Journal of Physics: Conference Series

PAPER • OPEN ACCESS

Lane Detection of Smart Car based on Deep Learning

To cite this article: Dongdong Yang *et al* 2021 *J. Phys.: Conf. Ser.* **1873** 012068

View the article online for updates and enhancements.

This content was downloaded from IP address 61.3.124.78 on 23/05/2021 at 18:41

Lane Detection of Smart Car based on Deep Learning

Dongdong Yang, Wenzhi Bao, Kai Zheng

Data Science and Engineering

East China Normal University

Shanghai, China

51194507017@stu.ecnu.edu.cn

Abstract. Deep learning brings higher accuracy to lane detection, and the speed of the model is becoming faster and faster. The lane detection is regarded as the classification task after image gridding, and a simplified lane structure loss function is proposed, which is more suitable for the car track. The lightweight SqueezeNet is used as the backbone network, and the deep learning model is applied to the intelligent car of raspberry pie. Through the experiment, the real-time detection speed reaches 23FPS, which can complete the task of lane tracking. Compared with traditional methods, deep learning model has better robustness.

1. Introduction

Lane detection is an important task in the field of driverless. According to the input of vision or laser sensor, lane position can be sensed in real time, which can play the role of lane control and auxiliary positioning.

Intelligent tracking car is a device that can travel along the lane detected in real time. As in the field of automatic driving, there are two research methods of lane detection by intelligent car. One is the traditional image processing method. Through image preprocessing, feature extraction, combined with Kalman filter[1] or Hough transform algorithm[2], the lane can be identified and fitted. Secondly, with the rise of deep learning, various methods based on convolutional neural network have achieved good performance in lane detection task in the field of automatic driving because of their strong feature extraction ability and robustness. Traditional image processing methods have relatively low hardware requirements and high FPS, but they are greatly affected by lighting and other environments, and the applicable scene is single. Deep learning method solves these problems, but because of its many network parameters and slow convergence of model training, it can rarely be directly moved to the raspberry pie computer of intelligent car[3].

Based on the idea of Ultra Fast Structure-aware Deep Lane Detection (UFLD)[4], this paper defines lane detection as a classification task, takes the lightweight SqueezeNet[5] as the backbone network, and defines the lane loss function suitable for car tracking, which achieves good speed and accuracy on the raspberry pie car.

2. Related works

Traditional methods usually do edge detection and lane fitting based on image information, and focus on the extraction of image features. Therefore, when the image features are not strong enough, or the environment has a great impact, such as the image contrast is low when the light is dark, and the car encounters obstacles to block part of the lane, the generalization performance of traditional methods is

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

not good enough. In addition, image processing is mostly carried out pixel by pixel, so it is difficult to improve the speed due to the large amount of calculation. Therefore, it is proposed to set the region of interest in advance to narrow the processing range.

In the development of deep learning, some methods based on neural network show some advantages in lane detection task. Seokju Lee et al.[6] designed VGPNet (vanishing point guided network) based on the vanishing point of the road, and also marked the lane grid, which aims to solve the problem of road marking and lane detection in extreme weather. Compared with the lane marked by pixels, the method can reduce the network calculation. SCNN (Spatial As Deep: Spatial CNN for Traffic Scene Understanding) proposed by Xinang Pan[7], enables image feature information to be transmitted from the top and down of the same layer, making full use of the prior information that lane is a long continuous structure. When obstacles block the lane, it can be inferred from the context information. Although the greater sense field is obtained, the speed is also slower. Zequn Qin et al. proposed Ultra Fast Structure-aware Deep Lane Detection, which defined lane detection as a collection of finding the location of some rows in the image. It is based on row-based classification and location selection, greatly reduces the computational complexity of the network, and the classifier uses global features, which solves the problem of insufficient sense field in lane detection.

3. Methods

Based on the research background of intelligent car tracking and the classification idea of UFLD, this paper proposes a lane detection method that can run on raspberry pie computer. In this section, we will introduce our method in detail, including formula and lane structure loss. To illustrate the problem better, there are some of the symbols used below.

Table 1. Notation.

Variable	Definition
H	Height of image
W	Width of image
h	Number of row anchors
w	Number of gridding cells
C	Number of lanes
P	Probability of each location
Loc	Locations of lanes
f	The classifier for selecting lane locations
X	The global features of image

3.1. Row-based classification

Suppose our car camera captures an image with the size of $h \times w$ as shown in an image, usually in the segmentation task, we need to deal with $h \times W$ classification problems, and the computational complexity is extremely high. We regard the task of lane detection as a set of finding the position of lane on some rows, that is, processing multi classification task row by row.

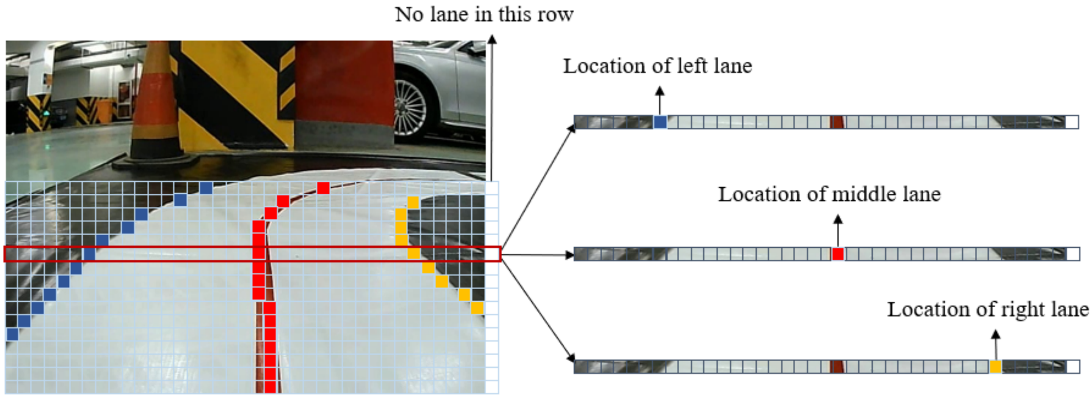

Figure 1. The gridding cells of road image.

First of all, we gridded the image. On each row, the position is divided into multiple units, and H rows are selected. Generally, h is far less than the height h of the image, and h can be set manually on demand. In this way, we reduced the number of classification from h×W to c×h, greatly reducing the computational complexity.

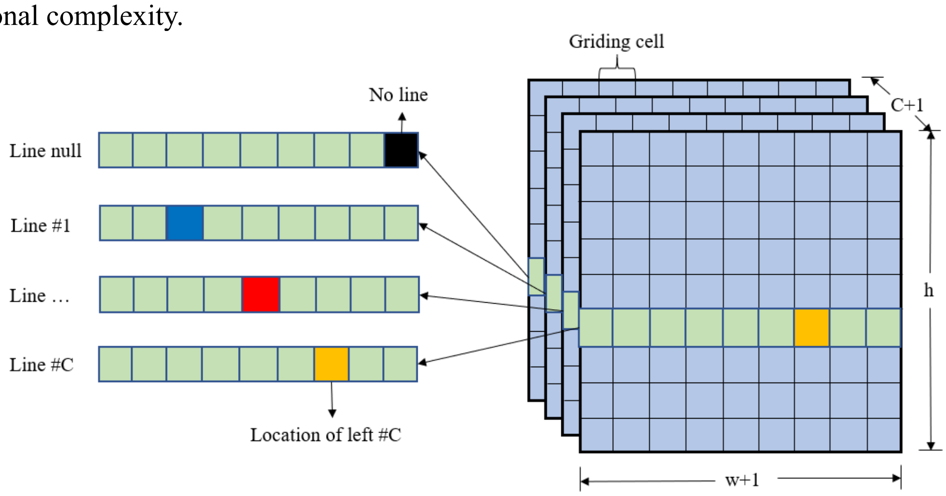

Figure 2. The formulation.

Assuming that the maximum number of lanes is C, the image is divided into h rows, X is the global image feature, and f^{ij} is the classifier used to select the position of the i-th lane and the j-th row to anchor the upper lane.

$$P_{i,j,:} = f^{ij}(X), s.t.\, i \in [1, C], j \in [1, h] \tag{1}$$

Where, $P_{i,j,:}$ is a vector of w+1 dimension (we use an additional dimension to indicate that there is no lane), which is expressed as the probability of w+1 grid cell selection for the i-th lane and the j-th row anchor. Assuming that $T_{i,j,:}$ is the label in the correct position, our classification loss function is as follows:

$$L_{cls} = \sum_{i=1}^{C} \sum_{j=1}^{h} L_{CE}(P_{i,j,:}, T_{i,j,:}) \tag{2}$$

L_{CE} is the cross entropy loss. From Eq.(1), our prediction probability is based on global features, so the lane position can be obtained according to all probability distributions.

3.2. The loss of lane structure

In addition to classification loss, the UFLD also proposes two loss functions based on Lane prior information.

One is that the lane is continuous, that is to say, the lane is continuous, and the points in the adjacent row anchor points should be close to each other. In the formula, the lane position is represented by the classification vector. Therefore, the continuity is achieved by constraining the distribution of the classification vector on the adjacent rows.

$$L_{sim} = \sum_{i=1}^{C}\sum_{j=1}^{h-1} \left\| P_{i,j,:} - P_{i,j+1,:} \right\|_1 \tag{3}$$

The other is that the real lane on road is straight most of the time, so the second-order difference equation is used to constrain the shape of the lane. For the straight case, the second-order difference of the predicted position of the adjacent three lanes in h is zero. Then the loss can be defined as:

$$Loc_{i,j} = \sum_{k=1}^{w} k \cdot softmax(P_{i,j,k}) \tag{4}$$

$$L_{shp} = \sum_{i=1}^{C}\sum_{j=1}^{h-2} \left\| (Loc_{i,j} - Loc_{i,j+1}) - (Loc_{i,j+1} - Loc_{i,j+2}) \right\|_1 \tag{5}$$

Considering that the long straight lane in the scene of lane tracked by the intelligent car is rare, and the computational complexity of the intelligent car is also reduced, we decide to abandon the second loss function. It is enough to use only the location similarity of adjacent rows for lane structure loss. So our loss function is as follows:

$$L_{total} = L_{cls} + \lambda L_{sim} \tag{6}$$

3.3. Backbone network

The reason why neural network model does not work well on the raspberry pie computer of intelligent car, is that the parameters of neural network are too many, and the model is too large. Compared with traditional methods, training and operation consume more computing resources. This paper uses lightweight network SqueezeNet to replace ResNet-34[8] in UFLD as the backbone network

SqueezeNet is a lightweight CNN model proposed by song Han. Its parameters are 50× less than that of AlexNet[9], but its performance is close to that of AlexNet. Under acceptable performance, the small model has many advantages over the large model:

1) More efficient distributed training, small model parameters are small, network traffic is reduced;
2) It is easy to update the model, the model is small, and the client program is easy to update;
3) It is easy to deploy on specific hardware, such as FPGA, because its memory is limited.

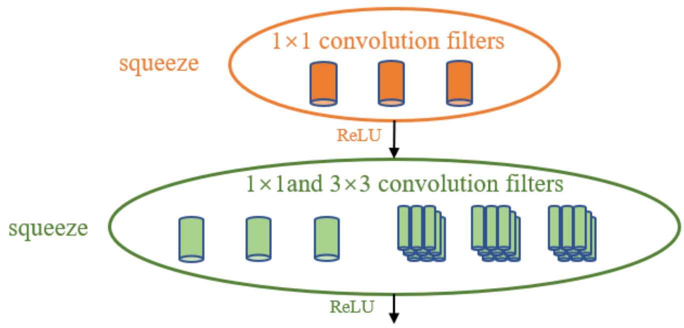

Figure 3. The network structure of SqueezeNet.

In the fire module[10], the expansion layer uses the hybrid convolution kernel 1×1 and 3×3, whose stride is 1. For the 1×1 convolution kernel, its output feature map is the same size as the original, but because it needs to concatenate with the feature map obtained by 3×3, So the 3×3 convolution is used to perform padding=1 operation. The whole SqueezeNet is built by stacking fire basic modules.

4. Experiment

In order to verify the effectiveness of the method, we conducted experiments on our own data sets. In this part, we will introduce our data set and experimental results.

4.1. Dataset

We set up the intelligent car tracking route in the underground garage, with three lanes. Then use Labelme[11] to label the lane, and save it in JSON format.

A total of 2770 jpg images are included in the data set, including normal illumination, insufficient illumination, and even ground reflection. Part of the lane are at the edge of the image, and only part of the outline is exposed. A brief description is shown in the table 2.

Table 2. Dataset.

Dataset	Frame	Train	Validation	Test	Resolution	Lane	environment
mydata	2770	990	200	1580	640×480	3	underground garage

4.2. Number of griding cells

Refer to the evaluation index of Tusimple dataset[12]. The accuracy is calculated by:

$$accuracy = \frac{\sum_{cell} C_{cell}}{\sum_{cell} S_{cell}} \tag{7}$$

C_{cell} is the number of correctly predicted grid cells, and S_{cell} is the total number of grid cells in the current view. We experiment with different number of grid cells to demonstrate the effect on the accuracy of the method. We use 10 to 100 grid cells in steps of 10 to do the experiment.

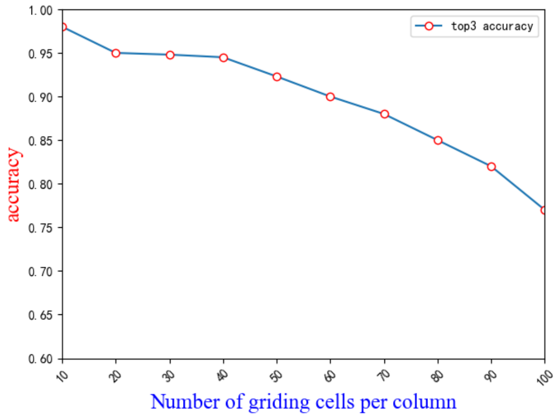

Figure 4. The accuracy under different numbers of gridding cells.

The less the number of grid cells, the higher the accuracy, but the grid is too large to represent the lane position. When the number of mesh cells is less than or equal to 40, the accuracy does not decrease obviously.

4.3. Results

Our models are trained on PyTorch [13] and NVIDIA GTX 2070 GPU, and tested with NVIDIA GTX 2070 GPU and Raspberry 4B CPU. Compared with the results of the UFLD, our accuracy is slightly decreased, but the detection speed is significantly improved.

Table 3. Comparison with other method.

Method	Accuracy	FPS	Environment
UFLD	95.87	322	GTX 1080Ti GPU
Ours	94.46	438	GTX 2070 GPU
Ours	94.46	23	Raspberry 4B CPU

From table 3, the accuracy is 95.87%, which is 1.41% lower than that of UFLD. Under NVIDIA GTX 2070 GPU, our model can run to 438 frames, and the real-time detection speed on raspberry pie 4B can reach 23FPS.

The CPU of raspberry pie 4B is arm cortex-a72 1.5GHz. Compared with GTX graphics card which is more suitable for neural network operation, the detection speed is greatly reduced, which is within the expected range. The driving speed of the intelligent car is 20cm/s, so the detection speed of 23FPS can ensure the tracking task of the car.

5. Conclusion

Based on the classification idea of UFLD, we simplify the loss function suitable for the track and use SqueezeNet instead of ResNet-34 as the backbone network. The deep learning model is applied to the intelligent car tracking task of raspberry pie. Compared with the traditional method, the method has better adaptability to the environment of incomplete lane and poor illumination while ensuring good accuracy and speed (23 FPS). In the future, we plan to run the model on the GPU of raspberry pie 4B.

References

[1] Welch, G. . (2001). Kalman filter. Siggraph Tutorial.
[2] Duda, R. O. , & Hart, P. E. . (1972). Use of hough transformation to detect lines and curves in pictures. Communications of the ACM, 15(1), 11-15.
[3] Hasani, R. F. . (2014). Car security system using raspberry pi. Dr Mohammad Iqbal.
[4] Qin, Z. , Wang, H. , & Li, X. . (2020). Ultra fast structure-aware deep lane detection.
[5] Iandola, F. N. , Han, S. , Moskewicz, M. W. , Ashraf, K. , Dally, W. J. , & Keutzer, K. . (2016). SqueezeNet: alexnet-level accuracy with 50x fewer parameters and <0.5mb model size.
[6] Lee S,Kim J,Yoon J S, et al.VPGNet: Vanishing Point Guided Network for Lane and Road Marking Detection and Recognition[C]//IEEE International Conference on Computer Vision(ICCV).IEEE,2017.
[7] Pan X, Shi J, Luo P, et al. Spatial as deep: Spatial cnn for traffic sceneunderstanding[C]//Thirty-Second AAAI Conference on Artificial Intelligence. 2018.
[8] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016).
[9] Krizhevsky, A. , Sutskever, I. , & Hinton, G. . (2012). ImageNet Classification with Deep Convolutional Neural Networks. NIPS (Vol.25). Curran Associates Inc.
[10] Yuan, C. , Wu, Y. , Qin, X. , Qiao, S. , Pan, Y. , & Huang, P. , et al. (2019). An effective image classification method for shallow densely connected convolution networks through squeezing and splitting techniques. Applied Intelligence.
[11] Russell, B. C. , Torralba, A. , Murphy, K. P. , & Freeman, W. T. . (2008). Labelme: a database and web-based tool for image annotation. International Journal of Computer Vision, 77(1-3).
[12] TuSimple: Tusimple benchmark. https://github.com/TuSimple/tusimple-benchmark, accessed November, 2019.
[13] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017).