit many tasks, such as camera calibration (Cipolla, Drummond, and Robertson 1999; Antone and Teller 2000; Grammatikopoulos, Karras, and Petsa 2007), wireframe parsing (Zhou et al. 2019b; Zhou, Qi, and Ma 2019), 3D reconstruction (Guillou et al. 2000), photo forensics (O'brien and Farid 2012), object detection (Hoiem, Efros, and Hebert 2008), autonomous driving (Lee et al. 2017), and visual SLAM (Davison et al. 2007; Li et al. 2019a).

As shown in Figure 1, traditional vanishing point detection methods (Kogecka and Zhang 2002; Tardif 2009) generally include three steps: line/contour detection, line clustering/classification, and vanishing point regression. In recent years, deep learning approaches (Borji 2016; Zhai, Workman, and Jacobs 2016; Chang, Zhao, and Itti 2018; Zhang et al. 2018; Zhai, Workman, and Jacobs 2016; Kluger et al. 2017; Zhou et al. 2019a; Liu, Zhou, and Zhao 2021; Tong et al. 2022; Lin et al. 2022) have shown great potential in this task. Some of the most recent state-of-the-art works suffer from some problems. NeurVPS (Zhou et al. 2019a) and Lin et al. (Lin et al. 2022) need to sample a large number of candidate points in the Gaussian sphere and classify whether they are vanishing points which is slow during inference. TLC (Tong et al. 2022) uses Transformer to classify straight lines into four categories, requiring additional line category annotations. Post-processing is also required to enhance performance. Although these works are capable of end-to-end training, they still require additional post-processing steps to produce the vanishing point as the final output. In other words, they are not entirely end-to-end because the vanishing point is not directly predicted when an image is inputted.

DETR (Carion et al. 2020) is a highly influential work in the field of object detection. It uses a transformer encoder-decoder architecture for end-to-end object detection. In DETR, object queries are treated as learnable queries, allowing the model to make predictions in a parallel manner, which improves the efficiency. It also uses a set loss function to handle the problem of object permutation, thus eliminates many hand-designed components such as NMS (Non-Maximum Suppression). Some follow-up works such as Deformable DETR (Zhu et al. 2020) and DAB-DETR (Liu et al. 2022) have been proposed to improve its convergence and performance. There are also works on adapting DETR for other applications (Xu et al. 2021; Misra, Girdhar, and Joulin 2021; Prangemeier, Reich, and Koeppl 2020). Drawing inspiration from these studies, we modified DETR and its variations to address the problem of vanishing point detection.

In this paper, we propose VPDETR, an end-to-end framework for vanishing point detection. Our modification focuses on the decoder part, utilizing anchor vanishing point positional encoding as queries and updating point coordinates layer by layer. Our method is based on set prediction, therefore overcomes the limitations of the Manhattan world assumption, i.e. there are three orthogonal vanishing points in each image, enabling it to predict vanishing points for non-Manhattan world data as well. Specifically, for Manhattan world data, we introduce cross-prediction loss and orthogonal loss to improve the results. Moreover, our method achieves fast inference speed of 16 FPS, outperforming state-of-the-art (SOTA) method (Zhou et al. 2019a)

*Corresponding Author.

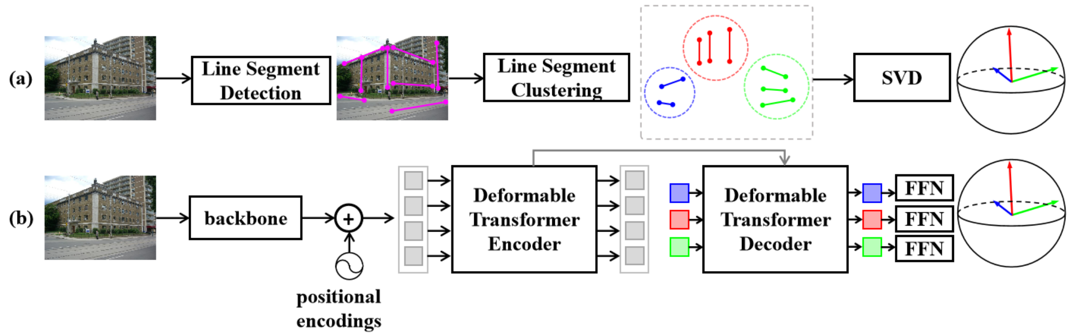

Figure 1: Pipeline comparison between: (a) Traditional vanishing point detection framework, (b) Our proposed Vanishing Point DEtection TRansformers (VPDETR).

that rely on sampling candidate points and classifying them one by one (0.5 FPS).

Our work presents the following contributions: (1) We introduce VPDETR, a novel fully end-to-end framework for vanishing point detection, which is capable of handling both Manhattan and non-Manhattan world datasets by treating vanishing point detection as a set prediction problem. (2) For the Manhattan world assumption, we propose a cross-prediction loss and orthogonal loss to improve accuracy. (3) Our experiments demonstrate that VPDETR achieves a good balance of accuracy and inference speed, comparable to state-of-the-art models. And we provide ablation experiments to showcase the effectiveness of the proposed components.

Related Work

Vanishing Point Detection. Vanishing point detection is a fundamental problem in computer vision that locates the point of convergence of parallel lines in an image. The problem was first introduced in (Barnard 1983), and has since been tackled using various approaches. The dominant approach for vanishing point detection is line-based, which generally consist of three steps. Firstly, line detection (Canny 1986; Von Gioi et al. 2008) or contour detection (Arbelaez et al. 2010) is performed. Secondly, the parametric lines are clustered using various algorithms such as RANSAC (Bolles and Fischler 1981; Wu et al. 2021), Hough transform (Lutton, Maitre, and Lopez-Krahe 1994), J-Linkage (Tardif 2009), EM (Kogecka and Zhang 2002; Košecká and Zhang 2002), and dual space (Lezama et al. 2014). Finally, geometry is used to estimate the vanishing point.

Recently, learning-based methods (Borji 2016; Chang, Zhao, and Itti 2018; Zhang et al. 2018; Shi et al. 2019; Zhai, Workman, and Jacobs 2016; Kluger et al. 2017; Li et al. 2021; Kluger et al. 2020) have been shown to be effective for estimating the vanishing point in images with complex backgrounds and cluttered scenes. NeurVPS (Zhou et al. 2019a) performs best, which introduces conic convolution to enhance the detection accuracy, albeit at the expense of slow inference speed. To address this issue, Liu et al.(Liu, Zhou, and Zhao 2021) proposes an efficient conic convolution in VaPiD. Lin et al. (Lin et al. 2022) incorporates the Hough transform and the Gaussian sphere into deep neural networks to improve model generalization. TLC (Tong et al. 2022) is the first to apply Transformer (Vaswani et al. 2017) to vanishing point detection, utilizing both the geometric and contextual features of lines to improve the accuracy.

DETR and Its Variants. Carion et al. (Carion et al. 2020) introduced DETR (DEtection TRansformer), a Transformer-based end-to-end object detector that eliminates the need for hand-designed components such as anchor design and NMS. Several studies delve into a deeper understanding of the decoder queries in DETR. Many papers associate queries with spatial position from different angles. Deformable DETR (Zhu et al. 2020) predicts 2D anchor points and employs a deformable attention mechanism that focuses on specific sampling points around a reference point. Efficient DETR (Yao et al. 2021) selects the top K positions from the dense prediction of the encoder to improve decoder queries. DAB-DETR (Liu et al. 2022) extends the representation of queries from 2D anchor points to 4D anchor box coordinates, and enabling dynamic updates of boxes in each layer of the decoder. More recently, DN-DETR (Li et al. 2022) introduced a denoising training method to accelerate DETR training. And DINO (Zhang et al. 2022) further introduces three tricks to get better performance.

Meanwhile, there have also been studies exploring the adaptation of DETR for other applications such as line segment detection (Xu et al. 2021), 3D object detection (Prangemeier, Reich, and Koeppl 2020), and instance segmentation (Prangemeier, Reich, and Koeppl 2020). In this work, we have sensibly adapted DETR to make it appropriate for vanishing point detection tasks, resulting in precise predictions and a truly end-to-end output.

Method

Overview

Our model includes a CNN backbone, Transformer encoders and decoders, and multilayer perceptron (MLP) prediction heads for vanishing points and confidence score. The main improvement of our model is in the decoder part, as illustrated in Figure 2.

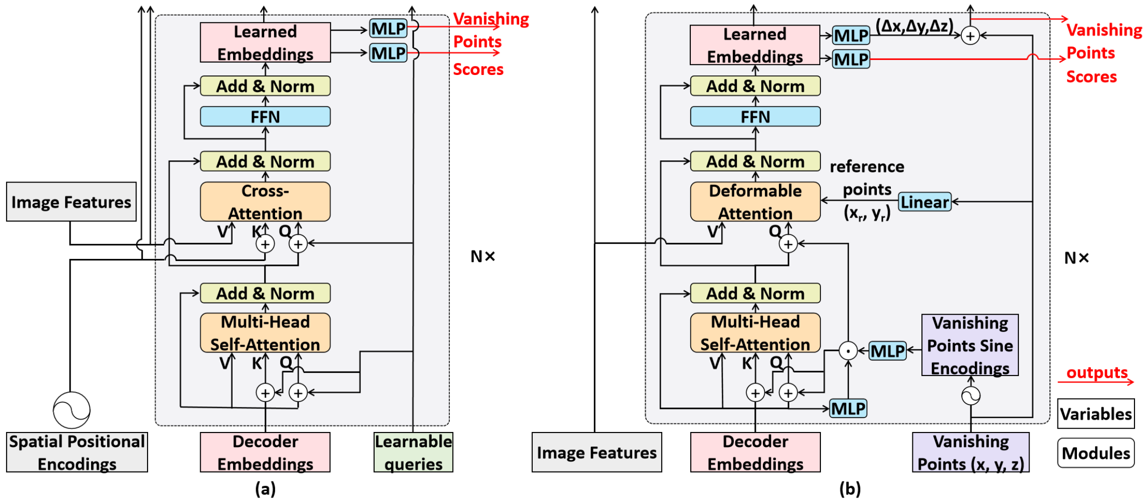

Figure 2: From left to right: (a) Decoder framework of vanilla DETR, (b) Decoder framework of our proposed VPDETR.

Given an image, the features are extracted using a CNN backbone and refined by the Transformer encoders. The decoder then employs dual queries, which consist of both positional queries (anchor points) and content queries (decoder embeddings), to determine the vanishing points that correspond to the anchor points. The dual queries are iteratively refined across multiple layers, slowly approaching the target ground-truth vanishing points. Finally, the outputs of the final decoder layer are utilized to predict confidence scores and vanishing points through prediction heads. A bipartite graph matching is then conducted to calculate the loss, following the procedure used in DETR.

Anchor Point Learning and Iterative Updating

Inspired by (Liu et al. 2022), we propose to directly learn anchor points and derive positional queries from these anchors.

We use the Gaussian sphere representation of the vanishing point, the details of which can be found in Section 3.2 of NeurVPS (Zhou et al. 2019a). We denote $A_q = (x_q, y_q, z_q)$ as the q-th anchor point, where $x_q, y_q, z_q \in \mathbb{R}$, take values in $[-1, 1]$. The positional query (Meng et al. 2021; Liu et al. 2022) P_q associated with it can be obtained from

$$P_q = \mathrm{MLP}^p(\mathrm{Cat}(\mathrm{PE}(x_q), \mathrm{PE}(y_q), \mathrm{PE}(z_q))) \tag{1}$$

where $P_q \in \mathbb{R}^D$, Cat means concatenate, and PE means sinusoidal positional encoding:

$$\mathrm{PE}(x)_{2i} = \sin\left(\frac{x}{T^{2i/D}}\right), \quad \mathrm{PE}(x)_{2i+1} = \cos\left(\frac{x}{T^{2i/D}}\right) \tag{2}$$

where T is temperature, a hyper-parameter, and i denotes the index in the vector. In our implementations, PE maps a float to a vector with $D/2$ dimensions as $\mathrm{PE}: \mathbb{R} \to \mathbb{R}^{D/2}$, and MLP^p projects a $3D/2$ dimensional vector into D dimensions $\mathrm{MLP}^p: \mathbb{R}^{3D/2} \to \mathbb{R}^D$. Following (Liu et al. 2022), we learn a $\mathrm{MLP}^s: \mathbb{R}^D \to \mathbb{R}^D$ to obtain a scale vector conditional on the content query C_q and multiply it with the positional encoding element by element:

$$\tilde{P}_q = P_q \cdot \mathrm{MLP}^s(C_q) \tag{3}$$

In the self-attention module, queries, keys, and values are formed as:

$$Q_q = C_q + \tilde{P}_q, \quad K_q = C_q + \tilde{P}_q, \quad V_q = C_q \tag{4}$$

after the self-attention operation, skip-connection and normalization, we can get the new content information. Next, we use the deformable attention module proposed in Deformable DETR (Zhu et al. 2020) to perform cross-attention operations, as illustrated in Figure 2. Finally, the learned embeddings E_i^l are fed into the score branch and the prediction branch, i.e., two MLP, to obtain the confidence scores and the residuals of vanishing points, where the subscript i take values between 1 and number of queries, and the superscript l means that the feature is output by the l-th decoder layer.

$$score_i^l = \mathrm{MLP}^{score}(E_i^l) \tag{5}$$

$$(\Delta x^l, \Delta y_i^l, \Delta z_i^l) = \mathrm{MLP}^{pred}(E_i^l) \tag{6}$$

$$(x_i^l, y_i^l, z_i^l) = (x_i^{l-1} + \Delta x^l, y_i^{l-1} + \Delta y_i^l, z_i^{l-1} + \Delta z_i^l) \tag{7}$$

where (x_i^l, y_i^l, z_i^l) is the i-th prediction of the l-th decoder layer, and for initialization, (x_0^l, y_0^l, z_0^l) are randomly sampled in $(-1, 1)$. $score_i^l$ represents the prediction quality, the smaller the angle between the prediction and the ground truth, the higher the score.

Bipartite Matching

Our VPDETR predicts a set of N vanishing points $\left\{\hat{\mathbf{VP}}_i^l = (x_i^l, y_i^l, z_i^l); i = 1, \ldots, N\right\}$ in one pass through the l-th decoder layer, with N set to 20 in this study. For simplicity, we omit the superscript l. We conduct a set-based bipartite matching between the predicted vanishing points and ground-truth targets to determine whether a prediction is associated with a real vanishing point and will be involved in the calculation of the loss function during the training stage.

Assume there are M target vanishing points $\{\mathbf{VP}_j; j = 1, \ldots, M\}$, we optimize a bipartite matching objective on a permutation function $\sigma(\cdot) : \mathbb{Z}_+ \to \mathbb{Z}_+$ which maps prediction indices $\{1, \ldots, N\}$ to potential target indices $\{1, \ldots, N\}$, including $\{1, \ldots, M\}$ for matched predictions, and $\{M+1, \ldots, N\}$ for unmatched predictions:

$$\begin{aligned}\mathcal{L}_{\text{match}} = \sum_{i=1}^{N} \mathbf{1}_{\{\sigma(i) \leq M\}} \Big[& \lambda_1 d\left(\hat{\mathbf{VP}}_{\sigma(i)}, \mathbf{VP}_i\right) \\ & + \lambda_2 a\left(\hat{\mathbf{VP}}_{\sigma(i)}, \mathbf{VP}_i\right) \Big]\end{aligned} \tag{8}$$

$$\sigma^* = \arg\min_{\sigma} \mathcal{L}_{\text{match}} \tag{9}$$

where $d(\cdot, \cdot)$ represents L1 distance between vectors, and $a(\cdot, \cdot)$ represents the angle between vectors. $\mathbf{1}_{\{\cdot\}}$ is an indicator function. The optimal permutation σ^* is computed using a Hungarian algorithm, mapping M positive prediction indices to ground-truth indices $\{1, \ldots, M\}$.

Losses

We compute losses based on the optimal permutation σ^* from the bipartite matching procedure introduced in Sec , in which $\{i; \sigma^*(i) \leq M\}$ represents positive predictions indices.

Regression Loss. We use a simple L_1 distance loss as a baseline of regression loss:

$$\mathcal{L}_{\text{reg}} = \sum_{i=1}^{M} d\left(\hat{\mathbf{VP}}_{\sigma^*(i)}, \mathbf{VP}_i\right) \tag{10}$$

where $d(\cdot, \cdot)$ represents L1 distance between vectors.

Cross-prediction loss. For datasets that fits the Manhattan world assumption, we design a novel cross-prediction loss. Denote the normalized predicted vanishing point $\hat{\mathbf{VP}}_{\sigma^*(i)}$ that matches the ground-truth $\mathbf{VP}_i$ as $\tilde{\mathbf{VP}}_i$:

$$\tilde{\mathbf{VP}}_i = \frac{\hat{\mathbf{VP}}_{\sigma^*(i)}}{\|\hat{\mathbf{VP}}_{\sigma^*(i)}\|} \tag{11}$$

then the cross-prediction loss is formulated as:

$$\begin{aligned}\mathcal{L}_{\text{cross}} = & \, dist\left(\mathbf{VP}_1, \tilde{\mathbf{VP}}_2 \times \tilde{\mathbf{VP}}_3\right) \\ & + dist\left(\mathbf{VP}_2, \tilde{\mathbf{VP}}_3 \times \tilde{\mathbf{VP}}_1\right) \\ & + dist\left(\mathbf{VP}_3, \tilde{\mathbf{VP}}_1 \times \tilde{\mathbf{VP}}_2\right)\end{aligned} \tag{12}$$

the intuition is that according to the orthogonality of the vanishing points in Manhattan world assumption, any two predicted vanishing points should be able to obtain another vanishing point by cross product. The order of the two elements participating in the cross product will cause the result to have different directions, therefore:

$$dist(\mathbf{a}, \mathbf{b}) = min(d(\mathbf{a}, \mathbf{b}), d(-\mathbf{a}, \mathbf{b})) \tag{13}$$

where $\mathbf{a}$ and $\mathbf{b}$ are two vectors. Eq. 13 solves the wrong direction problem caused by cross product.

Orthogonal Loss. For datasets in the Manhattan world, we additionally propose an orthogonal loss:

$$\mathcal{L}_{\text{orth}} = \left|\tilde{\mathbf{VP}}_1 \cdot \tilde{\mathbf{VP}}_2\right| + \left|\tilde{\mathbf{VP}}_2 \cdot \tilde{\mathbf{VP}}_3\right| + \left|\tilde{\mathbf{VP}}_3 \cdot \tilde{\mathbf{VP}}_1\right| \tag{14}$$

where $\cdot$ is inner-product. That is, the three predicted vanishing points should be orthogonal, the larger the inner product, the larger the loss value.

Classification Loss. The classification loss is based on binary cross-entropy loss (BCE), the input logits $(score_i, conf_i)$ are predicted in Eq. 5, and the target is determined by the angle between predicted vanishing points and the ground-truth. Denote the ground-truth value of the minimum angle with the predicted vanishing point $\hat{\mathbf{VP}}_i$ as $\mathbf{VP}_{\rho(i)}$. The corresponding targets are designed as:

$$score_i^t = \begin{cases} 0.01 * (degree_i - 10^\circ)^2 & \text{if } degree_i < 10^\circ \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where

$$degree_i = a\left(\mathbf{VP}_{\rho(i)}, \hat{\mathbf{VP}}_i\right) \tag{16}$$

The classification loss is formed as:

$$\mathcal{L}_{\text{cls}} = \sum_{i=1}^{N} \text{BCE}\left(score_i, score_i^t\right) \tag{17}$$

The final loss function is

$$\mathcal{L}_{\text{total}} = \lambda_{reg}\mathcal{L}_{\text{reg}} + \lambda_{cls}\mathcal{L}_{\text{cls}} \tag{18}$$

as a baseline for general datasets, and

$$\mathcal{L}_{\text{total}} = \lambda_{cross}\mathcal{L}_{\text{cross}} + \lambda_{orth}\mathcal{L}_{\text{orth}} + \lambda_{reg}\mathcal{L}_{\text{reg}} + \lambda_{cls}\mathcal{L}_{\text{cls}} \tag{19}$$

for Manhattan world datasets.

Experiments

Datasets. Since our method is based on set predictions, it can be applied both to datasets that following the Manhattan world assumption such as SU3 (Zhou et al. 2019b), ScanNet (Dai et al. 2017), YUD (Denis, Elder, and Estrada 2008), and SVVP (Tong et al. 2022), and to non-Manhattan world dataset such as NYU Depth (Silberman et al. 2012). The SU3 dataset comprises 23k synthetic outdoor images generated by a procedural photo-realistic building generator, with vanishing points computed directly from the CAD models of the generated buildings. The ScanNet dataset comprises 189,916 RGB-D images of real-world indoor scenes for training and 53,193 for validation. Vanishing points in ScanNet are estimated from surface normals following (Zhou et al. 2019a), which makes them less accurate than other datasets. YUD (Denis, Elder, and Estrada 2008) contains 102 images of indoor and outdoor scenes. SVVP (Tong et al. 2022) is a real-world street view dataset that contains 500 images. We also evaluate our model on the non-Manhattan

Method	SU3 (Zhou et al. 2019b)			ScanNet (Dai et al. 2017)				
	$AA@3°$	$AA@5°$	$AA@10°$	$AA@3°$	$AA@5°$	$AA@10°$	FPS	$End-to-End$
J-linkage (Tardif 2009)	69.2	77.0	84.4	27.8	41.7	57.7	1.2	×
Simon et al. (2018)	70.2	77.9	85.1	25.7	39.9	56.6	0.6	×
Wu et al. (2021)	74.8	79.5	83.9	22.9	36.8	54.0	23	×
Lu et al. (2017)	81.4	87.8	93.0	35.6	53.2	71.6	25	×
Li et al. (2019b)	59.1	66.9	74.6	35.0	50.2	66.9	25	×
CONSAC (Kluger et al. 2020)	77.9	85.2	91.0	31.1	46.1	62.4	2	×
TLC (Tong et al. 2022)	91.3	94.6	97.1	*36.2*	53.9	72.6	25	×
NeurVPS (Zhou et al. 2019a)	<u>94.4</u>	<u>96.5</u>	<u>98.2</u>	36.1	*54.3*	*74.9*	0.5	∗
Lin et al. (2022)	84.0	90.3	95.1	32.9	52.0	72.7	5.5	∗
$ours$	*92.8*	*95.7*	*97.8*	<u>36.8</u>	<u>55.4</u>	<u>75.6</u>	16	✓
$ours^*$	**96.4**	**97.8**	**98.9**	**41.7**	**60.0**	**78.7**	16	✓

Table 1: Comparison with SOTA on SU3 and ScanNet. Bold is highest, underlined is second highest, italics is third highest. ∗ in the column of "End-to-End" represents end-to-end trainable, but needs candidate points to be classified in the inference stage.

Method	SVVP (Tong et al. 2022)			YUD (Denis, Elder, and Estrada 2008)			
	$AA@3°$	$AA@5°$	$AA@10°$	$AA@3°$	$AA@5°$	$AA@10°$	FPS
J-linkage (Tardif 2009)	32.8	45.7	60.2	40.2	50.5	64.1	1.2
Simon et al. (2018)	45.4	59.6	73.2	40.1	58.2	77.5	0.6
Wu et al. (2021)	39.1	52.4	67.9	44.3	61.4	77.4	23
Lu et al. (2017)	*48.5*	*64.8*	*80.0*	58.0	73.2	86.2	25
Li et al. (2019b)	39.3	53.0	66.8	51.1	66.1	80.5	25
CONSAC (Kluger et al. 2020)	43.8	56.5	69.4	*62.1*	73.7	84.1	2
TLC (Tong et al. 2022)	<u>51.6</u>	<u>67.7</u>	<u>82.6</u>	<u>65.5</u>	<u>77.1</u>	<u>87.4</u>	25
NeurVPS (Zhou et al. 2019a)	27.1	40.4	55.3	39.9	50.3	65.0	0.5
Lin et al. (2022)	32.9	52.0	72.7	60.7	*74.3*	*86.3*	5.5
$ours$	41.6	60.3	78.9	42.3	61.4	80.3	16
$ours^*$	**66.9**	**83.3**	**91.6**	**69.1**	**81.3**	**90.7**	16

Table 2: Comparison with SOTA on YUD and SVVP. All learning-based models are pre-trained on SU3.

world dataset NYU Depth (Silberman et al. 2012), whose vanishing points varies from 1 to 8 in each image. And there are 1449 images in NYU Depth totally.

Evaluation. Following (Zhou et al. 2019a; Tong et al. 2022), we use 500 images for evaluation on SU3 and ScanNet. And on the SVVP and YUD datasets, since the number of images are too small, we use the model pre-trained on SU3. On these datasets, we calculate the percentage of predictions with an angle difference smaller than a certain threshold and compare the angle accuracy (AA) across different thresholds, as done in previous works (Zhou et al. 2019a; Liu, Zhou, and Zhao 2021; Tong et al. 2022; Lin et al. 2022). For the NYU Depth dataset, we follow (Kluger et al. 2020; Lin et al. 2022), which rank the predicted vanishing points by confidence and then use bipartite matching to compute angular errors for the top-k predictions, then we generate the recall curve and calculate the area under the curve (AUC) up to a specified threshold, such as 10 degrees.

In the inference stage, we first normalize the output value. For the general dataset, we select the prediction with confidence greater than a certain threshold as the output. For the Manhattan World dataset, we select the prediction with the highest score among all predictions as the first result. And as the score decreases, compare the predicted value with the existing results, and select the inner product smaller than a certain threshold such as 0.01 to add to the result until three vanishing points are selected as the final output. In addition, we provide $ours^*$ as the results of using 400 queries for prediction and bipartite graph matching for selection.

Implementation details. We implement our model on Nvidia RTX2080Ti for a fair comparison of inference speed. λ_{reg} in Eq. 18 and λ_{cross} in Eq. 19 is set to 5, and λ_{cls} and λ_{orth} is set to 1. ResNet-50 (He et al. 2016) pre-trained on ImageNet (Deng et al. 2009) is used as the backbone. Specifically, we use the C3 layer of the multi-scale feature map as the input of the Transformer, which will be discussed in ablation study. The number of queries is set as 20. By default, models are trained for 220 epochs and the learning rate is decayed at the 200-th epoch by a factor of 0.1. We trained our model using AdamW (Loshchilov and Hutter 2017) with base learning rate of 5×10^{-5}, and weight decay of 10^{-4}.

Comparison with State-of-the-Arts

We compare our model with J-Linkage (Tardif 2009), Simon et al. (Simon, Fond, and Berger 2018), Wu et al. (Wu et al. 2021), Lu et al. (Lu et al. 2017), Li et al. (Li et al.

Datasets	NYU Depth (Silberman et al. 2012)			
	top-k=#gt		top-k=#pred	
AUC	@5°	@10°	@5°	@10°
J-Linkage	49.30	61.28	*54.48*	68.34
T-Linkage	43.38	58.05	47.48	64.59
CONSAC	*49.46*	65.00	54.37	*69.89*
CONSAC+DLSD	46.78	61.06	49.94	65.96
VaPiD	–	*69.10*		–
Lin et al.	**55.92**	69.57	57.19	71.62
Ours	54.56	**72.32**	**65.52**	**80.81**

Table 3: Non-Manhattan scenario. Here "top-k=#gt" indicates the k most confident predictions are used for evaluation, where k in the number of ground-truth vanishing points. For "top-k=#pred" all predictions are used for evaluation.

2019b), CONSAC (Kluger et al. 2020), NeurVPS (Zhou et al. 2019a), TLC (Tong et al. 2022), and Lin et al. (Lin et al. 2022) on SU3, ScanNet, SVVP and YUD. And on the non-Manhattan dataset NYU Depth, we compare with J-Linkage (Tardif 2009), T-Linkage (Magri and Fusiello 2014), CONSAC (Kluger et al. 2020), VaPid (Liu, Zhou, and Zhao 2021) and Lin et al. (Lin et al. 2022). Notice among all these methods, J-Linkage, T-Linkage, Contrario-VP, and Quasi-VP are not learning-based methods, and TLC is learning-based method, these method relays on line segment detection (Von Gioi et al. 2008). CONSAC needs line segments as inputs. NeurVPS and Lin et al. are end-to-end trainable, but they rely on sampling candidate points hierarchically on the Gaussian sphere to obtain candidate points, and then classify whether these candidate points are vanishing points. Only our method is truly end-to-end, that is, after entering the image, it can directly return its vanishing points. The comparison results are listed in Table 1, Table 2, and Table 3.

Comparison results show that our methods achieves comparable or better performance than SOTA methods on SU3 (Zhou et al. 2019b), ScanNet (Dai et al. 2017), SVVP (Tong et al. 2022), and NYU Depth (Silberman et al. 2012) benchmarks, and the inference speed is 16 FPS, achieving a better balance of accuracy and speed. Furthermore, We also show the angle accuracy curves on SU3 and ScanNet for detail comparison in Figure 3 and Figure 4.

architecture	L_{orth}	L_{cross}	$AA@3°$	$AA@5°$
vanilla DETR			85.2	90.9
vanilla DETR	✓		86.1	91.5
vanilla DETR	✓	✓	86.5	91.8
VPDETR			86.0	91.5
VPDETR	✓		86.5	91.9
VPDETR	✓	✓	**88.1**	**92.8**

Table 4: Network structure and loss function ablation study, using the last layer feature map of ResNet-50.

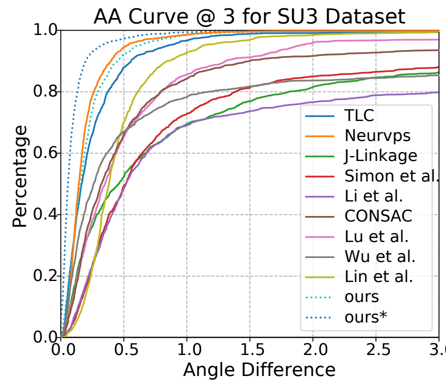

Figure 3: Angle accuracy curves for different methods on SU3.

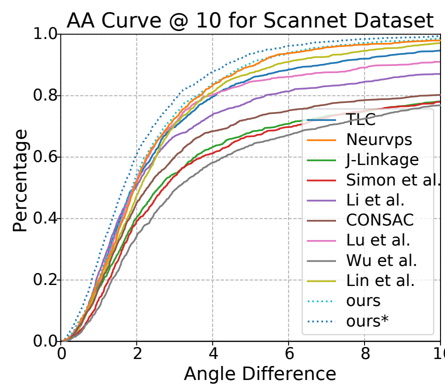

Figure 4: Angle accuracy curves for different methods on ScanNet.

Ablation Study

To verify the effectiveness of our design, we conduct extensive ablation experiments on the SU3 dataset. As shown in Table 4, our method outperforms vanilla DETR under the same setting. Adding the orthogonal loss and the cross-prediction loss can effectively improve the performance of the model, which is effective for both vanilla DETR and our VPDETR. And in Table 5, we tested the effect of using different levels of features on the model, and finally chose the C_3 level of features. As discussed by TLC (Tong et al. 2022), predicting vanishing points requires both semantic and geometric information, the C_3 level achieves a better balance. The resolution of C_5 and C_4 level is too low to lose too much geometric information. And experiments prove that using multi-scale features does not improve the accuracy.

Visualization

In order to study which regions of the input image our method pays more attention to for giving the final prediction, we draw the gradient norm of the final prediction with respect to each pixel in the image. The gradient norm reflects how much the result will change when a pixel is perturbed. So it can show us which pixels the model relies more on to

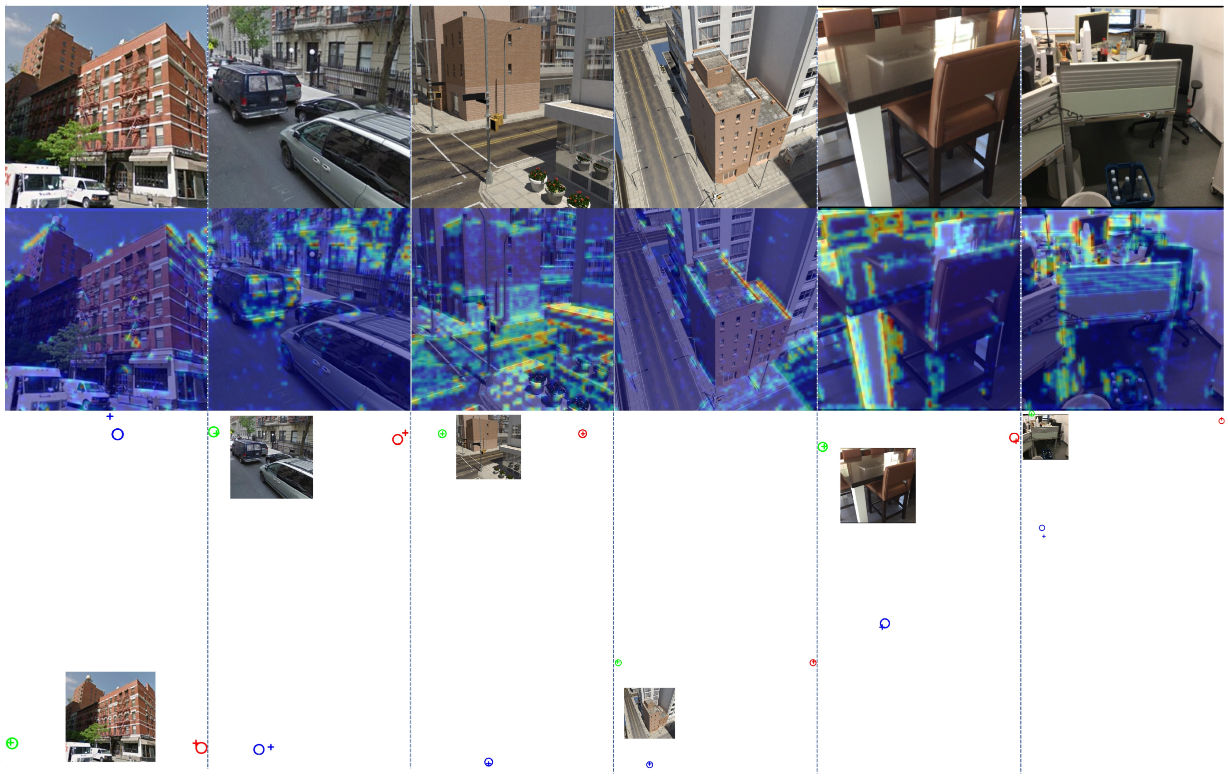

Figure 5: From top to bottom: (a) Original images, (b) The gradient norm of the final predictions with respect to each pixel in input images (c) Visual examples of vanishing point detection results of our model, the ground truth vanishing points are marked with '○', and the predicted ones are marked with '+'. The first two columns are from SVVP, the second two columns are from SU3, and the third two columns are from ScanNet.

Feature level					
C_5	C_4	C_3	$AA@3°$	$AA@5°$	$AA@10°$
✓			88.1	92.8	96.4
	✓		92.2	95.3	97.6
		✓	**92.8**	**95.7**	**97.8**
✓	✓		89.7	93.8	96.9
✓	✓	✓	92.5	95.5	97.7

Table 5: Feature level ablation study, C_5 is the last layer feature map of ResNet-50, C_4 is the penultimate layer, C_3 is the third to last layer. If more than one is checked, it means using multi-scale features.

make the final prediction.

As shown in Figure 5, in order to predict vanishing points, our model will pay more attention to places rich in straight line information, such as road markings, edges of road tiles, outlines of buildings, edges of furniture, corners, etc. This is consistent with the focus of traditional methods, which is to use the clues of the straight line to infer the vanishing point, but our method does not need to explicitly extract the straight line, but through training the model to automatically focus on the area with rich straight line information. Through observation, we found that our model may pay attention to some information that we usually don't notice, such as the texture of the road surface and the table.

In addition, we also show the position relationship between the predicted vanishing point and the ground truth in the Figure 5.

Conclusion

This paper proposes VPDETR, a novel end-to-end framework for vanishing point detection that is capable of handling both Manhattan and non-Manhattan world datasets. To improve model accuracy for Manhattan world datasets, we introduce an orthogonal loss and a cross-prediction loss. Our method achieves a good balance of accuracy and inference speed. Our method has limitations, such as the general performance on the YUD dataset due to the domain gap problem. We hope that our method can serve as a new paradigm of vanishing point detection, and inspire thinking about how to infer vanishing points of an image end-to-end, accurately, and quickly.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) (No. 62371009 and No. 61971008).

References

Antone, M. E.; and Teller, S. 2000. Automatic recovery of relative camera rotations for urban scenes. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition.*, volume 2, 282–289. IEEE.

Arbelaez, P.; Maire, M.; Fowlkes, C.; and Malik, J. 2010. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5): 898–916.

Barnard, S. T. 1983. Interpreting perspective images. *Artificial intelligence*, 21(4): 435–462.

Bolles, R. C.; and Fischler, M. A. 1981. A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *IJCAI*, volume 1981, 637–643.

Borji, A. 2016. Vanishing point detection with convolutional neural networks. *arXiv preprint arXiv:1609.00967*.

Canny, J. 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6): 679–698.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.

Chang, C.-K.; Zhao, J.; and Itti, L. 2018. Deepvp: Deep learning for vanishing point detection on 1 million street view images. In *2018 IEEE International Conference on Robotics and Automation*, 4496–4503. IEEE.

Cipolla, R.; Drummond, T.; and Robertson, D. P. 1999. Camera Calibration from Vanishing Points in Image of Architectural Scenes. In *BMVC*, volume 99, 382–391. Citeseer.

Dai, A.; Chang, A. X.; Savva, M.; Halber, M.; Funkhouser, T.; and Nießner, M. 2017. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5828–5839.

Davison, A. J.; Reid, I. D.; Molton, N. D.; and Stasse, O. 2007. MonoSLAM: Real-time single camera SLAM. *IEEE transactions on pattern analysis and machine intelligence*, 29(6): 1052–1067.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Denis, P.; Elder, J. H.; and Estrada, F. J. 2008. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *Proceedings of the European Conference on Computer Vision*, 197–210. Springer.

Grammatikopoulos, L.; Karras, G.; and Petsa, E. 2007. An automatic approach for camera calibration from vanishing points. *ISPRS journal of photogrammetry and remote sensing*, 62(1): 64–76.

Guillou, E.; Meneveaux, D.; Maisel, E.; and Bouatouch, K. 2000. Using vanishing points for camera calibration and coarse 3D reconstruction from a single image. *The Visual Computer*, 16(7): 396–410.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hoiem, D.; Efros, A. A.; and Hebert, M. 2008. Putting objects in perspective. *International Journal of Computer Vision*, 80(1): 3–15.

Kluger, F.; Ackermann, H.; Yang, M. Y.; and Rosenhahn, B. 2017. Deep learning for vanishing point detection using an inverse gnomonic projection. In *German Conference on Pattern Recognition*, 17–28. Springer.

Kluger, F.; Brachmann, E.; Ackermann, H.; Rother, C.; Yang, M. Y.; and Rosenhahn, B. 2020. Consac: Robust multi-model fitting by conditional sample consensus. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4634–4643.

Kogecka, J.; and Zhang, W. 2002. Efficient computation of vanishing points. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 1, 223–228. IEEE.

Košecká, J.; and Zhang, W. 2002. Video compass. In *Proceedings of the European Conference on Computer Vision*, 476–490. Springer.

Lee, S.; Kim, J.; Shin Yoon, J.; Shin, S.; Bailo, O.; Kim, N.; Lee, T.-H.; Seok Hong, H.; Han, S.-H.; and So Kweon, I. 2017. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *Proceedings of the IEEE international conference on computer vision*, 1947–1955.

Lezama, J.; Grompone von Gioi, R.; Randall, G.; and Morel, J.-M. 2014. Finding vanishing points via point alignments in image primal and dual domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 509–515.

Li, F.; Zhang, H.; Liu, S.; Guo, J.; Ni, L. M.; and Zhang, L. 2022. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13619–13627.

Li, H.; Chen, K.; Kim, P.; Yoon, K.-J.; Liu, Z.; Joo, K.; and Liu, Y.-H. 2021. Learning icosahedral spherical probability map based on bingham mixture model for vanishing point estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5661–5670.

Li, H.; Xing, Y.; Zhao, J.; Bazin, J.-C.; Liu, Z.; and Liu, Y.-H. 2019a. Leveraging structural regularity of Atlanta world for monocular SLAM. In *2019 International Conference on Robotics and Automation*, 2412–2418. IEEE.

Li, H.; Zhao, J.; Bazin, J.-C.; Chen, W.; Liu, Z.; and Liu, Y.-H. 2019b. Quasi-globally optimal and efficient vanishing point estimation in Manhattan world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1646–1654.

Lin, Y.; Wiersma, R.; Pintea, S. L.; Hildebrandt, K.; Eisemann, E.; and van Gemert, J. C. 2022. Deep vanishing point detection: Geometric priors make dataset variations vanish. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6103–6113.

Liu, S.; Li, F.; Zhang, H.; Yang, X.; Qi, X.; Su, H.; Zhu, J.; and Zhang, L. 2022. DAB-DETR: Dynamic Anchor Boxes are Better Queries for DETR. In *International Conference on Learning Representations*.

Liu, S.; Zhou, Y.; and Zhao, Y. 2021. Vapid: A rapid vanishing point detector via learned optimizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 12859–12868.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lu, X.; Yaoy, J.; Li, H.; Liu, Y.; and Zhang, X. 2017. 2-line exhaustive searching for real-time vanishing point estimation in manhattan world. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 345–353. IEEE.

Lutton, E.; Maitre, H.; and Lopez-Krahe, J. 1994. Contribution to the determination of vanishing points using Hough transform. *IEEE transactions on pattern analysis and machine intelligence*, 16(4): 430–438.

Magri, L.; and Fusiello, A. 2014. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3954–3961.

Meng, D.; Chen, X.; Fan, Z.; Zeng, G.; Li, H.; Yuan, Y.; Sun, L.; and Wang, J. 2021. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3651–3660.

Misra, I.; Girdhar, R.; and Joulin, A. 2021. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2906–2917.

O'brien, J. F.; and Farid, H. 2012. Exposing photo manipulation with inconsistent reflections. *ACM Trans. Graph.*, 31(1): 4–1.

Prangemeier, T.; Reich, C.; and Koeppl, H. 2020. Attention-based transformers for instance segmentation of cells in microstructures. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 700–707. IEEE.

Shi, Y.; Zhang, D.; Wen, J.; Tong, X.; Zhao, H.; Ying, X.; and Zha, H. 2019. Three orthogonal vanishing points estimation in structured scenes using convolutional neural networks. In *2019 IEEE International Conference on Image Processing (ICIP)*, 3537–3541. IEEE.

Silberman, N.; Hoiem, D.; Kohli, P.; and Fergus, R. 2012. Indoor segmentation and support inference from rgbd images. In *Proceedings of the European Conference on Computer Vision*, volume 7576, 746–760.

Simon, G.; Fond, A.; and Berger, M.-O. 2018. A-contrario horizon-first vanishing point detection using second-order grouping laws. In *Proceedings of the European Conference on Computer Vision*, 318–333.

Tardif, J.-P. 2009. Non-iterative approach for fast and accurate vanishing point detection. In *2009 IEEE 12th International Conference on Computer Vision*, 1250–1257. IEEE.

Tong, X.; Ying, X.; Shi, Y.; Wang, R.; and Yang, J. 2022. Transformer Based Line Segment Classifier With Image Context for Real-Time Vanishing Point Detection in Manhattan World. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6093–6102.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Von Gioi, R. G.; Jakubowicz, J.; Morel, J.-M.; and Randall, G. 2008. LSD: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4): 722–732.

Wu, J.; Zhang, L.; Liu, Y.; and Chen, K. 2021. Real-time vanishing point detector integrating under-parameterized ransac and hough transform. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3732–3741.

Xu, Y.; Xu, W.; Cheung, D.; and Tu, Z. 2021. Line segment detection using transformers without edges. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4257–4266.

Yao, Z.; Ai, J.; Li, B.; and Zhang, C. 2021. Efficient detr: improving end-to-end object detector with dense prior. *arXiv preprint arXiv:2104.01318*.

Zhai, M.; Workman, S.; and Jacobs, N. 2016. Detecting vanishing points using global image context in a non-manhattan world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5657–5665.

Zhang, H.; Li, F.; Liu, S.; Zhang, L.; Su, H.; Zhu, J.; Ni, L. M.; and Shum, H.-Y. 2022. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*.

Zhang, X.; Gao, X.; Lu, W.; He, L.; and Liu, Q. 2018. Dominant vanishing point detection in the wild with application in composition analysis. *Neurocomputing*, 311: 260–269.

Zhou, Y.; Qi, H.; Huang, J.; and Ma, Y. 2019a. Neurvps: Neural vanishing point scanning via conic convolution. *Advances in Neural Information Processing Systems*, 32.

Zhou, Y.; Qi, H.; and Ma, Y. 2019. End-to-end wireframe parsing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 962–971.

Zhou, Y.; Qi, H.; Zhai, Y.; Sun, Q.; Chen, Z.; Wei, L.-Y.; and Ma, Y. 2019b. Learning to reconstruct 3d manhattan wireframes from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7698–7707.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.