End-to-End Real-Time Vanishing Point Detection with Transformer

Xin Tong*, **Shi Peng, Yufei Guo, Xuhui Huang**

Intelligent Science & Technology Academy of CASIC
xin_tong@pku.edu.cn, pengshi1828@163.com, yfguo@pku.edu.cn, starhxh@126.com

Abstract

In this paper, we propose a novel transformer-based end-to-end real-time vanishing point detection method, which is named Vanishing Point TRansformer (VPTR). The proposed method can directly regress the locations of vanishing points from given images. To achieve this goal, we pose vanishing point detection as a point object detection task on the Gaussian hemisphere with region division. Considering low-level features always provide more geometric information which can contribute to accurate vanishing point prediction, we propose a clear architecture where vanishing point queries in the decoder can directly gather multi-level features from CNN backbone with deformable attention in VPTR. Our method does not rely on line detection or Manhattan world assumption, which makes it more flexible to use. VPTR runs at an inferring speed of 140 FPS on one NVIDIA 3090 card. Experimental results on synthetic and real-world datasets demonstrate that our method can be used in both natural and structural scenes, and is superior to other state-of-the-art methods on the balance of accuracy and efficiency.

Introduction

A vanishing point (VP) is the intersection of projections of a set of parallel lines in the world. The coordinates of vanishing points determine the direction of 3D lines in the world and bridge the information between the 2D image and 3D space. Vanishing point detection is an important and classical problem in computer vision. It has been widely applied in camera calibration, 3D reconstruction, SLAM, and autonomous driving.

Traditional vanishing point detection methods usually detect straight lines first, then cluster them into several groups and locate the vanishing points based on the geometric knowledge. Most previous learning-based methods learn the geometric knowledge from annotated images. By explicitly importing the geometric priors into the Neural Network (NN) models, recent learning-based methods achieve impressive success in improving predicting accuracy and data efficiency. For instance, Zhou *et al.* (Zhou et al. 2019a) scan the vanishing points on the Gaussian hemisphere with conic convolution. Lin *et al.* (Lin et al. 2022) first map the features to Hough histogram and then to the Gaussian hemisphere for interpretable prediction. These methods often rely on selecting vanishing point proposals, which means that a predicted vanishing point should be one of these candidates. Therefore, the trade-off of prediction accuracy and inferring speed should be carefully considered. A dense point sampling on the Gaussian sphere will always bring a high detection accuracy, but lead to a low inferring speed on the contrary. Kluger *et al.* (Kluger et al. 2020) present to learning the sampling probabilities of RANSAC for grouping lines corresponding to the same vanishing point. Tong *et al.* (Tong et al. 2022) propose to use a Transformer-based approach to group line segments in the images with image context for vanishing point detection. These methods provide convincing performance when lines can be detected in the images. However, they often fail to detect vanishing points in scenes with insufficient lines.

We find that it is still hard to use a unified model to efficiently detect vanishing points in complex cases, e.g., when lines are difficult to detect or the number of vanishing points is unknown. Inspired by the success of end-to-end object detection approaches (Redmon et al. 2016; Carion et al. 2020), we tackle the challenge by modeling vanishing point detec-

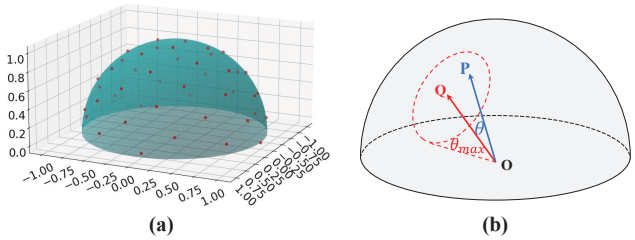

Figure 1: We pose vanishing point detection as an end-to-end point object detection task on the Gaussian hemisphere. (a) The hemisphere is divided into several overlapped sphere caps which are centered by anchors sampled using Fibonacci lattice (lvaro González 2010). In our model, each anchor will finally predict a confidence score for containing a vanishing point and a position of the possible vanishing point. (b) The anchor Q with a small angle distance from a vanishing point P should be responsible for detecting that vanishing point, e.g., $\theta < \theta_{max}$.

*Corresponding author.

tion as an end-to-end point object detection problem and using a Transformer-based model to directly regress the locations, as shown in Fig. 1. Meanwhile, we consider two improvements for vanishing point detection upon the modeling power of general-purpose Transformer architectures including defining the supervision manner and making use of low-level features.

In this paper, we propose a novel vanishing point detection method that can deal with both structural and natural scenes in real-time. To obtain both semantic features and geometric information for accurate detection, we extract multi-scale and multi-level image features from different layers of the backbone. We bridge the Transformer decoder and the backbone together without Transformer encoder. Thus, vanishing point queries of the decoder can be directly applied to the multi-level image feature maps to efficiently gather the features for detecting vanishing points. We model vanishing point detection as a point detection task on the Gaussian hemisphere. The hemisphere is divided into small sphere caps with predefined anchors and each vanishing point query is responsible for detecting the vanishing point located in a certain sphere cap. We are then able to apply confidence loss and position loss to supervise the model prediction.

Our main contributions can be summarized as follows: (1) We pose vanishing point detection as an end-to-end object detection problem by modeling the task as point object detection on the Gaussian hemisphere with spherical region division. (2) We propose a novel network architecture named Vanishing Point TRansformer (VPTR), which can directly predict the locations of vanishing points from given images. In VPTR, decoder bridge is used to enable the queries to directly gather multi-level image features from the backbone. We also design losses to train our model in both Manhattan and non-Manhattan scenes. (3) The proposed method runs at 140 FPS on an NVIDIA 3090 GPU for vanishing point detection. Extensive experiments show that our method can get better performance compared to other state-of-the-art methods on the balance of accuracy and efficiency. (4) Our method does not rely on line detection or Manhattan world assumption. Thus it can be used in both structural and natural scenes, and detect a varying number of vanishing points without architecture modification.

Related Works

Vanishing Point Detection. Traditional methods of vanishing point detection often aim to group the line segments and estimate the intersection of the lines. The seminal work is introduced in (Barnard 1983). Then many works tackle the problem by using Gaussian sphere (Collins and Weiss 1990; Straforini, Coelho, and Campani 1993), Manhattan world assumption (Bazin et al. 2012; Mirzaei and Roumeliotis 2011; Lu et al. 2017), Hough transformation (Almansa, Desolneux, and Vamech 2003), Branch-and-Bound (Bazin et al. 2012; Li et al. 2019; Ge et al. 2021), etc. Line-based methods usually start with line detection (Canny 1986; Von Gioi et al. 2008). Then the parametric lines are clustered using Hough transformation (Lutton, Maitre, and Lopez-Krahe 1994), RANSAC (Bolles and Fischler 1981; Wu et al. 2021),

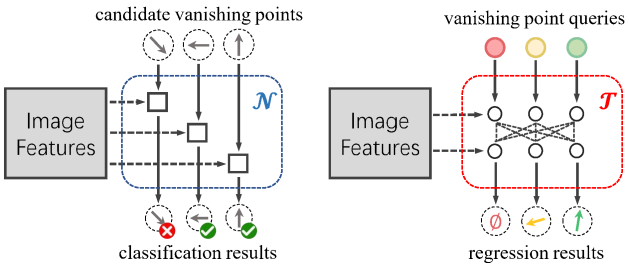

Figure 2: Difference between the pipelines of recent learning-based vanishing point detection methods (left) and our proposed method (right). Recent methods often classify the predefined candidate vanishing points independently. Thus, an accurate detection needs a large number of candidates, which will cause a heavy calculation burden. Our method models the task as a regression problem and provides a new way to predict the vanishing point at any location.

J-Linkage (Tardif 2009), EM algorithm (Denis, Elder, and Estrada 2008), dual space (Lezama et al. 2014).

Learning-based methods try to learn vanishing point detection from annotated datasets. In (Chang, Zhao, and Itti 2018; Zhang et al. 2018; Shi et al. 2019), CNN models are used to classify or regress the vanishing points. Global image context is extracted to generate horizon line candidates in (Zhai, Workman, and Jacobs 2016). Kluger *et al.* (Kluger et al. 2020) use neural networks to update the conditional sampling probabilities for line clustering. Bingham mixture model is applied in estimating vanishing points in (Li et al. 2021). Tong *et al.* (Tong et al. 2022) apply Transformer to classify the line segments in a non-iterative way for vanishing point detection. Some recent methods design neural networks with geometric priors to score the candidate vanishing points. Zhou *et al.* (Zhou et al. 2019a) present conic convolution to extract and gather features along the structural lines. Lin *et al.* (Lin et al. 2022) incorporate Hough Transform and Gaussian sphere in learning-based vanishing point detection. The difference between these methods and our method is illustrated in Fig. 2. Our method regresses the vanishing points from images directly with Transformer.

Object Detection. General Object Detection aims to identify and localize the objects appearing in the images. Methods using separate modules to generate region proposals are termed as two-stage detectors. Some classical models find object proposals in images during the first stage with Selective Search (Girshick et al. 2014; Girshick 2015), region proposal network (RPN) (Ren et al. 2015; He et al. 2017; Lin et al. 2017) and then classify and further localize them in the second. One-stage detection methods provide efficient ways that classify and localize objects in a single shot using dense sampling. In YOLO (Redmon et al. 2016) and its subsequent versions, the input image is divided into grids and the cell that the object's center locates in is responsible for detecting it. Some methods propose to represent the objects using their center points with sizes (Zhou, Wang, and Krähenbühl 2019), corner points (Law and Deng 2018) or extreme points

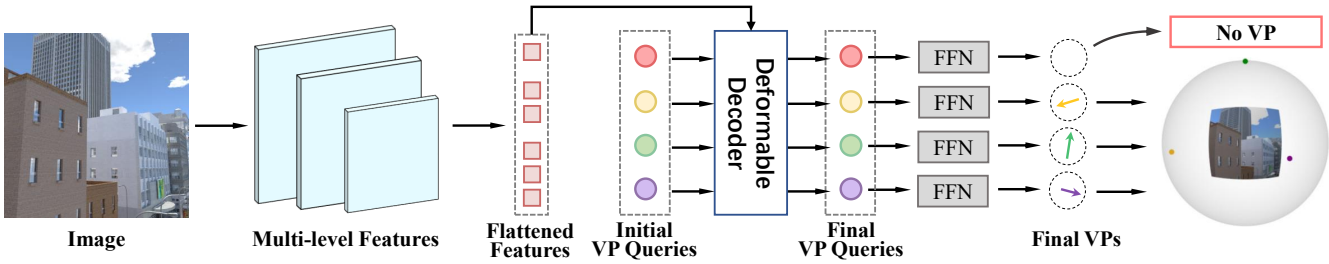

Figure 3: Overview of the proposed Vanishing Point TRansformer (VPTR). An image is first fed into a CNN backbone and multi-level feature maps are extracted from different layers. Then, initial vanishing point queries gather multi-level features across the feature maps with deformable decoder. Each query is only responsible for detecting the vanishing point within a small sphere cap. Finally, the vanishing points are detected from the final vanishing point queries by feed-forward networks (FFNs). VPTR directly regresses the locations of vanishing points from given images in an end-to-end manner.

(Zhou, Zhuo, and Krahenbuhl 2019). We model vanishing point detection as a one-stage detection problem, which can be seen as finding vanishing points on the Gaussian hemisphere.

Visual Transformers. Transformer-based models are widely used in computer vision tasks recently because of their strong representation ability. Dosovitskiy *et al.* (Dosovitskiy et al. 2020) use transformer (ViT) in classification by dividing an image into patches and Strudel *et al.*(Strudel et al. 2021) extend it to semantic segmentation. Carion *et al.* (Carion et al. 2020) apply transformer (DETR) in end-to-end object detection. Zhu *et al.*(Zhu et al. 2020) further propose deformable DETR whose attention modules only attend to a small set of keys. Swin Transformer (Liu et al. 2021) builds hierarchical feature maps by merging image patches and computes the representation with shifted windows. Transformers are also used in image super-resolution (Yang et al. 2020), pose estimation (Huang et al. 2020), tracking (Chen et al. 2021), etc. Transformer is first used to cluster the line segments corresponding with the same vanishing points in (Tong et al. 2022). Our model is designed based on general-proposed Transformer detector while having its own enhanced property for vanishing point detection.

Algorithm

Overview

The overview of our algorithm is depicted in Fig. 3. We introduce our method from three aspects including problem modeling, network architecture and training supervision. Our method can directly regress the locations of vanishing points from a given image.

End-to-end Vanishing Point Detection

In this section, we introduce our modeling of end-to-end vanishing point detection. The key insights are region division and query-based vanishing point prediction, which are introduced below.

Region Division. Inspired by the efficient detectors (Redmon et al. 2016) which divide the image into grids, we attempt to employ region division in vanishing point detection. However, different from classical object detection tasks where targets are always located in the image range, vanishing points may appear anywhere on the image plane, even at infinity. To overcome this problem, we detect vanishing points on the Gaussian hemisphere similar to (Zhou et al. 2019a; Lin et al. 2022), and present an anchor-based region division method on the hemisphere. Specifically, M points are sampled on the Gaussian hemisphere as anchors using Fibonacci lattice (lvaro González 2010). The m-th anchor (x_m, y_m, z_m) can be represented as

$$\begin{cases} y_m = m/M \\ z_m = \sqrt{1-y_m^2}\sin\left((\sqrt{5}-1)\pi m\right) \\ x_m = \sqrt{1-y_m^2}\cos\left((\sqrt{5}-1)\pi m\right) \end{cases} . \quad (1)$$

By setting a proper angle distance θ_{max} to the anchors, the hemisphere can be divided into M spherical caps with small overlaps. Each region is centered by the corresponding anchor and within a max angle distance θ_{max} from the anchor. In our modeling, each region is responsible for detecting the vanishing point falling into the region. Each vanishing point should be detected in at least one region.

Query-based Vanishing Point Prediction. We propose to directly regress the positions of vanishing points from images using Transformer. All the final vanishing points are produced from learnable vanishing point queries in Transformer framework. As we have defined M anchors on the hemisphere, we should finally predict M outputs from M vanishing point queries. The outputs and the anchors are in one-to-one correspondence. Each output contains a confidence score for containing a vanishing point and a regressed vanishing point with Gaussian sphere representation (Zhou et al. 2019a). Unlike classical object detection, the outputs do not need to contain category prediction. We also observe that the occlusion problem will not appear in vanishing point detection and two vanishing points will not be very close. Thus, with a proper angle distance θ_{max}, we assume that each anchor is responsible for detecting at most one vanishing point in our method.

VPTR Architecture

Our VPTR architecture is composed of a CNN backbone and a deformable Transformer decoder. We find that the low-

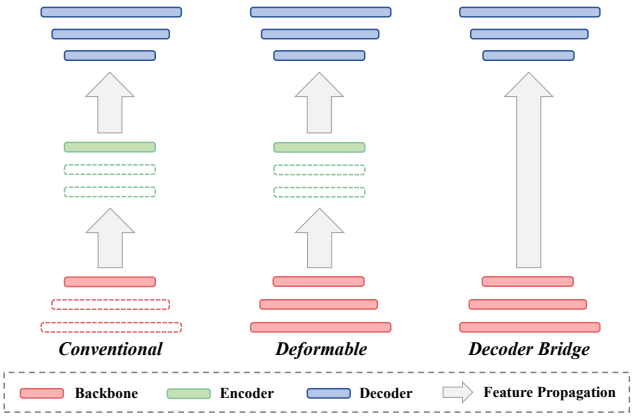

Figure 4: Illustration of feature propagation in different Transformer-based detectors. Only solid blocks can be transferred to the next parts. We bridge the decoder and the backbone directly without an encoder so that vanishing point queries can be applied directly to the multi-level image features with high resolution.

level image features from the backbone are hard to transfer to the decoder. As shown in Fig. 4, only features from the highest layer of backbone and encoders can be transferred to the next parts in conventional Transformer-based detectors. The feature maps from early layers cannot be used by the decoder since it is blocked by the encoder. These designs bring two major problems for vanishing point detection. One is that the decoder cannot obtain enough geometric features for accurate detection since most geometric information is in low-level features. The other is that the high computational cost of the encoder limits the size of the input feature maps.

We directly bridge the deformable decoder and the backbone to efficiently transfer multi-level image features for vanishing point detection. Vanishing point queries can gather global information from image features with high resolution. The deformable Transformer decoder we used consists of 6 deformable decoder layers. We do not use Transformer encoder layers in VPTR, since we find it does not bring accuracy improvement in our experiments. The deformable attention module in the decoder only attends to a small set of key sampling points on the image feature maps. We use learnable attention position in the module to adaptively search the global information related to vanishing points. The module can be represented as

$$\mathbf{y}_{hlqk} = \sum_{h=1}^{H} \boldsymbol{W}_h \Big(\sum_{l=1}^{L}\sum_{k=1}^{K} A_{hlqk} \cdot \boldsymbol{W}'_h \mathbf{F}_l(\boldsymbol{p}_{hlqk})\Big), \quad (2)$$

where q, h, l, k index the query, attention head, image feature level and sampling points respectively. $\boldsymbol{p}_{hlqk}$ is the learnable position and A_{hlqk} is the corresponding attention weight. We refer the readers to Zhu *et al.*(Zhu et al. 2020) for more details about deformable Transformer decoder.

Training Supervision

During training, we use a confidence loss and a position loss to supervise our vanishing point detection model. Specifically, in the confidence loss, we use binary cross entropy loss to supervise whether an anchor should predict a vanishing point according to the training image, e.g., a ground truth vanishing point is close enough to the anchor. If an anchor is responsible for detecting a vanishing point during detection, we call it a positive anchor. Otherwise, we call it a negative anchor. As the number of negative anchors is considerably larger than positive anchors, we randomly select some of the former for training, e.g., keeping the number of positive and negative anchors the same. The confidence loss $\mathcal{L}_{conf}$ can be represented as

$$\mathcal{L}_{conf} = -\frac{1}{M_{\pm}} \sum_{i=1}^{M_{\pm}} (\hat{c}_i \log c_i + (1-\hat{c}_i)\log(1-c_i)), \quad (3)$$

where $M_{\pm}$ is the total number of the positive and selected negative anchors. $\hat{c}_i$ is the ground truth indicates that whether the i-th anchor should respond to detect a vanishing point and is annotated as $+1$ or 0. c_i is the predicted probability of the i-th anchor.

In the position loss, we minimize the Euclidean distance between the predicted vanishing points and ground truth ones on the Gaussian hemisphere. This loss is only applied to the positive anchors since the prediction of negative anchors should be meaningless. The presented position loss $\mathcal{L}_{pos}$ can be represented as

$$\mathcal{L}_{pos} = \sum_{i=1}^{M_{+}} \|\boldsymbol{v_i} - \hat{\boldsymbol{v}}_{\boldsymbol{i}}\|_2, \quad (4)$$

where M_{+} is the number of positive anchors, $\boldsymbol{v_i}$ represents a predicted vanishing point from the positive anchor and $\hat{\boldsymbol{v}}_{\boldsymbol{i}}$ is the corresponding ground-truth.

For the cases following Manhattan world assumption, we also present a Manhattan loss to make the predicted vanishing points as orthogonal as possible. The Manhattan loss can be represented as

$$\mathcal{L}_{Man} = \|\boldsymbol{R}_{vp}^T \boldsymbol{R}_{vp} - \boldsymbol{I}\|_F, \quad (5)$$

where $\boldsymbol{R}_{vp}$ is the matrix consisting of three vanishing points represented on the Gaussian hemisphere. $\boldsymbol{I}$ is the identity matrix with the same size of $\boldsymbol{R}_{vp}$.

The total loss can be defined as the sum of the above loss terms, which can be written as

$$\mathcal{L}_{total} = \lambda_c \mathcal{L}_{conf} + \lambda_p \mathcal{L}_{pos} + \lambda_M \mathcal{L}_{Man}, \quad (6)$$

where $\lambda_c, \lambda_p, \lambda_M$ are factors for adjusting the loss terms. Specially, λ_M is set to 0 in non-Manhattan scenes. Notice that each query is responsible for detecting the vanishing point located in a certain pre-defined region. The correspondence between the predictions and the ground truths is determined and known. Thus, we do not use bipartite matching in calculating the losses.

Experimental Results

In this section, we first describe the implementation details of the VPTR. Then we compare it with the state-of-the-art vanishing point detection approaches on both synthetic and

Method	Applicability			SU3			ScanNet			FPS
	Non-Man	Real-time	Line-free	AA@3°	AA@5°	AA@10°	AA@3°	AA@5°	AA@10°	
J-linkage	✓			81.6	86.9	91.6	13.9	23.8	36.9	2.5
Simon *et al.*	✓			69.0	76.9	84.3	11.8	21.1	35.1	2.0
Li *et al.*		✓		73.4	78.4	82.9	13.6	23.6	37.0	25
CONSAC	✓			76.3	81.3	85.4	13.3	23.2	36.3	4
NeurVPS	✓		✓	**94.4**	**96.5**	**98.2**	23.6	41.3	64.0	1.1
TLC		✓		91.0	94.4	97.1	17.8	31.3	48.4	25
Lin *et al.*	✓	✓	✓	84.0	90.2	95.0	**24.7**	42.0	63.7	9.2
Ours	✓	✓	✓	88.5	92.9	96.3	24.2	**42.2**	**64.5**	**140**

Table 1: Comparison results on SU3 (Zhou et al. 2019b) and ScanNet (Dai et al. 2017) datasets. We compare our method with J-linkage (Tardif 2009), Simon *et al.* (Simon, Fond, and Berger 2018), Li *et al.* (Li et al. 2019), CONSAC (Kluger et al. 2020), NeurVPS (Zhou et al. 2019a), TLC (Tong et al. 2022) and Lin *et al.* (Lin et al. 2022). Our method gets better performance on the balance between accuracy and efficiency.

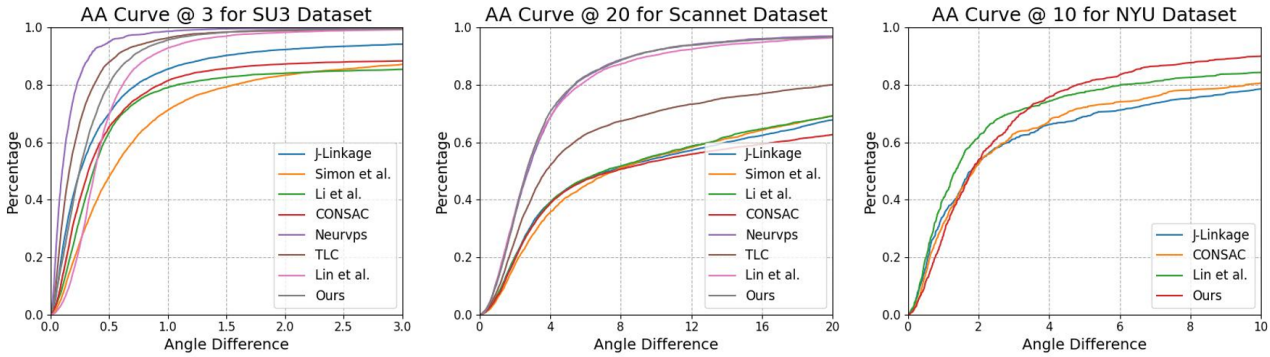

Figure 5: Angle accuracy curves for different methods on SU3 (Zhou et al. 2019b), ScanNet (Dai et al. 2017) and NYU (Silberman et al. 2012) datasets.

real-world datasets with quantitative experiments. In the ablation study, we demonstrate the effectiveness of the components in our method. We also conduct parameter study to select better hyperparameters or show the robustness.

Experimental Setup

Datasets We conduct our experiments in four publicly available datasets including SU3 dataset (Zhou et al. 2019b), ScanNet dataset (Dai et al. 2017), Natural Scene dataset (Zhou, Farhat, and Wang 2017) and NYU dataset (Silberman et al. 2012). SU3 dataset and ScanNet datasets follow the Manhattan world assumption, where there should be three orthogonal vanishing points in each image. SU3 is a photo-realistic dataset that contains 23000 synthetic outdoor images. ScanNet dataset is a real-world dataset and captures indoor scenes. It provides 189916 training images, 53193 validation images and 20942 test images. The Natural Scene dataset contains 2,275 images of real-world natural scenes. We divide them into 2,000 training images and 275 test images following (Zhou et al. 2019a). There are 1449 images in NYU dataset. Straight lines in Natural Scene dataset are hard to detect at most time and the number of vanishing points in NYU varies from 1 to 8 across the images, which make these datasets challenging to vanishing point detection.

For SU3 and ScanNet datasets, we randomly select 2300 and 2000 images for evaluation respectively following (Lin et al. 2022) to keep a relatively large testing set. NYU dataset is split into 1000, 224 and 225 images for training, validating and testing the models following (Kluger et al. 2020; Lin et al. 2022). We use the ground truth focal length on SU3 and ScanNet datasets. The focal lengths of Natural Scene and NYU datasets are set to the same as ScanNet dataset. We study the influence of changing the focal length in the experiments.

Implementation details Our training and evaluation are implemented in PyTorch. In training, We use AdamW as the model optimizer and set weight decay as 10^{-4}. We train the model for 60 epochs. The initial learning rates are set to 10^{-4} for backbone and 10^{-3} for others. Learning rates are reduced by a factor of 10 in epoch 30 and 45. We use a batch size of 16 and the size of the input images is set to 512×512. The model on SU3, ScanNet and Natural Scene datasets are trained from scratch. For the model on NYU, we train it on ScanNet for a warm-up since the number of images is relatively small.

We divide the hemisphere with $N = 256$ anchors and use 256 queries in VPTR. λ_c, λ_p are set to 1 and 5, and λ_M is set to 1 in Manhattan scenes. During inferring, K vanishing

Method	Natural Scene dataset		
	AA@1°	AA@2°	AA@10°
Zhou *et al.*	18.5	33.0	60.0
NeurVPS	29.1	50.3	**85.5**
Ours	**30.4**	**52.0**	84.3

Table 2: Comparison results for different methods on Natural Scene dataset (Denis, Elder, and Estrada 2008).

Method	NYU dataset		
	AA@3°	AA@5°	AA@10°
J-linkage	39.3	49.8	62.0
CONSAC	38.3	49.9	63.3
Lin *et al.*	**46.2**	**57.4**	69.5
Ours	38.0	52.9	**69.7**

Table 3: Comparison results for different methods on NYU dataset (Silberman et al. 2012).

point with high confidence scores are selected as final predictions. For all experiments, we use $M = 256$ anchors to divide the Gaussian Sphere space. Each anchor is responsible for predicting the vanishing point within an angle distance $\theta_{max} = 0.157$.

Metrics We evaluate all methods by measuring the angle difference between the predicted and the ground truth vanishing points on the Gaussian sphere following (Zhou et al. 2019a; Lin et al. 2022; Tong et al. 2022). The percentage of predictions whose angle difference is smaller than the given thresholds are counted. By generating the angle accuracy (AA) curves via different thresholds, AA@θ is defined as the area under the curve between $[0, \theta]$ divided by θ.

Comparison with the SOTA

We first conduct our comparison on two large benchmarks which follows the Manhattan world assumption including SU3 dataset (Zhou et al. 2019b) and ScanNet dataset (Dai et al. 2017). We compare our method with the state-of-the-art methods including J-Linkage (Tardif 2009), Simon *et al.* (Simon, Fond, and Berger 2018), Li *et al.* (Li et al. 2019), CONSAC (Kluger et al. 2020), NeurVPS (Zhou et al. 2019a), TLC (Tong et al. 2022) and Lin *et al.* (Lin et al. 2022). J-Linkage, Simon *et al.* and Li *et al.* are optimization-based methods. CONSAC, NeurVPS, TLC, Lin *et al.* and our proposed method are learning-based methods. The comparison results are listed in Table 1. We also show the angle accuracy curves for detail comparison in Fig. 5. NeurVPS achieves the highest detection accuracy benefiting from the scanning framework, while suffering from calculation speed limitation. Comparison results show that our method achieves comparable performance with previous SOTA methods on the benchmarks, and keeps the fastest inferring speed of 140 FPS. We also compare the applicability of these approaches. CONSAC and TLC need detected lines as model input. Lin *et al.* method and our proposed method can be applied in more complex cases.

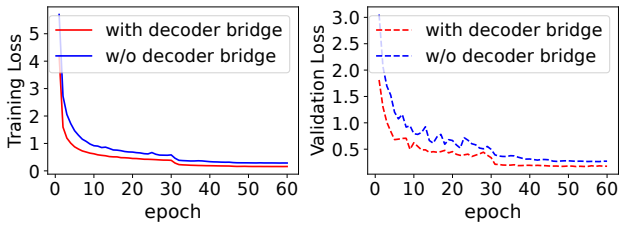

Figure 6: Training and validation curves on the models with and without decoder bridge. The model with decoder bridge shows a faster and better convergence.

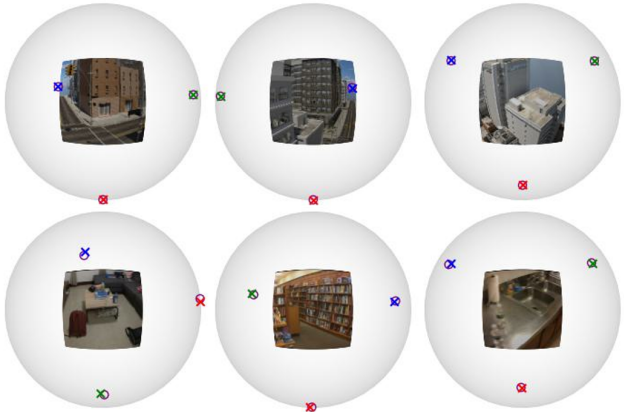

Figure 7: Visual examples on SU3 dataset (top) and ScanNet dataset (bottom). The locations of ground truth vanishing points are marked with '$\circ$', and the predicted vanishing points are marked with '$\times$'.

For evaluating our method in challenging cases, the comparison is also conducted on Natural Scene dataset, where lines are hard to detect. We compare our method with Zhou *et al.* and NeurVPS in Table 2. Our method can achieve comparable performance with the previous SOTA method. We also compare our method on NYU dataset in which the number of vanishing points varies across images. Competitive performance can be obtained from our method.

Visual examples of vanishing point detection results of our method are shown in Fig. 7 for SU3 and ScanNet dataset, and in Fig. 8 for Natural Scene dataset. Our method can produce convincing predictions in both real-world and synthetic images of a variety of scenes.

Ablation Study

To verify the effectiveness of components and find the influence of the hyperparameters in our proposed method, we conduct an ablation and parameter study of our network architecture. The results are presented in Table 4. The ablation study is conducted on SU3 dataset and the angle difference results are reported.

As a baseline Transformer-based vanishing point detector, we use deformable DETR with 6 encoder layers and 6 decoder layers. We modify the prediction head as described in the algorithm to fit our modeling and use confidence loss

Decoder Bridge	Manhattan Loss	Multi-level features	Anchors & Queries	Referring Points	AUC AA@0.5°	AA@1°	AA@3°
-	-	-	256	32	32.8	56.2	83.4
✓	-	-	256	32	43.1	65.4	87.3
✓	✓	-	256	32	44.0	66.9	88.1
✓	✓	✓	256	32	45.8	68.0	**88.5**
✓	✓	✓	128	32	44.1	66.7	87.9
✓	✓	✓	512	32	43.7	66.6	88.1
✓	✓	✓	256	8	45.0	67.3	88.2
✓	✓	✓	256	128	**45.9**	**68.1**	88.4

Table 4: Ablation and parameter study of our method on SU3 dataset (Zhou et al. 2019b). We first construct a baseline method according to our modeling, and then gradually add different components. Experimental results show that bridging the decoder with backbone, applying Manhattan loss and using multi-level strategy can all boost the performance of our approach. We also select the number of anchors and referring points according to the parameter study on them.

f scale	0.5	0.75	1	1.5	2
AA@1°	68.4	68.2	68.0	68.4	67.6
AA@3°	88.6	88.6	88.5	88.8	88.4
AA@5°	92.9	93.0	92.9	93.1	92.9

Table 5: Robustness to the change of focal length. f scale represents the factor we multiply by the calibrated focal length provided by the dataset. The performance of our method changes little when f varies from $\times 0.5$ to $\times 2$, which shows that our method is robust on focal length changing within a large range.

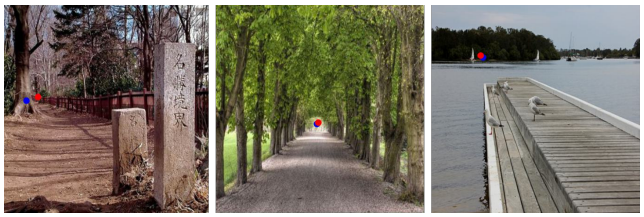

Figure 8: Visual examples on Natural Scene dataset. The red dots represent the ground truth vanishing points and the blue dots represent the predicted ones.

and position loss in training. We find our pipeline is feasible for vanishing point detection. By bridging the decoder directly to the backbone, the performance can be significantly improved. It verifies that the queries of decoder require low-level features for accurate vanishing point detection. We also compare the training and evaluation loss from the models with and without the decoder bridge. As shown in Fig. 6, the model with decoder bridge is easier to converge during training. Then, we add Manhattan loss for training in Manhattan scenes. Experimental results show that adding orthogonal constraints can improve the precision in Manhattan cases. To further utilize the low-level image features such as the features from early layers of the backbone, we make the queries gathering multi-level image features adaptively. The performance is further improved.

Moreover, we vary the number of vanishing point queries (anchors) and referring points to show the influence of these hyperparameters. We find changing the number of referring points has little impact on detection performance. The query number may be appropriately set to 256 since increasing or decreasing it will lead to a lower accuracy.

Since an uncalibrated camera may be used to get the images, e.g., the focal length f is unknown, we also conduct an experiment to study the influence of focal length in our method. We resize the true f by a scale factor. Then new vector presentations of vanishing points can be obtained and models can be trained with them. We evaluate the vanishing point detection in the original camera space and report the result in Table 5. We find the performance of our method changes little when f varies from $\times 0.5$ to $\times 2$.

VPTR is hard to predict accurate vanishing points when there are seldom images that can be used in training, such as on York Urban Dataset (YUD) which contains 102 images. Using geometric prior for data-efficient training or applying transferring learning technology may help to improve performance in this case. We leave it as future work.

Conclusion

We propose a novel clear and efficient vanishing point detection method named Vanishing Point TRansformer (VPTR), which can directly regress their locations from images. In VPTR, we first pose the detection as a point object detection problem on the Gaussian hemisphere with sphere region division algorithm. To efficiently utilize the low-level geometric information and high-level semantic information, we bridge the decoder to the image backbone without encoder and queries can gather multi-level image features directly in VPTR. We also present new losses to supervise our training in both Manhattan and non-Manhattan scenes. Extensive experiments on synthetic and real-world datasets show that our method can predict vanishing points in both structural and natural scenes efficiently. Moreover, our method can run at 140 FPS during inferring. To the best of our knowledge, VPTR is the fastest learning-based vanishing point detection method achieving similar performance.

References

Almansa, A.; Desolneux, A.; and Vamech, S. 2003. Vanishing point detection without any a priori information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4): 502–507.

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

Bazin, J.-C.; Seo, Y.; Demonceaux, C.; Vasseur, P.; Ikeuchi, K.; Kweon, I.; and Pollefeys, M. 2012. Globally optimal line clustering and vanishing point estimation in manhattan world. In *Conference on Computer Vision and Pattern Recognition*, 638–645. IEEE.

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. Citeseer.

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 *International Conference on Robotics and Automation (ICRA)*, 4496–4503. IEEE.

Chen, X.; Yan, B.; Zhu, J.; Wang, D.; Yang, X.; and Lu, H. 2021. Transformer tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8126–8135.

Collins, R. T.; and Weiss, R. S. 1990. Vanishing point calculation as a statistical inference on the unit sphere. In *ICCV*, volume 90, 400–403.

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.

Denis, P.; Elder, J. H.; and Estrada, F. J. 2008. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *European conference on computer vision*, 197–210. Springer.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Ge, W.; Song, Y.; Zhang, B.; and Dong, Z. 2021. Globally Optimal and Efficient Manhattan Frame Estimation by Delimiting Rotation Search Space. In *ICCV*, 15213–15221.

Girshick, R. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 1440–1448.

Girshick, R.; Donahue, J.; Darrell, T.; and Malik, J. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

Huang, L.; Tan, J.; Liu, J.; and Yuan, J. 2020. Hand-Transformer: Non-Autoregressive Structured Modeling for 3D Hand Pose Estimation. In *European Conference on Computer Vision*, 17–33. 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.

Law, H.; and Deng, J. 2018. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, 734–750.

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, 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 *ICCV*, 5661–5670.

Li, H.; Zhao, J.; Bazin, J.-C.; Chen, W.; Liu, Z.; and Liu, Y.-H. 2019. 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, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2117–2125.

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 (CVPR)*, 6103–6113.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *ICCV*.

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 *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.

lvaro González. 2010. Measurement of Areas on a Sphere Using Fibonacci and Latitude–Longitude Lattices. *Mathematical Geosciences*, 42(1): 49–64.

Mirzaei, F. M.; and Roumeliotis, S. I. 2011. Optimal estimation of vanishing points in a manhattan world. In *2011 International Conference on Computer Vision*, 2454–2461. IEEE.

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779–788.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

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 *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, 746–760. Springer.

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 (ECCV)*, 318–333.

Straforini, M.; Coelho, C.; and Campani, M. 1993. Extraction of vanishing points from images of indoor and outdoor scenes. *Image and Vision Computing*, 11(2): 91–99.

Strudel, R.; Garcia, R.; Laptev, I.; and Schmid, C. 2021. Segmenter: Transformer for Semantic Segmentation. *ICCV*.

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 (CVPR)*, 6093–6102.

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 *ICCV*, 3732–3741.

Yang, F.; Yang, H.; Fu, J.; Lu, H.; and Guo, B. 2020. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5791–5800.

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, 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, X.; Wang, D.; and Krähenbühl, P. 2019. Objects as points. *arXiv preprint arXiv:1904.07850*.

Zhou, X.; Zhuo, J.; and Krahenbuhl, P. 2019. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 850–859.

Zhou, Y.; Qi, H.; Huang, J.; and Ma, Y. 2019a. Neurvps: Neural vanishing point scanning via conic convolution. *arXiv preprint arXiv:1910.06316*.

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.

Zhou, Z.; Farhat, F.; and Wang, J. Z. 2017. Detecting dominant vanishing points in natural scenes with application to composition-sensitive image retrieval. *IEEE Transactions on Multimedia*, 19(12): 2651–2665.

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*.