IEEE | Transactions on Intelligent Transportation Systems

Ripple-GAN: Lane line detection with Ripple Lane Line Detection Network and Wasserstein GAN

Journal:	*Transactions on Intelligent Transportation Systems*
Manuscript ID	T-ITS-19-06-0591.R2
Manuscript Type:	Regular Papers
Date Submitted by the Author:	18-Dec-2019
Complete List of Authors:	Zhang, Youcheng; Tsinghua University, electronic egineering Lu, Zongqing; Tsinghua University, Department of electronic engineering Ma, Dongdong; Tsinghua University, Department of electronic engineering Xue, Jing-Hao; University College London, Department of Statistical Science Liao, Qingmin; Tsinghua University, Department of electronic engineering
Keywords:	Lane line detection, multi-target segmentation, RiLLD-Net, Ripple-GAN
Abstract:	With artificial intelligence technology being advanced by leaps and bounds, intelligent driving has attracted a huge amount of attention recently in research and development. In intelligent driving, lane line detection is a fundamental but challenging task particularly under complex road conditions. In this paper, we propose a simple yet appealing network called Ripple-Net, to exploit quick connections and gradient maps for effective learning of lane line features. Ripple-Net can handle most common scenes of lane line detection. Then, in order to address challenging scenarios such as occluded or complex lane lines, we propose a more powerful network called Ripple-GAN, by integrating Ripple-Net, confrontation training of Wasserstein generative adversarial networks, and multi-target semantic segmentation. Experiments show that, especially for complex or obscured lane lines, Ripple-GAN can produce a superior detection performance to other state-of-the-art methods.

SCHOLARONE™
Manuscripts

PLEASE KEEP CONFIDENTIAL

Ripple-GAN: Lane line detection with Ripple Lane Line Detection Network and Wasserstein GAN

Youcheng Zhang, Zongqing Lu*, Dongdong Ma, Jing-Hao Xue, Qingmin Liao

***Abstract*—With artificial intelligence technology being advanced by leaps and bounds, intelligent driving has attracted a huge amount of attention recently in research and development. In intelligent driving, lane line detection is a fundamental but challenging task particularly under complex road conditions. In this paper, we propose a simple yet appealing network called Ripple Lane Line Detection Network (RiLLD-Net), to exploit quick connections and gradient maps for effective learning of lane line features. RiLLD-Net can handle most common scenes of lane line detection. Then, in order to address challenging scenarios such as occluded or complex lane lines, we propose a more powerful network called Ripple-GAN, by integrating RiLLD-Net, confrontation training of Wasserstein generative adversarial networks, and multi-target semantic segmentation. Experiments show that, especially for complex or obscured lane lines, Ripple-GAN can produce a superior detection performance to other state-of-the-art methods.**

***Index Terms*—Lane line detection, multi-target segmentation, RiLLD-Net, Ripple-GAN**

I. Introduction

IN the current era, cars have become an indispensable means of transportation. Increasingly complex traffic conditions make the advanced driver assistant system (ADAS) an essential tool for people's safe travel by car. The ADAS uses a variety of sensors installed on the car to collect environmental data inside and outside the vehicle and perform relevant data processing and analysis tasks, such as identification and detection of static and dynamic objects, such that the driver can detect potential dangers in the quickest time. Lane line detection is a critical and challenging part of the ADAS. With the help from automatic lane line detection, drivers can better understand the conditions of current road and prepare for different road trends in advance. When the driver is fatigued or does not concentrate on driving, lane change maneuvers may cause traffic accidents. Such traffic accidents can be well prevented by properly detecting the current road and alarming the vehicle deviation. Hence the lane line detection plays an important role in autonomous driving technology.

To build a lane line detection system, many challenges need to be overcome, such as different types of lane marks, occlusion, defect and interference of lane lines. To address these challenges, many lane line detection methods have been proposed. These methods can be roughly divided into two categories: traditional methods and deep learning methods.

The traditional lane line detection methods usually use visual characteristics of lane lines as features, such as high-contrast colors, linear shapes, and presence in specific areas of the image, to distinguish the lane lines from other objects in the image. These methods often contains the following steps: image preprocessing, feature extraction, and lane line detection. The image preprocessing step helps to reduce computational time and improve the performance of algorithms [1]. In this step, researchers may change the color space, for instance from RGB to YCbCr [2], select regions of interest (ROI) [3]–[5], or use some filters such as the median filter [6] and the finite impulse response (FIR) filter [7] to eliminate interference. In the step of feature extraction, brightness-related characteristics, linearity and spectrum are three commonly used features in the lane line detection. The Canny and Sobel operators are often used for edge detection [2], [8]. The local maxima of intensity, gradient maps [9] and histograms [10] are also used to provide useful lane line information. In the step of lane line detection, the Hough transform (HT) is a basic and general way to estimate the parameters of lane marks [11], [12]. [13] first divided the ROI of road image into the straight region and the curve region, and then established the straight model by using the HT, as well as the curve model by using the lane-line continuity and tangent. As the vanishing point contains global information of a road, [14], [15] use it to guide lane line detection. However, these traditional methods usually are developed under some strong assumptions and take specific features and measures for specific problems in a low-level feature space, and thus often their applications are limited and their performances are susceptible to interference.

To avoid the drawbacks caused by the use of hand-crafted features, learning-based approaches have been proposed in the lane line detection. [16] used Gaussian mixture models and hidden Markov models to predict the forthcoming vehicle trajectory by multiple iterations. Recently, with the rise of deep learning in the computer vision field, more and more neural network-based lane line detection methods have emerged. Convolutional neural networks (CNNs) are used to directly detect the lane line from images [17]–[19]. [20] and [13] share similar ideas of treating different types of lane lines separately. [21] used two CNNs, with the second network to determine the location and shape of each lane based on the edge proposals produced by the first network. To improve the performance of CNN, other techniques have been melded with it, for example, CNNs were blended with other neural networks such

This work was partly supported by the National Natural Science Foundation of China (No.61771276), and the Special Foundation for the Development of Strategic Emerging Industries of Shenzhen (No. JCYJ20170817161056260). (Corresponding author: Zongqing Lu)

Y. Zhang, Z. Lu, D. Ma, Q. Liao are with the Department of Electronic Engineering and the Shenzhen International Graduate School, Tsinghua University, China

J.-H. Xue is with the Department of Statistical Science, University College London, UK

PLEASE KEEP CONFIDENTIAL

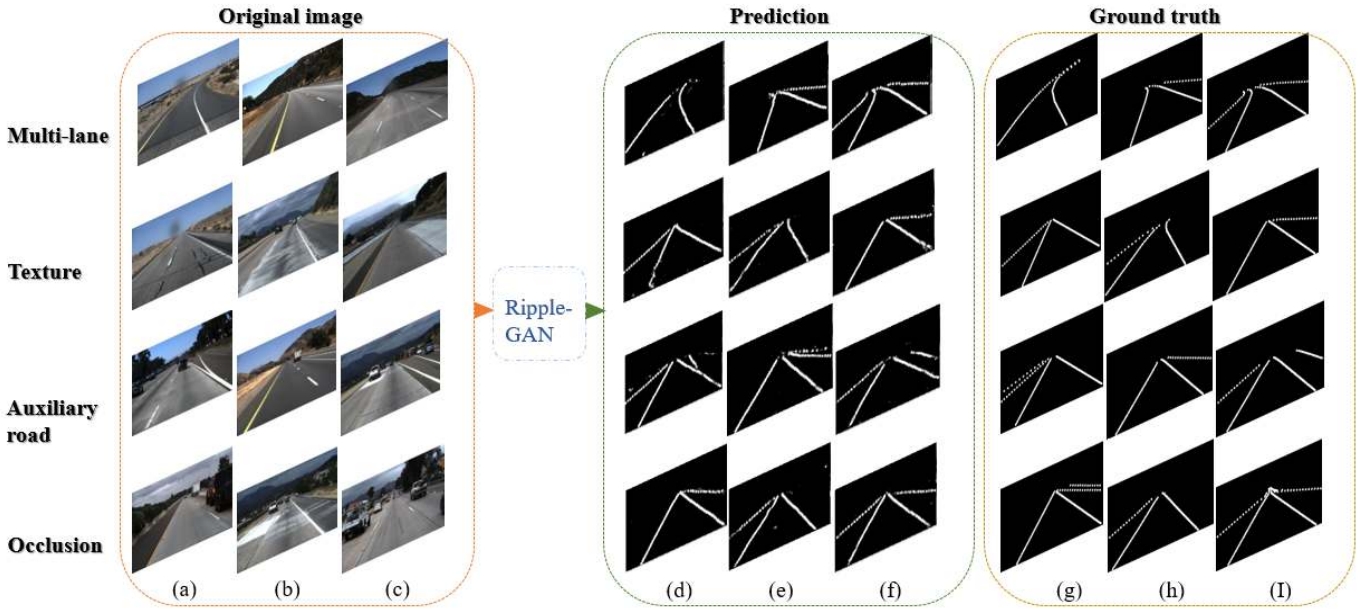


Fig. 1. Lane line detection by our proposed Ripple-GAN in four different scenarios: multiple lanes, pavement textures, auxiliary road, and occlusions. All predicted images are not post-processed.

as recurrent neural network (RNN) [22]. In this combination, CNN is responsible for providing the geometric characteristics of lane lines for RNN, and RNN is responsible for detecting lines. There are also other studies combining CNN with hand-crafted features. For instance, [23] proposed a dual-view CNN (DVCNN), of which the input part includes both front-view and top-view images. In general, learning-based approaches demonstrated good performance of lane line detection on simple road scenes, but they still face the challenges including noise interference, shadow, different weather conditions, etc.

Most traditional methods rely too much on hand-crafted features and often have poor performance in unfamiliar scenes. For deep learning based methods, there is a trade-off between the performance and the network complexity. If the network is too shallow, the detection performance will be poor because the feature learning is insufficient, but if the network is too deep, the performance of network may still be poor because it would be difficult to train the network satisfactorily. Moreover, complex road conditions including occlusion problems are the cases where both methods cannot handled well. In this paper, motivated by the desire to overcome these issues and improve the state-of-the-art work for lane line detection, we propose a new simple network called RiLLD-Net, and then based on RiLLD-Net develop a more powerful network called Ripple-GAN: the simple RiLLD-Net can correctly detect lane lines in most common scenes, and the Ripple-GAN can handle challenging scenes, such as incomplete and complex lane lines, by integrating the strengths of both RiLLD-Net and generative adversarial learning. Some examples of lane lines detected by the proposed Ripple-GAN in four different challenging scenes are shown in Fig. 1. It can be seen from the results that, even without post-processing, Ripple-GAN performs very well in those four scenarios. Especially when the lane line is complexly distributed, Ripple-GAN can predict the actual lane lines missed in the labeled ground truth, e.g. in Fig. 1(a) the line of an auxiliary road.

The main contributions of this paper are as follows:

- We propose RiLLD-Net, a new, simple network that passes feature maps between modules at various distances. In this way, the network can be learned more efficiently and the characteristics of each stage can be fully utilized. Combined with gradient information, the RiLLD-Net can highlight the lane line properties as well as to remove the interference, and thus detect the lane line effectively for most common scenes.
- We then propose a more powerful RiLLD-Net-based lane line detection network called Ripple-GAN, to further improve the detection performance under many challenging scenarios with only partial lane line information available. In the Ripple-GAN, we combine the RiLLD-Net, Wasserstein generative adversarial network (WGAN), and semantic segmentation. It can be verified that integrating generative adversarial training and multi-target segmentation can help the RiLLD-Net cope well with occluded and complex scenes.

The rest of the paper is organized as follows. Section II reviewed the existing lane line detection work and briefly clarifies the superiority of our approach. The proposed lane line detection networks, RiLLD-Net and Ripple-GAN, are introduced in Section III. In Section IV, the experimental details and results are presented, and the effectiveness of our methods is validated. Finally, some conclusions and future work are presented in Section V.

II. Closely-related Work

A. Semantic segmentation

In intelligent driving, the task of semantic segmentation is to group pixels in the scene into different classes for better

understanding of the driving situation. Usually it treats the entire road as a whole [24], [25], and CNN-based methods are now its mainstream for lane line detection. [26] proposed the Spatial CNN (SCNN) for lane line detection. SCNN sequentially sends and accumulates the slices of a 3D input tensor into convolution layers, and thus is suitable for structures with strong spatial relationship, such as continuous shape structure. However, SCNN suffers from the presence of dashed lines, occlusions and defects in the lane lines on the road, or some interference with a similar spatial structure to lane lines. Moreover, it is limited to detecting only a predefined, fixed number of lanes. LaneNet [27] is another effective semantic segmentation-based lane line detection method. However, the performance of LaneNet relies on some post-processing parts of the network for pixel embedding and curve fitting, and thus is in turn largely affected by the LaneNet's segmentation part, which also cannot cope well with the challenging scenes of partial lane information above-mentioned for SCNN. In [28], it was suggested that a segmentation approach is not effective for detecting elongated thin lane boundaries, and thus the lane detection problem was posed as a CNN regression task to predict the coordinates of 15 points on each lane line. However, it is not practical to report only the points as the result of lane line prediction, and the number and size of points all affect the training of the network. Moreover, not only does the network need to find the location of the lane line, but also the locations of the points have to be predicted accurately, which introduces unnecessary extra complexity.

B. *Wasserstein GAN*

Adversarial training in the generative adversarial network (GAN) can be helpful to learn high-level images features of lines, such as line connectedness when some lane lines are only partially available or completely occluded, as well as for semantic segmentation [29]. However, it may cause the gradient of generator to vanish when a perfect discriminator exists. To solve this problem, the Wasserstein distance-based GAN (WGAN) was proposed in [30]. Soon after, WGAN-GP [31] was proposed to make the WGAN training more stable by putting Lipschitz constraints into the loss. A GAN-based work related to this paper is the EL-GAN in [32] for lane segmentation. In a similar nature to the original GAN, the EL-GAN is trained by using an "embedding loss" on the predictions and labels, while the detection performance was improved by some post-processing. Different from EL-GAN, we select WGAN-GP [31] and modify its structure for lane line detection. To fully exploit it, we design two discriminators, one for semantic segmentation into three targets (background, lanes, and lane lines), and the other for emphasizing the lane lines. Also, noise is added to the input for exploit the imagination (generative) ability of WGAN to deal with the missed lane lines.

III. The Proposed Method

In this section we present the details of our proposed RiLLD-Net and Ripple-GAN.

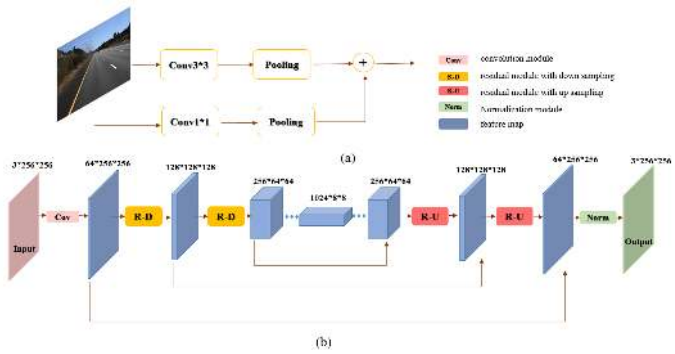


Fig. 2. (a) A residual module with pooling. (b) Overall structure of the RiLLD-Net: R-D refers to the residual module with downsampling; R-U refers to the residual module with upsampling; blue modules are feature maps.

A. *RiLLD-Net*

Inspired by the merits of U-Net [33] and deep residual learning [34] for image segmentation and recognition, we propose RiLLD-Net for lane line detection. The RiLLD-Net is a simple, new network blending the ideas of the encoder-decoder (context-localization) structure of U-Net, the residual module with skip connections, and quick connections between encoders and decoders. As illustrated in Fig. 2, connections between multiple feature layers are like the ripple in the network pond, hence we name the network RiLLD-Net.

In the RiLLD-Net, feature connections exist not only between the encoders and decoders via quick connections, but also within each of them via skip connections. The quick connection between a encoder and a decoder helps the decoder to fix well the details of the target lines. The skip connection within the residual module between adjacent feature layers allows the network to handle different features differently and learn deeper features of lane lines. RiLLD-Net uses skip connections on all sampling paths. This is different from only applying residual modules to the downsampling path to take the place of some convolution operations, or only applying to the bottleneck layer (the last level of downsampling path). This helps RiLLD-Net achieve efficient learning and full use of features. As for Ripple-GAN, we add Gaussian noise so that the support set of original data can be extended to the entire space. The skip connections spread the influence of Gaussian noise on the data distribution to the training process of the entire network. Therefore, GAN can be more efficiently and stably trained. That is, the design of the RiLLD-Net structure is conducive to a good performance of Ripple-GAN.

Fig. 2(a) shows the residual module of RiLLD-Net, and Fig. 2(b) represents the overall structure of RiLLD-Net, where the orange module labeled R-D refers to the residual module with downsampling and the pink module labeled R-U refers to the residual module with upsampling. We note that in RiLLD-Net, different from [35], we did not use the Dense block. Although high-density feature reuse can help the network process features, we prefer a simpler and more efficient network structure that facilitates real-time lane line detection with acceptable detection performance. Experiments in section IV show that our proposed network can fulfil this criterion.

PLEASE KEEP CONFIDENTIAL

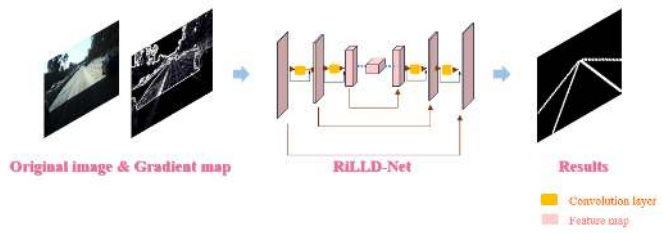


Fig. 3. The diagram of the RiLLD-Net for lane line detection. The input of this network includes the original image and a gradient map.

Using the RiLLD-Net, we develop a simple network for lane line detection, as shown in Fig. 3. The network is based on a 10-layer RiLLD-Net and includes both the original image and a gradient map as input. This is tailored for lane line detection as the gradient of the image can highlight the lane lines and suggest the directional characteristics of them. The gradient map is obtained by processing the original image with the Sobel operator. Using both the gradient and color information together facilitates the highlighting of lane lines and the removal of redundant interference information in the background. For simplicity and efficiency, here we only include MSE into the loss function of RiLLD-Net:

$$Loss_{mse} = E[(I_G - I_R)^2], \tag{1}$$

where I_G is the output image and I_R is the real labeled image.

B. Ripple-GAN

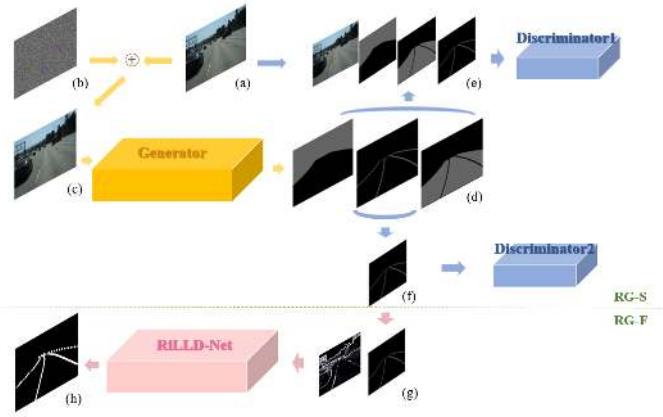


Fig. 4. The diagram of the Ripple-GAN for lane line detection. The part above the green line is RG-S: (a) Original image. (b) Gaussian noise. (c) Image with Gaussian noise added. (d) Three-target segmentation results. (e) Original image & segmented results. (f) Segmented lane line result. The part below the green line is RG-F: (g) Gradient map & segmented lane line result. (h) Final prediction result.

In experiments, we find that the RiLLD-Net can produce excellent results for lane line detection in most common scenes, but not outstanding under complex road conditions with only partial lane line information available. Therefore, to address this issue, we combine RiLLD-Net with generative adversarial learning to exploit the generative ability of the WGAN, proposing a new network termed Ripple-GAN. The diagram of Ripple-GAN for lane line detection is shown in Fig. 4, mainly including two parts: the RG-S network (segmentation part of Ripple-GAN) to produce the sketch of lane, and then the RG-F network (fine-tune part of Ripple-GAN) to fine tune the RG-S output.

1) RG-S part: A diagram of RG-S is shown above the green line in Fig. 4, including the following components.

Gaussian noise addition. We add white Gaussian noise to the original image to form the input of the network. In this way, the support set of input data distribution will be closer to the support set of Gaussian distribution, and we can better exploit the generative ability of the generator to complete defective or occluded lane lines. Also, this operation breaks block structures in the original image and help the following convolution layers to enhance elongated structures, making the lane lines more prominent in the obtained feature maps (see more discussion in Section IV-G).

Three-target segmentation. In RG-S, we design the generator to achieve three-target segmentation for background, lanes and lane lines. In traffic scenes, lane lines only account for a very small part of the whole picture, hence a method of directly predicting the lane line pixels is insufficient for accurate detection. Such lane lines could be easily ignored by a generator as their influence on the loss function is small, so lane lines have to be emphasized during the adversarial training. For this reason, we have added lanes and background into semantic segmentation in RG-S: Lane information can regulate the shapes of lane lines, and the background can narrow down the ROI areas of lane lines in the image and eliminate the interference from the background on lane line detection, as shown in Fig. 4(d).

Gradient map. In RG-F, we input the gradient map together with lane lines segmented by RG-S into the RiLLD-Net to predict the final results. In addition to emphasizing the lane line information, the gradient map in RG-F can provide supplementary scene information.

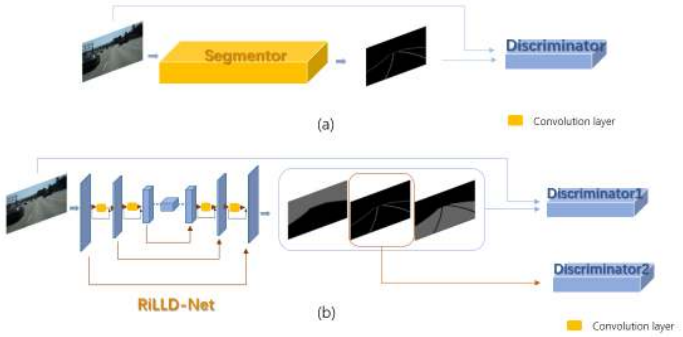


Fig. 5. (a) A traditional GAN-based segmentation network. (b) The proposed new RiLLD-Net-based GAN in RG-S.

RiLLD-Net-based GAN. Traditional GAN-based semantic segmentation networks treat the segmentor as the generator, as shown in Figure 5(a). The performance of such a network can be improved by jointly considering its generative, segmentation, and discriminative abilities. To this end, we propose a new RiLLD-Net-based GAN structure for the RG-S, as shown in Fig. 5(b). RG-S implements a RiLLD-Net-based generator to simultaneously predict the three types of targets, and then

designs two discriminators: the input of the first discriminator is the original training image and the three-target segmentation map; the input of the second discriminator is the generated binary map of lane lines. In this way, not only the relationship between the original scene and what the generator produced is strengthened by the first discriminator, but also the lane lines are highlighted by the second discriminator.

Loss function. The generator loss function of RG-S consists of two terms: one term called L_{low} measures the fitness between the low-level pixel-wise prediction and the label; the other term called L_{high} preserves the higher-level consistency conditioned on the input image and binary map:

$$L_G(x,y;\theta_G,\theta_{D_1},\theta_{D_2}) = \lambda_1 L_{low}(G(x;\theta_G),y) + \lambda_2 L_{high}(x,G(x;\theta_G);\theta_{D_1},\theta_{D_2}), \quad (2)$$

where x and y are the input image and the true segmentation map; θ_G, θ_{D_1} and θ_{D_2} are the parameters of the generator and two discriminators; $G(x;\theta_G)$ represents a transformation on the input image x, imposed by the generator network parameterized by θ_G; and λ_1 and λ_2 are the relative weights of the two terms.

The loss term L_{low} contained two parts, the mean square error (MSE) and the Dice distance, as shown in (4):

$$Dice(G(x;\theta_G),y) = \frac{2|G(x;\theta_G)\cap y|}{|G(x;\theta_G)|+|y|}, \quad (3)$$

$$L_{low}(G(x;\theta_G),y) = \beta_1||G(x;\theta_G)-y|| + \beta_2(1-Dice(G(x;\theta_G),y)), \quad (4)$$

where β_1 and β_2 are the relative weights. Minimizing MSE can reduce the Euclidean distance between the real data and the fake data generated by generator. The Dice distance, as shown in (3), is commonly used in semantic segmentation, measuring the similarity between two data sets, the prediction results $G(x;\theta_G)$ and the real data set y.

The loss term L_{high} is to make the data produced by the generator more similar in distribution to the true data. It is often formulated by a binary cross entropy loss between zero and the binary prediction results of the discriminator. However, the gradient of the generator would be explosive if the performance of discriminator is sufficiently good. Hence we use the Wasserstein loss to promote the network convergence and remove the sigmoid layer from the discriminators:

$$L_{high} = -E[D_1(x,G(x;\theta_G);\theta_{D_1})] - E[D_2(\widetilde{x};\theta_{D_2})], \quad (5)$$

where $D_1(x,G(x;\theta_G);\theta_{D_1})$ is the first discriminator, which compares the segmented three-target images of the faked and real images; and $D_2(\widetilde{x};\theta_{D_2})$ is the second discriminator, which focuses on distinguishing the forged lane lines $\widetilde{x}$ and the binary map of true lane lines $\widetilde{y}$. Moreover, during the training process, RG-S benefits from WGAN-GP by applying weight clipping to the loss term, which revises the loss terms of the two discriminators to be

$$L_{D_1} = -E[D_1(x,y;\theta_{D_1})] + E[D_1(x,G(x;\theta_G);\theta_{D_1})] + \gamma_1 E[||\bigtriangledown_\zeta D_1(\zeta;\theta_{D_1})||_p - 1]^2, \quad (6)$$

$$\zeta = (x,y) + \alpha_1[(x,G(x;\theta_G)) - (x,y)], \quad (7)$$

where (x,y) is a concatenation of x and y, and so does $(x,G(x;\theta_G))$; and

$$L_{D_2} = -E[D_2(\widetilde{y};\theta_{D_2})] + E[D_2(\widetilde{x};\theta_{D_2})] + \gamma_2 E[||\bigtriangledown_\eta D_2(\eta;\theta_{D_2})||_p - 1]^2, \quad (8)$$

$$\eta = \widetilde{y} + \alpha_2(\widetilde{x} - \widetilde{y}), \quad (9)$$

where α is a random number between 0 and 1, and p refers to the p-norm.

2) RG-F part: In the RG-S network we add Gaussian noise into the original image, such that the generator can complete partial lane lines in the scenario. However, during the training process, the imagination of the generator may produce the deviated prediction results. Hence, we design a simple fine-tune part in Ripple-GAN called RG-F, to refine the prediction results of lane lines. RG-F is shown below the green line in Fig. 4. It consists of a simple shallow RiLLD-Net. The input of RG-F includes the lane lines obtained from three-target segmentation results by RG-S. In the meantime, there should also be some information from the original scene to guide RG-F to refine the lane lines. Hence, we use both the gradient maps of original images and the extracted lane lines as the input part of RG-F. The loss function of RG-F is just the mean square error (MSE) as in (1) for simplicity. Because of the simple structure and complexity of RG-F, the overall complexity of the Ripple-GAN has not been increased much from using a two-stage network structure.

In summary, with gradient maps, the RiLLD-Net has good performance in lane line detection for most common scenes, and by further exploiting the strength of adversarial learning, the Ripple-GAN is able to handle more challenging situations in lane line detection, for example, when the lane line is damaged or obstructed. In the next section we shall verify the superiority of Ripple-GAN.

IV. Experimental Results and Analysis

In this section, we first introduce the TuSimple dataset and the training setup used in the experiments. Then we investigate the effects of various components (three-target prediction, Gaussian noise addition and gradient map) on the performance of the Ripple-GAN. Finally, we compare our Ripple-GAN with other state-of-the-art methods.

A. Datasets and experimental settings

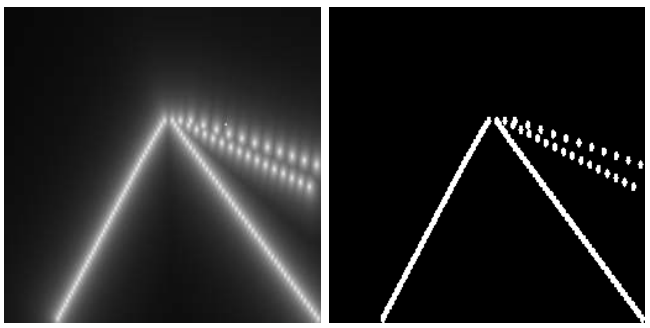


Fig. 6. A generated heatmap (left) and its binarized version (right).

1) TuSimple dataset: Up to present, there is few lane line detection dataset for deep learning methods. The TuSimple lane dataset[1] is a large scale dataset published for a lane detection challenge. There are series of images of size 720×1280 taken on the US highway, under medium to good weather conditions during daytime. The training set consists of over four sequences divided into many different sections. Each section contains twenty frames and the last frame is given the annotation. Totally, 3262 training images and 2782 test images are annotated. From the train set, 604 images were used as a validation set. The annotation indicating x-position of the lane lines at a number of discretized y-position comes in a json format and records 2-lane to 5-lane in different scenarios. To generate the labels for our proposed network, we use the given points to generate a heatmap of the lane line according to the distance between the pixels. This strategy eliminates the fitting process of the lane line and can directly generate the lane line labels by binarization, as shown in Fig. 6.

2) Architecture and training setup: All the input images are scaled to a resolution of 256×256 and normalized.

In the RiLLD-Net, the depth is 10. It is trained by using Adam with a batch size 16 and a learning rate 1e-4 until convergence. Its architecture is detailed in Table I.

In the Ripple-GAN, the two parts, RG-S and RG-F, are trained in succession. The RG-S network is trained first, and then the RG-F network is trained by using the training set and the validation set tested by the RG-S network. Both parts are trained with a RiLLD-Net with depth of ten. The batch size is 8 for RG-S and 16 for RG-F. The two parts are all trained by using Adam at the 1e-4 learning rate until convergence. In (2), λ_1 and λ_2 are set to 1; in (4), β_1 and β_2 are set to 100; and in (6) and (8), γ_1 and γ_2 are set to 10.

TABLE I
ARCHITECTURE IN DETAIL OF RILLD-NET.

Instruction	Layer name	Size of feature map
Input	Generator. Input	(batch size, 64,256,256)
Downsample	Generator. Res1	(batch size, 128,128,128)
	Generator. Res2	(batch size, 256,64,64)
	Generator. Res3	(batch size, 512,32,32)
	Generator. Res4	(batch size, 1024,16,16)
	Generator. Res5	(batch size, 1024,8,8)
Upsample	Generator. Res6	(batch size, 1024,16,16)
	Generator. Res7	(batch size, 512,32,32)
	Generator. Res8	(batch size, 256,64,64)
	Generator. Res9	(batch size, 128,128,128)
	Generator. Res10	(batch size, 64,256,256)
Normalize	Generator. OutputN	(batch size, 128,256,256)
Output	Generator. Output	(batch size, 3,256,256)

B. Analysis of three-target prediction

Different from other segmentation-based lane line detection methods, our Ripple-GAN predicts the lane line, lane and background simultaneously. In this section, we explore how the three segmentation tasks jointly improve the performance of each other.

[1]https://github.com/TuSimple/tusimple-benchmark/issues/3

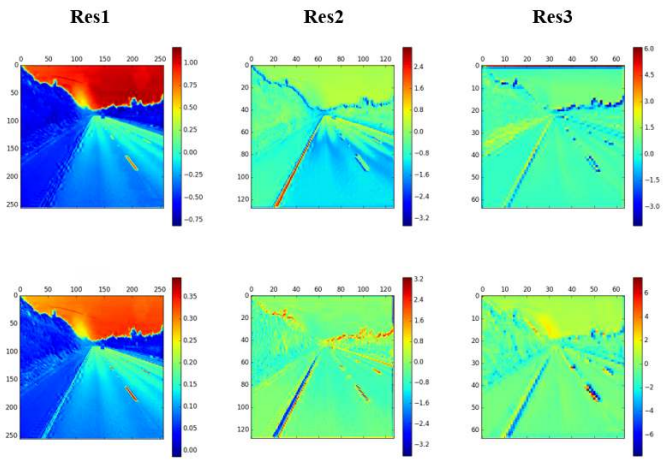


Fig. 7. Feature maps at three residual modules for one-target segmentation (upper row) or three-target segmentation (lower row). In order to display the maps clearly, no noise is added to the input here.

The feature maps of first three residual modules of RG-S are shown in pseudo color in Fig. 7, with the upper row obtained from the one-target (line) segmentation and the lower row for the three-target (background, lane, line) segmentation. It can be seen that the left and right boundaries of the road in the lower row are clearer than those in the upper row, so do the lane lines on the road. There is also a tendency in the three-target segmentation to predict the position of the straight lane line through the dotted lane line. These indicates that adding the background and lanes as the segmentation targets has a positive effect on the lane line detection. The segmentation of background can make the foreground (i.e. the area where the lane line and the lane are located) and the background separate better, and the segmentation of lanes can make the lane line segmentation more accurate. The performances of Ripple-GAN in the cases of three-target segmentation and one-target segmentation are shown in Table II. The amount of Gaussian noise is consistent in both cases.

TABLE II
PERFORMANCE OF RIPPLE-GAN WITH THREE-TARGET SEGMENTATION AND ONE-TARGET SEGMENTATION.

Segmentation	Recall	FN	FP
one-target	0.9677	0.0407	0.0049
three-target	**0.9728**	**0.0289**	**0.0048**

C. Analysis of Gaussian noise addition

In the input part of RG-S, we add Gaussian noise into the original image. Now we shall verify that the addition of Gaussian noise allows the network to fully exploit the imagination ability of GAN to address detection tasks in unfamiliar complex scenes, as well as a positive effect of Gaussian noise on improving the convergence of the network.

1) Influence of Gaussian noise on the network performance: Fig. 8 shows the feature maps of the first three residual modules of RG-S, in the case with or without Gaussian noise added. It can be seen from the feature maps that: with

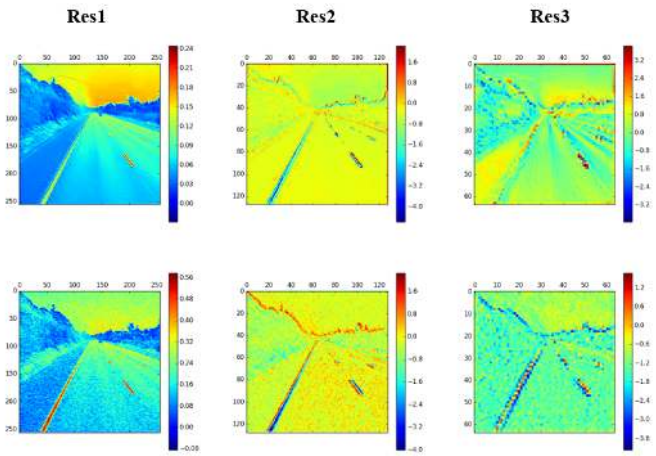


Fig. 8. Feature maps at residual modules without Gaussian noise (upper row) or with Gaussian noise added (lower row).

Gaussian noise added in the input, the feature maps become more blurring after convolutions, and in the meantime the lane lines become more significant, compared with the case of no Gaussian noise added. This indicates that the Gaussian noise addition can help the network quickly learn the characteristics of the lane lines.

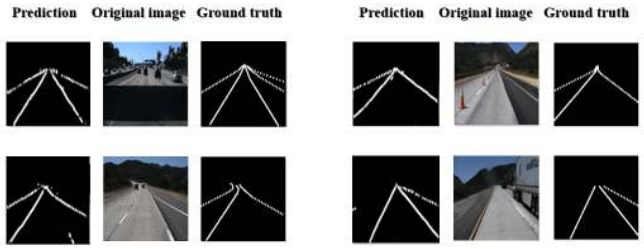


Fig. 9. Prediction results in four typical challenging scenarios: light changes, textures similar to lane lines on the road, special lane line distribution, and dynamic background occlusion.

We also selected from the test set some complex scenes: scenes with missing lane line information, scenes with complex lane lines, and scenes with lane-line-like texture interference on the road. We tested the performance of Ripple-GAN with Gaussian noise in these challenging scenarios, and some results without any post-processing are in shown in Fig. 9. As can be seen, the Ripple-GAN with Gaussian noise can cope well with occlusion or complex road conditions, and help eliminate interference caused by different road textures.

To quantitatively evaluate the effect of Gaussian noise addition on the performance of Ripple-GAN, we selected 185 representative challenging scenes (e.g. large-area occlusion, complicated road conditions, and different pavement textures) from the test set to form a challenging test set. The Ripple-GANs with Gaussian noise of different standard deviations were tested on this data set. The test results are listed in Table III, where standard deviation equal to zero denotes the case without Gaussian noise. The results show that the performance of Ripple-GAN with Gaussian noise is better than

TABLE III
PERFORMANCE OF RIPPLE-GAN WITH GAUSSIAN NOISE OF DIFFERENT STANDARD DEVIATION ON A CHALLENGING TEST SET. THE STANDARD DEVIATION EQUAL TO 0 INDICATES THAT GAUSSIAN NOISE IS NOT INVOLVED IN THE TRAINING PROCESS.

Standard deviation	Recall	FN	FP
0.0	0.9478	0.0683	0.0103
0.1	**0.9569**	0.0599	**0.0082**
0.3	0.9502	0.0748	0.0105
0.5	0.9438	0.0838	0.0104
0.7	0.9386	0.1203	0.0160
1.0	0.9518	**0.0548**	0.0141

TABLE IV
PERFORMANCE OF RIPPLE-GAN WITH GAUSSIAN NOISE OF FINE-GRAINED STANDARD DEVIATION ON A CHALLENGING TEST SET.

Standard deviation	Recall	FN	FP
0.01	0.9613	0.0464	**0.0061**
0.03	0.9631	0.0532	0.0085
0.07	**0.9665**	**0.0332**	0.0084
0.08	0.9617	0.0459	0.0062
0.09	0.9631	0.0532	0.0085
0.10	0.9569	0.0599	0.0082
0.11	0.9592	0.0586	0.0089
0.12	0.9631	0.0396	0.0105
0.13	0.9564	0.0599	0.0113

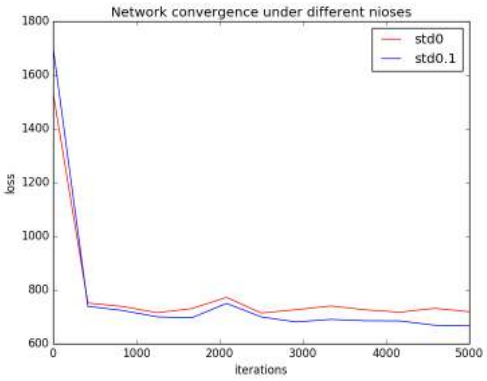
that of Ripple-GAN without Gaussian noise on the challenging test set. Gaussian noise with a standard deviation of 0.1 makes the biggest improvement of network performance in terms of recall. To further explore the relationship between the Gaussian noise addition and the network performance, we fine-grained the interval between the standard deviations of Gaussian noise around 0.1. The fine-grained test results are listed in Table IV. It can be seen that we can even obtain better test results around the noise standard deviation of 0.1.

2) Influence of Gaussian noise on the network training efficiency: From the experiments, we found that adding Gaussian noise not only improves the ability of the Ripple-GAN to detect lane lines, but also helps to speed up the network convergence. As analyzed in Section III-B1, the addition of Gaussian noise can make the network be more stably trained, and thus accelerate network convergence. The convergences (loss curves) of the network, under Gaussian noise with five standard deviations (std) from 0 to 1 are tested. The loss function values of the first 13 epochs were recorded.

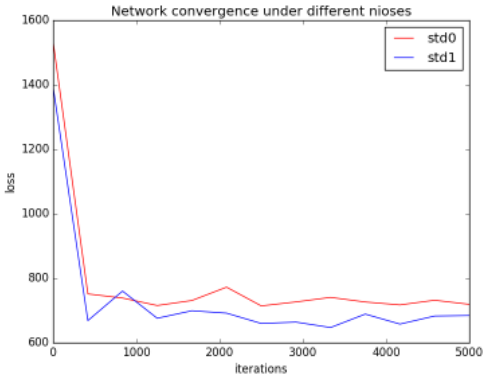
For illustration, in Fig. 10 we plot the change in loss after adding Gaussian noise (with standard deviation 0.1 vs without noise, or with standard deviation 1 vs without noise). It can be seen that, although due to the random initialization of the parameters these loss curves have different initial values, all losses have a big drop after the first epoch, and then the networks with Gaussian noise added have generally lower loss than the networks without Gaussian noise added.

D. Analysis of gradient map

Adding the gradient map into the input of the network helps supplement the input with the scene information and highlight the texture. In order to verify the effectiveness of the gradient map on improving the network performance, we compared the



(a) Loss under no noise (std0) or standard deviation 0.1 (std0.1)



(b) Loss under no noise (std0) or standard deviation 1 (std1)

Fig. 10. The convergence of the network under Gaussian noise with different standard deviations (std).

performances of Ripple-GAN between the situations of input with and without the gradient map. The experimental results are listed in the Table V. The amount of Gaussian noise is consistent in both cases.

TABLE V
RESULTS OF GRADIENT MAP ABLATION EXPERIMENT.

Gradient map	Recall	Precision	F_1
no	0.9669	0.9801	0.9734
yes	**0.9728**	**0.9806**	**0.9767**

We can observe that, after the gradient map was added, all the prediction recall, precision and F_1 score (a function of recall and precision) of Ripple-GAN have increased. That is, the gradient map has made a certain contribution to the improvement of network performance. More broadly speaking, the addition of manually-crafted features can help train neural networks more effectively.

E. Evaluation on the TuSimple database

Now we introduce some measures for lane line detection and compare our proposed two networks with the outstanding methods in the TuSimple Lane Line competition.

1) Metrics: In the Tusimple lane detection competition, the predicted lane line position is compared with 56 points of the true values to calculate the accuracy, the false positive rate (FP) and the false negative rate (FN). As the proportion of the pixels occupied by the lane line is a minority, in order to highlight the effect of lane line detection, the denominator in accuracy contains only the true values of all lane lines, and the numerator is the correctly predicted lane line points. That is, the accuracy here is actually the recall. In order to express accurately and avoid confusion, here we use the term 'recall'. Precision is also used as one of the evaluation measures.

Recall. The recall of each lane line is defined as

$$recall = \frac{TP}{ground\ truth} = \frac{TP}{FN + TP}, \tag{10}$$

where TP and FN are pixel counts in the true positive and false negative regions, respectively.

Precision. The precision of each lane line is defined as

$$precision = \frac{TP}{FP + TP}, \tag{11}$$

where TP and FP are pixel counts in the true positive and false positive regions, respectively.

MIoU. The mean intersection over union (MIoU) is a standard measure to calculate the ratio of intersections and unions between the set of ground truth and the set of predicted values. The MIoU metric for a single test image is defined as

$$MIoU = \frac{1}{k}\sum_{i=0}^{k-1}\frac{TP_i}{FN_i + FP_i + TP_i}, \tag{12}$$

where k is number of classes and in our case $k = 2$ representing lane lines and background.

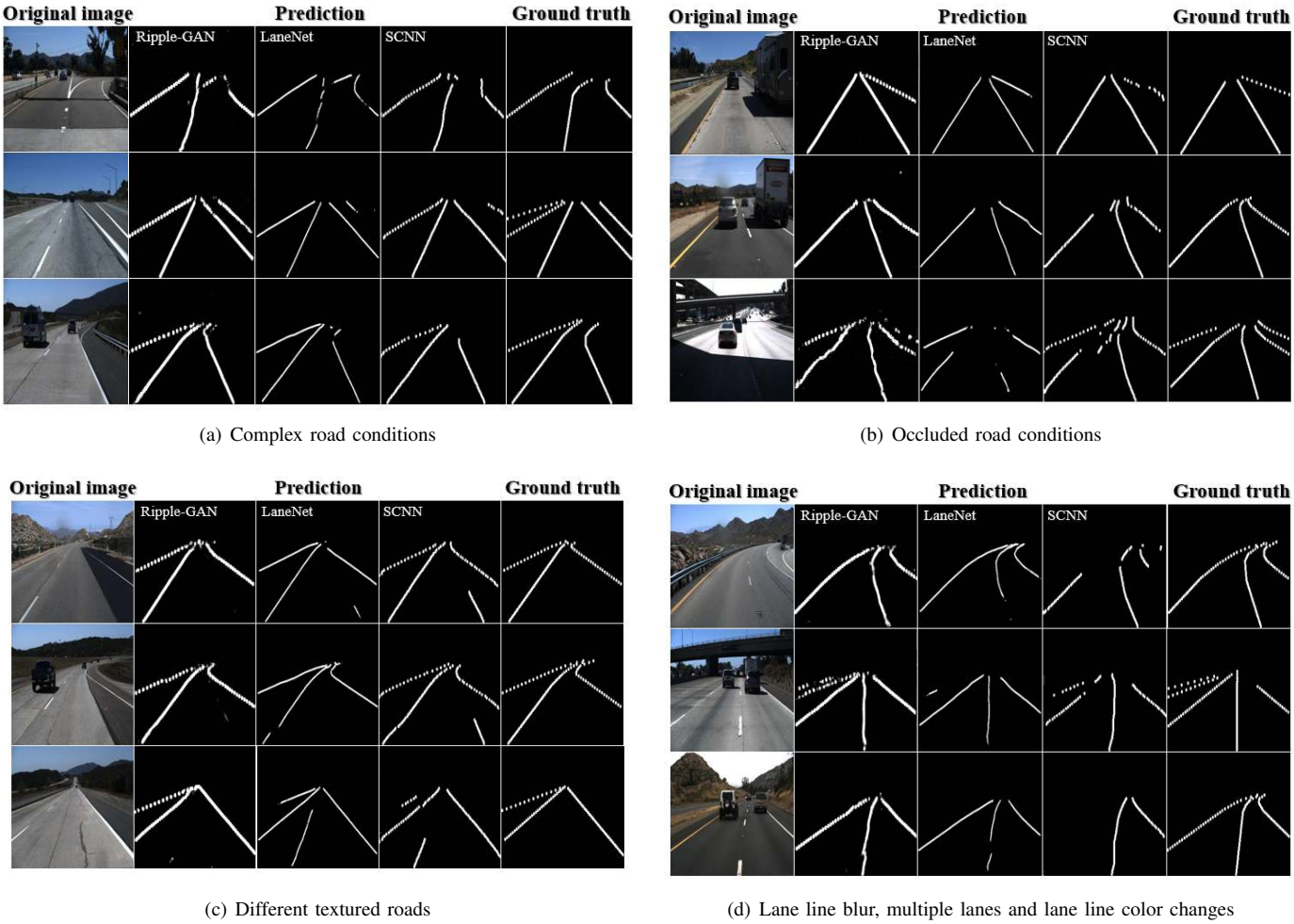
F_1 scores. F scores is an indicator used in statistics to measure the accuracy of a binary model. It simultaneously takes into account the precision and recall. F_1 score can be regarded as a harmonic average of the precision and recall, with a maximum of 1 and a minimum of 0. It is defined as

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}. \tag{13}$$

2) Comparison with state-of-the-art methods: To further verify the effectiveness of our proposed Ripple-GAN, we compare it with several state-of-the-art methods: SCNN, EL-GAN and LaneNet. In order to maintain consistency in the comparison, we perform a simple post-processing of the results: a number of key points are extracted from each lane line as representative. The comparison results is listed in Table VI, in terms of recall, FN, FP, precision and F_1 score.

TABLE VI
PERFORMANCE ON THE TUSIMPLE DATABASE. TOP TWO RESULTS ARE HIGHLIGHTED IN BOLDFACE.

Models	Recall	FN	FP	Precision	F_1
SCNN [26]	**0.9653**	**0.0180**	0.0617	0.7964	0.8727
EL-GAN [32]	0.9639	0.0336	0.0412	0.8540	0.9056
LaneNet [27]	0.9638	**0.0244**	0.0780	0.7554	0.8470
RiLLD-Net	0.9513	0.0502	0.0189	0.9264	0.9386
Ripple-GAN no noise	0.9652	0.0301	**0.0123**	**0.9515**	**0.9538**
Ripple-GAN with noise	**0.9728**	0.0289	**0.0048**	**0.9806**	**0.9767**



(a) Complex road conditions

(b) Occluded road conditions

(c) Different textured roads

(d) Lane line blur, multiple lanes and lane line color changes

Fig. 11. Comparison of lane line detection results by different methods in challenging scenarios.

As can be seen from the Table VI, the recall of RiLLD-Net is not as high as EL-GAN and LaneNet, but it has smaller FN and FP, which makes the F_1 score of RiLLD-Net higher than EL-GAN and LaneNet. As adversarial training in Ripple-GAN improves RiLLD-Net by being more strict on whether a pixel is detected as on a lane line, the recall of Ripple-GAN is increased, the FN and FP of Ripple-GAN is greatly reduced, and thus the precision and F_1 score of Ripple-GAN exceed the other three state-of-the-art methods. Moreover, after adding Gaussian noise to the input of Ripple-GAN, the network can better deal with complex and obstructed road conditions, hence overall the Ripple-GAN with noise performs the best and achieves a very high F_1 score. Examples of lane line detection results by different methods in different challenging scenarios are shown in Fig. 11.

We also compare the difference between the binary map of ground truth and the predicted result at the pixel level, and calculate the precision, recall and MIoU, as shown in Fig. 12 for the RiLLD-Net, the Ripple-GAN without noise and the Ripple-GAN with noise. We can observe that the three networks have similar precisions at the pixel level. However, both the RiLLD-Net and the Ripple-GAN predict lane lines with high recall and MIoU, and the Ripple-GAN with Gaussian noise remains the best performer.

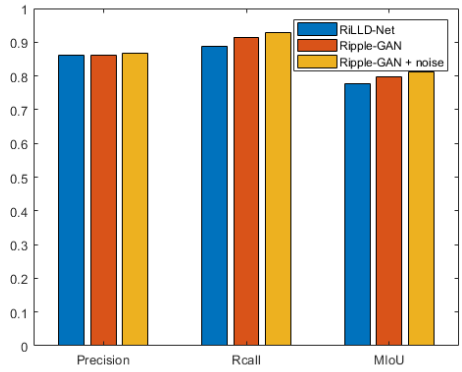


Fig. 12. Precision, recall and MIoU of test results of the proposed methods.

F. Algorithm complexity

1) Computational efficiency: The computational efficiencies of RiLLD-Net and Ripple-GAN are listed in the Table VII.

Compared with Ripple-GAN, RiLLD-Net is of simpler structure, and thus its detection is faster. Although its detection

TABLE VII
COMPUTATIONAL EFFICIENCIES OF RILLD-NET AND RIPPLE-GAN

Models	Training time / epoch	Test time / frame	Recall
RiLLD-Net	187.5s (4GPU)	0.2s (1GPU)	95.13
Ripple-GAN	566.7s (4GPU)	0.5s (1GPU)	97.28

recall is 95.13%, not as high as that of Ripple-GAN (97.28%), RiLLD-Net can perform rapid detection while ensuring a good accuracy when a vehicle drives at high speeds under relatively simple road conditions. The driving will usually slow down when the road conditions are complicated, and in this case it is appropriate to choose Ripple-GAN, which is good at handling complicated road conditions.

2) Space complexity: The Ripple-GAN algorithm consists of two parts, RG-S and RG-F, with an overall parameter size of 426.5M. The numbers of weights for RG-S and RG-F are 234.7M and 191.8M, respectively. RG-F and RiLLD-Net are equal in the space complexity.

G. Discussion

In this subsection, we discuss why our algorithm can be superior to other lane line detection algorithms, from the following aspects. Firstly, the RiLLD-Net with multi-layer feature interaction enables the network to learn the effective and comprehensive characteristics of the target in a timely manner, thereby improving the accuracy of the detection.

Secondly, adding the scene gradient feature map is equivalent to enhancing the importance of this feature in the neural network. This strategy can be applied to a variety of detection tasks, if such a manually extracted feature holds a strong correlation with the detection target.

Thirdly, multi-task detection is an approach to effectively improving the accuracy of lane detection. In a scene, the proportion of the lane line is very small. Since the loss function is calculated over the whole graph, even if the entire lane line is detected incorrectly, the contribution to the loss function will not be large. This makes the network fail to achieve the expected segmentation of lane lines even if it converges. Hence we added the background and lane, which are associated with the lane line, as the segmentation targets. These two targets have much larger proportions in the scene and thus are relatively easier targets of segmentation. The correct segmentation of the background can reduce the pixels outside the road that are misclassified as lane lines; the correct segmentation of the lane can further ensure the segmentation of the lane lines. In short, the correct segmentation of the surrounding pixels can assist the small target detection.

Finally, it can be seen from the feature map of convolution layer that adding Gaussian noise can enhance the texture of the strip in the picture, make the area with homogeneous structure such as sky and road less structural and be discarded in the subsequent processing of the network. Thus the linear structure can be highlighted. As seen from the second layer feature map in Fig. 8, the outline of the background mountain and the lane line are enhanced by the addition of noise into the network input. Later, deeper networks capable of learning more advanced features can eliminate the effects of noise on the background and filter out contours other than lane lines.

The original GAN uses the Jensen-Shannon (JS) divergence as the optimization objective, but when there exists a perfect discriminator, the discriminator gradient disappears and the update of the generator cannot get enough gradient, resulting in poor performance of the generator [30]. WGAN uses the Wasserstein distance to solve the shortcomings of the JS divergence, but there are still cases where the training is unstable and the network is not easy to converge.

To fix the instability issue, [36] suggests adding continuous noise to the inputs of the discriminator to smoothen the data distribution. It says that noisy samples can be guided to the direction of the real data manifold during the training. Inspired by this, in Ripple-GAN, we add moderate noise directly to the original data distribution. This operation can extend the support set of original data distribution to the entire space, ensuring that the new data distribution and the target data distribution have a shared support set. Experiments show that the network performance is indeed improved after adding Gaussian noise (see Section IV-C).

V. CONCLUSION

In this paper, we proposed an effective lane line detection method called Ripple-GAN. We first proposed RiLLD-Net, a simpler and basic network structure of Ripple-GAN. The RiLLD-Net can learn features efficiently, and enhancing RiLLD-Net with gradient features from the scene can make RiLLD-Net an efficient and decent detector in most traffic scenarios. To cope with challenging situations of complex, incomplete or obscured lane lines, we blended RiLLD-Net with the idea of WGAN to develop Ripple-GAN. In Ripple-GAN, the generator is a multi-target segmentation network and Gaussian noise is added into the input of the network, equipping Ripple-GAN with the ability to cope with detection tasks in more challenging road conditions. The proposed Ripple-GAN has achieved satisfactory results on the TuSimple dataset, and its F_1 score is higher than those of the existing methods. We also verified the positive impacts of multi-tasking and noise on lane line detection by experiments.

It is a bigger challenge to detect lane line when the road surface is completely or partly occluded, for example, a street surface with dim lights. This is a direction of our future work.

REFERENCES

[1] S. P. Narote, P. N. Bhujbal, A. S. Narote, and D. M. Dhane, "A review of recent advances in lane detection and departure warning system," *Pattern Recognition*, vol. 73, pp. 216–234, 2018.
[2] J.-G. Wang, C.-J. Lin, and S.-M. Chen, "Applying fuzzy method to vision-based lane detection and departure warning system," *Expert Systems with Applications*, vol. 37, no. 1, pp. 113–126, 2010.
[3] D. C. Andrade, F. Bueno, F. R. Franco, R. A. Silva, J. H. Z. Neme, E. Margraf, W. T. Omoto, F. A. Farinelli, A. M. Tusset, S. Okida, M. M. D. Santos, A. Ventura, S. Carvalho, and R. d. S. Amaral, "A novel strategy for road lane detection and tracking based on a vehicles forward monocular camera," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1497–1507, 2019.
[4] V. Gaikwad and S. Lokhande, "Lane departure identification for advanced driver assistance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 910–918, 2015.

PLEASE KEEP CONFIDENTIAL

[5] C. Lee and J. Moon, "Robust lane detection and tracking for real-time applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 12, pp. 4043–4048, 2018.
[6] H. Xu and H. Li, "Study on a robust approach of lane departure warning algrithm," in *International Conference on Signal Processing Systems*, vol. 2. IEEE, 2010, pp. V2–201.
[7] X. Wang, Y. Wang, and C. Wen, "Robust lane detection based on gradient-pairs constraint," in *Proceedings of the 30th Chinese Control Conference*, 2011, pp. 3181–3185.
[8] Q. Lin, Y. Han, and H. Hahn, "Real-time lane departure detection based on extended edge-linking algorithm," in *Second International Conference on Computer Research and Development*, 2010, pp. 725–730.
[9] D.-C. Tseng and C.-W. Lin, "Versatile lane departure warning using 3D visual geometry," *International Journal of Innovative Computing, Information and Control*, vol. 9, no. 5, pp. 1899–1917, 2013.
[10] K.-Y. Chiu and S.-F. Lin, "Lane detection using color-based segmentation," in *IEEE Intelligent Vehicles Symposium*, 2005, pp. 706–711.
[11] C. Jung and C. Kelber, "Lane following and lane departure using a linear-parabolic model," *Image and Vision Computing*, vol. 23, no. 13, pp. 1192–1202, 2005.
[12] Y. Wang, D. Shen, and E. K. Teoh, "Lane detection using spline model," *Pattern Recognition Letters*, vol. 21, no. 8, pp. 677–689, 2000.
[13] H. f. Wang, Y. f. Wang, X. Zhao, G. p. Wang, H. Huang, and J. Zhang, "Lane detection of curving road for structural high-way with straight-curve model on vision," *IEEE Transactions on Vehicular Technology*, 2019.
[14] Y. Su, Y. Zhang, T. Lu, J. Yang, and H. Kong, "Vanishing point constrained lane detection with a stereo camera," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2739–2744, 2018.
[15] U. Ozgunalp, R. Fan, X. Ai, and N. Dahnoun, "Multiple lane detection algorithm based on novel dense vanishing point estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 3, pp. 621–632, 2017.
[16] W. Wang, D. Zhao, W. Han, and J. Xi, "A learning-based approach for lane departure warning systems with a personalized driver model," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 10, pp. 9145–9157, 2018.
[17] S. Lee, J. Kim, J. Shin Yoon, S. Shin, O. Bailo, N. Kim, T.-H. Lee, H. Seok Hong, S.-H. Han, and I. So Kweon, "VPGNet: Vanishing point guided network for lane and road marking detection and recognition," in *ICCV*, 2017.
[18] V. Mistry and R. Makwana, "Survey: Vision based road detection techniques," *International Journal of Computer Science and Information Technologies*, vol. 5, pp. 4741–4747, 2014.
[19] P.-C. Wu, C.-Y. Chang, and C. H. Lin, "Lane-mark extraction for automobiles under complex conditions," *Pattern Recognition*, vol. 47, no. 8, pp. 2756 – 2767, 2014.
[20] Y. Huang, S. Chen, Y. Chen, Z. Jian, and N. Zheng, "Spatial-temproal based lane detection using deep learning," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, 2018, pp. 143–154.
[21] Z. Wang, W. Ren, and Q. Qiu, "LaneNet: Real-time lane detection networks for autonomous driving," *arXiv:1807.01726*, 2018.
[22] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, "Robust lane detection from continuous driving scenes using deep neural networks," *arXiv:1903.02193*, 2019.
[23] B. He, R. Ai, Y. Yan, and X. Lang, "Accurate and robust lane detection based on dual-view convolutional neutral network," in *IEEE Intelligent Vehicles Symposium*, 2016, pp. 1041–1046.
[24] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.
[25] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.
[26] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial CNN for traffic scene understanding," in *AAAI*, 2018.
[27] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans, and L. V. Gool, "Towards end-to-end lane detection: an instance segmentation approach," in *IEEE Intelligent Vehicles Symposium*, 2018, pp. 286–291.
[28] S. Chougule, N. Koznek, A. Ismail, G. Adam, V. Narayan, and M. Schulze, "Reliable multilane detection and classification by utilizing CNN as a regression network," in *ECCV Workshops*, 2018.
[29] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," *arXiv:1611.08408*, 2016.
[30] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv:1701.07875*, 2017.
[31] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of Wasserstein GANs," in *Advances in Neural Information Processing Systems*, 2017, pp. 5767–5777.
[32] M. Ghafoorian, C. Nugteren, N. Baka, O. Booij, and M. Hofmann, "EL-GAN: Embedding loss driven generative adversarial networks for lane detection," in *ECCV Workshops*, 2018.
[33] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015, pp. 234–241.
[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.
[35] S. Jegou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *CVPR Workshops*, 2017.
[36] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv:1701.04862*, 2017.

Youcheng Zhang received the B.Eng. degree in communication engineering from Beijing Institute of Technology in 2017. She is currently working towards the Ph.D. degree in the Department of Electronic Engineering, Tsinghua University. Her research interests include image processing, pattern recognition and artificial intelligence.

Zongqing Lu received the Ph.D. degree in signal processing from Xidian University in 2007. He is an Assistant Professor in the Department of Electronic Engineering, Tsinghua University. His research interests include image processing and machine learning.

Dongdong Ma received the M.S. degree in control theory and control engineering from Xi'an Jiaotong University in 2015. He is currently pursuing the Ph.D. degree in the Department of Electronic Engineering, Tsinghua University. His research interests include medical and polarization image analysis.

Jing-Hao Xue received the Dr.Eng. degree in signal and information processing from Tsinghua University in 1998 and the Ph.D. degree in statistics from the University of Glasgow in 2008. He is an Associate Professor in the Department of Statistical Science, University College London. His research interests include statistical classification, high-dimensional data analysis, pattern recognition and image analysis.

Qingmin Liao received the Ph.D. degree in signal processing and telecommunications from the University of Rennes 1 in 1994. He is a Professor in the Department of Electronic Engineering and the Graduate School at Shenzhen, Tsinghua University. His research interests include image/video processing, transmission, analysis, biometrics and their applications.