Transformer Based Line Segment Classifier with Image Context for Real-Time Vanishing Point Detection in Manhattan World

Xin Tong, Xianghua Ying*, Yongjie Shi, Ruibin Wang and Jinfa Yang
Key Laboratory of Machine Perception (MoE)
School of Artificial Intelligence, Peking University
{xin_tong, xhying, shiyongjie, robin_wang, jinfayang}@pku.edu.cn

Abstract

Previous works on vanishing point detection usually use geometric prior for line segment clustering. We find that image context can also contribute to accurate line classification. Based on this observation, we propose to classify line segments into three groups according to three unknown-but-sought vanishing points with Manhattan world assumption, using both geometric information and image context in this work. To achieve this goal, we propose a novel Transformer based Line segment Classifier (TLC) that can group line segments in images and estimate the corresponding vanishing points. In TLC, we design a line segment descriptor to represent line segments using their positions, directions and local image contexts. Transformer based feature fusion module is used to capture global features from all line segments, which is proved to improve the classification performance significantly in our experiments. By using a network to score line segments for outlier rejection, vanishing points can be got by Singular Value Decomposition (SVD) from the classified lines. The proposed method runs at 25 fps on one NVIDIA 2080Ti card for vanishing point detection. Experimental results on synthetic and real-world datasets demonstrate that our method is superior to other state-of-the-art methods on the balance between accuracy and efficiency, while keeping stronger generalization capability when trained and evaluated on different datasets.

1. Introduction

Under the pinhole camera model, parallel world lines in 3D are projected into 2D image lines that converge on an image point, which is called the vanishing point (VP). Vanishing point detection is one of the most fundamental problems in computer vision. A fast and accurate vanishing point detection algorithm enables and enhances applications such as camera calibration, 3D reconstruction, object detection, wireframe parsing, and autonomous driving.

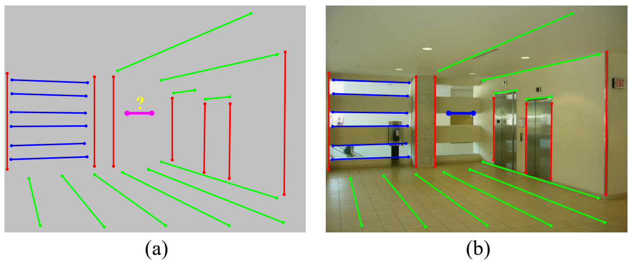

Figure 1. (a) Classical vanishing point detection methods usually depend on geometric prior for line segment clustering. However, it is not easy to determine which group the line segment (colored purple) should belong to only based on the geometric prior, since the line segment is close to both candidate horizontal vanishing points. (b) On the contrary, one can easily categorize the line segment when the image context is given. The example is from [12]. Combining image context, it may become easier and faster to cluster a line segment to a certain group. Motivated by this observation, we use both image context and geometric information in learning based line segment clustering for vanishing point detection in this work.

Classical vanishing point detection methods usually follow three steps including line detection, line clustering and vanishing point regression. With Manhattan world assumption, line segments are supposed to be clustered into three groups according to the three orthogonal vanishing points. Previous methods often consider the geometric prior that line segments in the same group should intersect at the same point (the vanishing point). Therefore, the image is often no longer used after the lines are extracted. However, as shown in Fig. 1, it is not easy to determine which group the line segment should belong to only based on the geometric prior, since the line segment is close to both candidate horizontal vanishing points. This ambiguity is eliminated when the image contexts are given, which shows the potential of the image contexts in improving the line clustering performance. Motivated by this observation, we employ image

*Corresponding Author

contexts with geometric features in learning based line segment clustering for vanishing point detection.

Recent learning based methods usually estimate vanishing points, horizon or score candidate vanishing points from the image contexts directly. In this work, we use networks to group line segments and remove outliers for vanishing point detection, which is demonstrated to be less likely to overfit on a certain dataset in our experiments. We present each detected line segment in the given image as a 1D feature vector. For accurately classifying the line segments, it is important to propagate and gather global information from all line segments. Inspired by the good performance of Transformers in many computer vision tasks, we use Transformer encoder architecture to efficiently capture non-local correlation over all line segments. By considering the feature vectors of line segments as a sequence of tokens, the Transformer based feature fusion module can be easily applied in our algorithm.

In this paper, we propose a novel method named Transformer based line segment classifier (TLC) for real-time vanishing point detection in Manhattan world. Our method is composed of two modules including a line segment descriptor and a feature fusion module. Given an image and line segments in it, TLC first represents each line segment as a 1D feature vector consisting of both local image contexts and geometric features with the line segment descriptor. The local image contexts are extracted from a Convolutional Neural Network (CNN) model followed by a novel line pooling operation. The geometric features are represented by one-hot encoding of the direction and positions of points uniformly sampled from the line segment. Then the feature vectors are processed with a clustering network and a scoring network, respectively. These networks are designed based on Transformer encoder architecture for efficient feature fusion. For each line segment, the clustering branch predicts the probability of each group it may belong to, and the scoring branch predicts a score meaning its confidence in detecting the corresponding vanishing point. Finally the locations of the vanishing points can be calculated using Singular Value Decomposition (SVD) based on the clustered line segments and the predicted scores. The proposed TLC can classify line segments for vanishing point detection in a non-iterative manner and be end-to-end trainable. Therefore, the inferring speed of the proposed method is very fast and further post-processing can be applied on the clustered line segments easily.

The contributions of our work can be summarized below: (1) This work is the first one that uses learning based method to classify line segments for vanishing point detection with both image contexts and geometric information. (2) For this goal, we propose a novel Transformer based Line segment Classifier (TLC). In TLC, we present a novel line segment descriptor to represent a line segment with a 1-D feature vector, and use a Transformer based module to fuse features of different line segments. Our method can group line segments according to the vanishing points and predict their confidence score to be inliers in a non-iterative manner. (3) The proposed method runs at 25 FPS on one NVIDIA 2080Ti card for vanishing point detection. Experimental results on synthetic and real-world datasets show that our method is superior to other state-of-the-art methods on the balance between accuracy and efficiency, while keeping stronger generalization capability when trained and evaluated on different datasets. We also construct a real-world street view vanishing point (SVVP) dataset[1] for further evaluating the proposed method.

2. Related Works

Vanishing Point Detection. Since the seminal work introduced in [2], various methods have been designed for vanishing point detection. Previous works tackle the problem by using Gaussian sphere [10,26,32], Manhattan world assumption [3, 24, 27], Hough transformation [1], Branch-and-Bound [3, 14, 22], etc. Line based methods are the most widely used approaches. They usually start with line detection [5, 36]. Then the images are usually left aside and the parametric lines are clustered using Hough transformation [25], RANSAC [4, 38], J-Linkage [33], EM algorithm [12], dual space [20]. In this work, we introduce image context in line clustering process and use learning based method to group the line segments corresponding to the same vanishing point.

Recently, learning based vanishing point detection methods achieve promising results benefiting from the strong ability to extract features of neural networks. In [7, 30, 42], CNN-based methods are used to classify or regress the vanishing point directly. In [41] global image context is extracted to guide the generation of horizon line candidates. Kluger *et al.* [17] use CNN to find vanishing points from inverse gnomonic image. Bingham mixture model is employed to estimate vanishing points in [21]. A neural network is used to update the conditional sampling probabilities for line clustering in [18]. Zhou *et al.* [43] propose conic convolution for vanishing point detection which can enforce feature extractions and aggregations along the structural lines. In this work, we apply Transformer based module to classify and score the line segments in a non-iterative way for vanishing point detection.

Line Representation. Lines have been widely used in learning based methods for various computer vision tasks, which are represented in different ways. Li *et al.* [23] propose a line proposal unit (LPU) to generate candidate traffic lines. Line pooling layer is presented in [19] that performs bilinear interpolation of each sampled location on

[1]https://github.com/tongxin94/svvp/

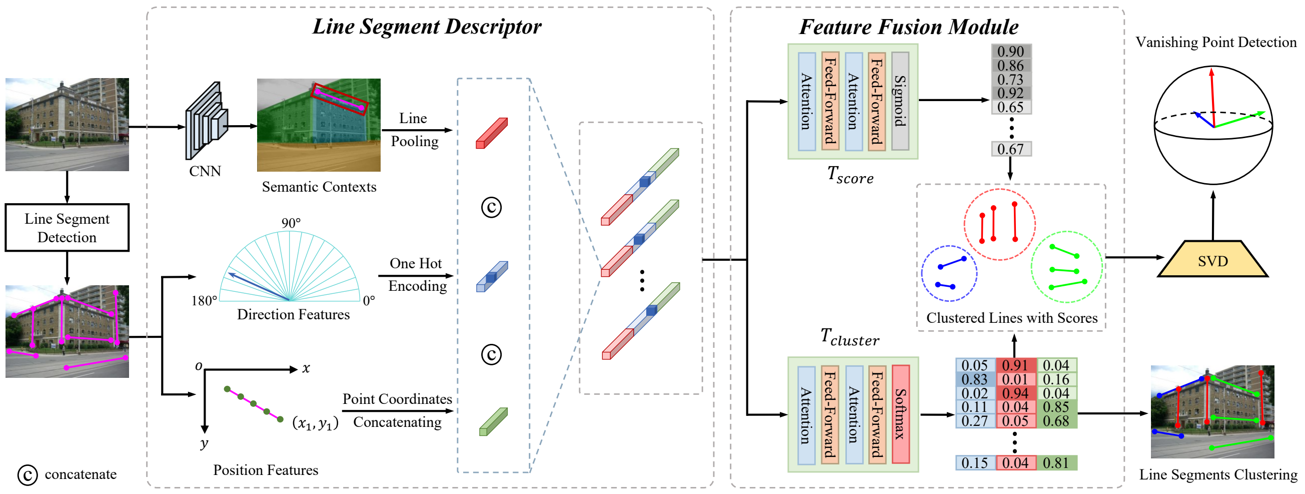

Figure 2. Overview of the proposed TLC. Given an image with Manhattan world assumption and the line segments in it, TLC first represents each line segment as a 1D feature vector which consists of the local image contexts and geometric information. Transformer based feature fusion module is used to capture non-local correlation of all the line segments. The module predicts the probability of each group and the confidence scores in estimating the vanishing point. Vanishing points can be obtained using SVD with each group of line segments and the corresponding scores.

the feature maps. Given image lines, [34] proposes a line descriptor that averages the feature vectors of the sampled points on the lines in visual SLAM. The endpoints, centroid and direction are used to represent a line segment in CONSAC [18]. For vanishing point detection, we present a novel line segment descriptor considering both the local image contexts extracted from CNN and geometric features including positions and directions.

Visual Transformer. Transformer is originated in neural language processing (NLP) [35] then widely applied in computer vision tasks because of its strong representation capabilities. Chen *et al.* [8] train a sequence transformer to auto-regressively predict pixels that can get comparable performance in image classification with CNN based methods. Dosovitskiy *et al.* [13] use transformer (ViT) in classification by dividing a image into patches and adding an extra learnable classification token in transformer for image classification. Carion *et al.* [6] apply transformer (DETR) in end-to-end object detection. Huang *et al.* [16] use a transformer based network to estimate 3D hand pose from point clouds. Transformer are also used in image super-resolution [39], image generation [29], video inpainting [40], tracking [9, 37] and hold a potential to be applied in more applications. In our method, we apply Transformer encoder architecture in line segment clustering and scoring for vanishing point detection.

3. Algorithm

3.1. Overview

The overview of our algorithm is shown in Fig. 2. Given an image with Manhattan world assumption and the line segments in it, our algorithm can cluster the line segments into three groups according to the three vanishing points and predict the location of each vanishing point. The proposed method, named Transformer based Line segment Classifier (TLC), is a neural network model composed of a line segment descriptor and a feature fusion module, which are introduced in detail in the following subsections.

3.2. Line Segment Descriptor

The line segment descriptor represents a line segment as a 1D vector of pre-defined size, in which local image contexts and geometric features are combined together. For geometric features, the position in the image and the direction information are used, which are represented as f_{pos} and f_{dir} respectively. We sample some points uniformly on the line segments. The coordinates of the points can be used to describe the position of the line segment. To describe the direction, we uniformly divide $0°$ to $180°$ degrees into several intervals and encode the direction with one-hot encoding according to which interval the direction is located.

We present a line pooling module to capture local image contexts of a line segment from the image, which is illustrated in Fig. 3. In the module, we consider using the semantic features around the line segments to describe them. We first use a CNN model to extract feature maps from the image. For each line segment, we then uniformly sample N points along it and get N feature vectors of a total size of $N \times C$. Bilinear interpolation is used to compute the values of each sampling point from the nearby grid points on the feature maps, since sampling points may not locate on the grid points. As we want to obtain a 1D fix-size representation of each line segment f_{con}, we finally apply a weighted

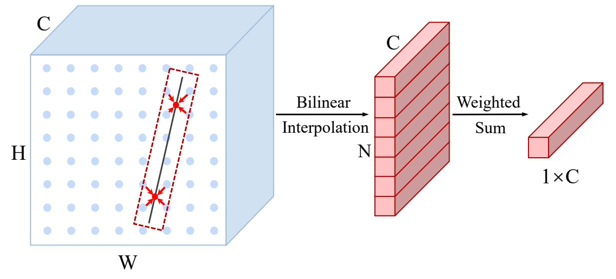

Figure 3. Illustration of the presented line pooling module. Given the feature maps extracted from a CNN model and a line segment, uniform sampling is applied along the line segment. Bilinear interpolation is used to compute the feature vector of each sampling point from the nearby grid points on the feature maps. All the feature vectors at sampling points are summed with learned weights. We finally get a 1D fix-size feature vector for each line segment.

summation to the feature vectors for squeezing them into a size of $1 \times C$. A similar architecture is also used in [34] for line matching. They average the feature vectors for final representation, while we use weighted summation for stronger representation ability.

The final representation of a line segment $f_{line} \in \mathbb{R}^{1\times C}$ is the concatenation of the above three features, which can be described as

$$f_{line} = [f_{con}, f_{pos}, f_{dir}]. \tag{1}$$

In practice, ResNet18 is used as the semantic feature extractor followed by a 1×1 convolutional layer to reduce the channel of feature maps to 32. We use 32 coordinates from 16 uniformly sampled points for representing the position and a one-hot vector with a size of 36 for representing the direction, respectively.

3.3. Feature Fusion Module

Our feature fusion module is used to predict the probability of each group and the confidence scores from the feature vectors of the line segments. The module is composed of two independent network branches $\mathcal{T}_{cluster}$ and $\mathcal{T}_{score}$. Starting from feature vectors of M line segments $f_{lines} \in \mathbb{R}^{M\times C}$, $\mathcal{T}_{cluster}$ is used to predict the probability p that the line segments correspond to the unknown-but-sought vanishing points, which is represented as

$$p = softmax(\mathcal{T}_{cluster}(f_{lines})). \tag{2}$$

Similarly, $\mathcal{T}_{score}$ is used to predict the confidence score s in estimating the vanishing point, which is represented as

$$s = sigmoid(\mathcal{T}_{score}(f_{lines})). \tag{3}$$

A line segment with high confidence score is supposed to have a greater probability of passing through the vanishing point (or closer to the vanishing point).

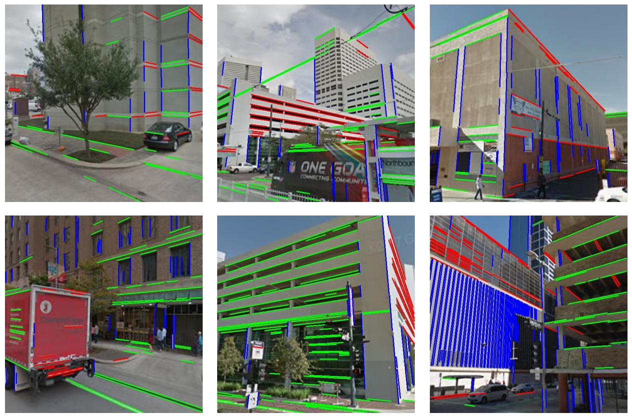

Figure 4. Visual examples of our constructed SVVP dataset. The line segments corresponding to the different vanishing points are colored blue, red and green, respectively.

The networks in the feature fusion module are designed based on Transformer encoder architecture, which is composed of an attention layer and a feed-forward network (FFN). The attention layer can capture non-local correlation from different line segments, which is represented as

$$\mathcal{A}tt(f) = softmax(\frac{Q(f) \cdot K(f)^T}{\sqrt{d_k}}) \cdot V(f), \tag{4}$$

where Q, K, V are linear layers for extracting the query, key and value vectors from feature vectors f of line segments, respectively. d_k is the dimension of $K(f)$ for normalization. The FFN $\mathcal{F}(\cdot)$ consists of a fully connected layer and residual connection. We do not use position embedding as classical Transformer since the final clustering results are supposed to be irrelevant to the order of the input line segments. Thus, the network architecture we use in the feature fusion module is represented as

$$\mathcal{T}(f) = \mathcal{F}(\mathcal{A}tt(f) + f). \tag{5}$$

3.4. Losses and Vanishing Points Estimation

We use cross entropy loss to classify the line segments according to the three unknown-but-sought vanishing points in Manhattan world, e.g., upwards first, then followed by left and right ones in horizontal. The line segment classification loss can help our method converge quickly and robustly, which can be written as

$$\mathcal{L}_{class} = -\frac{1}{M_{in}} \sum_{i=1}^{M_{in}} \sum_{c=1}^{3} g_i^c \log p_i^c, \tag{6}$$

where M_{in} is the number of inlier line segments, since we do not consider the outliers in this loss. p_i^c is the predicted probability of the i-th line segment that belongs to group c. g_i^c is the corresponding ground-truth label.

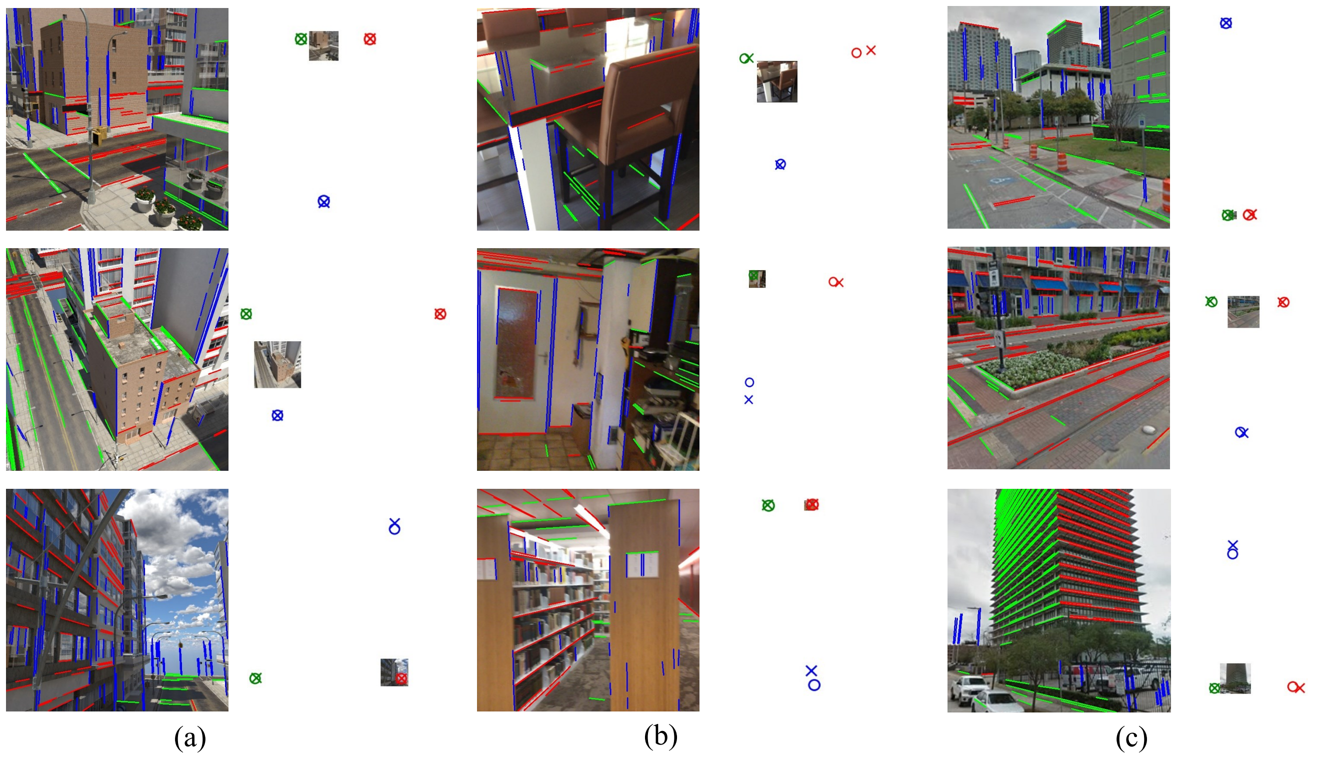

Figure 5. Visual examples of line segment clustering and vanishing point detection results of our method on (a) SU3 [44], (b) ScanNet [11] and (c) the SVVP datasets constructed by ourselves. The line segment clustering results are directly from the network output without post-processing. Different clusters of line segments and vanishing points are colored with red, green and blue, respectively. The locations of ground truth vanishing points are marked with '$\circ$', and the predicted vanishing points are marked with '$\times$'. Our method can produce accurate predictions in real-world and synthetic images of a variety of scenes.

We use BCE loss to supervise the $\mathcal{T}_{score}$ for removing outliers, which can be represented as

$$\mathcal{L}_{score} = -\frac{1}{M}\sum_{i=1}^{M}(y_i \log s_i + (1-y_i)\log(1-s_i)), \quad (7)$$

where M is the number of both inliers and outliers. s_i represents the predicted score of the i-th line segment and y_i is the corresponding ground-truth. The total loss can be defined as the sum of the above loss terms, which can be written as

$$\mathcal{L}_{total} = \mathcal{L}_{class} + \mathcal{L}_{score}. \quad (8)$$

We use Singular Value Decomposition (SVD) to calculate the vanishing points based on the clustered line segments and their confidence scores. A vanishing point can be represented as a normalized line direction vector, which is also called the Gaussian sphere representation. In homogeneous coordinates, line segments $L \in \mathbb{R}^{M\times 3}$ can be defined by the normal of the plane they form with the camera center [41]. The vector of the vanishing point can be got from $svd(L)$. Furthermore, a line segment with a high score is supposed to contribute more in the calculation. In the inferring phase, we use a hard selection to directly remove the line segments with a score under 0.5, since outliers may impact the results seriously. Thus, with the scores $s \in \mathbb{R}^{M\times 1}$, the vector of the vanishing point can be got from $svd(\mathbf{1}_{s>0.5}(s)L)$.

The vanishing points are computed independently. For uncalibrated images, a pre-defined approximate focal length (e.g., width of the image) can be used to estimate the intersections of the lines in each group. For calibrated images, the estimated vanishing points are supposed to be orthogonal to others (e.g., up to a tolerance γ). Otherwise, the best orthogonal pair can be selected and the third vanishing point can be calculated by the cross product.

4. Experimental Results

4.1. Experimental Setup

We conduct our experiments in three publicly available datasets including SU3 dataset [44], ScanNet dataset [11] and York Urban Dataset (YUD) [12]. All the datasets follow the Manhattan world assumption, where there should be three orthogonal vanishing points in each image. The SU3 dataset is a photo-realistic dataset that contains 23000 synthetic outdoor images. The vanishing points are directly

Method	SU3 [44]			ScanNet [11]			SVVP			FPS
	AA@3°	AA@5°	AA@10°	AA@3°	AA@5°	AA@10°	AA@3°	AA@5°	AA@10°	
J-linkage [33]	69.2	77.0	84.4	27.8	41.7	57.7	32.8	45.7	60.2	1.2
Simon *et al.* [31]	70.2	77.9	85.1	25.7	39.9	56.6	45.4	59.6	73.2	0.6
Wu *et al.* [38]	74.8	79.5	83.9	22.9	36.8	54.0	39.1	52.4	67.9	23
Lu *et al.* [24]	81.4	87.8	93.0	35.6	53.2	71.6	48.5	64.8	80.0	**25**
Li *et al.* [22]	59.1	66.9	74.6	35.0	50.2	66.9	39.3	53.0	66.8	**25**
CONSAC [18]	77.9	85.2	91.0	31.1	46.1	62.4	43.8	56.5	69.4	2
NeurVPS-SU3 [43]	**94.4**	**96.5**	**98.2**	17.4	26.8	41.0	27.1	40.4	55.3	0.5
NeurVPS-ScanNet [43]	57.2	72.6	85.5	36.1	**54.3**	**74.9**	35.5	53.7	73.5	0.5
Ours-SU3	91.3	94.6	97.1	36.0	53.4	71.6	**51.6**	**67.7**	**82.6**	**25**
Ours-ScanNet	88.9	92.8	95.8	**36.2**	53.9	72.6	49.2	65.0	80.2	**25**

Table 1. Comparison results on SU3 [44], ScanNet [11] and the SVVP datasets. We compare our method with J-linkage [33], Simon *et al.* [31], CONSAC [18], NeurVPS [43], Wu *et al.* [38], Lu *et al.* [24] and Li *et al.* [22]. Method-dataset represents the learning based method trained on the dataset. Our method gets better performance on the balance between accuracy and efficiency, while achieving stronger generalization capability compared to other learning based methods when trained and evaluated on different datasets.

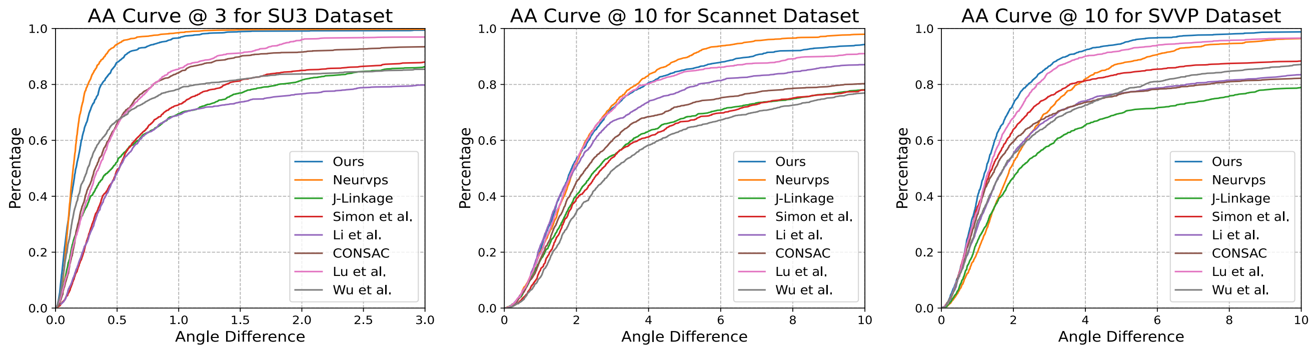

Figure 6. Angle accuracy curves for different methods on SU3 [44], ScanNet [11] and the SVVP datasets constructed by ourselves.

computed from the CAD models of the buildings. The ScanNet dataset is a real-world dataset and captures indoor scenes. It provides 189916 training images and 53193 validation images. We follow [43] to get the ground truth vanishing points. YUD includes 102 images of outdoor and indoor scenes with their ground truth VP triplets. For further evaluating our method, we also construct a real-world street view dataset for vanishing point detection, called SVVP. We get 500 images from Google Street View and label the vanishing points manually. Some examples are shown in Fig. 4.

For SU3 and ScanNet datasets, we use 500 images for evaluation following [43], and the others (in the training set) for training. For training our model, we generate ground truth classes of the line segments using the provided ground truth vanishing points. The ground truth class of a line segment is assigned according to the closest vanishing point to it. When a line segment is close to more than one vanishing point, we manually label the line segment. We use two thresholds δ_{in} and δ_{out} to discriminate the inliers and outliers. They are set to $1°$, $10°$ for SU3 dataset and $1.5°$, $15°$ for ScanNet dataset empirically. We find in ScanNet dataset, the inliers for the real vanishing points may be very few in some images. To keep the training images having enough inliers, we remove the images where the number of inliers are less than 30 in the training phase. In the evaluation of YUD and SVVP, we use the model trained with SU3 or ScanNet dataset, since there are not enough images for training in YUD and SVVP.

We use LSD [36] as the line segment detector for the proposed method. Our training and evaluation are implemented in PyTorch. For training, we use SGD optimizer. The learning rate is set to 0.005, while the momentum and weight decay are set to 0.9 and 0.0001 respectively. Due to GPU memory limitations, we use a batch size of 16 and the size of the input images is set to 512×512. In training, we randomly select 100 line segments in each image. For obtaining more accurate results, we also run a RANSAC based post-processing for 20 iterations per vanishing point before using SVD.

We evaluate all methods by measuring the angle difference between the predicted vanishing points and the ground truth on Gaussian sphere following [43]. The percentage

Method	YUD [12]		
	AA@3°	AA@5°	AA@10°
J-linkage [33]	40.2	50.5	64.1
Simon *et al.* [31]	40.1	58.2	77.5
Wu *et al.* [38]	44.3	61.4	77.4
Li *et al.* [22]	51.1	66.1	80.5
Lu *et al.* [24]	58.0	73.2	86.2
CONSAC [18]	62.1	73.7	84.1
NeurVPS-SU3 [43]	39.9	50.3	65.0
NeurVPS-ScanNet [43]	30.3	50.4	71.0
Ours-SU3	**65.5**	**77.1**	**87.4**
Ours-ScanNet	63.1	76.1	87.3

Table 2. Comparison results for different methods on YUD [12].

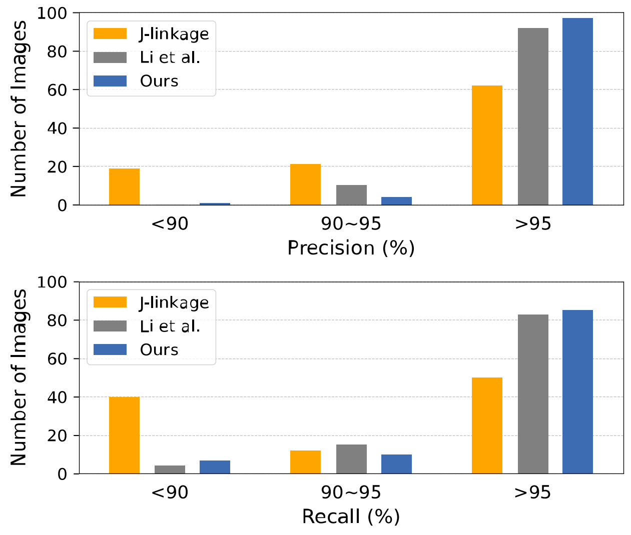

Figure 7. Quantitative comparisons in terms of precision and recall of line clustering on all the images of the YUD [12].

of predictions whose angle difference is smaller than some threshold can be counted. By plotting the angle accuracy (AA) curves via different thresholds, AA@θ is defined as the area under the curve between $[0, \theta]$ divided by θ. We also measure the clustering accuracy of the line segments of our method, which is the ratio of line segments correctly clustered by our method to the total line segments.

4.2. Comparison with the SOTA

We conduct our comparison on four benchmarks including SU3 dataset [44], ScanNet dataset [11], York Urban Dataset (YUD) [12] and the SVVP dataset constructed by ourselves. We show some visual examples of line segment clustering and vanishing point detection results of our method on these datasets in Fig. 5. Notice the clustering results of the line segments in the images are directly from the network outputs without post-processing. Our method can

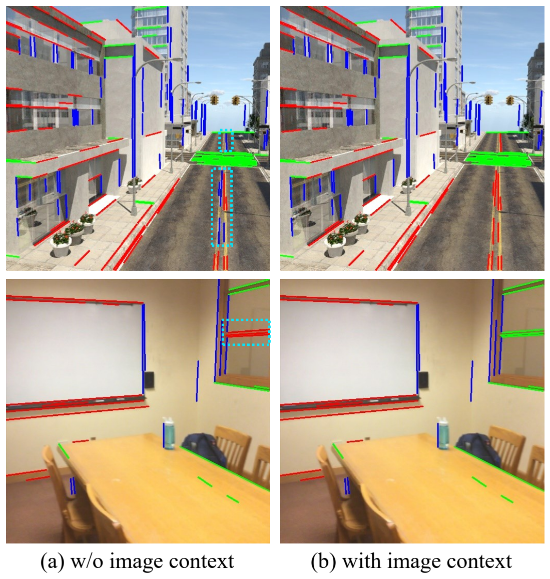

Figure 8. Line segments clustering results of two examples from SU3 [44] (top) and ScanNet [11] dataset (bottom) by our method (a) without using image context and (b) using image context. Image context is able to improve the clustering performance of the line segments, especially when the geometric prior is difficult to categorize them. The line segments enclosed by cyan bounding boxes denote the representative mistakenly clustered inliers.

produce accurate predictions in both real-world and synthetic images of a variety of scenes.

We compare our method with some state-of-the-art methods including J-Linkage [33], Simon *et al.* [31], Li *et al.* [22], Wu *et al.* [38], Lu *et al.* [24], CONSAC [18] and NeurVPS [43]. J-Linkage, Simon *et al.*, Wu *et al.*, Lu *et al.* and Li *et al.* are optimization based methods. The others are learning based methods. The comparison results are listed in Table 1 and Table 2. We also show the angle accuracy curves for detail comparison in Fig. 6. Comparison results show that our method achieves comparable or better performance than previous SOTA methods on the benchmarks, and keeps a inferring speed of 25 FPS. Moreover, our method achieves stronger generalization capability when trained and evaluated on different datasets.

We also report the precision and recall for line segment clustering on the images of YUD in Fig. 7. We use the manually extracted lines and their ground truth vanishing point association provided by YUD, similar to [22]. Our method performs better than other line based methods in the comparison.

4.3. Ablation Study

To verify the effectiveness of each module and further compare each module with its variants in our proposed method, we conduct an ablation study of our network architecture. The ablation study is conducted on SU3 dataset,

Position	Direction	Context Extractor	Line Pooling	Classification Network	Classification Acc
16 Points	36 Intervals	-	-	MLP based	82.1%
16 Points	36 Intervals	-	-	Transformer based	98.2%
16 Points	-	-	-	Transformer based	95.4%
16 Points	18 Intervals	-	-	Transformer based	97.8%
-	36 Intervals	-	-	Transformer based	84.9%
3 Points	36 Intervals	-	-	Transformer based	98.0%
32 Points	36 Intervals	-	-	Transformer based	98.2%
16 Points	36 Intervals	ResNet18	average pooling	Transformer based	98.7%
16 Points	36 Intervals	ResNet18	weighted sum	Transformer based	98.9%
16 Points	36 Intervals	RefineNet	average pooling	Transformer based	98.6%
16 Points	36 Intervals	RefineNet	weighted sum	Transformer based	**99.0%**

Table 3. Ablation study of our method on SU3 dataset [44]. We sample different number of points on each line segment and divide the angle space into different number of intervals. The classification accuracy of line segments is compared as the metric. '-' means the corresponding feature is not used. Experimental results show that both geometric features and image contexts are important for clustering the line segments. Using Transformer based architecture to capture global information from all line segments can significantly improve the line classification results, and employing image context in line classification can further boost the performance.

which is presented in Table 3. We report the results on the line segment classification accuracy directly from the network output.

As a baseline learning based line segment classifier, we use position and direction information to compose the feature vectors of line segments, and use MLP based network to classify them. By applying the feature fusion module described in Section 3.3, we find the classification performance can be significantly improved. Then, we vary the number of sampled point and angle intervals for getting the best hyper-parameter in our descriptor. Furthermore, we test two line pooling modules including average pooling and weighted sum. Two efficient context extraction backbones including ResNet18 [15] and RefineNet [28] are also tested. Experimental results demonstrate the effectiveness of TLC architecture. We show the improvement of our method by employing image contexts in composing the feature vector of line segments with a visual comparison in Fig. 8. Image context is able to improve the clustering performance of the line segments, especially when the geometric prior is difficult to categorize them.

The scoring network in TLC can improve the efficiency of post-processing by reducing the number of its iterations. We also test different architectures of the scoring network. We find Transformer based module can get 94% precision and 81% recall for outlier detection on SU3 dataset, while MLP based one produces approximately random results.

4.4. Discussions

Compared to the algorithm directly using the ground truth vanishing point as the supervisory signal, an advantage of our method is that post-processing such as RANSAC, EM can be easier to apply for further improving the performance. Another potential advantage is that our method is less sensitive to labeling errors, since small biases in labeling the vanishing points may not change the ground truth line classification labels. We also discuss the limitations of our method. First, our method is a line-based method, which cannot work when there are no detected lines in images. Second, our method can predict at most three vanishing points directly. Exploiting new clustering loss may make it possible to predict more vanishing points and our future works will focus on the problem.

5. Conclusion

In this paper, we propose a novel method named Transformer based line segment classifier (TLC) for real-time vanishing point detection. Our method exploits the image contexts in accurately grouping the line segments. We model the line classification problem as a sequence prediction problem. Each line segment is represented as a 1D feature vector and Transformer based architecture is used to exchange information for updating the features of the line segments. Given an image with Manhattan world assumption and the line segments in it, TLC can classify them into three groups according to the vanishing points and predict the outliers in a non-iterative way. Vanishing points can be estimated using SVD with each group of line segments. Our method achieves comparable or better performance compared with other state-of-the-art vanishing point detection methods on both synthetic and real-world datasets, and gets an inferring speed of 25 FPS.

Acknowledgments

This work was supported in part by State Key Development Program Grand No. 2020YFB1708002, and NNSFC Grant No. 61971008.

References

[1] Andrés Almansa, Agnes Desolneux, and Sébastien Vamech. Vanishing point detection without any a priori information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):502–507, 2003. 2

[2] Stephen T Barnard. Interpreting perspective images. *Artificial intelligence*, 21(4):435–462, 1983. 2

[3] Jean-Charles Bazin, Yongduek Seo, Cédric Demonceaux, Pascal Vasseur, Katsushi Ikeuchi, Inso Kweon, and Marc Pollefeys. Globally optimal line clustering and vanishing point estimation in manhattan world. In *CVPR*, pages 638–645. IEEE, 2012. 2

[4] Robert C Bolles and Martin A Fischler. A ransac-based approach to model fitting and its application to finding cylinders in range data. In *IJCAI*, volume 1981, pages 637–643. Citeseer, 1981. 2

[5] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986. 2

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 3

[7] Chin-Kai Chang, Jiaping Zhao, and Laurent Itti. Deepvp: Deep learning for vanishing point detection on 1 million street view images. In *ICRA*, pages 4496–4503. IEEE, 2018. 2

[8] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, pages 1691–1703. PMLR, 2020. 3

[9] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, pages 8126–8135, 2021. 3

[10] Robert T Collins and Richard S Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *ICCV*, volume 90, pages 400–403, 1990. 2

[11] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. 5, 6, 7

[12] Patrick Denis, James H Elder, and Francisco J Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *ECCV*, pages 197–210. Springer, 2008. 1, 2, 5, 7

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[14] Wuwei Ge, Yu Song, Baichao Zhang, and Zehua Dong. Globally optimal and efficient manhattan frame estimation by delimiting rotation search space. In *ICCV*, pages 15213–15221, 2021. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 8

[16] Lin Huang, Jianchao Tan, Ji Liu, and Junsong Yuan. Hand-transformer: Non-autoregressive structured modeling for 3d hand pose estimation. In *ECCV*, pages 17–33. Springer, 2020. 3

[17] Florian Kluger, Hanno Ackermann, Michael Ying Yang, and Bodo Rosenhahn. Deep learning for vanishing point detection using an inverse gnomonic projection. In *GCPR*, pages 17–28. Springer, 2017. 2

[18] Florian Kluger, Eric Brachmann, Hanno Ackermann, Carsten Rother, Michael Ying Yang, and Bodo Rosenhahn. Consac: Robust multi-model fitting by conditional sample consensus. In *CVPR*, pages 4634–4643, 2020. 2, 3, 6, 7

[19] Jun-Tae Lee, Han-Ul Kim, Chul Lee, and Chang-Su Kim. Semantic line detection and its applications. In *ICCV*, pages 3229–3237, 2017. 2

[20] José Lezama, Rafael Grompone von Gioi, Gregory Randall, and Jean-Michel Morel. Finding vanishing points via point alignments in image primal and dual domains. In *CVPR*, pages 509–515, 2014. 2

[21] Haoang Li, Kai Chen, Pyojin Kim, Kuk-Jin Yoon, Zhe Liu, Kyungdon Joo, and Yun-Hui Liu. Learning icosahedral spherical probability map based on bingham mixture model for vanishing point estimation. In *ICCV*, pages 5661–5670, 2021. 2

[22] Haoang Li, Ji Zhao, Jean-Charles Bazin, Wen Chen, Zhe Liu, and Yun-Hui Liu. Quasi-globally optimal and efficient vanishing point estimation in manhattan world. In *ICCV*, pages 1646–1654, 2019. 2, 6, 7

[23] Xiang Li, Jun Li, Xiaolin Hu, and Jian Yang. Line-cnn: End-to-end traffic line detection with line proposal unit. *IEEE Transactions on Intelligent Transportation Systems*, 21(1):248–258, 2019. 2

[24] Xiaohu Lu, Jian Yaoy, Haoang Li, Yahui Liu, and Xiaofeng Zhang. 2-line exhaustive searching for real-time vanishing point estimation in manhattan world. In *WACV*, pages 345–353. IEEE, 2017. 2, 6, 7

[25] Evelyne Lutton, Henri Maitre, and Jaime Lopez-Krahe. Contribution to the determination of vanishing points using hough transform. *IEEE transactions on pattern analysis and machine intelligence*, 16(4):430–438, 1994. 2

[26] Michael J Magee and Jake K Aggarwal. Determining vanishing points from perspective images. *Computer Vision, Graphics, and Image Processing*, 26(2):256–267, 1984. 2

[27] Faraz M Mirzaei and Stergios I Roumeliotis. Optimal estimation of vanishing points in a manhattan world. In *ICCV*, pages 2454–2461. IEEE, 2011. 2

[28] Vladimir Nekrasov, Thanuja Dharmasiri, Andrew Spek, Tom Drummond, Chunhua Shen, and Ian Reid. Real-time joint semantic segmentation and depth estimation using asymmetric annotations. In *ICRA*, pages 7101–7107. IEEE, 2019. 8

[29] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *ICML*, pages 4055–4064. PMLR, 2018. 3

[30] Yongjie Shi, Danfeng Zhang, Jingsi Wen, Xin Tong, He Zhao, Xianghua Ying, and Hongbin Zha. Three orthogonal vanishing points estimation in structured scenes using convolutional neural networks. In *ICIP*, pages 3537–3541. IEEE, 2019. 2

[31] Gilles Simon, Antoine Fond, and Marie-Odile Berger. A-contrario horizon-first vanishing point detection using second-order grouping laws. In *ECCV*, pages 318–333, 2018. 6, 7

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

[33] Jean-Philippe Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *ICCV*, pages 1250–1257. IEEE, 2009. 2, 6, 7

[34] Alexander Vakhitov and Victor Lempitsky. Learnable line segment descriptor for visual slam. *IEEE Access*, 7:39923–39934, 2019. 3, 4

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 3

[36] Rafael Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. Lsd: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):722–732, 2008. 2, 6

[37] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, pages 1571–1580, 2021. 3

[38] Jianping Wu, Liang Zhang, Ye Liu, and Ke Chen. Real-time vanishing point detector integrating under-parameterized ransac and hough transform. In *ICCV*, pages 3732–3741, 2021. 2, 6, 7

[39] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *CVPR*, pages 5791–5800, 2020. 3

[40] Yanhong Zeng, Jianlong Fu, and Hongyang Chao. Learning joint spatial-temporal transformations for video inpainting. In *ECCV*, pages 528–543. Springer, 2020. 3

[41] Menghua Zhai, Scott Workman, and Nathan Jacobs. Detecting vanishing points using global image context in a non-manhattan world. In *CVPR*, pages 5657–5665, 2016. 2, 5

[42] Xiaodan Zhang, Xinbo Gao, Wen Lu, Lihuo He, and Qi Liu. Dominant vanishing point detection in the wild with application in composition analysis. *Neurocomputing*, 311:260–269, 2018. 2

[43] Yichao Zhou, Haozhi Qi, Jingwei Huang, and Yi Ma. Neurvps: Neural vanishing point scanning via conic convolution. *NIPS*, 32, 2019. 2, 6, 7

[44] Yichao Zhou, Haozhi Qi, Yuexiang Zhai, Qi Sun, Zhili Chen, Li-Yi Wei, and Yi Ma. Learning to reconstruct 3d manhattan wireframes from a single image. In *ICCV*, pages 7698–7707, 2019. 5, 6, 7, 8