Computer Modeling in Engineering & Sciences CMES, vol.122, no.3, pp.1039-1053, 2020

A Lane Detection Method Based on Semantic Segmentation

Ling Ding[1, 2], Huyin Zhang[1, *], Jinsheng Xiao[3, *], Cheng Shu[3] and Shejie Lu[2]

Abstract: This paper proposes a novel method of lane detection, which adopts VGG16 as the basis of convolutional neural network to extract lane line features by cavity convolution, wherein the lane lines are divided into dotted lines and solid lines. Expanding the field of experience through hollow convolution, the full connection layer of the network is discarded, the last largest pooling layer of the VGG16 network is removed, and the processing of the last three convolution layers is replaced by hole convolution. At the same time, CNN adopts the encoder and decoder structure mode, and uses the index function of the maximum pooling layer in the decoder part to upsample the encoder in a counter-pooling manner, realizing semantic segmentation. And combined with the instance segmentation, and finally through the fitting to achieve the detection of the lane line. In addition, the currently disclosed lane line data sets are relatively small, and there is no distinction between lane solid lines and dashed lines. To this end, our work made a lane line data set for the lane virtual and real identification, and based on the proposed algorithm effective verification of the data set achieved by the increased segmentation. The final test shows that the proposed method has a good balance between lane detection speed and accuracy, which has good robustness.

Keywords: CNN, VGG16, semantic segmentation, instance segmentation, lane detection.

1 Introduction

The installation of the advanced driver assistance system [He, Shan and Song (2018)] (ADAS) with lane detection function on the vehicle not only prompts the driver to shift the lane in time, but also alerts the nearby vehicle to avoid in time. To a certain extent, accidents can avoid. The advanced driver assistance system is designed to help drivers work safely. It has many functions such as adaptive cruise control, blind spot detection, collision avoidance, and traffic sign detection. In addition, the system also includes the functions of lane line detection and lane departure monitoring. Modern cars combined with ADAS are becoming more standard. With the rapid development of driverless cars,

[1] School of Computer Science, Wuhan University, Wuhan, 430072, China.

[2] College of Computer Science and Technology, Hubei University of Science and Technology, Xianning, 437100, China.

[3] School of Electronic Information, Wuhan University, Wuhan, 430072, China.

[*] Corresponding Author: Huyin Zhang. Email: zhy2536@whu.edu.cn;

Jinsheng Xiao. Email: xiaojs@whu.edu.cn.

Received: 11 August 2019; Accepted: 12 January 2020.

CMES. doi: 10.32604/cmes.2020.08268 www.techscience.com/journal/CMES

the automatic lane keeping function in ADAS plays an increasingly important role, and the car correctly positioned to travel. In the lane, it provides an important basis for subsequent lane departure and trajectory planning decisions.

This paper proposes a robust lane detection method based on the current mainstream deep learning algorithm. This method mainly uses convolutional neural network, in which the encoder and decoder structure are added. Based on the existing algorithms, this paper improves the Encoder and Decoder parts respectively. The Encoder discards the fully connected layer of the VGG16 network and the last 2×2 maximum pooling layer, and the Encoder end three-volume layer is set to hole convolution. Decoder has two branches, one is the upsampling of Encoder, which implements semantic segmentation. In this paper, the indexing function of the pooling layer is used to perform upsampling in an unpooling manner. Each upsampling followed by multiple convolutional layers, and the standard cross entropy loss function is used to train the segmentation network. The other branch is the instance segmentation branch. The network generates the pixel vector is in the high-dimensional feature space. The discriminant loss function has combined with the semantic segmentation result to realize the instance segmentation. Finally, the instance detection of the lane line has realized by fitting.

In addition, this paper has made a research on the data set of the lane virtual line to distinguish the lane virtual reality detection. By increasing the decoder's semantic segmentation of the branch network output, the transition lane line and background are transformed into the solid lane. The accuracy and robustness of the dashed verified by actual tests, which is the same with the solid line identification of the proposed algorithm.

2 Related work

Regarding the detection of lane lines, the traditional method of detecting [Talib and Ramli (2015); Li, Long, Ming et al. (2014); Wu, Lin and Lee (2012); Yoo, Yang, Sohn et al. (2013)] mainly uses image processing technology to perform edge detection, thresholding processing and curve fitting on road images. The main steps are to pre-process the image, select the Region of Interest (ROI), and perform edge detection. After Hough transform, thresholding is performed, and then the result is processed by the straight line or curve fit. Common fitting methods mainly include least squares method, polynomial fitting, and random sample Consensus (RANSAC) algorithm fitting. Many scholars at home and abroad have a lot of research on this.

In 2014, the literature [Kim and Lee (2014)] proposed a Convolutional Neural Networks (CNN) combined with RANSAC lane detection method. Firstly, the original image is edge-detected and the lane information is enhanced. Then, in the simple road scene, the author thinks that the detection can have completed by using the RANSAC method. For complex road conditions such as shadows and fences, it is processed by CNN, the used the RANSAC method. The CNN network structure consists of three convolutional layers, two downsampling layers, a multi-layer perceptron, and three fully connected layers. The edge image of ROI is input, and the CNN network output only contains white lane lines and black background. The complexity judgment of the scene depends on the setting of the conditional threshold. The requirements of the different scenarios are different. At the same time, the CNN network structure is very simple, so the robustness of the whole

algorithm is not high.

The lane line detection [Xiao, Luo, Yao et al. (2018)] is based in part on visual sensors. First, the road information captured by the monocular camera on the windshield of the vehicle, and then the edge point information is extracted. Then, the lane line selected according to the custom parameter space voting method, and combined with the Extended Kalman Filter (EKF) pair. The coordinate parameters of the edge points of the lane line are tracked and estimated. Finally, the current vehicle position information is obtained according to the GPS positioning system, and the lateral offset between the vehicle and the current lane line is calculated to realize the early warning function. Parameter space voting is a classic method, but it is susceptible to interference points, which require high extraction of edge points and does not handle cornering well.

Traditional lane detection methods [Narote, Bhujbal, Narote et al. (2018)] rely on highly specialized, handcrafted features and heuristic constraints, often requiring various post-processing techniques for optimization, which are extremely unstable due to changes in road scenes.

In 2016, the literature [He, Ai, Yan et al. (2016)] obtained the corresponding top view from the front view of the image through the inverse perspective transformation method, and the candidate region and the candidate lane line through the hat-weighting filter. The double view network architecture of Double View Convolutional Neural Network designed, which input the original forward-looking image. The top view corresponds to the candidate region into the DVCNN network at the same time, and the last optimal lane obtained by using the global optimization function considering information including the length, number, probability, direction, and width of the lane line. The combined different views improve the accuracy of detection, but it also increases the speed of the algorithm. In considering speed, in 2017, Kim et al. [Kim, Kim, Jang et al. (2017)] proposed a fast learning algorithm based on Extreme Learning Machine (ELM) convolutional neural network and applied it to lane line detection. The input image before lane detection is enhanced by eliminating noise and obstacles, which are not related to the edge detection result. ELM is a fast learning method for calculating the network weight between the output layer and the hidden layer in one iteration. This approach reduces the learning time of the network, but the role of the network is focused on image enhancement.

In 2018, Liu et al. [Liu and Deng (2018)] proposed an RPP model based on single convolution visual road detection. Specifically, RPP is a deep full convolution residual partition network with pyramid pool. In order to improve the prediction accuracy of the KITTI-ROAD detection task, Liu proposes a new strategy by adding road edge tags and introducing appropriate data enhancements. It is an effective idea to use the semantic segmentation in deep learning to complete the detection of roads or lanes.

3 Proposed method

The algorithm is based on the road segmentation and lane detection algorithm [Oliveira, Burgard, Brox et al. (2016)], and draws on the use of the discriminant loss function [De, Neven and Van (2017)] in LaneNet [Neven, De, Georgoulis et al. (2018)] algorithm and the cavity convolution [Yu and Koltun (2016)] applied in the algorithm of DeepLabv1 [Chen, Papandreou, Kokkinos et al. (2014)] and LMD [Chen, Lo, Hang et al. (2018)].

Void convolution is the replacement of the common convolutional layer of the extracted feature part to expand the receptive field. The distinguishing feature of the discriminant loss function is that it is easy to integrate into different network structures [Ren, He, Girshick et al. (2015)], and the instance segmentation is realized by post-processing. In addition, there are some works on image denoising preprocessing presented in previous articles [Xiao, Tian, Zhang et al. (2018); Ding, Zhang, Xiao et al. (2018)].

3.1 Network structure

The algorithm adopts the Encoder-Decoder structure mode [Redmon, Divvala, Girshick et al. (2016)]. The Encoder part uses the VGG16 network as the base model to extract the lane line features, discards the fully connected layer of the VGG16 network, retains only the first four 2×2 maximum pooling layers in VGG16, and uses the holes. Convolution can expand the characteristics of the receptive field [Liu, Anguelov, Erhan et al. (2016)]. In this paper, the 11th, 12th, and 13th layers are set as a cavity convolution with a void ratio of 2. Decoder has two branches; one is the upsampling of Encoder, which implements semantic segmentation [Chen, Papandreou, Kokkinos et al. (2016)]. It mainly uses the index function of the largest pooling layer. It consists of four Upsampling layers and ten convolution layers. The Upsampling layer is used to better handle the gradient disappearance problem using the activation function ReLU, and the segmentation network is trained using the standard cross entropy loss function. The other branch is the instance segmentation branch. The segmentation loss function based on the distance metric learning is used to implement the instance segmentation on the generated pixel vector feature map. Finally, the instance detection of the lane line is completed by cluster fitting. The specific network structure and work flow chart are shown in Fig. 1:

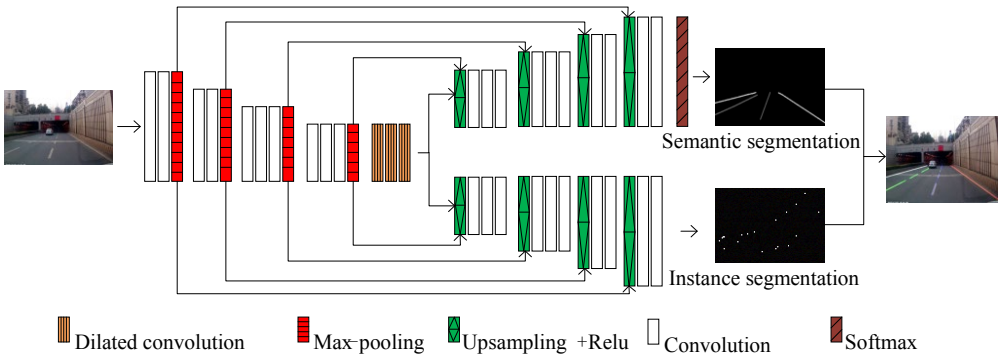

Figure 1: Network structure and workflow of this paper

3.2 Encoder

Encoder is the part of the algorithm network structure to extract image features. Based on vgg16, it is mainly composed of convolution layer and pooling layer. The specific parameters are shown in Tab. 1.

Table 1: Encoder network structure

Name	Kernel	Stride	Output
conv1_1	3×3	1	512×256×64
conv1_2	3×3	1	512×256×64
Pool1,max_indices	2×2	2	256×128×64
conv2_1	3×3	1	256×128×128
conv2_1	3×3	1	256×128×128
Pool2,max_indices	2×2	2	128×64×128
conv3_1	3×3	1	128×64×256
conv3_2	3×3	1	128×64×256
conv3_3	3×3	1	128×64×256
Pool3,max_indices	2×2	2	64×32×256
conv4_1	3×3	1	64×32×512
conv4_2	3×3	1	64×32×512
conv4_3	3×3	1	64×32×512
Pool4,max_indices	2×2	2	32×16×512
Dilated conv5_1	3×3	1	32×16×512
Dilated conv5_2	3×3	1	32×16×512
Dilated conv5_3	3×3	1	32×16×512

The input training sample resolution is adjusted to 512×256, convoluted with the filter bank to generate a set of feature maps, and then they are mass normalized, followed by the activation function linear rectification function. Then, the maximum pooling is used for 2-x downsampling, and the position of the largest eigenvalue in each pooled window is stored for each feature map before downsampling. In the last three convolutional layers, the maximum pooling operation is not continued, but a hole convolution [Chen, Papandreou, Schroff et al. (2017)] with a hole ratio of 2 is used instead of the ordinary convolution operation. Therefore, the resolution of the feature map at the end of the encoder network with a 2-x increase, the void convolution can expand the receptive field without any additional parameters and computational costs.

3.3 Remove the fully connected layer

In many common algorithms, Fully Connected Layers generally follow the convolutional and pooling layers, which act as a "classifier". The convolutional layer and the pooling layer map the input raw data to the hidden layer feature space for feature extraction, and the fully connected layer maps the learned features to the sample mark space. As shown in Fig. 2, the nodes a, b, and c of the fully connected layer are respectively connected to the nodes X, Y, and Z of the upper layer, and function to synthesize the previously extracted features. Nevertheless, at the same time, in view of the fact that all the nodes are connected, the parameters of the full connection layer generally account for the

largest proportion in the network structure. For example, in the familiar VGG16, for the input of 224×224×3, the first fully connected layer FC6 has 4096 nodes, and the upper layer of FC6 is the fifth largest pooling layer with 7×7×512, which has a total of 25088 nodes, then it means that 4096×25088 weights are needed, which consumes a huge amount of memory.

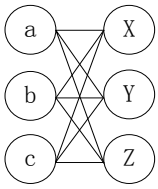

Figure 2: Schematic diagram of the full connection layer

In the semantic segmentation FCN network and the literature [Oliveira, Burgard, Brox et al. (2016)] network, the full connection layer is used, which reduces the filters number of the fully connected layer from 4096 to 1024. However, even the parameters of the fully connected layer are still redundant. In order to maintain the resolution of the feature map and consider the running speed of the algorithm, this paper chooses to discard the operation of the fully connected layer directly, thereby greatly reducing the number of parameters, and the speed comparison is confirmed by the final comparison experiment.

3.4 Increase the cavity convolution

The existence of downsampling makes the running filter have a larger receptive field, which is beneficial to collect more context information and improve the accuracy of segmentation. However, the result of semantic segmentation requires the same resolution as the input, which means that powerful downsampling will require the same powerful upsampling. On the other hand, downsampling will also lose edge resolution if the feature resolution is reduced. The important spatial information, and the operability of the original information for the lost information is greatly reduced. In this regard, the proposed and well-conceived cavity convolution [Yu and Koltun (2016)] avoids these problems. The cavity convolution provides an effective mechanism to control the field of view. It can expand the receptive field of the filter without downsampling to include larger Contextual information.

3.5 Semantic segmentation

The semantic segmentation [Chen, Papandreou, Schroff et al. (2017); Chen, Zhu, Papandreou et al. (2018); Peng, Zhang, Yu et al. (2017)] part mainly realizes the segmentation of the lane line and the background. By upsampling the encoder, the output has the same resolution as the input data. Upsampling in computer vision generally includes three methods, which are namely bilinear interpolation, inverse pooling, and deconvolution. The main idea of bilinear interpolation is to perform linear interpolation in two directions, respectively. Deconvolution is the inverse of convolution operation. Compared with the former two, the parameters in the deconvolution process need to train. In theory, if the convolution kernel parameters are set properly, deconvolution can

achieve anti-pooling. Anti-pooling operations tend to be more efficient in terms of memory usage because they only need to store fewer indexes. The de-pooling and deconvolution diagram is shown in Fig. 3, where Fig. 3(a) shows the maximum pooling and the corresponding de-pooling operation, and Fig. 3(b) shows the convolution and the corresponding deconvolution operation.

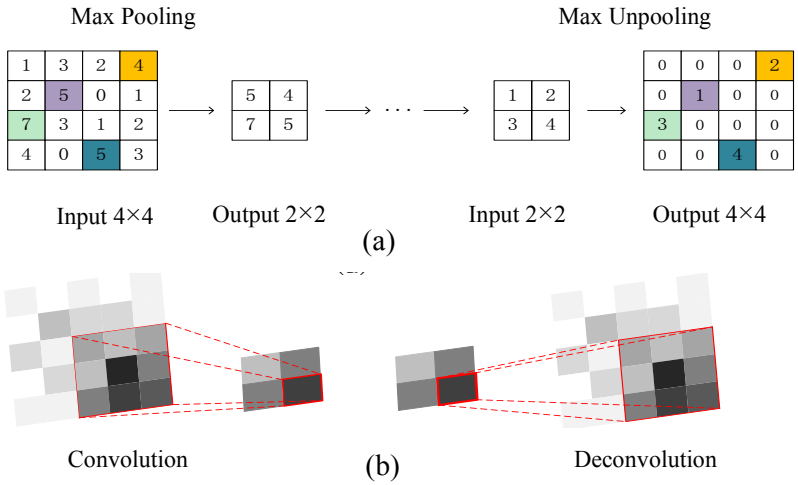

Figure 3: Comparison of anti-pooling and deconvolution

The role of semantic segmentation is mainly to provide a mask for instance segmentation. The case segmentation involves the post-processing of clustering. If clustering acts overall image, it will consume a lot of time, and the mask provided by semantic segmentation can ignore the proportion of the scale. Large background information can speed up clustering.

3.6 Data imbalance

Data imbalance means that the proportion of each category varies greatly. If the data is not balanced, such as category 1 accounting for 1% and category 2 accounting for 99%, then the network model biased prediction result to category 2 will get the highest. The rate is accuracy, but the effect is not good in practical applications.

To solve the data imbalance problem, the class weight adds to weight the cross entropy, as shown in Eq. (1).

$$w_{class} = \frac{1}{ln(c + p_{class})} \tag{1}$$

Among them, p is the probability that the corresponding category appears in the overall sample, c is a hyperparameter, which is set to 1.03.

3.7 Instance segmentation

The instance segmentation branch network has realized by discriminating the loss function. The discriminant loss function contains three items, which are the variance term,

the distance term, and the regular term. Both the variance term and the distance term have a certain range of distances, which is manifested in that the embedded pixels are no longer subjected to tension when they are within the center δ_v of the cluster, which means that the embedded pixels do not have to be aggregated to a single point. Similarly, the embedded pixels are no longer subject to thrust when they are farther away from the center of the cluster than $2\delta_d$. Where δ_v and δ_d are the thresholds associated with the training samples, based on the distance between the instances.

The specific calculation of the discriminant loss function is shown in the following equations:

$$L_{var} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i=1}^{N_c}[\| \mu_c - x_i \| - \delta_v]_+^2 \tag{2}$$

$$L_{dist} = \frac{1}{C(C-1)}\sum_{C_A=1}^{C}\sum_{C_B=1}^{C}[2\delta - \| \mu_{C_A} - \mu_{C_B} \|]_+^2, (C_A \neq C_B) \tag{3}$$

$$L_{reg} = \frac{1}{C}\sum_{C=1}^{C}\| \mu_C \| \tag{4}$$

$$\mu_C = \frac{1}{N}\sum_{i=1}^{N_C} x_i \tag{5}$$

Eq. (2) represents the pulling force, and Eq. (3) represents the thrust, C represents the number of instances in the real label, N_c indicates the total number of pixels in an instance. x_i denotes the embedding vector generated by the pixel i mapping in the example, and μ_c is the center of the embedding vector of the mapping corresponding to all the pixels of the instance in the real label, that is, the embedded average value, which is calculated by the Eq. (5). Eq. (4) is a regular term to ensure that the distance between each cluster center mapped to the feature space and the origin does not become very far.

Finally, the loss function of the whole algorithm is the weighted sum of the respective loss functions of the two branch networks, and the weights are all 0.5.

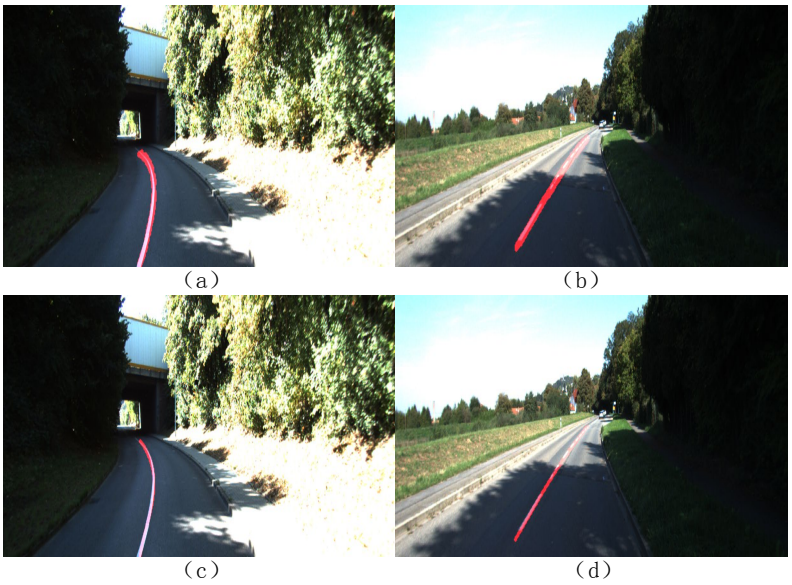

Figure 4: Clustering fit result graph

As shown in Fig. 4, Figs. 4(a) and 4(b) are effect diagrams after clustering, Figs. 4(c) and 4(d) are the results after corresponding fitting. It can be clearly seen from the figure that the clustering has completed the detection of the lane line [He, Gkioxari, Dollar et al. (2017); Kim and Park (2017)], but contains more background information, which can more accurately represent the specific position of the lane line after fitting.

4 Identification of lane lines

If only the lane line is identified, the final output of the segmentation network actually has two types, the lane line, and the image background. After the virtual reality detection function is added, the actual output of the segmentation network becomes three categories, which are namely the lane solid line, the lane dotted line, and the image background.

4.1 Building a data set

The dataset used in this paper is the labelme tool, which uses the monocular camera on the windshield of the vehicle to record video in the surrounding area including the high-speed section to obtain the data material. 50 video clips are sampled every 30 frames for each video. The image with no lane line is culled, the image size is 1280×720, and 1800 images are selected as the training set, and the data set is increased by random left and right flip.

4.2 Identification process

The algorithm in this paper uses the semantic segmentation method. It only needs to change the last output channel of the semantic segmentation from the original 2 channels to 3 channels to increase the recognition of a category. The specific operations are as follows:

(1) Change of data set: As shown in Fig. 5, in the Fig. 5(a), all the lane line pixel values are

255, and the pixel value is 0 for the background and in the Fig. 5(b), the pixel value is 255. The dotted line indicates that the pixel value is 100 for the solid line; the pixel value for 0 indicates the background. The training data Fig. 5(d) is corresponding to the example segmentation and the original Fig. 5(c) do not need to be changed. In actual training, 0, 1, and 2 correspond to the category background, solid line, and dotted line respectively.

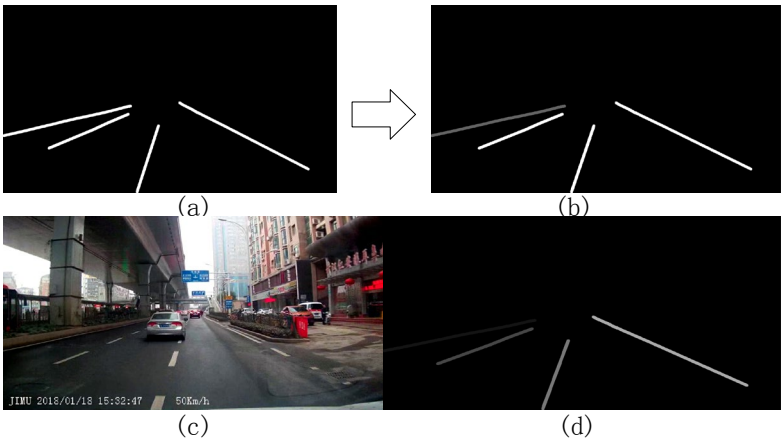

Figure 5: Virtual and real identification data set

(2) Algorithm change: The last convolutional layer of the semantic segmentation branch network adds an output channel; the feature vector of the lane dotted line and the lane solid line are respectively obtained on the instance segmentation map by the semantic segmentation mask map.

The final lane virtual recognition effect is shown in Fig. 6.

Figure 6: Lane virtual reality recognition effect map

Fig. 6(a) shows the original image to detect, and Fig. 6(b) shows the effect of the lane virtual reality recognition. The solid line drawn in the figure indicates the actual lane solid line, and the dotted line indicates the actual lane dotted line. As shown in Fig. 6, the effectiveness of the proposed algorithm in the lane virtual identification function is verified.

5 Experimental results

5.1 Experimental environment and network parameters

The experimental software environment of this article includes ubuntu16.04 (x64), python3.5, cuda-9.0, cudnn-7.0, TensorFlow 1.10. Hardware platform: GTX-1060GPU 6g memory. The processor is intel(R)core(TM)i5- 6500M CPU@3.20GHZ.

The parameters involved in this algorithm mainly have learning rate, the size is set to 0.0001, the training sample image resolution is 512×216, and the batch size is set to 4. The size of the Batch size cannot have continued by the limitation of GPU memory. The experimental data set respectively is a KITTI dataset and a self-made dataset. The KITTI dataset does not distinguish between virtual lanes and real lanes. The self-made dataset is used for lane virtual reality detection. The corresponding embedded pixel mapping feature spatial dimensions are 3 and 4, respectively, δ_d set to 3, δ_v set to 0.5. A suitable neural network optimization algorithm helps the model to produce better and faster results.

5.2 Experimental results and analysis

The algorithm is used to test the video sequences in various environmental scenarios, including daytime, high-speed, night, rainy days, and so on. These scenes also include corners, vehicle interference, occlusion, strong illumination, ground strip interference, and so on. In the complicated road conditions, 8 sets of video were used and the algorithm running speed and accuracy were used as evaluation criteria.

The test results are shown in each scene of the self-made data set, different colors represent different lane instances, solid lines indicate solid lanes, and dashed lines indicate lane dashed lines.

(a) (b) (c)

Figure 7: Test results in different environments: (a) Test results of curved road sections; (b) Test results of rainy days; (c) Test results of night

Fig. 7 shows the lane line recognition of good road conditions in different environments. Fig. 7(a) shows the correct detection results of the lanes of the multi-vehicle road segment, and lane line recognition. Fig. 7(b) shows the corners. The lane line is accurate detection results, which includes the effect of lane line detection in the rainy environment [Xiao, Zou, Chen et al. (2018)], and Fig. 7(c) shows the result of lane line recognition in the night environment.

Open source dataset test results: different colors represent different lane instances, solid

lines indicate solid lanes, and dashed lines indicate lane dashed lines.

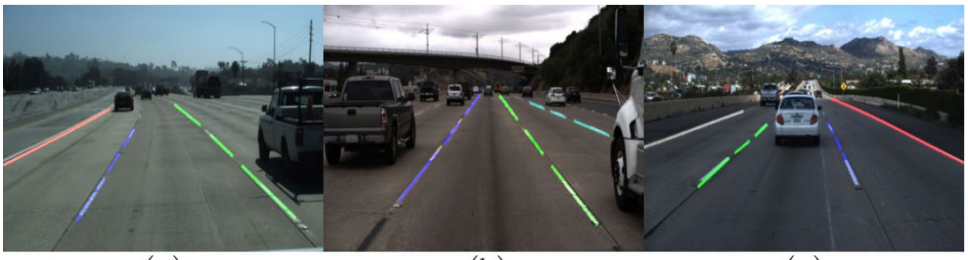

Figure 8: Test results of Tucson dataset

Fig. 8 shows the materials of the Tucson dataset are taken from the highway, but it is not easy to detect. The difficulty lies in the serious wear of the lane line, and the lane line features are not obvious, but the detection result of the algorithm is better.

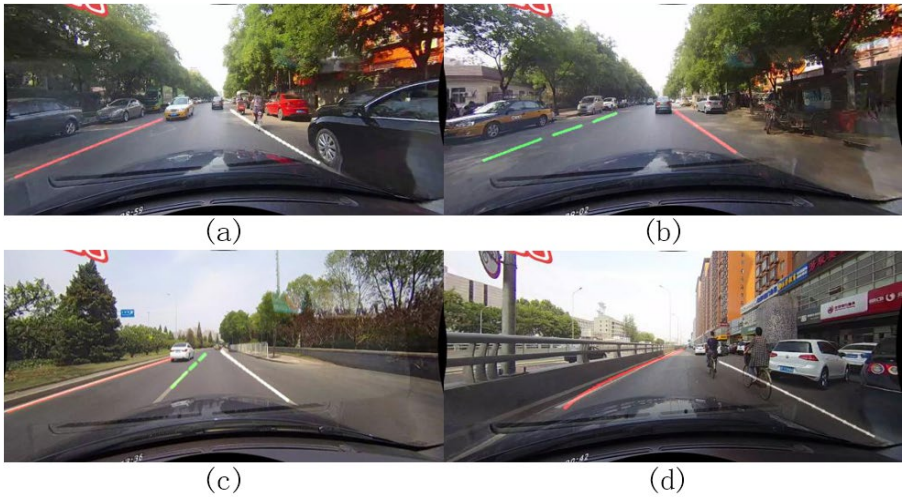

Figure 9: CULane data set test results

Fig. 9 shows the resolution of CULane dataset is relatively large, and it contains the interference information of many vehicle heads. In the case of good road conditions, the algorithm can still accurately identify the lane line.

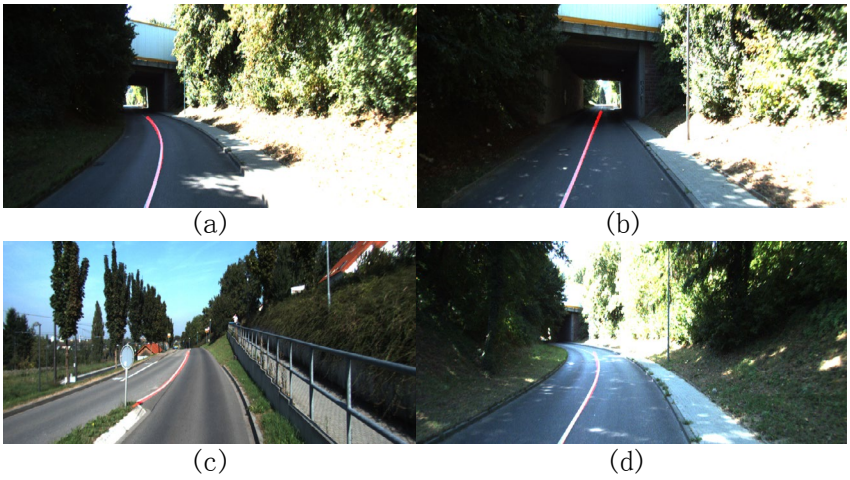

Figure 10: KITTI dataset detection

Fig. 10 shows the KITTI dataset provides two types of markings, the road, and the current lane. It is mainly used for the segmentation of the vehicle's travelable area. The lane line mainly exists in the middle of the road [Behrendt and Witt (2017)]. The method accurately identifies the lane line.

6 Conclusion

This paper proposes a robust lane detection method based on the current mainstream deep learning algorithm. The algorithm tested by different datasets and lanes in different weather environments, the accuracy and robustness of the proposed algorithm are verified. In the algorithm comparison experiment, the detection speed of the algorithm is faster than the algorithm in ref. [Oliveira, Burgard, Brox et al. (2016)], and the corresponding accuracy is almost the same, only about 1.39%. In contrast to the traditional method based on voting of ref. [Li, Zhou, Li et al. (2018)], the algorithm is much slower in detection speed, but it is much higher in detection accuracy, especially for the detection of false real lane lines and the detection of corners. To achieve effective detection, in contrast, the algorithm using deep learning does not have these problems, and can achieve accurate detection.

Acknowledgement: This work is supported by the National Natural Science Foundation of China (61772386); Joint fund project (nsfc-guangdong big data science center project), project number: U1611262, Hubei University of Science and Technology, Master of Engineering, special construction, project number: 2018-19GZ01, Hubei University of Science and Technology Teaching Reform Project, project number: 2018-XB-023, S201910927028.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Behrendt, K.; Witt, J. (2017): Deep learning lane marker segmentation from automatically generated labels. *International Conference on Intelligent Robots and Systems*, pp. 777-782.

Chen, L. C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A. L. (2014): Semantic image segmentation with deep convolutional nets and fully connected CRFs. arXiv: 1412.7062.

Chen, L. C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A. L. (2016): DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 40, no. 4, pp. 834-848.

Chen, L. C.; Papandreou, G.; Schroff, F.; Adam, H. (2019): Rethinking atrous convolution for semantic image segmentation. arXiv:1706.05587.

Chen, L. C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. (2018): Encoder-decoder with atrous separable convolution for semantic image segmentation. *European Conference on Computer Vision*, pp. 801-818.

Chen, P.; Lo, S.; Hang, H.; Chan, S.; Lin, J. (2018): Efficient road lane marking detection with deep learning. *International Conference on Digital Signal Processing*, pp. 1-5.

De, B. B.; Neven, D.; Van, G. L. (2017): Semantic instance segmentation with a discriminative loss function. arXiv: 1708.02551.

Ding, L.; Zhang, H.; Xiao, J.; Li, B.; Lu, S. et al. (2018): An improved image mixed noise removal algorithm based on super-resolution algorithm and CNN. *Neural Computing and Applications*, vol. 31, no. 1, pp. 325-336.

He, A.; Shan, Y.; Song, D. B. (2018): The influence of in-car ADAS alarm sound on driver behavior. *Automobile Practical Technology*, pp. 38-39.

He, B.; Ai, R.; Yan, Y.; Lang, X. (2016): Accurate and robust lane detection based on dual-view convolutional neutral network. *IEEE Intelligent Vehicles Symposium*, pp. 1041-1046.

He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. (2017): Mask R-CNN. *International Conference on Computer Vision*, pp. 2961-2969.

Kim, J.; Kim, J.; Jang, G. J.; Lee, M. (2017): Fast learning method for convolutional neural networks using extreme learning machine and its application to lane detection. *Neural Networks*, vol. 87, pp. 109-121.

Kim, J.; Lee, M. (2014): Robust lane detection based on convolutional neural network and random sample consensus. *International Conference on Neural Information Processing*, pp. 454-461.

Kim, J.; Park, C. (2017): End-to-end ego lane estimation based on sequential transfer learning for self-driving cars. *IEEE Conference on Computer Vision and Pattern Recognition Workshops,* pp. 30-38.

Li, Q.; Chen, L.; Li, M.; Shaw, S. L.; Nüchter, A. (2013): A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios. *IEEE Transactions on Vehicular Technology*, vol. 63, no. 2, pp. 540-555.

Li, Q.; Zhou, J.; Li, B.; Guo, Y.; Xiao, J. (2018): Robust lane-detection method for low-speed environments, *Sensors*, vol. 18, no. 12, pp. 4274.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Berg, A. (2016): SSD: single shot multiBox detector. *European Conference on Computer Vision*, pp. 21-37.

Liu, X.; Deng, Z. (2018): Segmentation of drivable road using deep fully convolutional residual network with pyramid pooling. *Cognitive Computation*, vol. 10, no. 2, pp. 1-10.

Narote, S. P.; Bhujbal, P. N.; Narote, A. S.; Dhane, D. M. (2018): A review of recent advances in lane detection and departure warning system. *Pattern Recognition*, vol. 73, pp. 216-234.

Neven, D.; Brabandere, B. D.; Georgoulis, S.; Proesmans, M.; Gool, L. V. (2018): Towards end-to-end lane detection: an instance segmentation approach. *IEEE Intelligent Vehicles Symposium*, pp. 286-291.

Oliveira, G. L.; Burgard, W.; Brox, T. (2016): Efficient deep models for monocular road segmentation. *International Conference on Intelligent Robots and Systems*, pp. 4885-4891.

Peng, C.; Zhang, X.; Yu, G.; Luo, G.; Sun, J. (2017): Large kernel matters-improve semantic segmentation by global convolutional network. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4353-4361.

Redmon, J.; Divvala, S. K.; Girshick, R. B.; Farhadi, A. (2016): You only look once: unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779-788.

Ren, S.; He, K.; Ross, G.; Sun, J. (2015): Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149.

Talib, M. L.; Ramli, S. (2015): A review of multiple edge detection in road lane detection using improved Hough transform. *Advanced Materials Research*, vol. 1125, pp. 541-545.

Wu, C. F.; Lin, C. J.; Lee, C. Y. (2012): Applying a functional neurofuzzy network to real-time lane detection and front-vehicle distance measurement. *IEEE Transactions on Systems Man & Cybernetics Part C*, vol. 42, no. 4, pp. 577-589.

Xiao, J.; Luo, L.; Yao, Y.; Zou, W.; Reinhard, K. (2018): Lane detection based on road module and extended Kalman filter. *PSIVT 2017*, pp. 382-395.

Xiao, J.; Tian, H.; Zhang, Y.; Zhou, Y.; Lei, J. (2018): Blind video denoising via texture-aware noise estimation. *Computer Vision and Image Understanding*, vol. 169, pp. 1-13.

Xiao, J.; Zou, W.; Chen, Y.; Wang, W.; Lei, J. (2018): Single image rain removal based on depth of field and sparse coding. *Pattern Recognition Letters*, vol. 116, pp. 212-217.

Yoo, H.; Yang, U.; Sohn, K. (2013): Gradient-enhancing conversion for illumination-robust lane detection. *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1083-1094.

Yu, F.; Koltun, V. (2016): Multi-scale context aggregation by dilated convolutions. arXiv: 1511.07122.