



Lane line detection based on the codec structure of the attention mechanism

Qinghua Zhao¹ · Qi Peng¹ · Yiqi Zhuang¹

Received: 20 December 2021 / Accepted: 3 April 2022 / Published online: 9 May 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract

For self-driving cars and advanced driver assistance systems, lane detection is imperative. On the one hand, numerous current lane line detection algorithms perform dense pixel-by-pixel prediction followed by complex post-processing. On the other hand, as lane lines only account for a small part of the whole image, there are only very subtle and sparse signals, and information is lost during long-distance transmission. Therefore, it is difficult for an ordinary convolutional neural network to resolve challenging scenes, such as severe occlusion, congested roads, and poor lighting conditions. To address these issues, in this study, we propose an encoder–decoder architecture based on an attention mechanism. The encoder module is employed to initially extract the lane line features. We propose a spatial recurrent feature-shift aggregator module to further enrich the lane line features, which transmits information from four directions (up, down, left, and right). In addition, this module contains the spatial attention feature that focuses on useful information for lane line detection and reduces redundant computations. In particular, to reduce the occurrence of incorrect predictions and the need for post-processing, we add channel attention between the encoding and decoding. It processes encoding and decoding to obtain multidimensional attention information, respectively. Our method achieved novel results on two popular lane detection benchmarks (CULane F1-measure 76.2, TuSimple accuracy 96.85%), which can reach 48 frames per second and meet the real-time requirements of autonomous driving.

Keywords Lane detection · Autonomous driving · Channel attention · Spatial attention

1 Introduction

In modern society, cars have become the basic means of transportation for people. Therefore, as the number of cars increases and traffic conditions become increasingly complex, advanced driver assistance systems (ADAS) are becoming critically important. ADAS utilize a variety of sensors installed in the car and rely on the timely collection of environmental data from inside and outside the car to alert the driver of possible dangers. Lane detection is an important and challenging task for ADAS. With the help of lane line detection, drivers can better understand current road conditions and act in challenging scenarios, such as severe obstructions, extreme weather conditions, and tunnel access. Furthermore, lane line detection can help reduce the

occurrence of traffic accidents. Therefore, lane line detection plays an important role in autonomous driving technology.

Many studies have been conducted on lane marking [5, 9, 10, 15, 16, 23, 25, 28, 32, 35, 38, 43], most of which are based on two-stage semantic segmentation methods [14, 21, 32]. In the first stage, the image is classified pixel-by-pixel and each pixel is assigned a label, to specify whether it belongs to the lane line. However, this method has a fatal disadvantage—it does not consider the relationship between each pixel and requires a second stage of post-processing. For example, geometric constraints, such as structural or similarity constraints, are added to the lane lines, and the parameters need to be manually adjusted to match the first stage. Therefore, this method suffers from robustness and scalability problems in different environments and datasets. An alternative approach was provided by some researchers who tried to convey the spatial information via feature maps, such as spatial convolutional neural network (SCNN) [33] and recurrent feature-shift aggregator (RESA) [53]. SCNN replaces the conventional layer-by-layer convolutions with

✉ Qi Peng
qipeng08@gmail.com

¹ School of Microelectronics, Xidian University, Xi'an, China

slice-by-slice convolutions, which allows information to be transferred across rows and columns. However, because sequential information transfer between adjacent rows or columns requires several iterations and a relatively long inference time, it is difficult to achieve satisfaction with the real-time requirements of autonomous driving. RESA slices the feature maps vertically and horizontally to collect different sliced feature maps at different step sizes, enabling each pixel to collect global information. However, the lane lines do not extend to the entire image boundary; therefore, this method has redundant computations that affect the inference time. This method requires several iterations of the loop, which may lead to the loss of information during long propagation.

In our study, for the relationship between pixels, inference time, and information loss in the propagation process, we propose a codec structure based on the attention mechanism. First, because pixel-by-pixel classification fails to consider the relationship between each pixel and the real-time problem, we designed a spatial recurrent feature-shift aggregator (SPRESA) module with spatial attention to aggregate the adjacent pixels and focus on the spatial information near the lane lines. Finally, to reduce the probability of losing information during long-distance propagation, channel attention was added between the encoder and the decoder. The encoding and decoding modules extract multi-scale semantic information through the channel attention and perform feature fusion through a skip connection; thus, the decoding module contains rich spatial information and channel information.

Figure 1 shows the architecture of our lane line detection network, which contains three modules, namely, the encoder, SPRESA, and channel attention-guided decoding modules, and two branches are obtained: the segmentation and existence branches. Further details are presented in Sect. 3. The main contributions of our study can be summarized as follows:

We propose a novel lane line detection method based on a codec structure with an attention mechanism.

A SPRESA module is proposed to link the spatial relationships between the pixels and devote more attention to

the vicinity of the lane lines by spatial attention, reducing the inference time and improving the speed.

The decoding part is guided using the channel attention, which not only fuses the low-level information of the encoding module with the high-level information of the decoding module, but also provides rich lane line information for the existence and segmentation of the branch line. It reduces the model's misjudgment of lane lines and improves the accuracy of the lane line prediction.

We evaluated our method on two popular lane detection benchmarks (CULane and TuSimple), achieving novel performances (76.2 F1-measure on CULane; 96.85% accuracy on TuSimple). Therefore, it can be used as a strong benchmark to facilitate future research on lane line detection.

The remainder of this paper is organized as follows. Section 2 discusses related studies. In Sect. 3, we explain our methodology in three parts: the encoder module, SPRESA module, and decoder module. Section 4 compares the experimental results of our proposed method with those of other algorithms and presents a large number of visualization results. Finally, Sect. 5 concludes the paper.

2 Related studies

Lane detection plays an important role in autonomous driving. Many lane line detection methods have been proposed, which can be broadly classified into two categories: traditional methods and deep learning methods.

2.1 Traditional method

Traditional lane line detection methods [1, 7, 15, 18, 42, 48] typically include the following steps: image preprocessing, feature extraction, lane line detection, and lane tracking. The main purpose of image preprocessing is to eliminate irrelevant information in the image and store useful real information. For this step, the image may be enhanced to facilitate the extraction of lane information, generate a region of interest (ROI), and remove non-lane information. Few studies [3, 6, 48] generated ROIs by selecting the lower part of the image to reduce redundant information, while others [24, 40, 50] used vanishing points. After the ROI is generated, an inverse transmission transform [8, 19], median filter [51], and finite impulse response filters [46] are used to reduce the noise to facilitate the extraction of lanes and remove non-lane information. In traditional methods, lane lines are usually modeled in the form of lines [27], parabolas [22, 47], and hyperbolas [50]. In addition, lane tracking generally uses the Kalman filter and particle filter detection methods [27, 39] for post-processing. However, these traditional methods have been developed based on certain strong assumptions and take specific measures for specific problems. When faced with challenging scenarios, such as bad

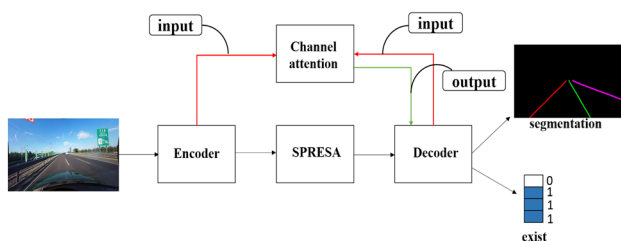


Fig. 1 Simplified network architecture of the proposed method

weather, bad lighting, and road congestion, they have poor generalization ability.

2.2 Deep learning methods

In recent years, with the development of convolutional neural networks (CNNs), an increasing number of studies have applied deep learning methods to lane line detection [15, 33, 36], which addresses the challenges faced by the traditional methods, including the fine-tuning of the steps in ROI generation, filtering, and tracking. In deep learning, lane line detection is considered a segmentation problem, where the input is an image and the output is a segmentation map. Thus, it involves dense pixel-by-pixel prediction and is difficult to guarantee in real time.

Huval et al. [17] predicted the two endpoints of the local line segment in a sliding window using fully convolutional network (FCN) regression. VPGNet [23] uses vanishing points to assist in the monitoring and positioning of traffic routes. Neven et al. [32] proposed LaneNet, which performs detection in two stages: (i) lane edge proposal generation and (ii) lane location. Deeplanes [13] used two horizontally mounted downward cameras to estimate the position of the lane point; however, because of the orientation problem of the camera, this method has significant limitations and cannot fully utilize all the information in the scene.

Qin et al. [37] expressed lane detection as a row-based selection problem. The use of row-based selection reduces the computational cost of the lane detection tasks. In addition, the self-attention distillation (SAD) method is used for the lane detection task, allowing the model to self-learn with any number of additional labels [15]. Wang et al. [45] proposed a multitasking approach that considered the localization capability of the lane lines. Its main idea was to integrate the semantic information extracted by CNNs with the localization capability provided by manual features and to predict the vanishing point location.

Zhang et al. [52] used generative adversarial networks with a multi-target semantic segmentation approach to detect lane lines, which improved the detection accuracy for complex or ambiguous lane scenes. Su et al. [41] proposed a structural information-guided lane line detection framework, called SGNet, that can accurately describe lane lines and classify and locate an uncertain number of lane lines. Li [26] used cascaded CNNs to perform instance segmentation on the lane boundaries followed by a linear classifier for lane classification.

3 Proposed method

In this section, we describe the proposed lane line detection network in detail (Fig. 2). The architecture consists of three parts: the encoder module, which is used to extract

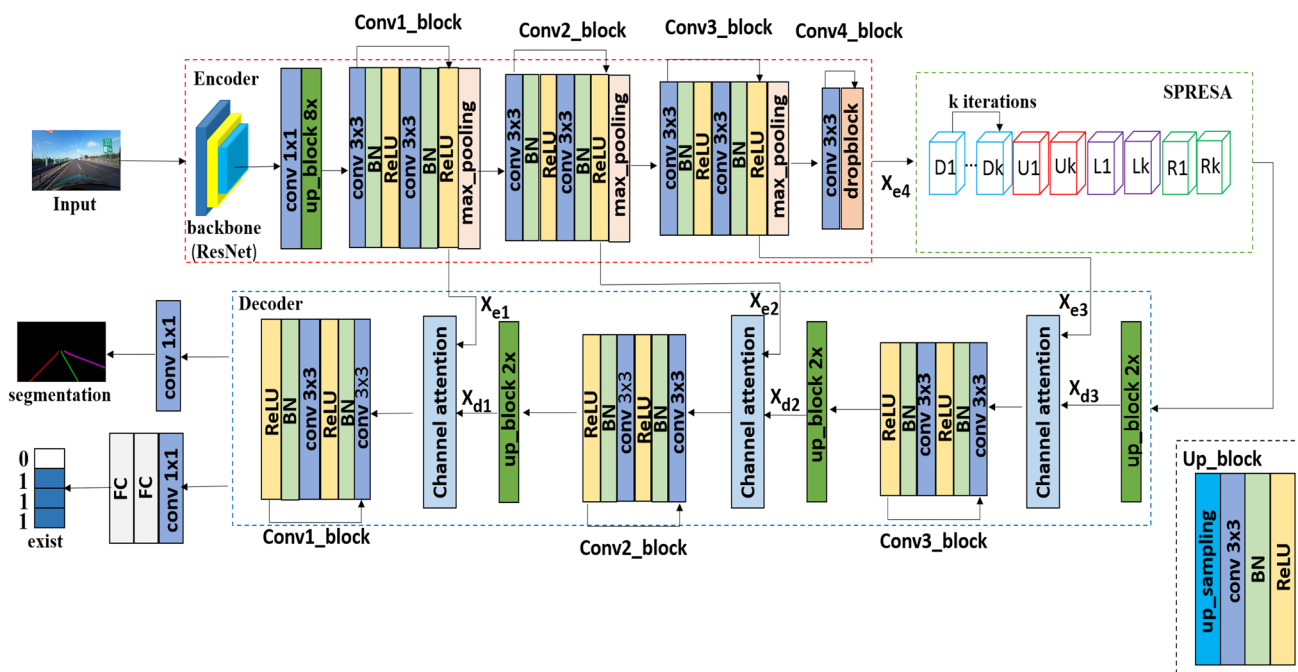


Fig. 2 Overall network architecture, which comprises three parts: (i) an encoder module that extracts feature information initially, wherein X_{e1} , X_{e2} , X_{e3} , and X_{e4} are the extracted low-level features; (ii) a SPRESA module that moves the sliced feature maps cyclically in the vertical and horizontal directions to aggregate spatial informa-

tion; and (iii) a decoder module that aggregates low-level and high-level features, wherein X_{d1} , X_{d2} , and X_{d3} are the high-level features. Up_block is an up-sampling block. In the figure, $2 \times$ and $8 \times$ represent up-sampling factors of 2 and 8, respectively

Table 1 Modules comprising the proposed network, along with their layer details and dimension information

	Layer Input data	Output size
Encoder	ResNet	2048*36*100
	Conv 1×1	128*36*100
	Up_block	64*288*800
	Conv1_block	32*288*800
	Max pooling	32*144*400
	Conv2_block	64*144*400
	Max pooling	64*72*200
	Conv3_block	128*72*200
	Max pooling	128*36*100
	Conv4_block	128*36*100
	Dropblock	128*36*100
	SPRESA	D0-U0-R0-L0
D1-U1-R1-L1		128*36*100
D2-U2-R2-L2		128*36*100
D3-U3-R3-L3		128*36*100
Decoder	Up_block	128*72*200
	Channel attention	128*72*200
	Conv3_block	128*72*200
	Up_block	64*144*400
	Channel attention	64*144*400
	Conv2_block	64*144*400
	Up_block	32*288*800
	Channel attention	32*288*800
Output	Segmentation	5*288*800
	Conv 1×1	5*36*100
	Exist	5*18*50
	Conv 1×1	4500->128
	Average pooling	128->4
	fully connected fully connected	

features from the input image; the SPRESA module, which moves the sliced feature maps cyclically in the vertical and horizontal directions to aggregate spatial information; and

the decoding part, where the low-level features of the encoding module and the high-level semantic information of the decoding module are fused through channel attention. The dimension information of each module is shown in Table 1. Thus, the decoding module provides richer spatial and channel feature information for the lane line existence and probability distribution.

3.1 Encoder

The encoder backbone uses the common lightweight model, ResNet. Taking the CULane dataset using ResNet_50 and ResNet_34 as an example, their details are shown in Table 2. In this study, we used the ResNet (CULane—ResNet_50, TuSimple—ResNet_34) to extract features preliminarily. It can be seen from Table 2 that after the input image is processed by ResNet_34 and ResNet_50, a feature map with a size 1/8 that of the input image is obtained. In the ResNet, we use a 1×1 convolution to reduce the number of output channels to 128. Thereafter, it is restored to the input image size via up-sampling. Next, feature maps of different sizes are obtained through four operation blocks, which can be easily concatenated using the decoder module. Among them, the first three operation blocks consist of a convolutional layer, a batch normalization and an activation function are followed immediately by a max pooling layer, while the last operation block is the matching decoding module feature map, which includes only one convolutional layer and one Dropblock layer [11].

The encoder module is further explained as follows: after the initial feature extraction of the input image by ResNet, four operation blocks are utilized to further enrich the encoding information of the low-level features. Subsequently, to avoid the influence of a few abnormal samples on the entire model, the feature map after the convolution

Table 2 Network structure of ResNet_34 and ResNet_50

Layer name	Output size	34-layer	50-layer
Conv1_block	144×400	Conv, 7×7, 64, stride 2	
Conv2_block	72×200	Max pool, 3×3, stride 2	
		$\left(\begin{matrix} \text{conv, } 3 \times 3, 64 \\ \text{conv, } 3 \times 3, 64 \end{matrix} \right) \times 3$	$\left(\begin{matrix} \text{conv, } 1 \times 1, 64 \\ \text{conv, } 3 \times 3, 64 \\ \text{conv, } 1 \times 1, 256 \end{matrix} \right) \times 3$
Conv3_block	72×200	$\left(\begin{matrix} \text{conv, } 3 \times 3, 128 \\ \text{conv, } 3 \times 3, 128 \end{matrix} \right) \times 4$	$\left(\begin{matrix} \text{conv, } 1 \times 1, 128 \\ \text{conv, } 3 \times 3, 128 \\ \text{conv, } 1 \times 1, 512 \end{matrix} \right) \times 4$
Conv4_block	36×100	$\left(\begin{matrix} \text{conv, } 3 \times 3, 256 \\ \text{conv, } 3 \times 3, 256 \end{matrix} \right) \times 6$	$\left(\begin{matrix} \text{conv, } 1 \times 1, 256 \\ \text{conv, } 3 \times 3, 256 \\ \text{conv, } 1 \times 1, 1024 \end{matrix} \right) \times 6$
Conv5_block	36×100	$\left(\begin{matrix} \text{conv, } 3 \times 3, 512 \\ \text{conv, } 3 \times 3, 512 \end{matrix} \right) \times 3$	$\left(\begin{matrix} \text{conv, } 1 \times 1, 512 \\ \text{conv, } 3 \times 3, 512 \\ \text{conv, } 1 \times 1, 2048 \end{matrix} \right) \times 3$

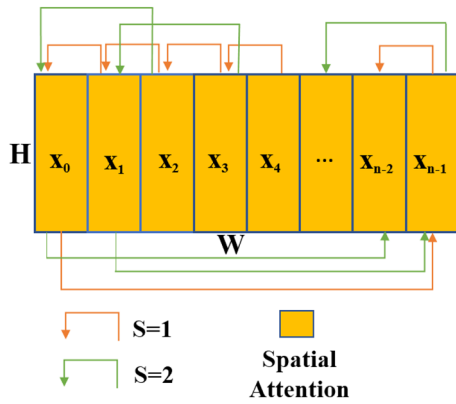


Fig. 3 SPRESA module, which transfers information from right to left with step lengths of one and two, respectively. The size $H \times W$ feature map is from the last operation block of the encoding module, $X_{0 \dots n-1}$ are the feature maps obtained by slicing in the vertical direction. Yellow shading is the spatial attention mechanism

operation is normalized (BatchNorm). This can effectively prevent the explosion of the gradient and will place part of the sample below zero or above zero according to the gradient, which effectively matches the filtering effect of the

activation function. Our method employs the rectified linear unit (ReLU) nonlinear activation, which can significantly improve the expression ability of the neural networks in this model. Assuming x^l as the feature map of layer l , the expression of the ReLU activation function is depicted in Eq. (1):

$$f(x_{i,c}^l) = \max(0, x_{i,c}^l), \tag{1}$$

i denotes the feature map space dimension; c denotes the channel dimension, and l denotes the number of layers to which the feature map belongs. After convolution and activation, the feature map expression is given by Eq. (2):

$$x_c^l = f\left(\sum_n x_n^{l-1} * k_{n,c}\right) \tag{2}$$

k is the convolution kernel, f_{nl} is the number of l -layer feature maps, and $*$ represents the convolution operation. Furthermore, to prevent overfitting problems in the training process, the dropout layer is added after activating the function [41]. Finally, through the processing of the max pooling layer, we obtain different scale feature maps that are spliced with the decoder module.

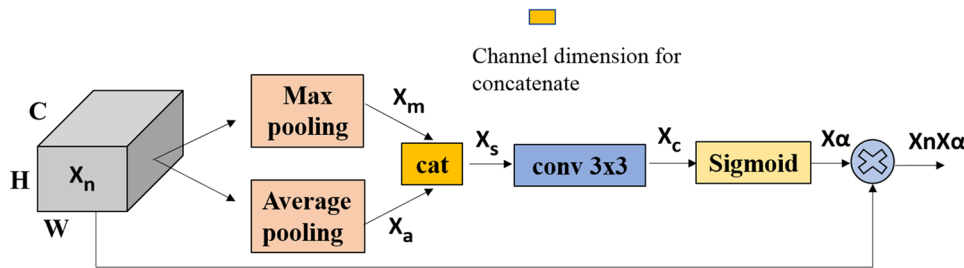
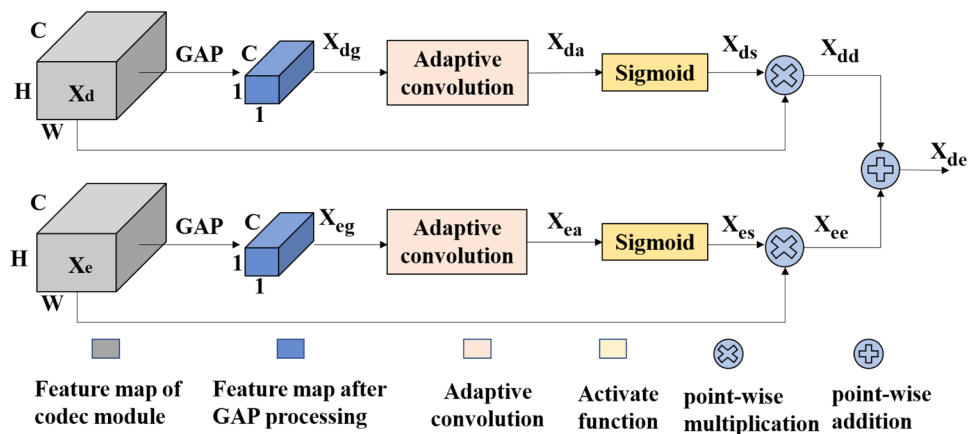


Fig. 4 Spatial attention structure: X_n is the feature map in the vertical or horizontal direction; X_m and X_a are the feature maps of max pooling and average pooling, respectively; X_s is the feature map after X_m

and X_s are concatenated in the channel direction; X_c is the feature map after X_s convolution; X_a is spatial attention feature; and $X_n X_n$ is the input of the decoding module

Fig. 5 Channel attention structure: X_e is the encoder module feature map, X_d is the decoder module feature map, X_{dg}/X_{eg} are the feature maps of global average pooling, X_{dd}/X_{ea} are the feature maps of adaptive convolution, X_{ds}/X_{es} are the channel attention feature maps, X_{dd}/X_{ee} are the feature maps of point-wise multiplication, and X_{de} is the multidimensional channel attention map



3.2 SPRESA module

In CNNs, spatial information is lost owing to the convolutional step size and repeated use of the pooling layer. To prevent this, researchers [30, 34] increased the receptive field size through the deconvolution operation, but this is computation-intensive. Zheng et al. [53] therefore proposed a RESA module that cyclically moves the slice feature maps in the vertical (top to bottom and bottom to top) and horizontal (left to right and right to left) directions, which allows each pixel to collect global information. However, for the entire image, lane lines only occupy a part of the image and do not extend to the image boundary. Therefore, there is redundancy in their method. Considering this, we added spatial attention to the RESA module and named it SPRESA.

Spatial attention mechanisms can be adapted to aggregate information useful to lane lines. Taking information transfer from right to left as an example, the overall structure is shown in Fig. 3. Each feature map circulates from right to left in step sizes of one and two, respectively. At the first iteration and step $S=1$, X_i can receive the information passed from X_{i+1} . Because it is a circular movement, the features at the tail can receive the feature information from the other side, i.e., X_{n-1} can receive the feature information from X_0 . At the second iteration and with step $S=2$, X_i can receive features from X_{i+2} . For example, X_0 can receive information from X_2 at the second iteration, whereas X_0 and X_2 have already received information from X_1 and X_3 in previous iterations, respectively. Therefore, now it takes only two iterations for X_0 to receive information from X_1 , X_2 , and X_3 . The other iterations are similar to this. Thus, only K (Fig. 2) iterations are required for each piece of the feature image (X_i) to obtain the entire feature image information. In addition, we added the spatial attention feature to each message delivery, as shown by the yellow shading in Fig. 3. The spatial attention structure is illustrated in Fig. 4.

In Fig. 4, the sliced feature map X_n (Fig. 3) is used as the input feature map for the module. Firstly, for X_n , max pooling and average pooling are carried out based on the channel direction to obtain two feature maps X_m and X_a of size $H \times W \times 1$. Max pooling encodes the most significant part of the feature map, whereas average pooling encodes the global statistical information. They are then concatenated and aggregated into the channel information of a feature map, i.e., X_s . Therefore, these two features can be used simultaneously to obtain a more effective lane line feature descriptor. The next step is to apply a convolution operation (conv 3×3)

to reduce the number of channels to one ($H \times W \times 1$), which focuses on useful information for the lane lines. Moreover, a nonlinear activation function (Sigmoid) is employed after the convolutional layer to generate the spatial attention feature (X_α), which makes the decision function discriminative. Finally, X_α is multiplied elementwise with the input feature map X_n to generate the input feature map required by the decoding module.

3.3 Decoder

Although the SPRESA module obtains rich spatial information, for the lane line marker detection, the information of small objects with large variability will be lost in the long-distance transmission process. Therefore, information interactions between channels are important. In addition, lane marker detection is a multi-classification problem; therefore, we use multidimensional attention to learn semantic information in images. Consequently, we add the channel attention between encoding and decoding.

Figure 5 shows the attention module. We use the processing of the encoder module as an example to explain the attention module. First, a $C \times H \times W$ (encode module feature maps X_{e1} , X_{e2} , and X_{e3} , as shown in Fig. 2) feature map is processed via the global average pooling (GAP) to obtain a $C \times 1 \times 1$ feature map (X_{eg}). Second, it performs adaptive convolution processing to obtain a $C \times 1 \times 1$ channel attention feature map (X_{ea}); this allows a layer with a larger number of channels to perform more cross-channel interactions [44]. Finally, an activation function is added to increase the generalization ability, and then the feature map X_{es} is multiplied point-wise with the input feature map X_e to generate the $H \times W \times C$ size feature map (X_{ee}). The procedure utilized by the decoder module is similar to the above procedure. The size function of the adaptive convolution kernel is given by Eq. (3):

$$k = \varphi(C) = \left\lfloor \frac{\log_2(C)}{\gamma} + \frac{b}{\gamma} \right\rfloor, \quad (3)$$

where C is the number of channels and γ and b are set to two and one, respectively. There are three identical blocks in the decoder, each consisting of a convolutional layer and an up-sampling layer that increases the size of the feature map, as shown in Fig. 2. The convolutional layer operation is the same as that of the encoding module. As shown in Fig. 2, the feature map is processed by the SPRESA module, which is sequentially convolved and up-sampled to obtain a feature

Table 3 Details of the experimental datasets

Dataset	#Frame	Train	Validation	Test	Resolution	#Lane	#Scenarios	Environment
TuSimple	6408	3268	358	2782	1280 × 720	≤ 5	1	Highway
CULane	1,33,235	88,880	9675	34,680	1640 × 590	≤ 4	9	Urban and highway

map corresponding to the size of the encoding module. Next, the channel attention mechanism is used to process it, which is the same as the encoding module, as shown in Fig. 5. Finally, the attention maps of the codec channels obtained above are fused with a multidimensional attention feature map. Consequently, the decoding module can provide abundant channel and spatial information for the subsequent lane line existence and segmentation probability map, thereby reducing the fault prediction and post-processing of lane lines.

4 Experiments

In this section, we report the evaluation conducted of EDNet on two public datasets: TuSimple and CULane. Section 4.1 outlines the evaluation metric used for each dataset in the official evaluation methods. Section 4.2 gives the details of the implementation. Section 4.3 shows the evaluation results of EDNet; Sect. 4.4 presents an ablation study of the attentional mechanism on the inference time, FP, FN, and F1-measure.

Table 4 Comparison with state-of-the-art results on the TuSimple dataset

Method	Accuracy	FP	FN
ResNet-18 [15]	92.69	0.0948	0.0822
ResNet-34 [15]	92.84	0.0918	0.0796
LaneNet [32]	96.38	0.0780	0.0244
EL-GAN [10]	96.39	0.0412	0.0336
FCN-Instance [16]	96.5	0.0851	0.0269
SCNN [33]	96.53	0.0617	0.0180
R-18-SAD [15]	96.02	0.0786	0.0451
R-34-SAD [15]	96.24	0.0712	0.0344
EDNet-34 (Our method)	96.85	0.0390	0.0290

The optimal results for each indicator is indicated in bold

4.1 Datasets

To validate our model, we conducted experiments on two widely used benchmark datasets: TuSimple and CULane. The CULane dataset was acquired from six different vehicles driven by six different drivers on Beijing roads. Over 55 h of video was collected, and 1,33,235 frames were extracted. It consists of nine different scenes: normal, crowd, night, no line, shadow, arrow, dazzle night, curve, and crossroad of urban roads. The TuSimple dataset consists of 6408 road images on US highways, where the images are under different weather conditions. The details of the datasets are presented in Table 3.

The official evaluation metrics differ for the two datasets. For the TuSimple dataset, the main evaluation criterion is the accuracy rate, defined by the following equation according to the average number of correct points:

$$\text{accuracy} = \frac{\sum_{\text{clip}} C_{\text{clip}}}{\sum_{\text{clip}} S_{\text{clip}}} \tag{4}$$

C_{clip} is the number of correct predictions by the model, and S_{clip} is the total number of ground truths. The false negative (FN) and false positive (FP) rates are also calculated by the following equations:

$$\text{FP} = \frac{F_{\text{pred}}}{N_{\text{pred}}}, \tag{5}$$

$$\text{FN} = \frac{M_{\text{pred}}}{N_{\text{gt}}}, \tag{6}$$

F_{pred} denotes the number of wrongly predicted lanes; N_{pred} denotes the number of predicted lanes; M_{pred} denotes the number of missed lanes; and N_{gt} denotes the number of ground-truth lanes.

For the CULane dataset, according to [33], assuming a width of 30 pixels for each traffic lane, the intersection-over-union (IoU) is calculated between the model predictions and

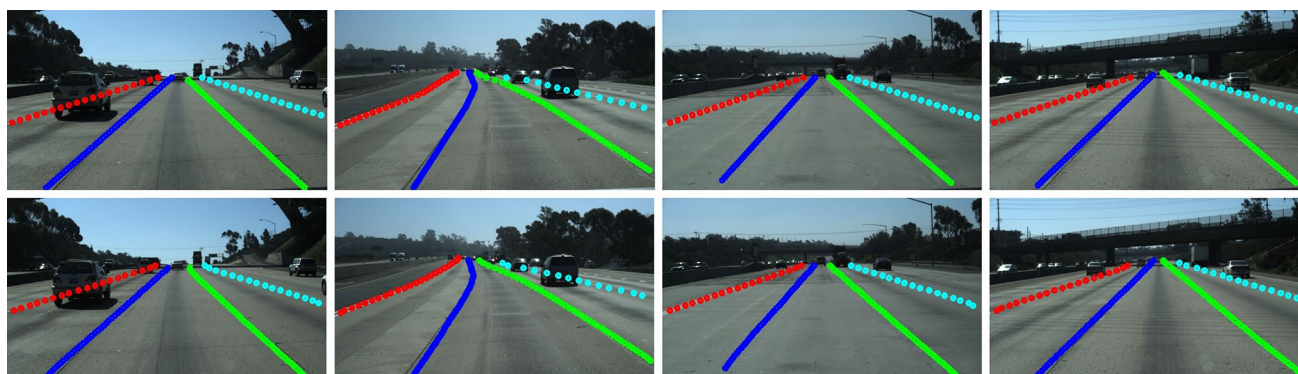


Fig. 6 Visualization results obtained on the TuSimple dataset: (first row) ground truth; (second row) model prediction results

Table 5 Test results of different methods on the CULane dataset

Category	Res50 [48]	Res34-VP [24]	SCNN [28]	Res34-SAD [15]	Res34-Ultra [33]	PINet [35]	Curve-Lane-L [50]	EDNet-50 (our method)
Normal	87.4	90.4	90.6	89.9	90.7	90.3	90.7	92.6
Crowded	64.1	69.2	69.7	68.5	70.2	72.3	72.3	74.3
Night	60.6	63.8	66.1	64.6	66.7	67.7	68.2	70.3
No-line	38.1	43.1	43.4	42.2	44.4	49.8	49.4	48.0
Shadow	60.7	62.5	66.9	67.7	69.3	68.4	70.1	74.5
Arrow	79.0	83.5	84.1	83.8	85.7	83.7	85.8	87.4
Hlight	54.1	61.4	58.5	59.9	59.5	66.3	67.7	66.3
Curve	59.8	64.7	64.4	66.0	69.5	65.6	68.4	70.3
Crossroad	2505	2141	1990	1960	2037	1427	1746	1780
Total	66.7	70.9	71.6	70.7	72.3	74.4	74.8	76.2
Runtime (ms)	–	26	116	<51	6	40	–	20.5

The optimal results for each indicator is indicated in bold

‘–’ indicates unknown results; for crossroad, only the FP is shown, The bold section shows the optimal results for each indicator

the ground truth. Furthermore, using the F1-measure as an evaluation metric, the expression can be presented as follows:

$$\text{F1-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad (8)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (9)$$

TP denotes true positive, meaning that the predicted IoU is higher than the threshold (0.5), FP denotes false positive, and FN denotes false negative.

4.2 Implementation details

Regarding the processing of the dataset, our method is consistent with that of [15], adjusting the CULane image to 288×800 and the TuSimple image to 368×640 . We used the SGD optimizer [2] with a momentum of 0.9 and a weight decay of $1e-4$ to train the network model. The learning rates of CULane and TuSimple were set to $2.5e-2$ and $2.0e-2$, respectively. We used the warm-up strategy [12] for the first 500 batches and then applied a polynomial learning rate decay strategy [31] with power set to 0.9. The loss function was the same as that of SCNN [33], which consists of the segmentation binary cross-entropy loss and existence classification loss. Considering the label imbalance between the background and lane markings, the segmentation loss of the background was multiplied by 0.4. For the dataset, training

batches and epochs were treated in a similar manner as that in [53], with the CULane batch size set to eight and epoch to 12, and the TuSimple batch size to eight and epoch to 150. The model was trained with an NVIDIA 3090 GPU (24 GB Memory) on an Ubuntu 20.04 system. All models were implemented using PyTorch1.9.

4.3 Results

To validate the effectiveness of our method, we performed extensive comparisons with several state-of-the-art methods. For the TuSimple dataset, seven methods were used for comparison, namely, Resnet-18 [15], ResNet34 [15], LaneNet [32], EL-GAN [10], FCN-Instance [16], SCNN [33], R-18-SAD [15], and R-34-SAD [15]. We used ResNet34 as our backbone and named our method EDNet-34; the results are shown in Table 4. Figure 6 shows the partial visualization results on the TuSimple dataset. As shown in Table 4, EDNet-34 achieved an accuracy of 96.85%. We also analyzed FP and FN, which yielded competitive performances. This means that for small objects with large variability, our method can predict more accurately than the other methods.

The CULane dataset was used to compare several advanced lane line detection methods, namely, ResNet50 [4], Res34-VP [29], SCNN [33], ResNet34-SAD [15], Res34-Ultra [37], PINet [20], and CurveLane [49]. The training and test images used in the above method were the same. We used ResNet50 as our backbone and named our method EDNet-50. Table 5 shows the results. Figure 7 shows the results of the nine scene visualizations for the CULane dataset. As shown in Table 5, in various scenes of the CULane dataset, our method had a competitive advantage over the other methods in terms of F1-measure and speed. This

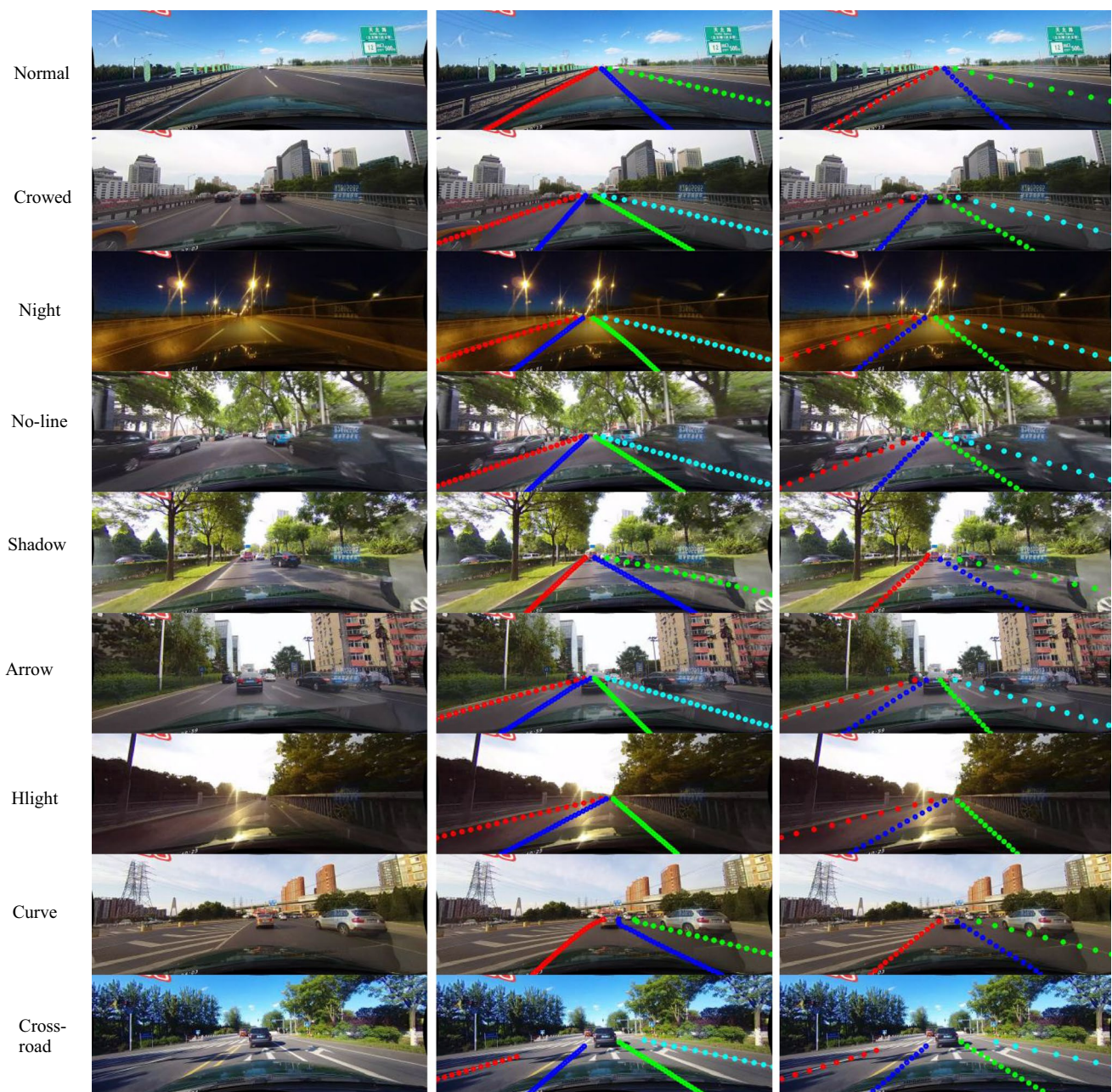


Fig. 7 Visualization results of various scenes in the CULane dataset: (first column) original image; (second column) ground truth; (third column) prediction result

proves the effectiveness of the proposed channel and spatial attention. In addition, we achieved the fastest 48 fps, which shows that our proposed architecture incurs a low computational cost and can be applied in autonomous driving systems.

The visualization results in Figs. 6 and 7 show that our proposed method can accurately fit lane lines in various scenarios. To further verify the feasibility of our algorithm, the

qualitative results of our algorithm and other algorithms on the CULane dataset are shown in Fig. 8. As shown in the figure, the Res_50 methods cannot maintain the continuity and smoothness of the lane lines owing to heavy occlusion by other vehicles or strong light interference. Conversely, SCNN achieves partial performance improvement by passing information between pixel rows and columns, but the prediction results are still suboptimal. In Fig. 8, it can also be observed that SCNN can have over-predicted or

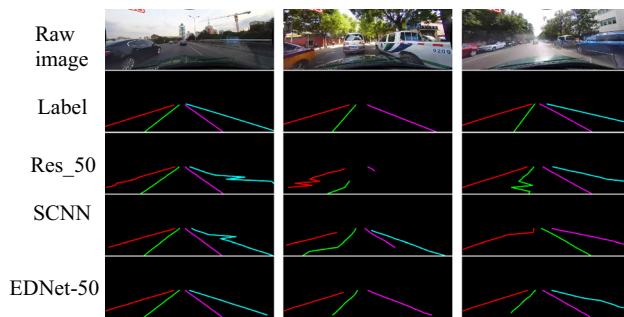


Fig. 8 Qualitative comparison results of Res-50, SCNN, and EDNet on the CULane dataset

Table 6 Experimental results with and without channel attention

	Precision	Recall	F1-measure
Without channel attention	77.3	74.5	75.9
With channel attention	77.6	74.8	76.2

The optimal results for each indicator is indicated in bold

Table 7 Experimental results with and without spatial attention

	Inference time (ms)
Without spatial attention	29.2
With spatial attention	20.5

The optimal results for each indicator is indicated in bold

under-predicted lane lines. This indicates that lane line information may be lost during long propagation and multiple iterations. In these methods, EDNet-50 passes and aggregates information in the horizontal and vertical directions in different steps. In addition, we add spatial attention to allow the network to automatically focus on the lane information. Consequently, our method is robust overall and can get close to the label lane lines in various scenarios.

4.4 Ablation study

The role of the channel attention mechanism is to assign different weights to each channel, allowing the network to focus on important features and suppress unimportant features. Therefore, channel attention can filter out irrelevant information and noise in the low-level features of the encoding module and the high-level semantic information of the decoding module. In addition, channel attention can fuse the feature information of the two to prevent information from being lost during long-distance propagation.

We conducted experiments to determine whether the model had channel attention; the results are presented in Table 6. Compared with the non-channel attention cases, after adding the channel attention, the precision, recall, and F1-measure improved. Considering that the RESA [53] module has redundant calculations during information transmission, we added spatial attention to this module, focusing on the information useful to lane lines, and we performed the appropriate experiments. The results are listed in Table 7. After adding spatial attention, the model inference time decreased. All experiments were set up following the implementation details. The above results indicate that the proposed method is quite effective.

5 Conclusion

In this study, we proposed EDNet, a novel encoder–decoder architecture based on the attention mechanism for lane detection. The encoding module extracts rich low-level information from the input image. In addition, we introduced the SPRESA module, which aggregates effective spatial information with a reduced inference time. To reduce the probability of information loss in the transmission process, we added channel attention to process the low-level information of the encoding module and the feature map of the decoding module, respectively. Consequently, the decoding module obtains rich spatial and channel information, which reduces the false prediction and post-processing of lane line detection. Based on the results of extensive experiments conducted, we verified that our proposed method achieved the most advanced performance on two popular lane detection benchmark datasets (TuSimple and CULane).

Funding National Natural Science Foundation of China (Grant no. 61601354).

References

1. Borkar, A., Hayes, M., Smith, M.T.: A novel lane detection system with efficient ground truth generation. *IEEE Trans. Intell. Transp. Syst.* **13**(1), 365–374 (2011)
2. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: *Proceedings of COMPSTAT'2010*. Physica-Verlag HD, pp 177–186 (2010)
3. Cáceres Hernández, D., et al.: Real-time lane region detection using a combination of geometrical and image features. *Sensors* **16**(11), 1935 (2016)
4. Chen, L.-C., et al.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(4), 834–848 (2017)

5. Chougule, S., et al.: Reliable multilane detection and classification by utilizing CNN as a regression network. In: Proceedings of the European conference on computer vision (ECCV) workshops (2018)
6. Deng, J., Han, Y.: A real-time system of lane detection and tracking based on optimized RANSAC B-spline fitting. In: Proceedings of the 2013 Research in Adaptive and Convergent Systems, pp 157–164 (2013)
7. Deusch, H., et al.: A random finite set approach to multiple lane detection. In: 2012 15th International IEEE Conference on Intelligent Transportation Systems. IEEE (2012)
8. Du, X., Tan, K.K.: Comprehensive and practical vision system for self-driving vehicle lane-level localization. *IEEE Trans. Image Process.* **25**(5), 2075–2088 (2016)
9. Garnett, Noa, et al.: "Real-time category-based and general obstacle detection for autonomous driving." *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017.
10. Ghafoorian, M., et al.: El-GAN: Embedding loss driven generative adversarial networks for lane detection. In: Proceedings of the European conference on computer vision (ECCV) Workshops (2018)
11. Ghiasi, G., Lin, T.-Y., Le, Q.V.: Dropblock: A regularization method for convolutional networks. arXiv preprint [arXiv:1810.12890](https://arxiv.org/abs/1810.12890) (2018)
12. Goyal, P., et al.: Accurate, large minibatch SGD: Training ImageNet in 1 hour. arXiv preprint [arXiv:1706.02677](https://arxiv.org/abs/1706.02677) (2017)
13. Gurchian, A., et al.: Deeplanes: End-to-end lane position estimation using deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2016)
14. Hillel, A.B., et al.: Recent progress in road and lane detection: a survey. *Mach. Vis. Appl.* **25**(3), 727–745 (2014)
15. Hou, Y., et al.: Learning lightweight lane detection CNNs by self-attention distillation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2019)
16. Hsu, Y-C, et al.: Learning to cluster for proposal-free instance segmentation. In: 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, (2018)
17. Huval, B., et al.: An empirical evaluation of deep learning on highway driving. arXiv preprint [arXiv:1504.01716](https://arxiv.org/abs/1504.01716) (2015)
18. Jung, H., Min, J., Kim, J.: An efficient lane detection algorithm for lane departure detection. In: 2013 IEEE Intelligent Vehicles Symposium (IV). IEEE (2013)
19. Jung, S., Youn, J., Sull, S.: Efficient lane detection based on spatiotemporal images. *IEEE Trans. Intell. Transp. Syst.* **17**(1), 289–295 (2015)
20. Ko, Y., et al.: Key points estimation and point instance segmentation approach for lane detection. In: *IEEE Transactions on Intelligent Transportation Systems* (2021)
21. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. *Adv. Neural. Inf. Process. Syst.* **24**, 109–117 (2011)
22. Kwon, S., et al.: Multi-lane detection and tracking using dual parabolic model. *Bull. Netw. Comput. Syst. Softw.* **4**(1), 65–68 (2015)
23. Lee, S., et al.: Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
24. Lee, C., Moon, J.-H.: Robust lane detection and tracking for real-time applications. *IEEE Trans. Intell. Transp. Syst.* **19**(12), 4043–4048 (2018)
25. Levi, D., et al.: StixelNet: a deep convolutional network for obstacle detection and road segmentation. *BMVC* **1**(2), 4 (2015)
26. Li, H., Li, X.: Flexible lane detection using CNNs. In: 2021 International Conference on Computer Technology and Media Convergence Design (CTMCD). IEEE (2021)
27. Liang, M., Zhou, Z., Song, Q.: Improved lane departure response distortion warning method based on Hough transformation and Kalman filter. *Informatica* **41**(3) (2017).
28. Liu, T., et al.: Lane detection in low-light conditions using an efficient data enhancement: Light conditions style transfer. In: 2020 IEEE Intelligent Vehicles Symposium (IV). IEEE (2020)
29. Liu, Y.-B., Zeng, M., Meng, Q.-H.: Heatmap-based Vanishing Point boosts Lane Detection. arXiv preprint [arXiv:2007.15602](https://arxiv.org/abs/2007.15602) (2020)
30. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)
31. Mishra, P., Sarawadekar, K.: Polynomial learning rate policy with warm restart for deep neural network. In: TENCON 2019–2019 IEEE Region 10 Conference (TENCON). IEEE (2019)
32. Neven, D., et al.: Towards end-to-end lane detection: an instance segmentation approach. In: 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, (2018)
33. Pan, X., et al.: Spatial as deep: Spatial CNN for traffic scene understanding. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
34. Papandreou, G., Kokkinos, I., Savalle, P.-A.: Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)
35. Paszke, A., et al.: Enet: A deep neural network architecture for real-time semantic segmentation. arXiv preprint [arXiv:1606.02147](https://arxiv.org/abs/1606.02147) (2016)
36. Phillion, J.: Fastdraw: Addressing the long tail of lane detection by adapting a sequential prediction network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2019)
37. Qin, Z., Wang, H., Li, X.: Ultrafast structure-aware deep lane detection. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16. Springer International Publishing (2020)
38. Romera, E., et al.: Erfnet: efficient residual factorized convnet for real-time semantic segmentation. *IEEE Trans. Intell. Transp. Syst.* **19**(1), 263–272 (2017)
39. Shin, B.-S., Tao, J., Klette, R.: A superparticle filter for lane detection. *Pattern Recogn.* **48**(11), 3333–3345 (2015)
40. Son, J., et al.: Real-time illumination invariant lane detection for lane departure warning system. *Expert Syst. Appl.* **42**(4), 1816–1824 (2015)
41. Su, J., et al.: Structure guided lane detection. arXiv preprint [arXiv:2105.05403](https://arxiv.org/abs/2105.05403) (2021)
42. Tan, H., et al.: A novel curve lane detection based on Improved River Flow and RANSA. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). IEEE (2014)
43. Van Gansbeke, W., et al.: End-to-end lane detection through differentiable least-squares fitting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (2019)
44. Wang, Q., et al.: Supplementary material for 'ECA-Net: Efficient channel attention for deep convolutional neural networks. Tech. Rep
45. Wang, Q., et al.: Multitask attention network for lane detection and fitting. In: *IEEE Transactions on Neural Networks and Learning Systems* (2020)
46. Wang, X., Yongzhong, W., Chenglin, W.: Robust lane detection based on gradient-pairs constraint. In: Proceedings of the 30th Chinese Control Conference. IEEE (2011)
47. Wang, Y., Dahnoun, N., Achim, A.: A novel system for robust lane detection and tracking. *Signal Process.* **92**(2), 319–334 (2012)

48. Wu, P.C., Chin-Yu, C., Chang, H.L.: Lane-mark extraction for automobiles under complex conditions. *Pattern Recogn.* **47**(8), 2756–2767 (2014)
49. Xu, H., et al.: Curvelane-nas: Unifying lane-sensitive architecture search and adaptive point blending. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV* 16. Springer International Publishing (2020)
50. Xu, S., et al.: Road lane modeling based on RANSAC algorithm and hyperbolic model. In: *2016 3rd International Conference on Systems and Informatics (ICSAI)*. IEEE (2016)
51. Xu, H., Li, H.: Study on a robust approach of lane departure warning algorithm. In: *2010 2nd International Conference on Signal Processing Systems*, vol. 2. IEEE (2010)
52. Zhang, Y., et al.: Ripple-GAN: lane line detection with ripple lane line detection network and Wasserstein GAN. *IEEE Trans. Intell. Transp. Syst.* **22**(3), 1532–1542 (2020)
53. Zheng, T., et al.: Resa: Recurrent feature-shift aggregator for lane detection. *arXiv preprint arXiv:2008.13719* (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Qinghua Zhao received the B.S. degree in Tianjin university of technology and education in 2020. In September 2020, He was admitted to Xidian university for a master's degree in Software Engineering. His current research interests include computer vision, self-driving, and deep learning.



Qi Peng (Member, IEEE) received the B.S. degree in electronics science and technology from Xi'an Jiaotong University, Xi'an, China, in 2009, and the Ph.D. degree in electronics science and technology from Tsinghua University, Beijing, China, in 2015. She is now with Xidian University, Xi'an, China. Her current research interests include the fields of UHF RFID system design, image processing, and wireless communication system. Dr. Peng was the Secretary of the 2011 IEEE International Symposium on Radio-Frequency Integration Technology (RFIT).



Yiqi Zhuang (Member, IEEE) received the B.S. and M.S. degrees in Microelectronics from Xidian University, and the Ph.D. degree in Optoelectronics from the Xi'an Institute of Optics & Precision Mechanics of Chinese Academy of Science, China. Now he is a professor School of Microelectronics in Xidian University, and the member of IEEE. His current research interests include design of radio-frequency and short-range Communication Integrated Circuits, system integration technology with a goal of digital, analog, RF and power circuits integrated on one chip, characterization methodology of micro-nanoelectronic devices and materials specially by means of noise and other non-linear parameters.