Scientific Research Publishing

Journal of Computer and Communications, 2023, 11, 47-72
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Lane Line Detection Based on Improved PINet

Xueyan Jiao, Yiqiao Lin, Lei Zhao

School of Computer Science and Technology, Shandong University of Technology, Zibo, China
Email: 1466075375@qq.com

How to cite this paper: Jiao, X.Y., Lin, Y.Q. and Zhao, L. (2023) Lane Line Detection Based on Improved PINet. *Journal of Computer and Communications*, **11**, 47-72.
https://doi.org/10.4236/jcc.2023.113005

Received: February 28, 2023
Accepted: March 28, 2023
Published: March 31, 2023

Abstract

Accurate perception of lane line information is one of the basic requirements of unmanned driving technology, which is related to the localization of the vehicle and the determination of the forward direction. In this paper, multi-level constraints are added to the lane line detection model PINet, which is used to improve the perception of lane lines. Predicted lane lines in the network are predicted to have real and imaginary attributes, which are used to enhance the perception of features around the lane lines, with pixel-level constraints on the lane lines; images are converted to bird's-eye views, where the parallelism between lane lines is reconstructed, with lane line-level constraints on the predicted lane lines; and vanishing points are used to focus on the image hierarchy, with image-level constraints on the lane lines. The model proposed in this paper meets both accuracy (96.44%) and real-time (30 + FPS) requirements, has been tested on the highway on the ground, and has performed stably.

Keywords

Lane Line Detection, Instance Segmentation, Accuracy, Real Time

1. Introduction

Lane lines are road signals that restrict vehicles from following a prescribed route. Lane line detection is mainly to discriminate between lane lines and road background and to determine the position relationship between the current vehicle and the lane lines. The elongated shape of the lane lines requires a model with strong high and low level feature fusion capability to obtain both global spatial structure and local location information; in the actual highway scenario, although there are specific standards for lane markings, the different curvature of the road makes the shape of the lane lines vary, and there are both solid and dashed lane lines, and there are often joined and separated lane lines in the highway scene, making the number of lane lines uncertain, which requires the

model to have strong robustness to complex high-speed scenes; vehicle deviation or lane change will also cause the lane where the vehicle is located to change, and the relative position between lane lines will also switch; in addition, the accuracy of lane line detection is also affected by weather (rain, snow, fog, etc.), lighting conditions (daytime, nighttime) or other conditions. In addition, the accuracy of lane line detection is also affected by weather (rain, snow, fog, etc.), lighting conditions (day and night) or other conditions (vehicle shading, broken lane lines, etc.).

For the special shape of the lane lines, a stacked hourglass network is selected as the backbone network to extract features in this experiment. In a conventional Convolutional Neural Network (CNN), the convolutional layer receives the input from the previous layer, uses convolutional operations and nonlinear activation, and sends the output to the next layer, and the whole process is performed sequentially. Because the lane lines have a coherent elongated structure with a strong shape prior but a weak appearance, it is difficult to combine the overall information using CNN neural networks. The stacked hourglass network can capture feature information at different scales, and the output feature map incorporates semantic features of different levels of hourglass networks and scales, which can better capture the spatial information between lane lines.

For complex highway scenes, in previous work, most of the lane line virtual and real attributes, potential parallel structures, and extinction point information in road scenes have been ignored, which are used in this experiment to improve the perception of lane lines and enhance the accuracy of detection. The dashed lines can be crossed to represent the possibility of lane change operations, while the solid lines cannot be crossed to limit the drivable area of the vehicle, and the real and imaginary lane lines are related to the planning problem of the vehicle's forward direction. The lane lines have a parallel relationship in the real world, and the camera's perspective makes it lose this structure, which can be reconstructed in the overhead view to better cluster the lane line instances. The extinction point is the distant intersection of lane lines, which can guide the detection of distant lane lines in the case of blurred or obscured distal ends, and is important reference information for the adjustment of camera parameters for a shooting.

2. Related Work

Many methods are related to the task of lane line detection, and the traditional methods mainly extract features manually to identify and segment out the lane line region, and color [1] [2] [3] and shape [4] [5] [6] [7] are the most common features of lane lines. Bingjie Bai *et al.* [8] used double thresholding to extract white and yellow lane line information in the image, detected lane line pixels, and used Hough transform to complete the Ping Liu *et al.* [9] used inverse perspective transformation to get the top view and used a clustering algorithm to discriminate the line shape. However, the traditional methods have a high

workload, poor robustness of hand-made models, limited processing scenes, difficulty in coping with complex road conditions with diverse scenes in the real world, and the light intensity and special weather (rain, snow, fog, etc.) can have a large impact on their accuracy.

The proposed deep learning method provides a new research idea for the above problem and becomes the mainstream approach nowadays. Although CNN shows strong performance learning ability in scene understanding, it still performs poorly for lane line instances with slender constructions and perhaps obscured incoherent lanes, so SCNN [10] treats lane line detection as a semantic segmentation task to obtain an accurate classification of whether each pixel in an image belongs to a lane, achieves information transfer across rows and columns, and publishes a large dataset of lane line CULane, but the network is computationally intensive; SAD [11] proposes a self-attentive distillation module based on it to obtain a more lightweight backbone network and improve the computational speed. These two methods have a relatively large bias in the detection results for distant lane lines and can only detect a predefined number of lane lines. Line-CNN [12] proposes a new anchor-based method inspired by Faster R-CNN [13], which introduces a set of rays to capture lane lines; SGNet [14] rethinks the current difficulties in the field of lane line detection and proposes an anchor algorithm guided by extinction points. However, it is difficult to handle lane lines with large curvature due to the poor flexibility of anchor shape fixation. Inspired by human visual perception, the detection of incoherent lane lines relies mainly on the contextual information of the scene as well as global information in the case of severe occlusion by oncoming vehicles or extremely dark or bright illumination effects, based on this observation, for the detection speed as well as complex and diverse road scenes, UFSD [15] proposes the method of row detection to predict the cells that may belong to lane lines in each row, which effectively reduces the computational effort and greatly improves the detection speed, and the lightweight version can reach 300 + FPS, but the accuracy decreases a bit compared to the segmentation approach and requires a post-processing process to cluster the lane lines. Traditional methods usually perform curve fitting with least squares after extracting the lane line features, and to achieve end-to-end lane line detection and improve the detection efficiency, PolyLaneNet [16] was first proposed to estimate lane lines using a deep neural network approach to regress the curve equation by inputting images taken by the front camera mounted on the vehicle and outputting a polynomial representing each lane line in the image parameters, however, the prediction bias is relatively large for straight lanes.

Most previous work has focused on the powerful learning capability of neural networks to learn and fit the shape of lane lines, while ignoring the structural information associated with lane lines in images. In this paper information closely related to lane lines, such as the virtual and real properties of lane lines, the parallel structure of lane lines, and the distant intersections of lane lines, is

utilized to constrain the perception of lane lines in multiple layers of structure.

3. Method

3.1. Image Annotation

In this paper, the image label data are stored in a JSON file, as shown in Figure 1. An image corresponds to a row of annotation information objects in the JSON file, and the annotation information consists of five key-value pairs to represent the information of lane line points (lanes, h_samples), the information of virtual and real attributes of lane lines (types), the information of extinction points of lane lines (vp_point), and the relative position information of the corresponding image (raw_file). The following describes the labeling method and the meaning of each piece of information.

- Lane Labeling

In this experiment, the lane lines are manually labeled as points, and each lane line is labeled with more than three points, and the lane lines are a collection of coordinates of a sequence of points, and the dashed lines are treated as a kind of solid lines. The manually annotated image information is shown in Figure 2 as xml information, an image corresponds to an "image" element containing image information, and the "id attribute in the "image" element is used to identify the different image tags, the "name" attribute is the name of the image, and the "height" and "width" attributes indicate the height and width of the image, respectively. Each "image" element contains several "polyline" sub-elements that


```
{ "lanes" : [[721, 732, 749, 768, 788, 809, 831, 856, 880, 905, 929, 954, 978, 1002, 1027, 1051, 1075, 1100, 1123, 1147, 1170, 1194, 1217, 1237, 1256, 1275, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2], [721, 733, 748, 767, 788, 809, 831, 871, 919, 964, 1006, 1047, 1087, 1127, 1167, 1205, 1244, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2], [679, 674, 673, 673, 679, 685, 691, 697, 704, 711, 718, 725, 731, 738, 745, 752, 760, 767, 774, 782, 789, 797, 804, 812, 819, 826, 834, 841, 848, 855, 862, 869, 876, 883, 891, 898, 905, 912, 919, 926, 933, 940, 947, 955, 962, 969, 976, 983, 990, 997], [639, 617, 603, 590, 578, 568, 559, 550, 541, 532, 523, 515, 506, 498, 489, 481, 474, 466, 458, 451, 443, 436, 428, 420, 413, 405, 398, 391, 384, 376, 369, 362, 355, 348, 341, 333, 326, 319, 312, 305, 298, 290, 283, 276, 269, 262, 254, 247, 240, 233], [599, 562, 535, 508, 482, 455, 431, 406, 382, 358, 334, 312, 289, 267, 244, 222, 200, 178, 156, 135, 113, 91, 69, 47, 26, 4, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2], [565, 504, 468, 428, 387, 343, 303, 265, 226, 190, 155, 120, 84, 49, 14, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2, -2]],

"h_samples": [220, 230, 240, 250, 260, 270, 280, 290, 300, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430, 440, 450, 460, 470, 480, 490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600, 610, 620, 630, 640, 650, 660, 670, 680, 690, 700, 710],

"types": [1, 0, 1, 1, 1, 0],

"vp_point": [782, 208],

"raw_file": "issues/cross_point/1655878248_4003772743_1655878248559252640.jpg"}
```


Figure 1. Image and annotation information.


```
<image height="720" id="156" name="1655878248_4003772743_1655878248559252640.jpg" width="1280">
  <polyline
  points="1279.7422904836785,472.2148971720667;1218.2828461716801,440.3609168884091;1105.48228069661,392.0178173990932;842.4941257422238,284.9730929904654;829.5098748489553,279.31
  </polyline>
  <polyline
  points="1279.8666316302665,389.5285554280527;1175.3174395549709,362.5523444171208;1036.149714020606,327.9613882698093;956.4523830104422,308.5489984379041;901.9199389259076,296.9
  </polyline>
  <polyline
  points="1004.6061744713718,719.1210817874373;835.9674553225957,481.9026168514924;749.5903718922615,365.4989772656828;715.4475432955539,315.4418770631539;696.8117132155379,288.45
  </polyline>
  <polyline
  points="227.4856577139922,719.5710251427491;402.4963139009041,475.731237985445;484.2547148854477,367.6466825325176;526.441648164113,317.1838436824396;557.7286082511614,283.87837
  </polyline>
  <polyline
  points="0.7567274450281,472.0185596778785;220.1153616006624,371.0785758056232;339.4595777922353,318.850962945524;448.0167493348372,273.0717708890254;503.0386956357515,252.191461
  </polyline>
  <polyline
  points="0.134422873661,364.6586886339365;99.734230857044,336.2727743291164;204.5211247958892,306.9909866277914;292.3212289179178,283.8396135112899;334.9100064973317,272.85406242
  </polyline>
</image>
```


Figure 2. XML annotation information.

represent the point information of each lane. The "point" attribute of the "polyline" element is a series of semicolon-separated point coordinates, which are used to represent the shape of the lane lines.

The coordinates of the points used to represent the lane lines in the XML file are relatively few, and a lot of detailed information about the shape of the lane lines will be lost. To further refine the lane line point labels, in this experiment, the lower half of the image is cut horizontally every 10 pixels from 220 to 710 as shown in Figure 3, and the lane line point is the intersection of the red labeled line in the horizontal direction of the image and the center line of the lane line represented by the coordinate points in XML so that all the lane lines in the image have the same vertical coordinates as the subscript points. The transformed lane information is saved in the JSON file, where "lanes" is the information of the horizontal coordinates of each lane, "lanes" is a two-dimensional array, and the number of lanes containing a one-dimensional array is the number of lanes in this image. "lanes" includes 6 one-dimensional arrays, which means that there are 6 lanes in the image pointed to by this annotation, and the data in each one-dimensional array that is greater than zero is the horizontal coordinate of the pixel point in this lane. "h_samples" is the vertical coordinate of the lane lines, "h_samples" is one-dimensional, and the length of each one-dimensional array of "lanes" is the same as The length of each one-dimensional array of "lanes" is the same as the length of "h_samples", and the combination of any one-dimensional array of "lanes" and "h_samples" can obtain the horizontal coordinates of all points of the For example, the coordinates of the first point of the first lane in the figure are (721, 220).

The comparison between XML annotation information and JSON annotation information is shown in Figure 4. It can be seen that the lane line points in the JSON file are denser, and the curvature information can be retained relatively more accurately in places where the curvature of the lane line is relatively large.

- Lane Line Attributes Labeling

In the process of manual labeling, there is no labeling classification of the lane

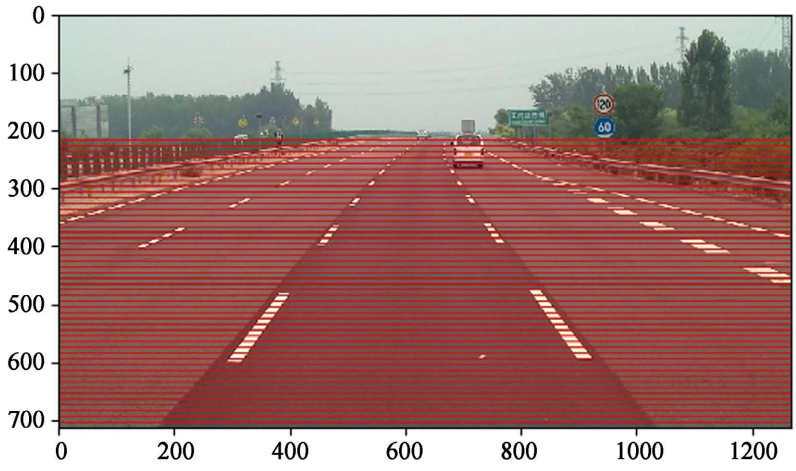


Figure 3. Horizontal cutting line.

Figure 4. XML annotation and JSON annotation visualization comparison.

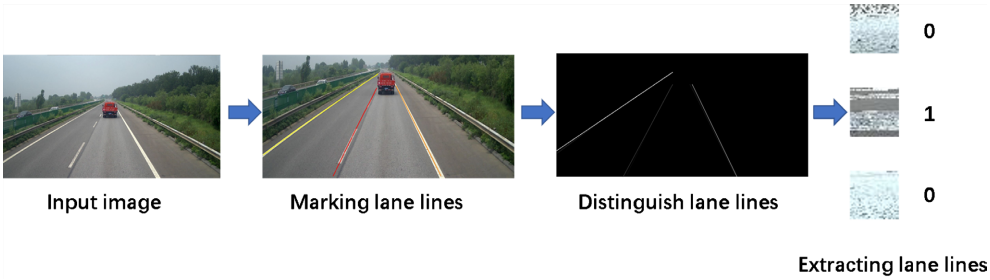


Figure 5. Lane line attributes dataset acquisition.

line's real and imaginary attributes, but the imaginary line can be crossed to represent the ability to change lanes, and the solid line cannot be crossed to limit the vehicle's drivable area, and the imaginary and real lane lines are related to the planning of the vehicle's forward direction.

In this experiment, we use a dummy real attribute classification network to accomplish the prediction and labeling. The data set acquisition process of the imaginary and real attribute classification network is shown in Figure 5. For each lane line, the length is no more than 500 pixels (labeled from 220 - 710 height), the lane lines are labeled by connecting the lane line points with lines of different color widths of 2 pixels, and the labeled image is converted into a grayscale map. In the grayscale map, the lane line pixel values are specified according to the labeling order, such as the background pixel value is 0, the first In the grayscale map, the pixel values of the lane lines are specified according to the order of labeling, such as the background pixel value is 0, the first labeled lane line pixel value is 1, the second labeled lane line pixel value is 2, and so on, until all lane lines in the labeled image are completed. The location information of each lane line point is obtained according to the different grayscale image pixel values, and all the pixel points of each lane line are extracted in the original input image by the location information index, and the lane lines are folded and stitched into a 32 * 32 pixel size image. These stitched images have certain features and can be recognized and classified by the neural network. The 3 lane lines in the figure can be extracted from 3 attribute images. The lane lines are solid lines with a label noted as 0, and the lane lines are dashed lines with a label noted as 1.

By the above lane line extraction method, the lane line attribute image data with the size of 32 * 32 is collected as shown in Table 1.

In the process of attribute classification training, two network structures were

designed. The first network structure is shown in Figure 6, where the input 32 * 32 lane line attribute image, the backbone network consists of three layers of convolution plus pooling, two layers of linear regression, and the output predicts the confidence that the input image is a solid line (0) or a dashed line (1). In this paper the network is defined as lane line attribute classification network A. In this network was trained for 50 iterations, 100 iterations, and 150 iterations, respectively, and the training process is shown in Figure 7.

The second network structure is shown in Figure 8, where a 32 * 32 lane line attribute image is input and two layers of convolution are performed before the pooling operation. The network also consists of three layers of convolution plus

Table 1. Image dataset of lane line attributes.

	0 (Full Line)	1 (Broken Line)
Training set	8871	8103
Test set	300	350

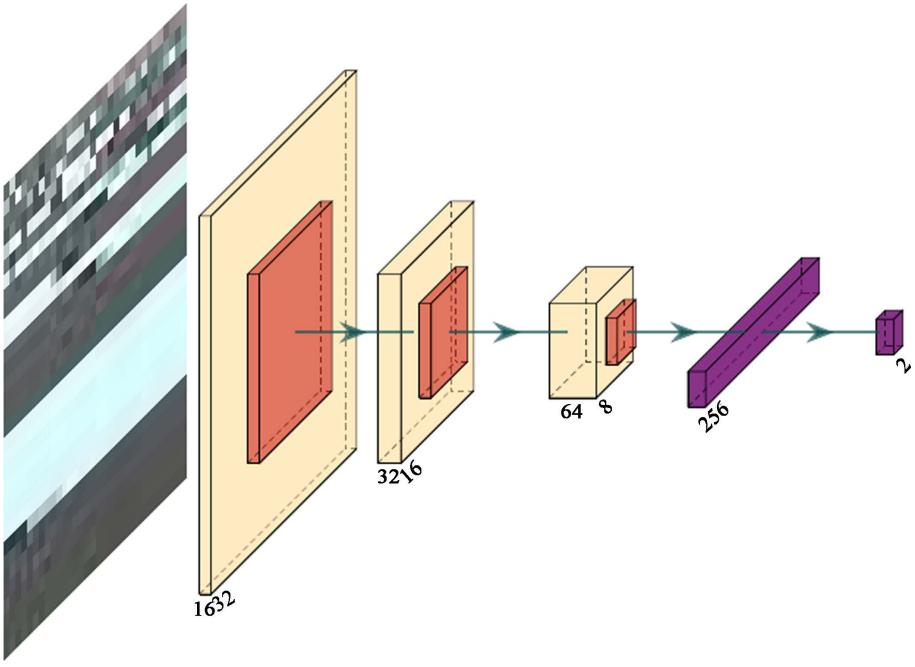


Figure 6. Lane line attribute classification network A.

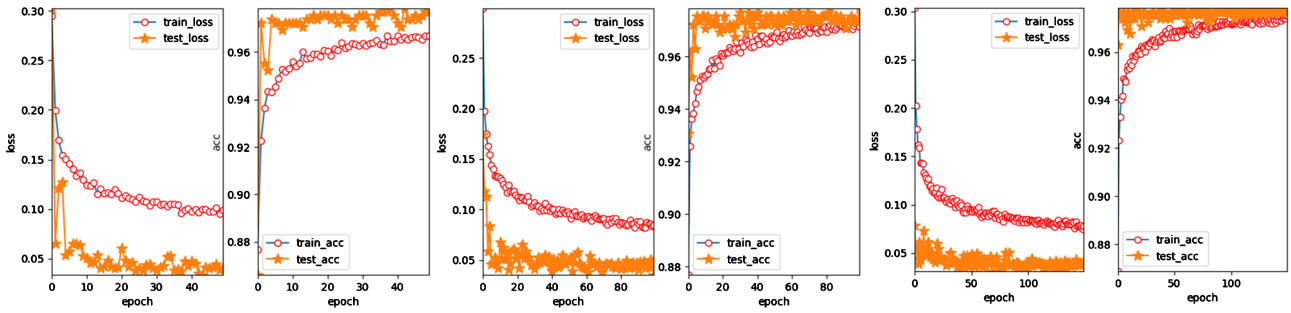


Figure 7. Network A training 50, 100 and 150 times.

pooling and two layers of linear regression, and the final output predicts the confidence level of whether the image is a dashed (1) or a solid (0) line. In this paper, the network is defined as the lane line attribute classification network B. In this network also 50 iterations, 100 iterations, and 150 iterations of training experiments are conducted, and the training process is shown in Figure 9.

There are three evaluations of the lane line attribute classification network result model, which are the solid line prediction accuracy (*Full_line_acc*), the dashed line prediction accuracy (*Broken_line_acc*), and the total accuracy (*Full_acc*), which are calculated as shown below.

$$Full\_line\_acc = \frac{T_{pred\_full}}{N_{um_{full}}} \tag{1}$$

$$Broken\_line\_acc = \frac{T_{pred\_broken}}{N_{um_{broken}}} \tag{2}$$

$$Full\_acc = \frac{T_{pred\_full} + T_{pred\_broken}}{N_{um_{full}} + N_{um_{broken}}} \tag{3}$$

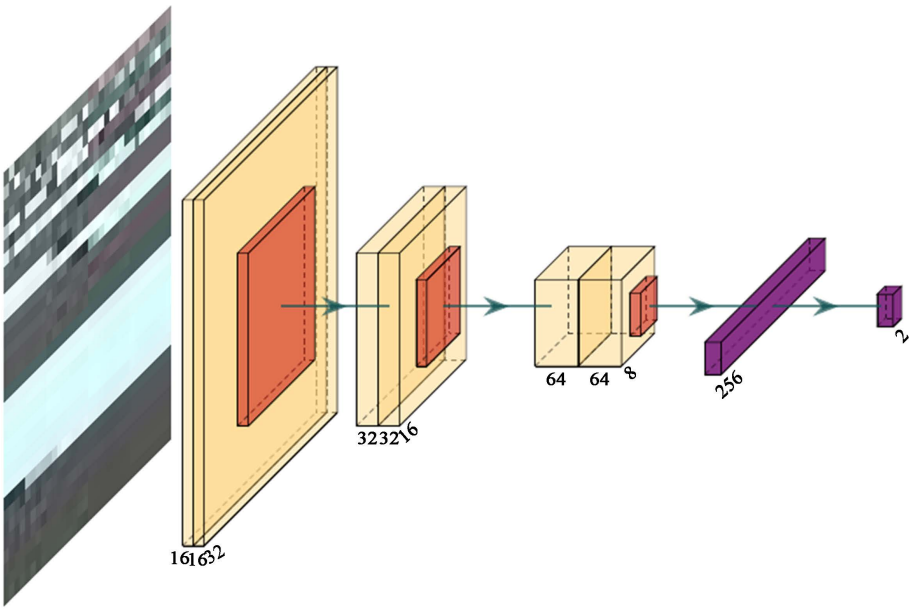


Figure 8. Lane line attribute classification network B.

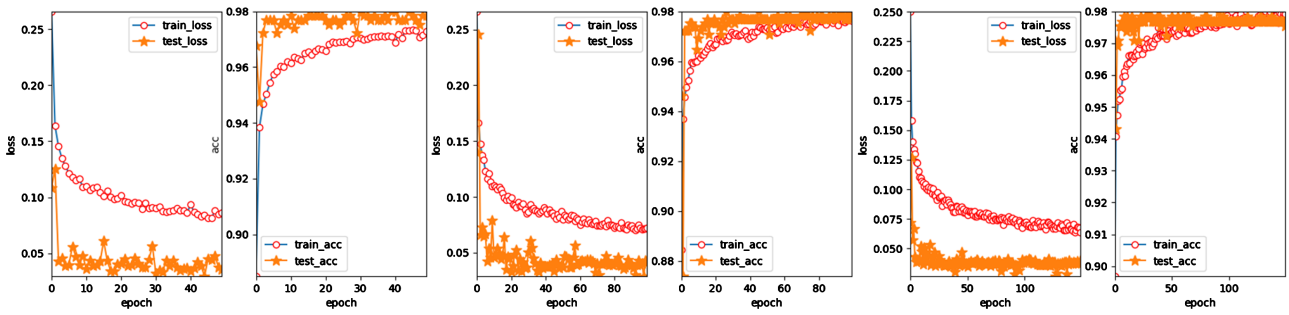


Figure 9. Network B training 50, 100 and 150 times.

where T_{pred_full} is the number of correctly predicted solid images, $N_{um_{full}}$ is the number of total solid images, and T_{pred_broken} is the number of correctly predicted dashed images, $N_{um_{broken}}$ is the number of total dashed images.

In each training process, the model with the smallest difference between the true value and the predicted value is retained as the resultant model, and the results of the resultant model evaluated on the test set are shown in Table 2.

In the In the above table, it can be found that after training network A iterations 100 and iterations 150 have the same result, which is better than the result of iteration 50, indicating that the model training has converged, and the best result of network A training is 99.22% of the total accuracy of the test set prediction. Network B is trained, and the same results are the same for 100 iterations and 150 iterations, and the best result of network B training is 99.38% correct prediction rate of the test set. Therefore, the model trained by network B can be used to predict and calibrate the attributes of the original image lane line dataset as the true value, and the labeling results are shown as "types" in the JSON file in Figure 1, with solid lines marked as 0 and solid lines marked as 1.

- Lane Line Vanishing Point Labeling

The lane line image is a 1280 * 720 size perspective image taken by the camera. The lane lines no longer remain parallel to each other in the figure but converge on the far side due to the perspective principle of small and large, and the lane lines or the extensions of the lane lines intersect at a point on the far side, which is the extinction point of the lane lines. In this experiment, the information of the lane line extinction point is labeled by two methods, and the results obtained by the two labeling methods are compared to select the more reasonable and accurate way to label the lane line training data set.

The first method is shown in the figure, predefining the vanishing point coordinates and mapping the captured image to an aerial view by performing an inverse perspective transformation (IPM) through the vanishing point coordinates, if the lane lines in the aerial view are parallel to each other, that is, the vanishing point position is correctly defined. Before explaining the implementation principle, it is necessary to understand the two coordinate systems involved in image processing.

Table 2. Comparison of experimental results.

Name of network	Epoch	Accuracy of full line prediction	Accuracy of broken line prediction	Total Accuracy
A	50	1.0000	0.9826	0.9906
	100	1.0000	0.9855	0.9922
	150	1.0000	0.9855	0.9922
B	50	1.0000	0.9854	0.9922
	100	1.0000	0.9884	0.9938
	150	1.0000	0.9884	0.9938

World coordinate system: represents the 3-D space of the real world, describing the camera position, with the origin in m depending on the situation.

Pixel coordinate system: represents the 2-D image space of the projected 3-D scene, the origin is the upper left corner of the image, unit pixel.

A schematic diagram of the relationship between the world coordinate system and the pixel coordinate system is shown in Figure 10.

The camera acquires an image point q with pixel coordinates (u,v), corresponding to the world coordinate point $Q(x,y,z)$. The process of shooting image mapping is to project it onto the $z=0$ plane of the 3D world space.

The mounting position of the camera of the shooting car is shown in Figure 11, and its world coordinates are (l,w,h), and the angle between the projection

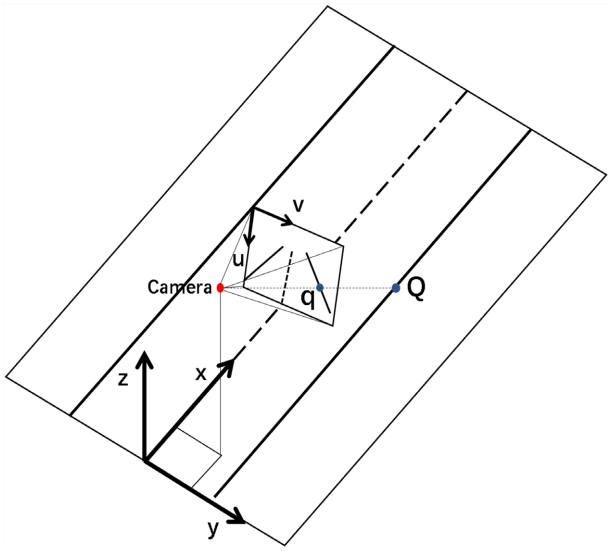


Figure 10. Diagram of the relationship between the world coordinate system and the pixel coordinate system.

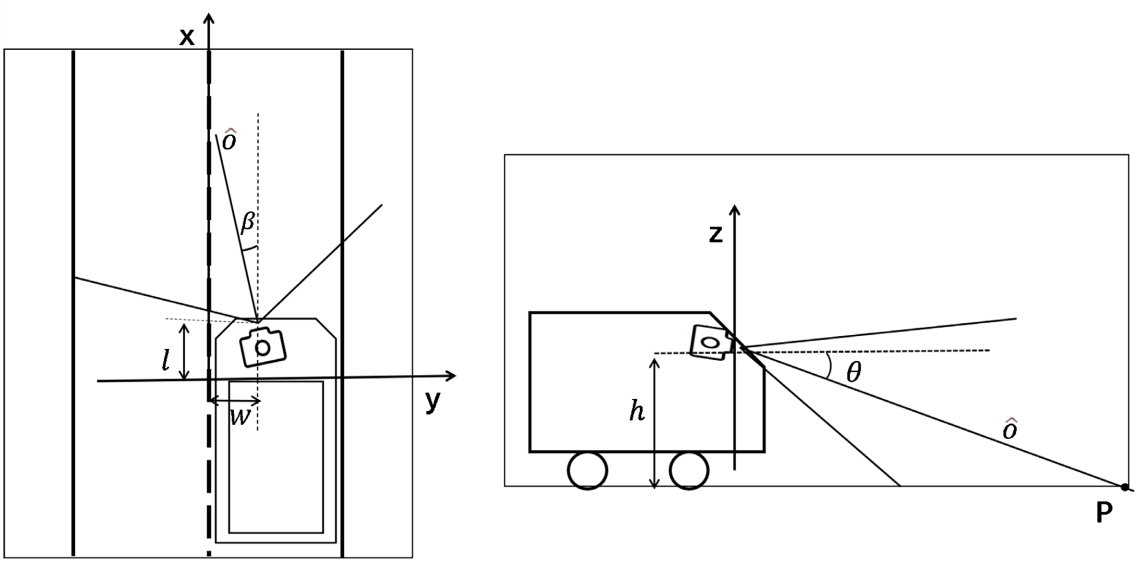


Figure 11. The left is the *xy*_plane view in world coordinates; the right is the *z*-plane view in world coordinates.

of the camera optical axis $\hat{o}$ on the $z=0$ plane and the x-axis is the translation angle β, as shown in the diagram on the right side of Figure 11, and the angle between the vertical line of the optical axis $\hat{o}$ and the z-axis is the tilt angle θ, in this paper β and θ can be calculated by predefined extinction point positions, the aperture angle of the camera is 2α, and the resolution of the camera is $m \times n$, then for the coordinate points (u,v) of the pixel coordinate system mapped to the coordinate points (x,y,z) of the world coordinate system the conversion relation [17] is:

$$x(u,v) = h \times \cot\left[(\theta-\alpha)+u\frac{2\alpha}{m-1}\right] \times \cos\left[(\beta-\alpha)+v\frac{2\alpha}{n-1}\right]+l \tag{4}$$

$$y(u,v) = h \times \cot\left[(\theta-\alpha)+u\frac{2\alpha}{m-1}\right] \times \sin\left[(\beta-\alpha)+v\frac{2\alpha}{n-1}\right]+w \tag{5}$$

where $u=0,1,\cdots,m-1$ and $v=0,1,\cdots,n-1$, any point (u,v) of the pixel coordinate system returns a coordinate point $(x,y,0)$ of the world coordinate system.

The coordinate transformation relationship of the inverse mapping process is shown below.

$$u(x,y,0) = \frac{\operatorname{arccot}\left[\frac{h\sin(x,y,0)}{y-w}\right]-(\theta-\alpha)}{\frac{2\alpha}{m-1}} \tag{6}$$

$$v(x,y,0) = \frac{\operatorname{arccot}\left[\frac{y-w}{x-l}\right]-(\beta-\alpha)}{\frac{2\alpha}{n-1}} \tag{7}$$

The inverse mapping process defined above removes perspective effects by correlating hypothetical real-world coordinate points $(x,y,0)$ with pixel points $(u(x,y,0),v(x,y,0))$ of the inverse mapped image in the form of a scan.

In this experiment, the coordinates of the predefined vanishing point are $(vptx,vpty)$, and the camera translation angle β and tilt angle θ are defined as follows, respectively.

$$\beta = -\frac{(vptx-m)\times\alpha}{m} \tag{8}$$

$$\theta = -\frac{(vpty-n)\times\alpha}{n} \tag{9}$$

As shown in Figure 12, the original image and the image generated after inverse mapping by calculating the camera translation angle β and tilt angle θ based on the predefined vanishing point positions are shown. The diagram on the left side of Figure 12 shows the perspective image taken by the camera, and the diagram on the right side of Figure 12 shows the bird's-eye view of the road area in front of the vehicle observed from a significant height generated by the inverse mapping transformation. If the lane lines in the aerial view are transformed into equally spaced parallel lines with the same structure as the lane lines in the real world, it is proved that the selected vanishing point position is correct and the vanishing point information is retained.

The second method is labeled by a computational method. As shown in Figure 13, the minimum adjacency matrix of each lane line is calculated to obtain the centerline perpendicular to the short edge, and each lane line is represented by the calculated centerline, and the average coordinates of the intersection of all centerlines are calculated as the coordinates of the vanishing point.

The above method for the normal high-speed scene calculation process is shown in Figure 14. Figure 14 (a1) is the originally captured image, Figure 14 (a2) is the minimum adjacency matrix for each lane line point, and Figure 14 (a3) calculates the centerline of each adjacency matrix, all centerlines do not necessarily intersect at one point, in this experiment, the average coordinates of all intersection points of all centerlines are calculated, as shown in Figure 14, there are three lane lines in the image, the lane lines intersect two by two, there can be at most three intersection points, the average coordinates of these three intersection points are calculated as the vanishing point coordinates of this image, Figure 14 (a4) visualizes the calculated vanishing point coordinates.

However, this calculated method does not perform well for other complex scenes. In order to improve the robustness of the method and to adapt more to real-world high-speed scenes, the scenes that do not perform well are targeted in

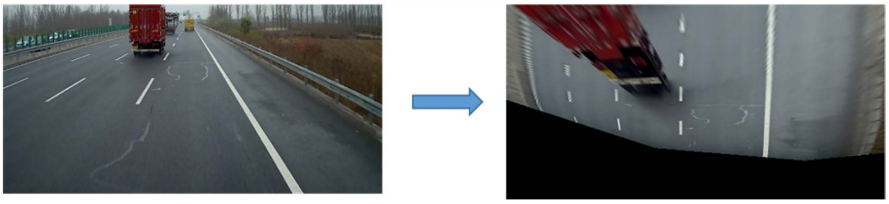


Figure 12. Left is the original image; right is the transformed bird's eye view.

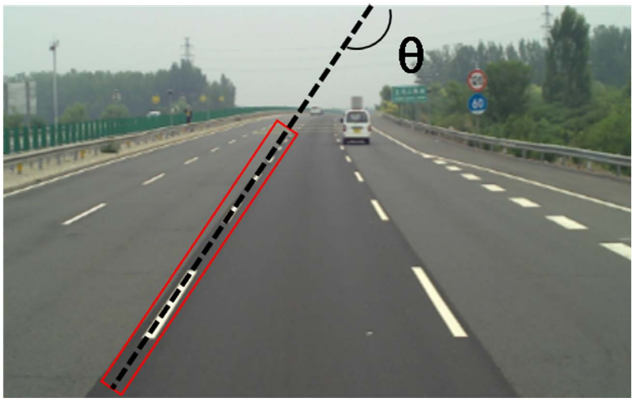


Figure 13. Lane line representation.

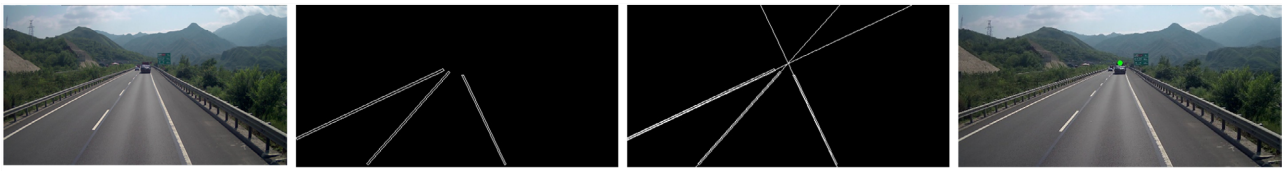


Figure 14. Normal high-speed scene.

this paper.

For a scene with only one lane, there is no intersection between the lane lines, as shown in Figure 15. Find out the width w, length h, the center point of the matrix ($upper\_x, upper\_y$) and the angle γ between the center line of the matrix and the positive direction of the image x-axis, then the center point of the short side of the distal end of the minimum adjacency matrix of the lane lines ($upper\_x, upper\_y$) can be obtained according to the following formula

$$upper\_x = center\_x - \frac{h}{2} \times \cos\gamma \tag{10}$$

$$upper\_y = center\_y - \frac{h}{2} \times \sin\gamma \tag{11}$$

For a scene with only one lane line in the image, the vanishing point is visualized in the image as shown in Figure 15 (a4).

For the ramp scene, the distal end of the lane line curved arc, the calculation method of the vanishing point does not meet the reality, to adapt to this scene, as shown in Figure 16, in this paper to intercept the proximal two-thirds of the lane line area, discard the distal end of the curved obvious one-third area, the location of the calculated vanishing point in the image visualization as shown in Figure 16 (a4).

For the lane line convergence intersection vanishing point calculation error is relatively large, the convergence intersection at the lane line and other lane lines distal end will overlap, as shown in Figure 17, lane line l_2 convergence lane line l_1, two-lane lines intersect at point c, in Figure 17 (a3) can be seen lane line l_2 and other lane lines intersection point will have significantly deviated from the other lane lines between the intersection point, this case to seek the average coordinates as the vanishing point To solve this interference, this paper additionally proposes a filtering method: with the help of the intersection point c of the lane lines at the convergence intersection, find the lane lines whose distance to point c is less than 10 pixels in the image, *i.e.*, lane line l_1 and lane line l_2,

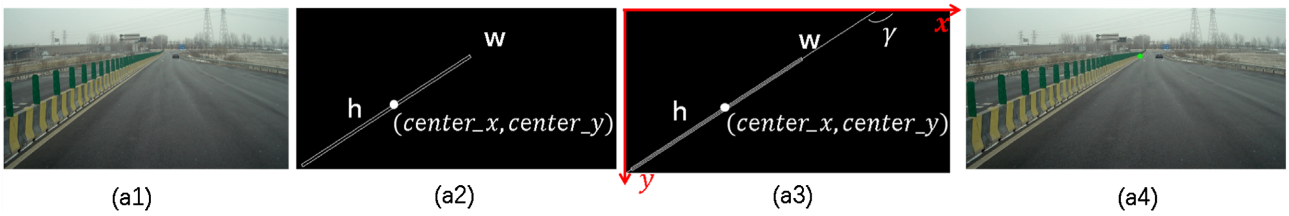


Figure 15. One lane line scene.

Figure 16. Ramp scene.

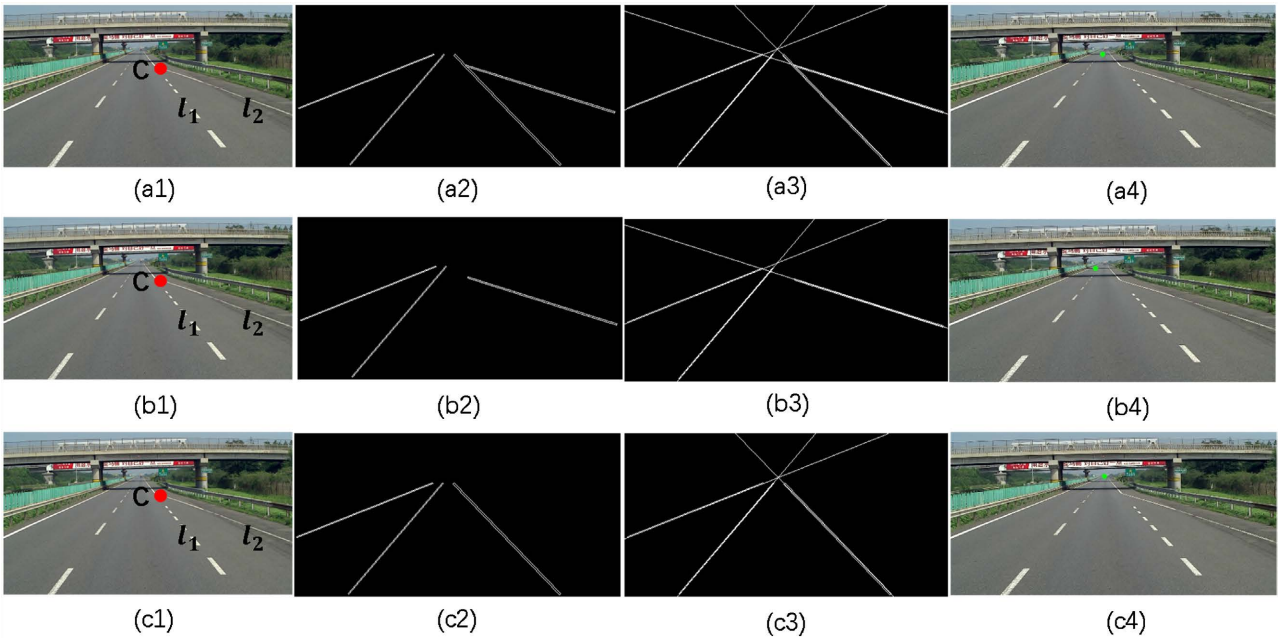


Figure 17. Converge into the intersection scene.

and filter these two lane lines respectively, as shown in **Figure 17** (a4). Similarly, filter lane line l_2 as shown in **Figure 17** (b2) and calculate the distance dis1 from the vanishing point location to the remaining three lane lines obtained after filtering lane line l_1. Similarly, filter lane line l_2 as shown in **Figure 17** (c2) and calculate the distance dis2 from the vanishing point location to the remaining three lane lines obtained after filtering lane line l_2. dis0, dis1, and dis2 are selected as the smallest result, whose corresponding vanishing point location is the vanishing point of this convergence scene, compare the vanishing point markers in **Figure 17** (a4), **Figure 17** (b4), and **Figure 17** (c4), obviously the vanishing point location in **Figure 17** (c4) is more in line with the actual cognition.

For the scene of high speed intersection, because the lane lines of the upper and lower high speed intersection and the direction of the lane lines of the normal driving road deviate more, so it will cause interference to the calculation of the vanishing point, as shown in **Figure 18** (a4), because of the influence of the lower high speed intersection, the position of the vanishing point deviates compared to the driver's visual center, and the distance dis0 from the vanishing point to the lane line is calculated by recording all the lane lines in the image. For the scene of highway intersection, a lane line filtering method is proposed in this paper: there are four lane lines l_1, l_2, l_3 and l_4 in the image, as shown in **Figure 18** (b3), **Figure 18** (c3), **Figure 18** (d3) and **Figure 18** (e3), filter these four lane lines respectively, use the remaining three lane lines to calculate the vanishing point, and record the calculated distance from the vanishing point to the remaining three lane Compare dis0, dis1, dis2, dis3 and dis4, and select the one with the smallest value as the vanishing point position of the image. As in **Figure 18** (e4), the location of the vanishing point after filtering out the lower highway lane line l_4 is more in line with the actual scene.

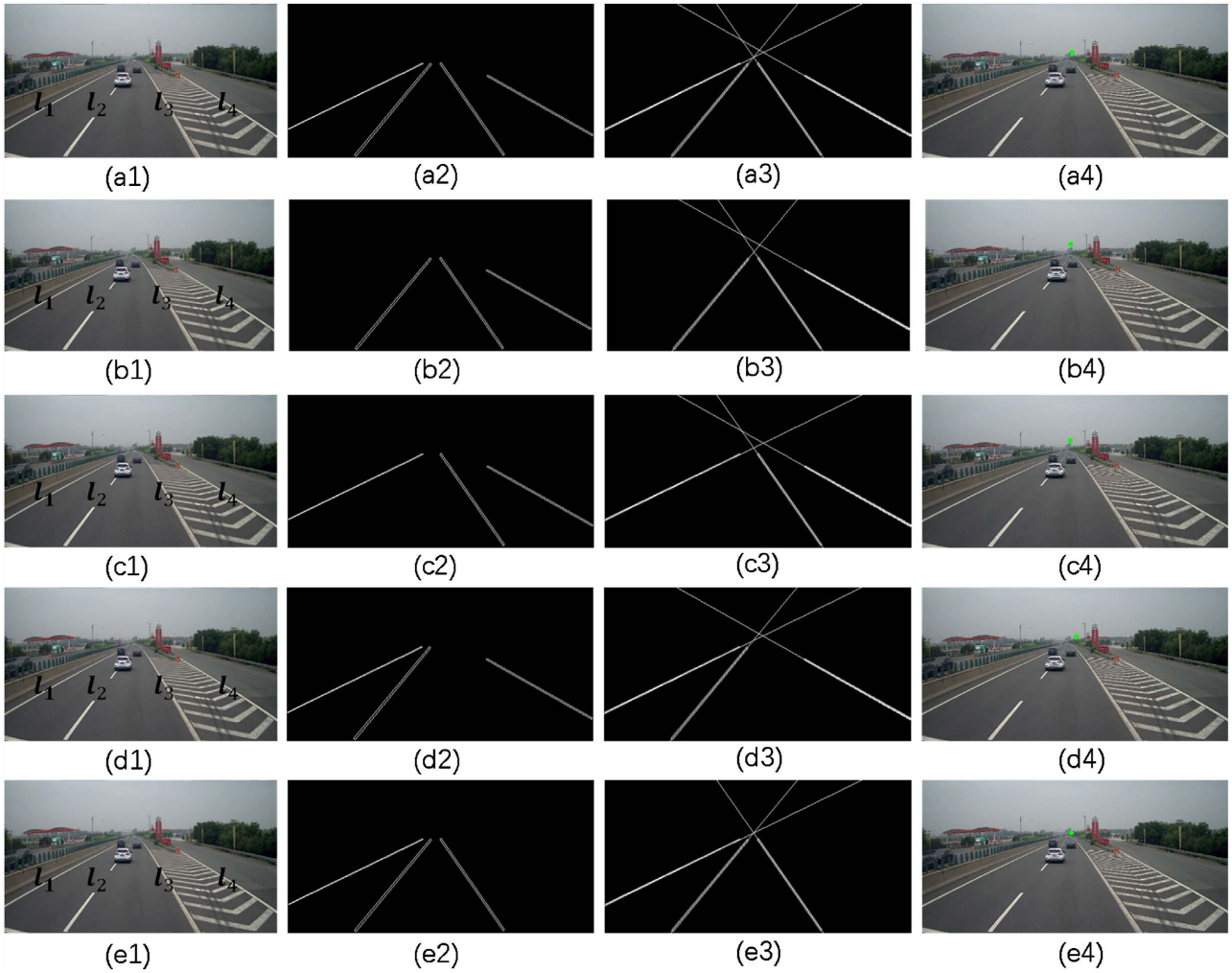


Figure 18. Highway intersection scene.

The first method of finding the coordinates of the vanishing point of the inverse perspective transformation is too subjective for each marker, each person has a different concept of parallel lane lines, resulting in a different adjustment size for the location of the extinguishing points that make the lane lines parallel to each other, and different markers have a larger error in the location of the vanishing point obtained for the same scene; in addition, in the ramp scene where the curvature of the lane lines is relatively large, itself in the conversion after the In addition, in ramp scenes with large curvature of lane lines, it is difficult to make the lane lines parallel to each other in the bird's-eye view after conversion. The second method of finding the vanishing point by calculation is more comprehensive for different scenes, and the error is smaller for the same scene with a fixed calculation method, so we use the calculation method to calibrate the true value of the vanishing point coordinates of the lane line data set.

3.2. Network Structure

In this paper, we consider lane line detection as an instance segmentation prob-

lem for continuous elongated objects and propose a lane line detection method based on an improved PINet [18] network. The structure of PINet training network is shown in **Figure 19**, the input size of 512 * 256 images and lane line labeling information, in the size Resizing layer, first through a series of convolution and pooling, the module for preliminary image feature extraction, and the output image of size 64 * 32 is sent to the Feature extraction layer, which is based on the Stacked Hourglass Network [19] and consists of two stacked hourglass network modules. The output of each Hourglass Network module is divided into three branches: Confidence branch, Offset branch, and lane line point instance classification branch (Feature). To improve the detection efficiency, the PINet network does not perform pixel-level discrimination and detection on the original 512 * 256 size image but performs lane line detection on the reduced 64 * 32 size image, so the predicted output of the network will have some deviation. To further improve the accuracy of lane line detection, the improved lane line detection network structure is shown in **Figure 20**. The 1280 * 720 image taken by the front camera and the labeling information (**Figure 1**) is input, the sky part above the image that is not related to lane lines is cropped, and the image size is resized to 512 * 256 after a series of data enhancements methods such as rotation, adding noise, mirror flip, etc. In addition to the lane detection branch, three other supervised and constrained branches are added, namely the lane attribute prediction branch (Type branch), the lane parallel structure prediction branch (Hara branch), and the extinction point prediction branch (Vanishing point branch). The improved network predicts the location and shape of the lane lines with pixel-level, lane line-level, and image-level constraints, and improves the prediction results by applying multi-layer structural constraints to the lane lines.

The network structure in this experiment includes a Resizing layer and a Feature extraction layer, and the Feature extraction layer is a two-layer stacked hourglass network. The details of the Resizing layer module network are shown in **Table 3**, where "layer" indicates the operation performed on the image,

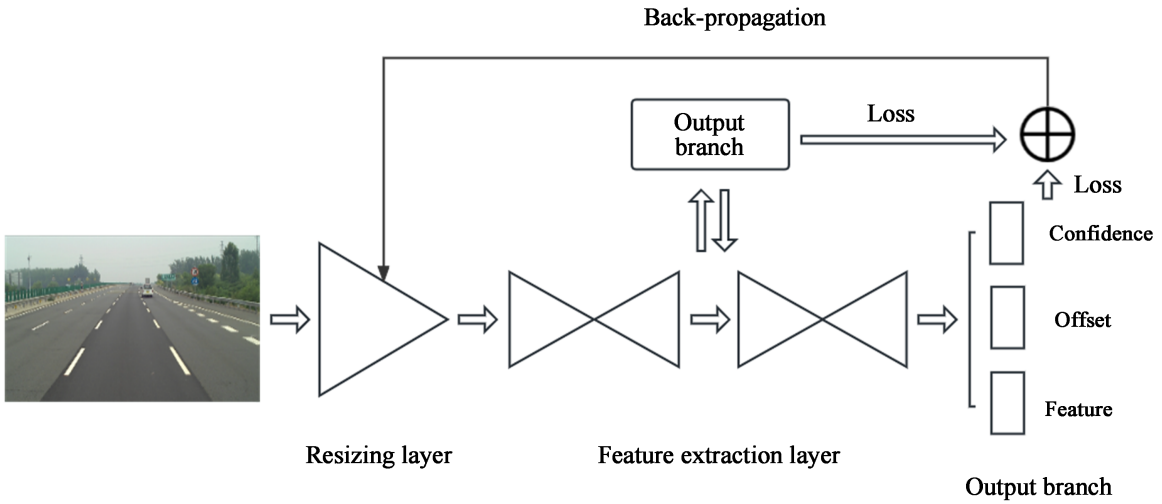


Figure 19. PINet network structure.

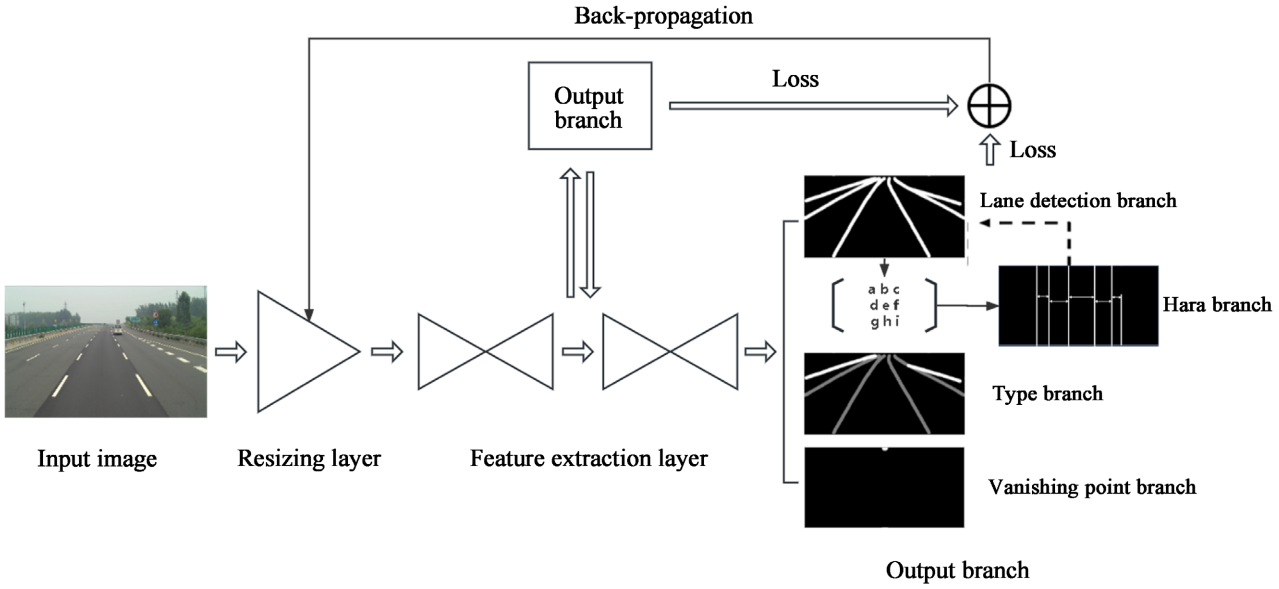


Figure 20. Improved network structure.

Table 3. Resizing layer implementation details.

Layer	Size/Stride/Padding	Output size
Input data		3 * 512 * 256
Conv + PRelu + bn	7/2/3	64 * 256 * 128
Conv + PRelu + bn	1/1/0	64 * 256 * 128
Conv + PRelu + bn	3/1/1	64 * 256 * 128
Conv + PRelu + bn	1/1/0	64 * 256 * 128
Max pooling	2/2	64 * 128 * 64
Conv + PRelu + bn	1/1/0	64 * 128 * 64
Conv + PRelu + bn	3/1/1	64 * 128 * 64
Conv + PRelu + bn	1/1/0	64 * 128 * 64
Max pooling	2/2	64 * 64 * 32
Conv + PRelu + bn	1/1/0	128 * 64 * 32
Conv + PRelu + bn	3/1/1	128 * 64 * 32
Conv + PRelu + bn	1/1/0	128 * 64 * 32

"Size", "Stride" and "Padding" indicates the convolution kernel size, convolution step, and image padding pixel value of the convolution network, or the pooling window size and sliding step of the window for the pooling operation, and "Output" indicates the number of channels * length * width of the output image. After a series of convolution and pooling, the image features are initially extracted and the size is reduced to 1/8 of the original size, *i.e.*, from 512 * 256 to 64 * 32, followed by connecting the feature extraction layers. The stacked hourglass network provides a repetitive top-down and bottom-up inference mechanism for the network to aggregate information across scales through a series of upsampling and downsampling, and the network output features retain the in-

formation of all layers and are consistent with the original input size of the hourglass network as 64 * 32, which can ensure that the lane line information is not lost compared to the traditional neural network that only uses the convolutional features of the last layer. As shown in Figure 20, the hourglass network is left-right symmetric, retaining the current scale information before performing downsampling, reducing the resolution of the features by convolution and pooling, and after reaching the minimum resolution, upsampling is performed by inverse convolution, and the data from the previous scale are summed after upsampling to achieve cross-scale feature fusion. The hourglass network structure is shown in Figure 21, such as c7 doubles the resolution by up-sampling, c4 goes through a 1 × 1 convolution to get c4a, c4a can be regarded as a copy of c4, which has the same size and dimension as c4, while the size of c4a is the same as c7 that was up-sampled, and the values are added directly to get c4b. By stacking the feature maps layer by layer in this way, the final c1b then retains the information of all layers. The output of the first layer of the hourglass network will not only be used as the input of the second layer of the hourglass network, but will also be used to determine the performance of the network model by comparing the loss of the difference between the predicted and true label values of the network through the loss function, and then find the optimization direction. the loss of the evaluation model is the sum of the losses calculated from the predicted and true values generated by all hourglass networks.

- Lane Line Prediction Branch

The lane line prediction task consists of three parts, namely the confidence branch, the offset branch, and the lane line point instance classification branch, to predict whether each pixel in the feature map is a lane line point, which pixel corresponds to the original image and which lane line it belongs to. The feature maps of each part are shown in Figure 22. In this experiment, the input image size is 512 * 256 and the hourglass network output feature map size is 64 * 32.

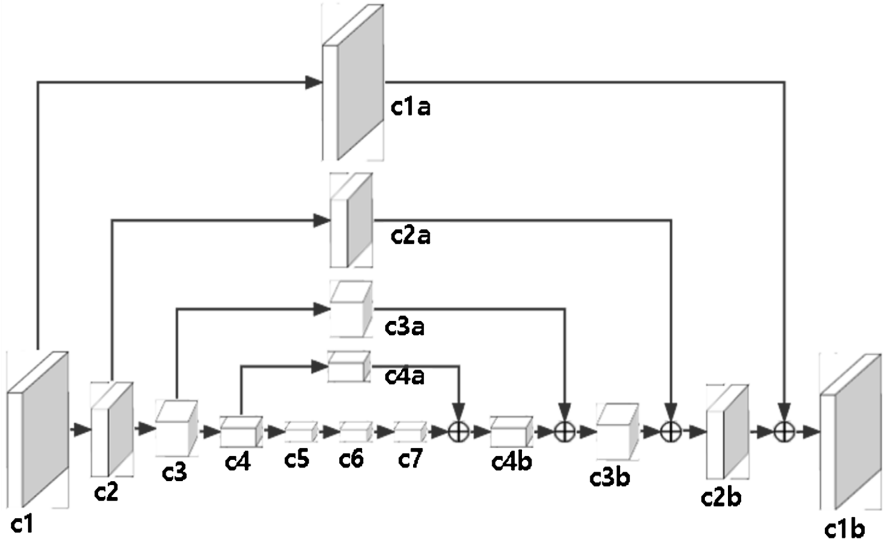
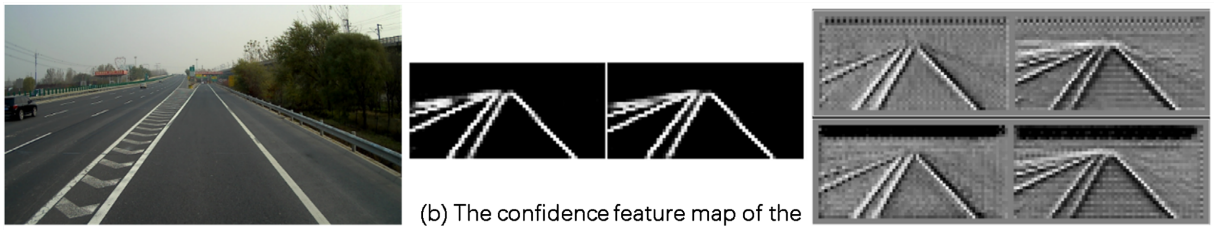


Figure 21. Hourglass network structure.



(a) Input image

(b) The confidence feature map of the output of the first layer of hourglass network in the upper row; the confidence feature map of the output of the second layer of hourglass network in the lower row

(c) The offset feature map of the output of the first layer of hourglass network in the upper row; the offset feature map of the output of the second layer of hourglass network in the lower row

Figure 22. Visualization feature map.

The feature map is 1/8 of the original image size, so one pixel of the feature map corresponds to a matrix of 8 * 8 pixels in the original input image. The confidence branch is responsible for semantic segmentation and predicts the confidence value of whether each pixel in the output feature map belongs to a lane line, if the pixel belongs to a lane line, the confidence value is close to 1, otherwise, the confidence value is close to 0. Because the size of the network input image and the network output feature map are not the same, to predict the location of the lane line points more accurately, the offset branch gives the predicted value of each lane point the output value is between 0 and 1, which is expressed as x-axis offset and y-axis offset, respectively. If the predicted x-axis offset is 0.5, then the predicted location of the point is 4 pixels away from the true value in the x-axis direction, and similarly, the predicted y-axis offset is 0.5, then the predicted location of the point is 4 pixels away from the true value in the y-axis direction. The lane line point instance classification branch predicts the high-dimensional feature (dimension 6 in this paper) of each lane line point, which is used to calculate the similarity between different lane line points. The implementation of the lane line point instance classification branch borrows from the 3D point cloud instance segmentation in SGPN [20] to predict which lane line point specifically belongs to a lane line. For each image, containing different lane line instances, according to the principle of similarity metric learning, points belonging to the same lane line instance should have similar features, and a similarity matrix is constructed to record the differences between the features of all point pairs to measure whether each point pair belongs to the same instance, as shown in **Figure 23**, for any two points (x_1, y_1) in a 64 * 32 size lane line image and (x_2, y_2), there is a corresponding coordinate (j, k) in the similarity matrix, where

$$j = x_1 + y_1 \times 64 \tag{1}$$

$$k = x_2 + y_2 \times 64 \tag{2}$$

An embedded feature value is defined at (j, k) in the similarity matrix to indicate whether the points (x_1, y_1) and (x_2, y_2) belong to the same instance, and if they belong to the same instance, the corresponding value of this feature

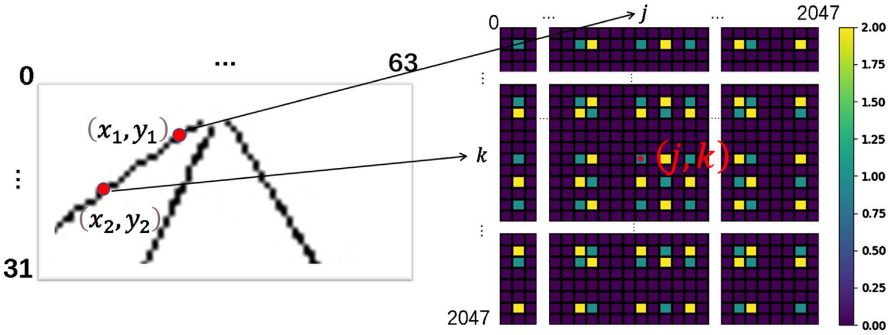


Figure 23. Relationship between similarity matrix and lane line points.

value is 1 (represented in green in the figure), otherwise, it is 2 (represented in yellow in the figure). The output predicts the instance classification features, predicts the relationship between each pixel and all other pixels, and by the similarity matrix constructed for pixels belonging to the same instance their features are expected to be closer, and for pixels of different instances the features are expected to be further apart. Then if the predicted embedding feature values of some lane line points are within a certain range, they can be considered as the same instance, and different instances are distinguished using this clustering process. The loss function $L_{feature}$ [18] of the lane line point instance classification branch is defined as:

$$L_{feature} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{N_e^2}\sum_{j=1}^{Ne}\sum_{k=1}^{Ne} f_i(j,k) \tag{3}$$

$$f_i(j,k) = \begin{cases} \left\| F_j - F_k \right\|_2 & \text{if } C_{jk} = 1 \\ \max\left(0, \lambda - \left\| F_j - F_k \right\|_2\right) & \text{if } C_{jk} = 2 \end{cases} \tag{4}$$

where N is the total number of images, i denotes the sequence number of images, Ne is the number of pixel frames in the feature map, F_j denotes the predicted embedding feature of pixel j in the feature map, and λ is a constant. C_{jk} denotes whether pixel j and pixel k are in the same lane instance, and if $C_{jk} = 1$, it means that these two pixels belong to the same instance, and if $C_{jk} = 2$, it means that these two pixels belong to different instances.

$L_{confidence}$, L_{offSet}, and $L_{feature}$ are the loss functions of the confidence branch, offset branch, and lane line point instance classification branch, respectively, so the loss of the lane line prediction branch is:

$$L_{lane} = aL_{confidence} + bL_{offSet} + cL_{feature} \tag{5}$$

- Prediction Of Lane Line Virtual And real Attributes

The real and imaginary attributes of lane lines determine the drivable range of vehicles as well as the forward direction in the real world. In the experiment, the network predicts the lane lines while predicting the imaginary and real attributes of lane lines, the location of lane line attributes in the feature map is the same as the location of lane lines, and the prediction of lane line attribute information can enhance the lane line feature information. In the experiment, the lane line

attribute prediction is implemented using the segmentation method, as shown in Figure 24, the lane line attributes truth map is assigned to each pixel grid attribute value, where the background value is 0, the dashed line value is 1, and the solid line value is 2. The lane line attribute prediction feature map outputted by the hourglass network is classified pixel by pixel to judge the existence of lane line key points by attribute information prediction at the pixel level to monitor the lane line attribute The prediction branch output feature map and the real and imaginary attributes of the lane line do the difference operation whose loss calculation function is:

$$L_{type} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{Ne}\sum_{j=1}^{Ne}\sum_{k=0}^{K-1}-\left[1-\exp\left(-y_{i,j,k}\log p_{i,j,k}\right)\right]^2\cdot 0.5\cdot y_{i,j,k}\log p_{i,j,k} \quad (6)$$

where N is the total number of images, Ne is the number of pixel cells in the feature map, and K is the number of attribute labels. The probability that the jth cell is predicted to be the kth label value is $p_{i,j,k}$, and $y_{i,j,k}$ is the label, *i.e.*, if the category of the jth cell is k, then $y_{i,j,k}=1$, otherwise, it is equal to 0.

- Prediction Of Parallel Properties Of Lane Lines

The lane lines exist parallel in the real world and lose this structure between the lane lines in the image due to the perspective principle of the camera, but the image can be converted to a bird's eye view by inverse perspective and modeled in the bird's eye view to reconstruct the parallelism between each other. the IPM transformation method can convert the original image to a bird's eye view by the perspective transformation matrix H. Due to the need to adapt to different highway scenes, the Therefore H matrix is not fixed, and in the experiments, the network predicts the parameters of the ideal perspective change based on the input image to adapt to the complex road scenes. The network predicts the lane lines, and for each lane line represented by a series of points, as shown in Figure 25, the minimum external matrix of the lane line points is found to obtain the centerline perpendicular to the short edge, and the clockwise angle between the positive x-axis and the centerline is θ. Then the predicted lane lines can be expressed as $ax+by+c=0$. As shown in Figure 26, assuming the predicted lane lines are L_1 and L_2, and project them into the bird's-eye view after the inverse perspective transformation, the corresponding instances of lane lines are l_1 and l_2, then l_1 and l_2 can be expressed as:

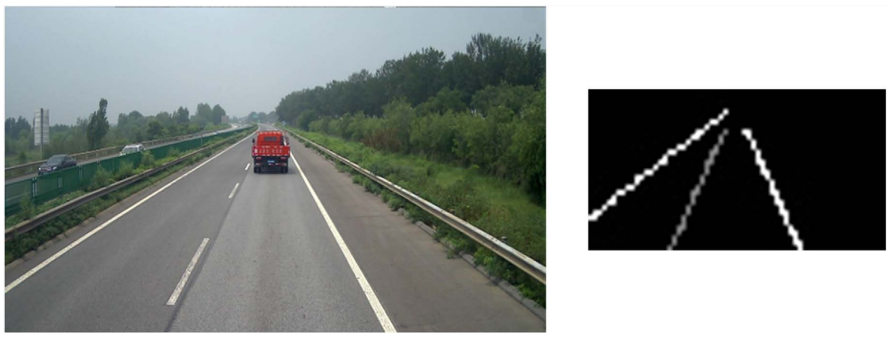


Figure 24. True value map of lane line virtual and real attributes.

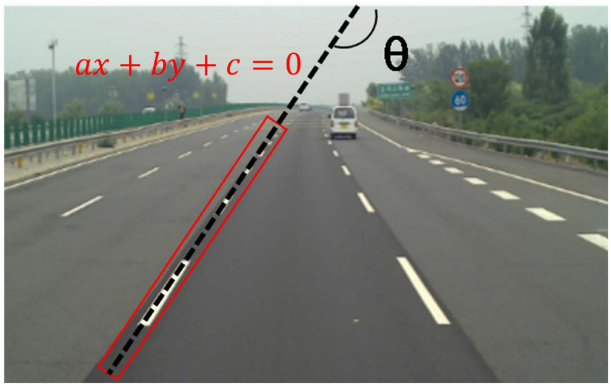


Figure 25. Lane-line representation.

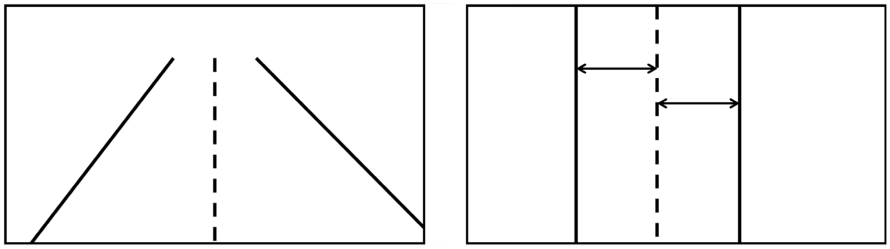


Figure 26. Bird's eye view conversion.

$$a_1 \cdot x + b_1 \cdot y + c_1 = 0 \tag{7}$$

$$a_2 \cdot x + b_2 \cdot y + c_2 = 0 \tag{8}$$

Since the lane lines in the bird's eye view have a parallel relationship with each other, the difference of x is always constant in the case of equal y. Therefore, $a_1 \cdot b_2 = a_2 \cdot b_1$, then it is possible to provide negative feedback to the network through L_H. The closer L_H is to 0, the more accurate the detection of lane lines is, thus improving the accuracy of lane line detection by constraining the relationship between lane lines.

$$L_H = \frac{1}{N}\sum_{i=1}^{N} \frac{\sum_{j=0,k=0,j\neq k}^{L_i-1} L1\left(a_j b_k - a_k b_j\right)}{\left[L_i \cdot (L_i - 1)\right]/2} \tag{9}$$

where N is the total number of images, L_i is the number of lane lines in the ith image, and $L1$ is the Smooth L1 loss function.

- Prediction of Lane Line Disappearance Point Location

The vanishing point, which is also the distant intersection of lane lines, is regarded as a special kind of key point to be predicted by feature maps in the network, and in this way, the task of predicting the vanishing point is used as a limiting and guiding factor in the training process to optimize the detection results of lane lines at the image level. After reading the JSON file to get the true value of the vanishing point, since the target of the vanishing point is too small and has only one coordinate information, a circle is drawn with the vanishing point as the center and 4 as the radius as shown in Figure 27, and all the coordinate information of the whole vanishing point marking circle is used as the

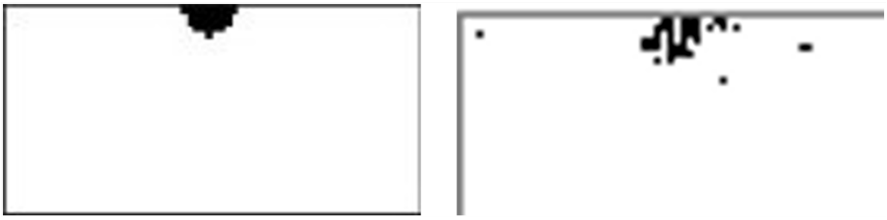


Figure 27. Left is the vanishing point true value graph, right is the vanishing point prediction graph.

true value and the feature map to do the pixel-level loss function calculation as follows.

$$L_{vp} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{Ne}\sum_{j=1}^{Ne}-\left[y\cdot\log(p)+(1-y)\cdot\log(1-p)\right] \tag{10}$$

where N is the total number of images, Ne is the number of pixel frames in the feature map, y is the extinction sample label, the positive sample label is 1 and negative sample label is 0, and p is the probability of predicting a positive sample.

4. Experimental Results and Analysis

4.1. Experimental Data

All data for this experiment come from Beijing Trunk Co., Ltd. on Beijing highways, and the datasets are all highways, which are more in line with the needs of in-house research than the TuSimple [21] dataset and the CuLane dataset, with 41,058 training data and 4171 validation data, including scenes of congestion, blocking blurred lane lines, and special weather (rain, snow, and fog).

4.2. Experimental Results

- Evaluation Metrics

There are two accuracy evaluation metrics for lane lines, *Accuracy*, *FPR*, and *FNR* on the Tusimple dataset and F_1 *Score* on the CULane dataset, each of the two evaluation metrics has its focus, and both metrics are used to evaluate the training model in this experiment.

Where the Accuracy is calculated as:

$$Accuracy = \Sigma_{im}\frac{C_{im}}{S_{im}} \tag{1}$$

C_{im} is the number of correctly predicted lane points, and S_{im} is the number of lane points in the real label. When the distance between the predicted points and the actual lane points is less than the set threshold, the lane points are considered to be correctly predicted. A lane line is considered correctly predicted when the percentage of correctly predicted points in a lane line exceeds 85%.

FPR is defined as follows:

$$FPR = \frac{Fpred}{Npred} \tag{2}$$

FNR is defined as follows:

$$FNR = \frac{Mpred}{Ngt} \tag{3}$$

Fpred is the total number of lane lines predicted incorrectly (false checks), *Npred* is the total number of lane lines predicted, *Mpred* is the number of lane lines not predicted but present (missed checks), and *Ngt* is the number of all lane lines in the label.

F_1Score, also known as the balanced F Score, considers both accuracy and recall and is a reconciliation balance between the two.

$$F_1Score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4}$$

$$precision = \frac{TP}{TP + FP} \tag{5}$$

$$recall = \frac{TP}{TP + FN} \tag{6}$$

The CULane dataset renders lane points as 30-pixel-wide lane lines and calculates the intersection-to-merge ratio (IoU) between the predicted lane lines and the real lane lines. IoU greater than a specific threshold (0.5) is considered as correct prediction (*TP*), IoU less than or equal to the threshold is considered as wrong detection (*FP*), and having real lane lines but no detection is considered as *FN*.

- Experiment Comparison

To demonstrate the effectiveness of the experimental approach in this paper, several experiments were conducted to compare the model in this paper with the PINet model, and to compare the performance variation of this approach in the test set. To explore the effectiveness of pixel-level, lane-level, and image-level constraint structures, this paper combines pixel-level perception with the Base model (Base + T) and adds the lane line constraints into the Base + T combination as Base + T + H. The results of the ablation experiments can be seen in Table 4, where the accuracy of lane line detection (*Accuracy* and F_1Score) is improving, proving that the method proposed in this paper is effective that used together can improve the performance of lane line detection.

Figure 28 shows the performance of this experimental model in different scenarios with different colors for different lane line instances. The top row shows the true value of the lane line labels, and the bottom row shows the predicted lane line results of this experimental model.

Table 4. Comparison of ablation experiments.

Network	*Accuracy* (%)	*FPR*	*FNR*	F_1 *Score*
Base	0.9224	0.1785	0.1133	0.8538
Base + T	0.9532	0.0956	0.0666	0.9220
Base + T + H	0.9652	0.0791	0.0567	0.9289
Ours	0.9644	0.0561	0.0502	0.9415

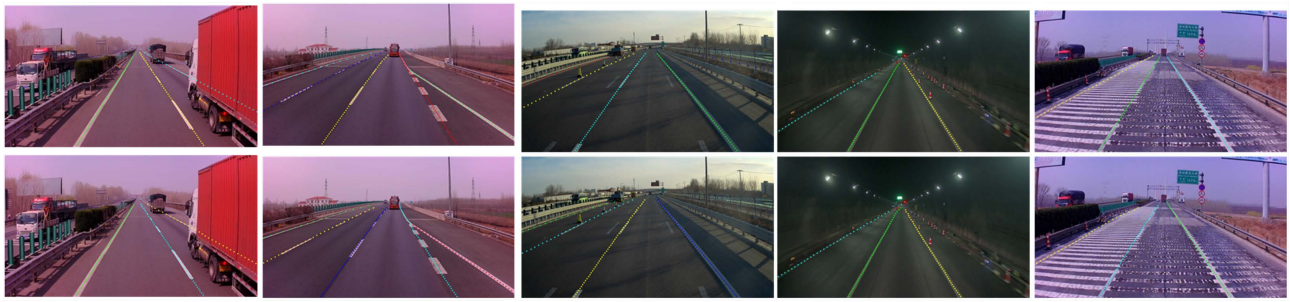


Figure 28. Experimental results.

4.3. Conclusion

In this paper, information closely related to lane lines such as the virtual and real properties of lane lines, the parallel structure of lane lines, and the distant intersection of lane lines are utilized to constrain the structure of lane line perception in multiple layers, and through ablation experimental comparison, it is shown that the method in this paper is effective and can improve the accuracy of lane line detection and significantly reduce the error detection rate, thus ensuring the safety of unmanned vehicles. However, there is still much room for improvement in the accuracy of the lane detection task. The current model performs well in most scenarios, but underperforms in complex scenarios where lanes cross and merge into and out of each other, and such complex road scenarios are also accident-prone, requiring more accurate predictions.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Chen, T., Zhang, H.D., Chen, D. and Tan, C. (2016) Lane Detection Based on High Priority Pixels and Tracking by Kalman Filter. *Automotive Engineering*, **38**, 200-205.

[2] Mammeri, A., Boukerche, A. and Tang, Z. (2016) A Real-Time Lane Marking Localization, Tracking and Communication System. *Computer Communications*, **73**, 132-143. https://doi.org/10.1016/j.comcom.2015.08.010

[3] Bertozzi, M. and Broggi, A. (1998) GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection. *IEEE Transactions on Image Processing*, **7**, 62-81. https://doi.org/10.1109/83.650851

[4] Lee, J.W. (2002) A Machine Vision System for Lane-Departure Detection. *Computer Vision & Image Understanding*, **86**, 52-78. https://doi.org/10.1006/cviu.2002.0958

[5] Cai, F.Y., Gao, L., Sun, X.Q., Chen, L. and Wang, H. (2018) A Lane Identification Method Based on Morphological Features. *China Mechanical Engineering*, **29**, 1863-1868.

[6] Qu, X., Yu, F. and Zhao, Y. (2018) Algorithm for Lane Detection Based on Hyperbolic Model. *Journal of Hubei University of Automotive Technology*, **32**, 52-55.

[7] Wang, X.-C., Lu, X.-C., Zhang, H.-S., Xiao, Y.-M. and Xie, M. (2018) A Lane Detection Method Based on Parallel Coordinate System. *Journal of University of Elec-*

tronic Science and Technology of China, **47**, 362-367.

[8] Bai, B., Han, J.F., Pan, S.H., Lin, C. and Yuan, H.D. (2013) Lane Detection Based on Dual-Threshold Segmentation. *Information Technology*, **37**, 43-45.

[9] Liu, P. and Sun, Y. (2019) Lane Detection Algorithm Based on Inverse Perspective Mapping. *Computer & Digital Engineering*, **47**, 678-681.

[10] Pan, X., Shi, J., Luo, P., Wang, X.G. and Tang, X.O. (2018) Spatial as Deep: Spatial CNN for Traffic Scene Understanding. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, 2-7 February 2018, 7276-7283. https://doi.org/10.1609/aaai.v32i1.12301

[11] Hou, Y., Ma, Z., Liu, C. and Loy, C.C. (2019) Learning Lightweight Lane Detection CNNs by Self Attention Distillation. *Proceedings of* 2019 *IEEE/CVF International Conference on Computer Vision* (*ICCV*), Seoul, 27 October-2 November 2019, 1013-1021. https://doi.org/10.1109/ICCV.2019.00110

[12] Li, X., Li, J., Hu, X. and Yang, J. (2019) Line-CNN: End-to-End Traffic Line Detection with Line Proposal Unit. *IEEE Transactions on Intelligent Transportation Systems*, **21**, 248-258. https://doi.org/10.1109/TITS.2019.2890870

[13] Girshick, R. (2015) Fast R-CNN. *Proceedings of* 2015 *IEEE International Conference on Computer Vision* (*ICCV*), Santiago, 7-13 December 2015, 1440-1448. https://doi.org/10.1109/ICCV.2015.169

[14] Su, J., Chen, C., Zhang, K., Luo, J.F., Wei, X.M. and Wei, X.L. (2021) Structure Guided Lane Detection. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence* (*IJCAI*-21), Montreal, 19-26 August 2021, 997-1003.

[15] Qin, Z., Wang, H.Y. and Li. X. (2020) Ultra Fast Structure-Aware Deep Lane Detection. In: Vedaldi, A., Bischof, H., Brox, T. and Frahm, J.M., Eds., *ECCV* 2020: *Computer Vision-ECCV* 2020, *Lecture Notes in Computer Science*, Vol. 12369, Springer, Cham, 276-291.

[16] Tabelini, L., Berriel, R., Paixão, T.M., Badue, C., De Souza, A.F. and Oliveira-Santos, T. (2020) PolyLaneNet: Lane Estimation via Deep Polynomial Regression. *Proceedings of* 2020 25*th International Conference on Pattern Recognition* (*ICPR*), Milan, 10-15 January 2021, 6150-6156. https://doi.org/10.1109/ICPR48806.2021.9412265

[17] Bertozzi, M. and Broggi, A. (2021) GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection. *IEEE Transactions on Image Processing*, **7**, 62-81. https://doi.org/10.1109/83.650851

[18] Ko, Y., Lee, Y., Azam, S., *et al.* (2021) Key Points Estimation and Point Instance Segmentation Approach for Lane Detection. *IEEE Transactions on Intelligent Transportation Systems*, **23**, 8949-8958.

[19] Newell, A., Yang, K. and Deng, J. (2016) Stacked Hourglass Networks for Human Pose Estimation. In: Leibe, B., Matas, J., Sebe, N. and Welling, M., Eds., *ECCV* 2016: *Computer Vision-ECCV* 2016, *Lecture Notes in Computer Science*, Vol. 9912, Springer, Cham, 483-499. https://doi.org/10.1007/978-3-319-46484-8_29

[20] Wang, W., Yu, R., Huang, Q. and Neumann, U. (2018) SGPN: Similarity Group Proposal Network for 3D Point Cloud Instance Segmentation. *Proceedings of* 2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), Salt Lake City, 18-23 June 2018, 2569-2578. https://doi.org/10.1109/CVPR.2018.00272

[21] TuSimple. (2017) Tusimple Lane Detection Challenge. https://github.com/TuSimple/tusimple-benchmark