ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240481

SinLane: Siamese Visual Transformer via Pyramid Feature Integration for Lane Detection

Zinan Lv[a], Dong Han[b,*], Wenzhe Wang[b,] and Danny Z. Chen[c]**

[a]Shanghai Jiao Tong University, Shanghai, China
[b]Zhejiang University, Hangzhou, China
[c]University of Notre Dame, Notre Dame, USA

Abstract. Lane detection is an important yet challenging task in autonomous driving systems. Based on the development of the Visual Transformer, early Transformer-based lane detection studies have achieved promising results in some scenarios. However, for complex road conditions such as uneven illumination intensity and heavy traffic, the performance of these methods remains limited and may even be worse than that of contemporaneous CNN-based methods. In this paper, we propose a novel Transformer-based end-to-end network, called SinLane, that attains the attention weights focusing on the sparse yet meaningful locations and improves the accuracy of lane detection in complex environments. SinLane is composed of a novel Siamese Visual Transformer structure and a novel Feature Pyramid Network (FPN) structure called Pyramid Feature Integration (PFI). We utilize the proposed PFI to better integrate global semantics and finer-scale features and to promote the optimization of the Transformer. Moreover, the designed Siamese Visual Transformer is combined with multiple levels of the PFI and is employed to refine the multi-scale lane line features output from the PFI. Extensive experiments on three benchmark datasets of lane detection demonstrate that our SinLane achieves state-of-the-art results with high accuracy and efficiency. Specifically, our SinLane improves the accuracy by over 3% compared to the current best-performing Transformer-based method for lane detection on CULane.

1 Introduction

Lane detection is a fundamental task for autonomous driving systems, which helps vehicles allocate themselves and plan the driving routes in a real-time state. In recent years, vision-based lane detection methods have achieved remarkable progress.

Early lane detection studies relied on hand-crafted features to extract information (e.g., color and texture), and applied methods like Hough transform [17] and Kalman filter [33] to filter out unreal or noncontinuous lanes. Afterward, convolutional neural network (CNN) based methods were introduced to the lane detection task, greatly improving the accuracy of prediction. These methods either generate segmentation results or regress curves to attain prediction. However, such methods may be ineffective and incapable of clustering global semantics information and struggle with complex environments, especially when involving strong light, shadows, and dense traffic. To deal with these issues, some works introduced soft attention-based methods [18] to extract more context information. However, such methods focus only on the spatial weight of features and neglect the dependencies among features. Recently, LSTR [19] employed a Transformer-based architecture, which greatly improved the speed of prediction, proving that the self-attention mechanism is quite suitable for the lane detection task.

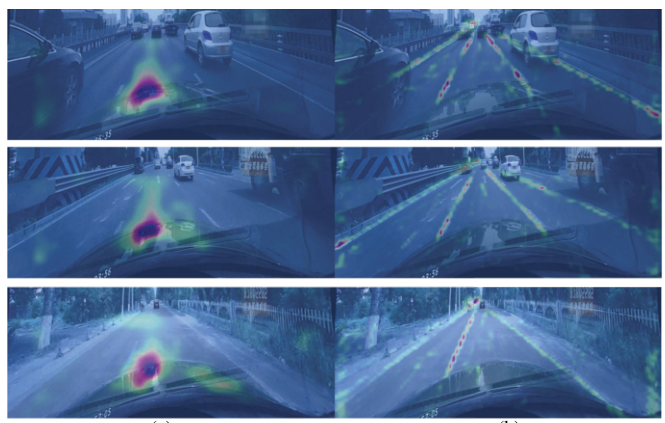
(a) (b)

Figure 1: Attention map examples of (a) LSTR [19] and (b) our proposed SinLane. The two models are both trained with the same number of epochs. The attention weights of LSTR concentrate on the middle area of the lane lines. On the contrary, the attention weights of our method are evenly distributed from top to bottom on each line on the road.

Although LSTR [19] gained promising results, some issues remain. As shown in Fig. 1, its attention weights designed for the slender structures are distributed unevenly in the map. This phenomenon causes the backbone to down-scale the input image to low-resolution feature maps when extracting features, which leads to a loss of low-level information. When the decoder of the Transformer calculates cross-attention of the detection sequence, the attention coefficient is required to be computed and optimized on the entire low-resolution feature maps, giving rise to difficulties for the model learning to focus on the sparse yet meaningful locations. This also results in taking a large number of training epochs to attain the relatively 'proper' attention weights.

Recently, Feature Pyramid Network (FPN) based methods [16], fusing multi-scale features through top-down and lateral connectivity, have achieved great success on several computer vision tasks. Contemporaneous CNN-based methods (e.g., [38]) often adopt FPN

* Corresponding Author. Email: phhandong@outlook.com
** Corresponding Author. Email: wangwnezhe@zju.edu.cn

structures to obtain multi-scale information. However, according to previous studies [11], only semantic information from the high-level feature layer in FPN is considered and passed to the low-level feature layer, resulting in neglecting the balance of features in multi-levels. Low-level features that are informative in image texture and semantics are not fully utilized. By and large, effective fusion of low-level features needs to be further explored.

In this paper, we propose a novel Transformer-based end-to-end network called SinLane to improve the accuracy of lane detection in complex environments. SinLane is composed of a novel Siamese Visual Transformer structure and a novel FPN structure called Pyramid Feature Integration (PFI). For effective hierarchical information extraction and fusion, inspired by the fusion factors in [11], our PFI is designed to balance and integrate hierarchical features extracted from different levels. The PFI promotes our Siamese Visual Transformer to optimize and regress lane line sequences based on global semantics and finer-scale features. Besides, we develop a novel encoder-decoder architecture for the Siamese Visual Transformer. The output sequence of the Transformer is refined by features from each level of the PFI to fuse high- and low-scale information.

We conduct experiments on three benchmark datasets (CULane [25], Tusimple [30], and LLAMAS [1]), which show that our model achieves state-of-the-art results on all the three datasets. Specifically, it yields over 3% improvement in accuracy compared to the known best-performing Transformer method [19] for lane detection on CULane.

Our main contributions are summarized as follows:

- We propose a novel FPN module, Pyramid Feature Integration (PFI), to fully integrate global semantics and finer-scale features.
- We design a Siamese Visual Transformer to refine multi-scale lane line features from the PFI.
- We achieve state-of-the-art results on three benchmark datasets, with over 3% improvement in accuracy compared to best-known Transformer methods on CULane.

2 Related Works

Early lane detection studies relied on hand-crafted features [6, 23], resulting in limited feature capturing, and thus were not effective for lane detection tasks in complex conditions.

To cope with complex environments, deep learning (DL) methods were introduced to the lane detection task. Segmentation-based methods [9, 13, 35, 36] were first applied to lane detection, whose detection outputs were based on per-pixel segmentation maps. Compared to traditional methods, CNN-based methods make it possible to capture more plentiful visual features and spatial structure information, and thus DL-based methods outperform traditional detection methods. However, per-pixel-based segmentation methods incur high computational costs have limited real-time capability, and also struggle with learning the long and thin characteristics of lane lines.

To address these issues, LaneNet [24] introduced a branched, multi-task architecture to cast the lane detection task as an instance segmentation problem. This method is more robust to variations in road conditions compared to the previous methods, but it is more time-consuming. RESA [37] was proposed to aggregate spatial information by shifting sliced feature maps, which obtains good real-time results but still fails under complex road conditions. Furthermore, the output lane lines of most of the above methods may not be continuous.

To attain more continuous lane lines with higher efficiency, in recent studies, curve-based methods [19, 28, 8, 4] viewed the lane detection task as a polynomial regression problem, and utilized parametric curves to fit lane lines. These methods depend heavily on the parameters of the curves (e.g., $x = ay^3 + by^2 + cy + d$, where (x, y) denotes the coordinates of a lane line pixel and a, b, c, and d are the parameters of a curve). PloyLaneNet [28] first proposed an end-to-end deep polynomial regression method that directly outputs parameters. To improve the stability and efficiency, BézierLaneNet [8] proposed a parametric Bézier curve to model the geometric shape of lane lines. However, even if with high efficiency, limited by the learning ability of global information, the accuracy of these curve-based methods is not satisfactory on large datasets, especially in complex road conditions.

After Transformer was introduced to the computer vision field [7], it has achieved impressive results in model inference speed as well as the acquisition of global information. DETR [3] obtained satisfactory results in object detection, which were better than some CNN-based methods. But, in the area of lane detection, Transformer-based methods [19, 27] still struggle to produce satisfactory results. The DETR-based method LSTR [19] is fast in inference but relatively low in accuracy, especially in some complex road environments. PriorLane [27] improved the accuracy of prediction compared to LSTR with pre-training and local prior. However, there is still a gap in accuracy between the contemporaneous Transformer-based methods and CNN-based methods.

In the detection tasks, the low-level layers are rich in geometric information but lack abstract semantic information, but deep layers do the opposite. For lane detection tasks, the unique long and thin shape of the lane lines and complex driving scenes pose high demand for the integration of local and global information. FPN [16] proposes a top-down feature pyramid architecture to merge low and high-level features. A bottom-up architecture is proposed in PANet [20] for better aggregation from low-level to high-level features. Kong [15] reformulates the FPN structure and applies global attention and local reconfiguration to fuse low-level representations with high-level semantic features. Cao [2] proposes a multi-branch and high-level semantic network to bridge the gap between multi-scale feature maps. Nas-FPN [10] and BiFPN [29] propose learnable fusion strategies which improve the effect of feature fusion from multi-scale. However, all these methods ignore the scale distribution of datasets and fail to fuse global and local information in complex self-driving scenes.

3 Method

In this section, we present the overall network architecture, the structures of Pyramid Feature Integration and Siamese Visual Transformer, and the training and inference details.

3.1 Architecture Design

The overall architecture of our SinLane network is shown in Fig. 2. Since a lane line has distinct structural characteristics, it can be represented by a series of key points sampled at equal distance along the y-axis, which can be expressed as:

$$l_i = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}. \quad (1)$$

Though the lane detection task can be viewed as a segmentation task, by means of key point representations of lane lines, it can be converted to a sequence prediction task, which has similarities to the

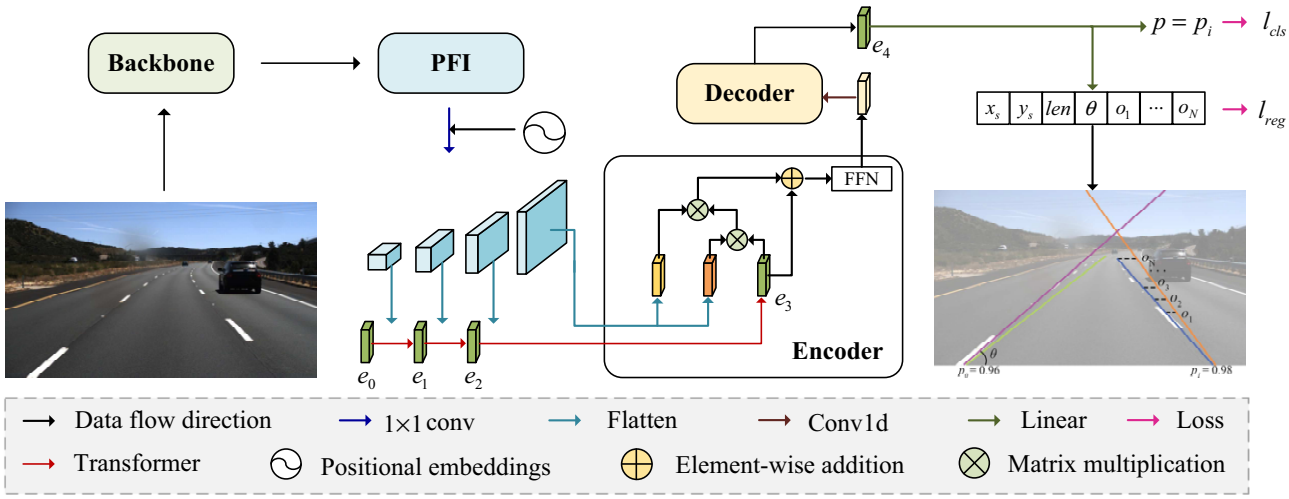

Figure 2: The overall architecture of our proposed SinLane network. The backbone first extracts multi-scale features from the input image. PFI is then applied to fully integrate global semantic information and local finer-scale features. Subsequently, Siamese Visual Transformer (Encoder and Decoder) generates lane sequences. Specifically, e_0 is the initial lane sequence, and e_1, e_2, and e_3 denote refined lane sequences optimized by different scales of feature maps from PFI.

task of object detection. Inspired by DETR [3], we propose an end-to-end Transformer-based method, SinLane, to generate lane predictions without complex post-processing steps such as Non-maximum Suppression (NMS) [31]. The main structure of our network can be divided into four parts, backbone (ResNet or DLA34), neck, head, and training objective, as follows:

- **Backbone**. Commonly used backbones such as ResNet, DLA34, and VGG can be adapted by our method.
- **Neck**. We propose PFI as the neck of our network to further integrate global semantic information and finer-scale features, allowing feature maps as input to the Transformer to incorporate more informative features, thus promoting the optimization of the Transformer. The output of the neck is top-to-bottom multi-scale features.
- **Head**. The head of our network is a Siamese Visual Transformer structure. It contains four weight-sharing Transformers. The input of each Transformer is composed of a sequence output from the former Transformer and specific-scale feature maps from PFI after flattening. The output of each Transformer is a fixed number of lane line sequences. Global features of different scales are learned by Siamese visual encoders and decoders.
- **Training objective**. The objective of the model can be divided into two parts: (1) classification and (2) regression. We supervise the generation of the fixed number of lane lines with the ground-truth lane lines based on attendance and their properties, respectively.

3.2 Pyramid Feature Integration

We develop PFI to integrate global semantic information and finer-scale features. Fig. 3 shows the detailed structure of PFI.

3.2.1 Fusion Factor

The traditional FPN structure [16] generates multi-scale features from high-level to low-level; but, it neglects the integration of features in different scales. According to the study in [11], the performance of FPN can be affected by two types of factors, (1) the down-sampling factor and (2) the fusion factors, between adjacent feature levels. Commonly, the larger the down-sampling factor, the better the performance of FPN, and the slower the network is. Due to the trade-off between time efficiency and accuracy, we use a suitable and fixed value as the proportion of down-sampling. The main focus of this section is on the fusion factors.

Fusion factors are the scale factors ($\alpha \in \{\alpha_0, \alpha_1, \alpha_2\}$ in Fig. 3) between the semantic features of adjacent levels, which can be expressed as:

$$P_k = f_{conv_1}\left(f_{conv_3}(P_k^{'}) + \alpha_{k-1} \times f_{inter}(P_{k-1})\right), \quad (2)$$

where $k \in \{1, 2, 3\}$, α_{k-1} is the fusion factor determined by the input features, P_k is the k-th scale output features integrated by the fusion factor, $P_k^{'}$ is the k-th scale output features from the backbone, $f_{conv_1}(\cdot)$ denotes 1×1 convolution, $f_{conv_3}(\cdot)$ represents 3×3 convolution, and $f_{inter}(\cdot)$ is $2 \times$ up-sampling.

To obtain suitable fusion factors for the input feature maps, we utilize a lightweight convolution module to learn the α_i's, which can be expressed as:

$$\alpha_{k-1} = Sigmoid(GAP(f_{conv_3}(P_k^{'}))), \quad (3)$$

where $GAP(\cdot)$ denotes global average pooling.

3.2.2 Integration Structure

After using the fusion factors to balance the features of adjacent high-level and low-level, we adopt an integration structure to further fuse them. For FPN-based information integration, there are some known methods such as PANet [20] and NAS-FPN [10]. When applied to the lane detection task, because lane lines in the images are usually long and thin, balancing global information and finer-scale features is relatively difficult for these methods.

Thus, we develop a new integration structure to integrate global semantic information and finer-scale features. First, we reshape the multi-scale features (which have been balanced by the fusion factors) to the same scale. Note that the feature shape is an adjustable parameter depending on the balance of accuracy and efficiency. Next, we

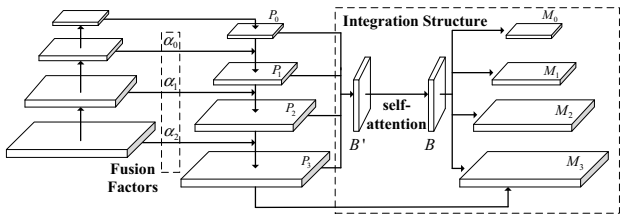

Figure 3: The architecture of our proposed PFI. The inputs of PFI are different scales of feature maps generated by the backbone.

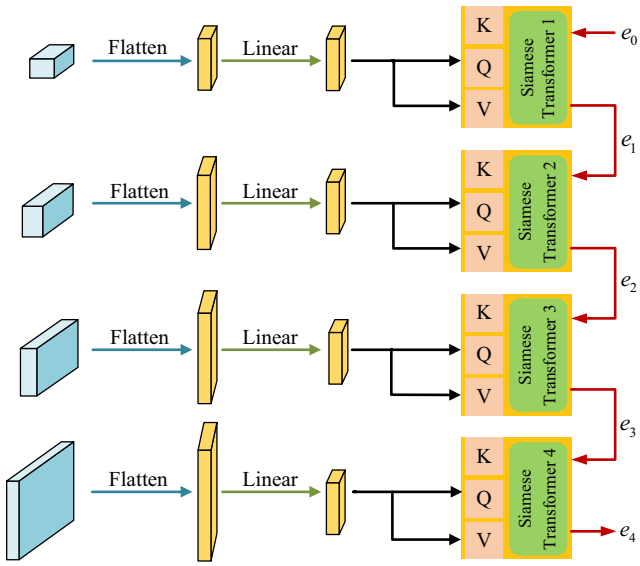

Figure 4: The main structure of the Siamese Visual Transformer.

average the reshaped features, which can be expressed as:

$$B^{'} = \frac{1}{l_{num}} \sum_{k=0}^{l_{num}-1} (Re(P_k)), \tag{4}$$

where $B^{'}$ denotes the output of the integration, l_{num} represents the number of scales that need to be integrated, and $Re(\cdot)$ denotes the reshape function.

To ensure that our PFI is a plug-and-play module and prevent the loss of multi-scale information from the backbone, we need to restore $B^{'}$ to multi-scale feature maps. We employ self-attention to better extract feature information from $B^{'}$, and then down- and up-sample the output B to $M_0, \ldots, M_3$, which share the same sizes as $P_0, \ldots, P_3$, respectively (see Fig. 3). Our PFI adopts the non-local method in [32] as the default attention module. Some other methods (e.g., HANet [5]) can also be supported here, but their lane detection results might not be as good. The whole process of B' to $M_0 - M_3$ can be expressed as:

$$M_i = f_{inter}(f_{sa}(B')) + P_i, \tag{5}$$

where $f_{inter}(\cdot)$ denotes the interpolation operation and $f_{sa}(\cdot)$ is a self-attention module.

Compared to previous works like PANet [20], our PFI exhibits two advantages:

(1) It is lightweight and plug-and-play, which can be easily implemented by parallel computation and applied to other tasks and networks;

(2) it is suitable for tasks that heavily rely on the fusion of global and local information, such as the lane detection task.

3.3 Siamese Visual Transformer

We propose the Siamese Visual Transformer to extract rich information from multi-scale feature maps. Fig. 4 shows the detailed Siamese Visual Transformer structure. The main structure is made up of four Siamese Visual Transformers with shared parameters [14].

3.3.1 Object Sequences

Because lane lines are thin, long, and have distinctive structural features, we use object sequences ($e_0, \ldots, e_3$ in Fig. 2) to represent them, which can help reduce computational costs and is easy to optimize for the Transformer. An object sequence can be expressed as:

$$e_i = \{x_s, y_s, len, \theta, o_1, o_2, \ldots, o_N\}_{p=p_i}, \tag{6}$$

where e_i is the i-th sequence of an input image, (x_s, y_s) denotes the starting point of the lane line (the point at the bottom of the lane line), len is the length of the lane line, θ denotes the starting angle of the lane line, p_i represents the probability that the i-th sequence is a lane line, and the j-th element in $\{o_j\}_{j=1}^{N}$ represents the offset of the j-th key point on the lane line. The initial object sequence e_0 is generated by embedding and is sequentially refined by multi-scale feature maps output by PFI through the Transformer.

By supervising the generation of the object sequences using the ground truth, we can force the output sequences to correspond to the actual lanes in the image one by one, so as to realize the detection of the lane lines.

3.3.2 Transformer Encoder

We divide the input feature maps from PFI into patches to reduce the burden of the Transformer calculation. The 3-D patches are then flattened to 2-D sequences with positional embeddings added. Different from the traditional Transformer encoder, we utilize a Siamese structure to enable the Transformer to learn richer multi-scale information with shared parameters. Specifically, we apply attention between the sequences from the input feature maps and those from the output of the upper-level Transformer (or the pre-generated sequence e_0) instead of the original self-attention. This process can be expressed as:

$$e_i = Softmax\left(\frac{e_{i-1}m_i^T}{\sqrt{d_i}}\right) m_i + e_{i-1}, \tag{7}$$

where m_i is the i-th flattened sequence with positional embeddings and d_i is the length of the sequence.

After the attention mechanism, a feed-forward network is used to obtain features that are nonlinear.

3.3.3 Transformer Decoder

We take the traditional Transformer decoder as our Transformer decoder, which utilizes multi-head self-attention mechanisms. The feed-forward network is made up of two fully connected layers: One is for generating the probability that the sequence is a lane line, and the other is for computing structural information about the lane line (see Eq. (6)).

Dataset	Train	Val	Test	Scene
CULane	88880	9675	34680	Urban & Highway
Tusimple	3268	358	2782	Highway
LLAMAS	58269	20844	20929	Highway

Table 1: Detailed description of the three datasets we use.

3.4 Training and Inference Details

3.4.1 Training Objective

The training objective can be divided into two parts: (1) the accuracy of the classification, and (2) the overlap of the prediction and ground truth. The objective of the classification accuracy can be described as:

$$\ell_{cls} = \sum_{i=1}^{L} \mathcal{L}\left(p_i, g_i\right), \tag{8}$$

where $\mathcal{L}(\cdot,\cdot)$ denotes the cross-entropy loss and L is the number of sequences. If the predicted lane line matches the ground truth, then g_i = 1 (0 otherwise).

The degree of overlap between the predicted lane line and ground truth can be evaluated in two aspects: the Euclidean distance between their starting points and the horizontal distance between their corresponding key points. The overlap objective can be written as follows:

$$\ell_{reg} = \sum_{i=1}^{L} \left(\left\| \hat{S}_i - S_i \right\|_2 + \frac{\lambda_{dis}}{N} \sum_{j=1}^{N} |\hat{y}_{ij} - y_{ij}| \right), \tag{9}$$

where $\hat{S}_i$ is the starting point of the predicted lane line, S_i is the starting point of the ground truth, $\|\cdot\|_2$ is L_2-norm, and λ_{dis} is an objective coefficient.

The training objective function is a weighted sum of the two objectives, which can be defined as:

$$\ell = \lambda_{cls}\ell_{cls} + \lambda_{reg}\ell_{reg}, \tag{10}$$

where λ_{cls} and λ_{reg} are two hyper-parameters for balancing the importance of the two objectives.

In particular, we compute the objective function ℓ after each decoder. The final objective function is the sum of l_{num} objective functions from each refined sequence ($e_0, \ldots, e_3$ in Fig. 2). In this way, more lane line information can be obtained for optimization.

3.4.2 Inference Details

Our method generates a fixed number of sequences corresponding to lane lines in the input image. Our method is an NMS-free method, and we use a one-to-one assignment during the inference stage.

4 Experiments

4.1 Datasets

To evaluate the effectiveness of our proposed SinLane method, we conduct experiments on three well-known public datasets: CULane [25], Tusimple [30], and LLAMAS [1]. CULane is a widely-used large dataset for lane detection including eight hard-to-detect scenarios in urban areas and on highways. Tusimple is also a widely-used dataset with images collected on US highways under clear weather. LLAMAS is a recently released dataset captured on highways, with unsupervised labels generated by high-definition maps. Details of these datasets are given in Table 1.

4.2 Evaluation Metrics

For CULane [25] and LLAMAS [1], we adopt the F1-measure proposed by SCNN [25] as the evaluation metric. Intersection-over-Union (IoU) between the ground truth (GT) label and the predicted lane line of the model is calculated to determine whether a sample is True Positive (TP), False Positive (FP), or False Negative (FN). The IoU and F1-measure are calculated as in the following formulas:

$$Precision = \frac{TP}{TP+FP}, \quad Recall = \frac{TP}{TP+FN}, \tag{11}$$

$$IoU = \frac{Intersection}{Union}, \tag{12}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{13}$$

For Tusimple [30], the evaluation metrics are composed of three official indicators: accuracy, False Positive Rate (FPR), and False Negative Rate (FNR). The accuracy is calculated as:

$$Accuracy = \frac{\sum_{clip} C_{clip}}{\sum_{clip} S_{clip}}, \tag{14}$$

where C_{clip} is the number of correct points and S_{clip} is the number of ground truth (GT) points in an input image. If the accuracy of a predicted lane is greater than 85%, it will be considered a True Positive (TP). The F1 score is also used in the evaluation.

4.3 Implementation Details

In the experiments, we adopt ResNet and DLA34 as the backbones for our model. For all the proposed models, the number of multi-scale layers is set to 4, and the number l_{num} of scales that need to be integrated is set to 4. For the object sequences, the number N of key points is set to 72, and the number of lane lines is set to 192. In terms of training objective calculation, the objective coefficients λ_{dis}, λ_{cls}, and λ_{reg} are set to 0.5, 2.0, and 1.0, respectively. For data augmentation, we adopt the affine transformation method (horizontal flip, rotation), and brightness and saturation addition method. All input images are resized to 800×320 pixels for both the training and testing stages. In the optimizing process, we adopt AdamW [22] and the cosine decay learning rate strategy [21] with the initial learning rate set to 6e-4. All the experiments are conducted on a machine with an NVIDIA RTX3090 GPU with 24GB memory.

4.4 Results

Results on CULane. We compare the results of recent known methods and our method on the CULane dataset, shown in Table 2. In terms of the total F1 score, our method achieves state-of-the-art results and yields big improvement compared to previous Transformer-based methods (see the middle section of Table 2). Among all the difficult scenarios, our "Hlight" result is over 8% higher than that of the best-known Transformer-based method, demonstrating that our proposed network can adapt well to complex environments. Compared with CLRNet, our SinLane performs better under complex road conditions such as "Hlight" and "Night".

Fig. 5 shows some visual results of several known methods and our proposed method. LSTR is an open-source Transformer-based method and CLRNet is the second-best-performing method on CULane. When road signs are missing or unclear, LSTR and CLRNet

Method	Backbone	Total	Normal	Crowded	Hlight	Shadow	Noline	Arrow	Curve	Cross	Night	FPS(↑)	GFlops(↓)
SCNN [25]	VGG16	71.60	90.60	69.70	58.50	66.90	43.40	84.10	64.40	1990	66.10	7.5	328.4
UFLD [26]	ResNet18	68.40	87.70	66.00	58.40	62.80	40.20	81.00	57.90	1743	62.10	**341**	8.4
UFLD [26]	ResNet34	72.30	90.70	70.20	59.50	69.30	44.40	85.70	69.50	2037	66.70	184	-
RESA [37]	ResNet34	74.50	91.90	72.40	66.50	72.00	46.30	88.10	68.60	1896	69.80	51	-
RESA [37]	ResNet50	75.30	92.10	73.10	69.20	72.80	47.70	88.30	70.30	1503	69.90	39	-
ADNet [34]	ResNet18	77.56	91.92	75.81	69.39	76.21	51.75	87.71	68.84	1133	72.33	87	-
ADNet [34]	ResNet34	78.94	92.90	77.45	71.71	79.11	52.89	89.90	70.64	1499	74.78	77	-
CondLane [18]	ResNet18	78.14	92.87	75.79	70.72	80.01	52.39	89.37	72.40	1364	73.23	201	10.2
CondLane [18]	ResNet101	79.48	93.47	77.44	70.93	80.91	54.13	90.16	75.21	1201	74.80	56	44.8
CLRNet [38]	ResNet18	79.58	93.30	78.33	73.71	79.66	53.14	90.25	71.56	1321	75.11	203	11.9
CLRNet [38]	ResNet101	80.13	93.85	78.78	72.49	82.33	54.48	89.79	75.57	1262	75.51	115	42.9
CLRNet [38]	DLA34	80.47	93.73	**79.59**	75.30	**82.51**	**54.58**	**90.62**	74.13	1155	75.37	131	18.5
LSTR [19]	ResNet18	68.72	86.78	67.34	56.63	59.82	40.10	78.66	56.64	1166	59.92	126	**2.9**
Laneformer [12]	ResNet18	71.71	88.60	69.02	64.07	65.02	45.00	81.55	60.46	25	64.76	-	-
Laneformer [12]	ResNet34	74.70	90.74	72.31	69.12	71.57	47.37	85.07	65.90	26	67.77	-	-
Laneformer [12]	ResNet50	77.06	91.77	75.41	70.17	75.75	48.73	87.65	66.33	19	71.04	-	-
PriorLane [27]	ResNet18	76.27	92.36	73.86	68.26	78.13	49.60	88.59	73.94	2688	70.26	-	-
SinLane	ResNet18	79.90	93.48	78.31	**75.96**	81.02	53.44	90.56	73.00	**1084**	75.41	225	14.2
SinLane	ResNet34	80.13	**93.98**	78.72	72.65	81.76	53.94	90.10	75.94	1185	75.41	173	23.6
SinLane	DLA34	**80.68**	93.90	79.45	75.89	81.59	54.19	90.51	**76.01**	1143	**75.98**	160	20.8

Table 2: Comparison results of recent methods and our method on the CULane dataset. In order to compare the computation speeds in the same environment, we remeasure FPS on the same machine with an RTX3090 GPU using open-source code (if code is available).

Method	Backbone	F1	Acc	FP	FN
SCNN	VGG16	95.97	96.53	6.17	**1.80**
RESA	ResNet34	96.93	96.82	3.63	2.48
UFLD	ResNet18	87.87	95.82	19.05	3.92
UFLD	ResNet34	88.02	95.86	18.91	3.75
PolyLaneNet	EfficientNetB0	90.62	93.36	9.42	9.33
LaneATT	ResNet18	96.71	95.57	3.56	3.01
LaneATT	ResNet122	96.06	96.10	5.64	2.17
ADNet	ResNet18	96.90	96.23	2.91	3.29
ADNet	ResNet34	97.31	96.60	2.83	2.53
CLRNet	ResNet18	97.41	96.84	2.28	1.92
CLRNet	ResNet101	97.68	96.83	2.37	2.38
LSTR	ResNet18	96.84	96.18	2.91	3.38
Laneformer	ResNet18	97.30	96.54	4.35	2.36
Laneformer	ResNet34	97.41	96.56	5.39	3.37
Laneformer	ResNet50	97.56	96.80	5.60	1.99
PriorLane	ResNet18	97.15	96.58	3.91	2.95
SinLane	ResNet18	**97.93**	96.81	2.03	2.11
SinLane	ResNet34	97.74	96.81	2.18	2.34
SinLane	ResNet101	97.92	96.77	**1.76**	2.40
SinLane	DLA34	97.55	**96.94**	3.03	1.85

Table 3: Comparison results on the Tusimple dataset.

Method	Backbone	F1@50	F1@75	GFlops
LaneATT	ResNet18	94.64	82.36	**9.3**
LaneATT	ResNet34	94.96	82.79	18.0
LaneATT	ResNet122	95.17	84.01	70.5
LaneAF	DLA34	96.90	84.71	23.6
CLRNet	ResNet18	96.96	85.59	11.9
CLRNet	DLA34	**97.16**	85.33	18.5
SinLane	ResNet18	96.75	84.57	14.0
SinLane	DLA34	97.07	**85.80**	20.7

Table 4: Comparison results on the LLAMAS dataset.

both struggle to capture lane lines. When the lane lines are blocked by vehicles or pedestrians, CLRNet is unable to recognize lane lines. Furthermore, these two methods are sensitive to light. Our method performs stably both in complex road conditions and at night, showing the robustness of our model.

Results on Tusimple. Comparison results of recent methods and our method on the Tusimple dataset are given in Table 3. The dataset consists of images captured in different weather conditions. Our method achieves state-of-the-art results in F1 score, Accuracy, and False Negative Rate (FN), demonstrating that our method can be adapted to both complex urban environments and simple highway scenarios.

PFI	Encoder	Decoder	F1 score
			68.40
✓			75.63 (+7.23)
✓	✓		78.94 (+10.54)
✓		✓	77.78 (+9.38)
	✓	✓	78.99 (+10.59)
✓	✓	✓	**79.90** (+11.50)

Table 5: Results of the overall ablation study on the CULane dataset with the same backbone ResNet18.

Note that because the dataset was collected on US highways where the road conditions are relatively simple (lane features are obvious and clear), the results of various methods are close.

Results on LLAMAS. Comparison results of recent methods and our method on the LLAMAS dataset are shown in Table 4. Our SinLane performs well and achieves state-of-the-art results in F1@75 score, and it achieves the second-best in F1@50 score. Due to the simplicity of the LLAMAS dataset (on highways), the results of various methods are close.

4.5 Ablation Study

To examine the effects of different modules in our method, we conduct ablation experiments.

4.5.1 Overall Ablation Study

To verify the role of each module in our proposed method, we carry out overall ablation experiments with the same baseline UFLD [26]. The results of the experiments are shown in Table 5. Our proposed PFI can greatly improve the accuracy of the detection of lane lines, which improves the F1 score from 68.40% to 75.63% with a 7.23% increase. Moreover, the Encoder module and the Decoder module further improve the F1 score by 3.31% and 2.15%, respectively. The combination of the Encoder and Decoder further improves the detection results by 0.96%, which achieves state-of-the-art results on CULane with ResNet18 as the backbone.

4.5.2 Ablation Study on Pyramid Feature Integration

Compared to the traditional FPN, our PFI can balance finer-scale features and global semantics for better prediction of lane lines.

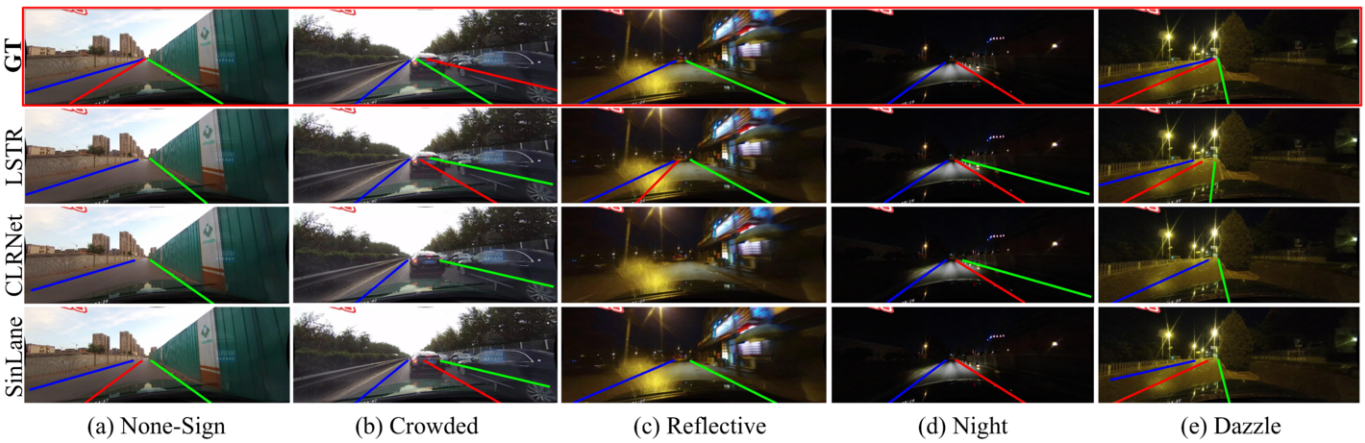

Figure 5: Visualization results of ground truth (GT), LSTR [19], CLRNet [38], and our SinLane method on the benchmark dataset CULane [25]. The results are generated using the same backbone ResNet18.

As a plug-to-play module, it can be applied to other networks easily. As shown in Table 6, we add our PFI to ConaLaneNet and CLRNet, which are curve-based methods. When PFI is added to ConaLaneNet and CLRNet, the total F1 scores increase considerably, which demonstrates the effectiveness of our proposed PFI module. In particular, the F1 scores of "Hlight", "Shadow", "Curve", and "Night" show great improvements, proving that our PFI can combine finer-scale features and global semantics to detect lanes in a better manner under complex scenarios.

Method	Neck	Total	Normal	Crowded	Hlight	Shadow
CondLane	FPN	78.07	92.31	76.7	71.09	76.62
	PFI	78.36	92.74	76.81	71.69	78.54
		(+0.29)	(+0.43)	(+0.11)	(+0.60)	(+1.92)
CLRNet	FPN	79.36	93.18	77.69	73.62	80.73
	PFI	79.66	93.45	**78.50**	74.29	**81.42**
		(+0.30)	(+0.27)	(+0.81)	(+0.67)	(+0.69)
SinLane (ours)	FPN	79.56	93.44	78.09	73.5	78.49
	PFI	**79.90**	**93.48**	78.31	**75.96**	81.02
		(+0.34)	(+0.04)	(+0.22)	(+2.46)	(+2.71)
Method	Neck	Noline	Arrow	Curve	Cross	Night
CondLane	FPN	51.14	88.67	67.95	1137	72.82
	PFI	52.04	89.08	70.15	1303	73.01
		(+0.90)	(+0.41)	(+2.2)	(+166)	(+0.19)
CLRNet	FPN	52.22	90.18	67.87	1101	74.93
	PFI	52.95	90.3	69.81	1263	**75.54**
		(+0.73)	(+0.12)	(+1.94)	(+162)	(+0.61)
SinLane (ours)	FPN	52.83	90.17	69.32	**1033**	74.82
	PFI	**53.44**	**90.56**	**73.00**	1084	75.41
		(+0.61)	(+0.39)	(+3.68)	(+51)	(+0.59)

Table 6: Results of the ablation study on PFI. The configurations of ConaLaneNet and CLRNet are the same as the original code provided by the authors with the same backbone ResNet18 except for the neck of the network.

4.5.3 *Ablation Study on Siamese Visual Transformer*

The main difference between our proposed network and the previous Transformer-based networks is that we propose Siamese Visual Transformer. To promote the optimization of Transformers, previous studies (e.g., Deformable DETR [39]) utilized multi-scale features. But, they only concatenated feature maps in different scales to integrate multi-scale features. In the process of concatenation, due to the different sizes of feature maps, some high-scale features can be lost, resulting in a loss of multi-scale features. Our proposed Siamese Visual Transformer structure can refine the output sequence from high-level to low-level, which can better utilize features in different scales with different sizes. We conduct ablation studies on CULane to evaluate the efficiency of Siamese Visual Transformer with the same backbone ResNet18. The results are shown in Table 7. Not using the Integration method means that a particular scale feature map output from PFI is directly fed into the Transformer. "3-D concat at PFI" denotes that multi-scale feature maps output from PFI are spliced into a tensor and then fed into the Transformer. "2-D concat before Encoder" denotes that the flattened multi-scale feature maps are spliced into a sequence and then perform self-attention in the process of the encoder. "2-D concat after Decoder" denotes that the flattened multi-scale feature maps use self-attention in the process of the encoder respectively and are spliced after the Transformer, which is similar to Deformable DETR [39]. As can be seen from Table 7, our method yields the best results among the above methods, demonstrating the effectiveness of our proposed Siamese Visual Transformer structure compared to those Transformer-based networks utilizing concatenation for feature integration.

PFI	Integration method	F1 score
-	-	72.53
✓	-	68.54
✓	3-D concat at PFI	73.38
✓	2-D concat before Encoder	74.68
✓	2-D concat after Decoder	75.01
✓	Siamese Visual Transformer	**79.90**

Table 7: Results of the ablation study on Siamese Visual Transformer on the CULane dataset with the same backbone ResNet18.

5 Conclusions

In this paper, we proposed a novel Transformer-based end-to-end network, called SinLane, for lane line detection. SinLane is composed of a novel Siamese Visual Transformer structure and a novel FPN structure called Pyramid Feature Integration (PFI). We showed that our proposed PFI can effectively integrate global semantics and finer-scale features and promote the optimization of the Transformer. Moreover, the designed Siamese Visual Transformer refines multi-scale lane line features output from our PFI. We evaluated our proposed method on three benchmark datasets, CULane, Tusimple, and LLAMAS. Experimental results demonstrated that our proposed SinLane achieved state-of-the-art results and improved the accuracy of lane line detection in complex environments. Specifically, it improved the accuracy by over 3% compared to the known best-performing Transformer-based method for lane line detection on the CULane dataset.

References

[1] K. Behrendt and R. Soussan. Unsupervised labeled lane markers using maps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[2] J. Cao, Y. Pang, S. Zhao, and X. Li. High-level semantic networks for multi-scale object detection. volume 30, pages 3372–3386, 2020. doi: 10.1109/TCSVT.2019.2950526.

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, et al. End-to-end object detection with Transformers. In *Computer Vision–ECCV 2020: 16th European Conference, 2020, Proceedings, Part I 16*, pages 213–229. Springer, 2020.

[4] H. Chen, M. Wang, and Y. Liu. BSNet: Lane detection via draw B-spline curves nearby. *arXiv preprint arXiv:2301.06910*, 2023.

[5] S. Choi, J. T. Kim, and J. Choo. Cars can't fly up in the sky: Improving urban-scene segmentation via height-driven attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9373–9383, 2020.

[6] P. Daigavane and P. Bajaj. Road lane detection with improved Canny edges using ant colony optimization. In *2010 3rd International Conference on Emerging Trends in Engineering and Technology*, pages 76–80, 2010. doi: 10.1109/ICETET.2010.128.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, Weissenborn, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Z. Feng, S. Guo, X. Tan, K. Xu, M. Wang, et al. Rethinking efficient lane detection via curve modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17062–17070, 2022.

[9] M. Ghafoorian, C. Nugteren, N. Baka, O. Booij, and M. Hofmann. EL-GAN: Embedding loss driven generative adversarial networks for lane detection. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[10] G. Ghiasi, T.-Y. Lin, and Q. V. Le. NAS-FPN: Learning scalable feature pyramid architecture for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7036–7045, 2019.

[11] Y. Gong, X. Yu, Y. Ding, X. Peng, J. Zhao, et al. Effective fusion factor in FPN for tiny object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1160–1168, 2021.

[12] J. Han, X. Deng, X. Cai, Z. Yang, H. Xu, C. Xu, and X. Liang. Laneformer: Object-aware row-column Transformers for lane detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 799–807, 2022.

[13] Y. Hou, Z. Ma, C. Liu, and C. C. Loy. Learning lightweight lane detection CNNs by self attention distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1013–1021, 2019.

[14] G. Koch, R. Zemel, R. Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2. Lille, 2015.

[15] T. Kong, F. Sun, C. Tan, H. Liu, and W. Huang. Deep feature pyramid reconfiguration for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[16] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, et al. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[17] G. Liu, F. Wörgötter, and I. Markelić. Combining statistical Hough transform and particle filter for robust lane detection and tracking. In *2010 IEEE Intelligent Vehicles Symposium*, pages 993–997. IEEE, 2010.

[18] L. Liu, X. Chen, S. Zhu, and P. Tan. CondLaneNet: A top-to-down lane detection framework based on conditional convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3773–3782, 2021.

[19] R. Liu, Z. Yuan, T. Liu, and Z. Xiong. End-to-end lane shape prediction with Transformers. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3694–3702, 2021.

[20] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018.

[21] I. Loshchilov and F. Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

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

[23] H. M. Mandalia and M. D. D. Salvucci. Using support vector machines for lane-change detection. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 49:1965 – 1969, 2005.

[24] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool. Towards end-to-end lane detection: An instance segmentation approach. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 286–291. IEEE, 2018.

[25] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang. Spatial as deep: Spatial CNN for traffic scene understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[26] Z. Qin, H. Wang, and X. Li. Ultra fast structure-aware deep lane detection. In *Computer Vision–ECCV 2020: 16th European Conference, 2020, Proceedings, Part XXIV 16*, pages 276–291. Springer, 2020.

[27] Q. Qiu, H. Gao, W. Hua, G. Huang, and X. He. PriorLane: A prior knowledge enhanced lane detection approach based on Transformer. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5618–5624. IEEE, 2023.

[28] L. Tabelini, R. Berriel, T. M. Paixao, C. Badue, A. F. De Souza, et al. PolyLaneNet: Lane estimation via deep polynomial regression. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6150–6156. IEEE, 2021.

[29] M. Tan, R. Pang, and Q. V. Le. Efficientdet: Scalable and efficient object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[30] Tusimple. TuSimple benchmark, 2020. https://github.com/TuSimple/tusimple-benchmark/, Accessed September, 2020.

[31] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001. doi: 10.1109/CVPR.2001.990517.

[32] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.

[33] P.-C. Wu, C.-Y. Chang, and C. H. Lin. Lane-mark extraction for automobiles under complex conditions. *Pattern Recognition*, 47(8):2756–2767, 2014.

[34] L. Xiao, X. Li, S. Yang, and W. Yang. ADNet: Lane shape prediction via anchor decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6404–6413, 2023.

[35] H. Xu, S. Wang, X. Cai, W. Zhang, X. Liang, et al. CurveLane-NAS: Unifying lane-sensitive architecture search and adaptive point blending. In *Computer Vision–ECCV 2020: 16th European Conference, 2020, Proceedings, Part XV 16*, pages 689–704. Springer, 2020.

[36] S. Yoo, H. S. Lee, H. Myeong, S. Yun, H. Park, J. Cho, et al. End-to-end lane marker detection via row-wise classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1006–1007, 2020.

[37] T. Zheng, H. Fang, Y. Zhang, W. Tang, Z. Yang, et al. RESA: Recurrent feature-shift aggregator for lane detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3547–3554, 2021.

[38] T. Zheng, Y. Huang, Y. Liu, W. Tang, Z. Yang, et al. CLRNet: Cross layer refinement network for lane detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 898–907, 2022.

[39] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable DETR: Deformable Transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.