

Article

Vision-Based Algorithm for Precise Traffic Sign and Lane Line Matching in Multi-Lane Scenarios

Kerui Xia ¹, Jiqing Hu ², Zhongnan Wang ² , Zijian Wang ² , Zhuo Huang ² and Zhongchao Liang ^{2,*} 

¹ Department of Electrical Engineering, Tsinghua University, Beijing 100089, China; keruixia@mail.tsinghua.edu.cn

² School of Mechanical Engineering and Automation, Northeastern University, Shenyang 110819, China; 2370139@stu.neu.edu.cn (J.H.); wangzhongnan@stumail.neu.edu.cn (Z.W.); wangzijian@stumail.neu.edu.cn (Z.W.); zhuohuang@stumail.neu.edu.cn (Z.H.)

* Correspondence: liangzc@me.neu.edu.cn

Abstract: With the rapid development of intelligent transportation systems, lane detection and traffic sign recognition have become critical technologies for achieving full autonomous driving. These technologies offer crucial real-time insights into road conditions, with their precision and resilience being paramount to the safety and dependability of autonomous vehicles. This paper introduces an innovative method for detecting and recognizing multi-lane lines and intersection stop lines using computer vision technology, which is integrated with traffic signs. In the image preprocessing phase, the Sobel edge detection algorithm and weighted filtering are employed to eliminate noise and interference information in the image. For multi-lane lines and intersection stop lines, detection and recognition are implemented using a multi-directional and unilateral sliding window search, as well as polynomial fitting methods, from a bird's-eye view. This approach enables the determination of both the lateral and longitudinal positioning on the current road, as well as the sequencing of the lane number for each lane. This paper utilizes convolutional neural networks to recognize multi-lane traffic signs. The required dataset of multi-lane traffic signs is created following specific experimental parameters, and the YOLO single-stage target detection algorithm is used for training the weights. In consideration of the impact of inadequate lighting conditions, the V channel within the HSV color space is employed to assess the intensity of light, and the SSR algorithm is utilized to process images that fail to meet the threshold criteria. In the detection and recognition stage, each lane sign on the traffic signal is identified and then matched with the corresponding lane on the ground. Finally, a visual module joint experiment is conducted to verify the effectiveness of the algorithm.

Keywords: computer vision; multi-lane recognition; traffic sign detection; matching algorithm



Citation: Xia, K.; Hu, J.; Wang, Z.; Wang, Z.; Huang, Z.; Liang, Z. Vision-Based Algorithm for Precise Traffic Sign and Lane Line Matching in Multi-Lane Scenarios. *Electronics* **2024**, *13*, 2773. <https://doi.org/10.3390/electronics13142773>

Academic Editor: Felipe Jiménez

Received: 4 June 2024

Revised: 8 July 2024

Accepted: 10 July 2024

Published: 15 July 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The rapid development of intelligent transportation systems (ITS) worldwide is paving the way for the emergence of fully autonomous vehicles. In order to guarantee the secure and effective operation of fully automated vehicles on the road, it is crucial to implement precise lane detection and traffic sign recognition technologies, which form part of advanced driver assistance systems (ADAS) [1–3]. Lane detection and traffic sign recognition are foundational for vehicles that can perceive their environment. The accuracy and robustness of these systems directly impact the safety and reliability of autonomous driving systems. Lane detection technology enables vehicles to maintain their lane positioning, while adaptive cruise control facilitates interactions with the traffic environment and aids in navigation. This enhances the safety and convenience of driving.

Among the numerous methods for lane detection, two particularly common techniques are Hough transform (HT) and bird's eye view (BEV) polynomial fitting. The HT is a feature extraction technique that transforms lines in the image space into the parameter space. This method has demonstrated robustness against noise and broken lines in the image; however,

it is associated with a high computational complexity. In addition, complex scenes require parameter adjustments to account for different types of lane lines, and the Hough transform requires longer processing times for real-time applications [4,5]. BEV polynomial fitting transforms the road image in front of the vehicle into a BEV, which is a perspective viewed from above, and then fits a polynomial curve to the lane lines in this view. This method simplifies the three-dimensional road scene into a two-dimensional plane, reducing the complexity of the problem. However, it requires accurate camera calibration and image transformation processing and can be affected by changes in the road environment, such as unclear or missing road markings [6].

To enhance the accuracy and robustness of lane detection, this paper employs the sliding window method. The sliding window method involves the movement of a window within a specific area of the image, typically the lower half, to identify the peaks of white pixels that determine the position of the lane lines. This method is straightforward and efficient, and capable of being executed swiftly, making it suitable for real-time lane detection [7,8]. However, the current lane detection methods were primarily designed for single-lane detection. In actual vehicle operation, the environment is typically multi-lane, making multi-lane detection a more practically significant issue. The primary purpose of lane line detection is to facilitate the measurement of vehicle distance and position, which is critical for path tracking [9] and advanced driver assistance systems (ADAS). The implementation of lane line distance measurement and positioning usually involves a variety of technical methods, including sensor fusion techniques [10], machine learning methods [11], and stereo vision methods [12,13]. The approach to distance measurement and positioning proposed in this article employs an innovative “standard line” positioning strategy. This method utilizes image processing techniques to analyze the detected real lane lines. Histogram techniques are applied to determine the position of the “standard line.” Subsequently, the horizontal coordinate position of the “standard line” in the pixel coordinate system is obtained. This allows the intelligent vehicle to determine its lane.

A further challenge arises in instances where the lane detection is unable to ascertain the occupied lane. Therefore, combining traffic signs with lane line detection is a more practical approach. The current common methods for traffic sign recognition include template-matching-based methods, feature-based methods, and deep-learning-based methods. Template-matching-based methods identify traffic signs by comparing images with predefined templates. They are simple to implement for recognizing known signs. However, they are susceptible to occlusions, deformations, or alterations in lighting conditions [14,15]. Feature-based methods extract image features for matching. These methods offer good robustness to rotation, scaling, and partial occlusion, but they have high computational complexity and are sensitive to noise and changes in lighting conditions [16]. Deep-learning-based methods, such as convolutional neural networks (CNNs), are capable of learning feature representations from training data. These methods are adept at handling complex image variations and exhibit strong generalization capabilities [17–20]. This paper employs the YOLO detection algorithm, a method rooted in deep learning that facilitates real-time image processing and that is perfectly aligned with scenarios necessitating immediate response times. In consideration of the impact of inadequate lighting conditions, the application of SSR enhancement is employed with the objective of enhancing the recognition accuracy in such scenarios, which represents a novel aspect of this paper.

Traffic signs are detected and then matched with the current lane detection. Unlike other single-lane studies [21], this paper achieves recognition and matching of multi-lane lines and traffic signs [22–25], ensuring the accuracy and practicality of the perception of unmanned driving vehicles. Based on the above discussion, the innovative aspects of this paper are as follows:

1. The sliding window method is employed to achieve multi-lane detection.
2. A custom dataset is employed to train the traffic sign recognition model, and SSR enhancement is applied to enhance the recognition accuracy under poor lighting conditions.
3. The system employs a matching multi-lane configuration, with the integration of multiple traffic signs.

The organization of the remainder of this article is as follows: Section 2 introduces the use of computer vision technology to achieve the detection and positioning of lane lines, as well as the identification and classification of traffic signs through the YOLO model, together with the matching of these signs with lane lines to provide the road information required for autonomous vehicles. Section 3 constructs and employs a custom multi-lane traffic sign dataset to train the YOLO object detection model, and enhances the accuracy of traffic sign detection under various lighting conditions through image enhancement techniques and model optimization methods. In Section 4, joint experimental validation confirms the performance of the algorithm under conditions simulating lane lines and traffic signs. Finally, Section 5 presents the research conclusions.

2. Multi-Lane Detection and Distance Localization Based on Computer Vision

2.1. Image Preprocessing

Due to the complex and dynamic nature of road environments, each frame of raw road images often contains a significant amount of irrelevant information and noise [26]. Image preprocessing is a crucial process that involves removing irrelevant details from images, enhancing image data, and improving lane line features. These steps are essential for subsequent image processing tasks.

The imaging process of a camera involves transforming a three-dimensional scene into a two-dimensional color image. Essentially, this process entails a series of coordinate transformations, wherein the world coordinate system is converted into the pixel coordinate system, as in Figure 1.

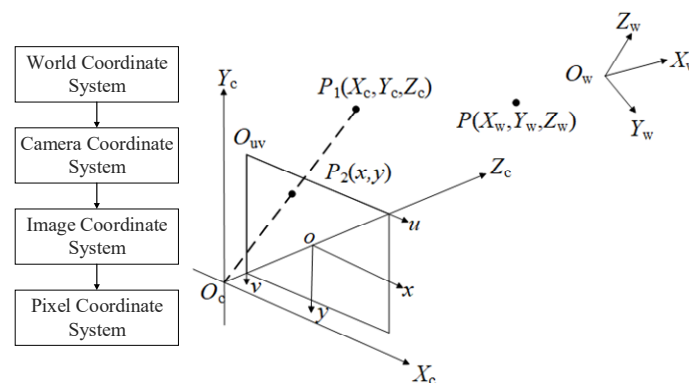


Figure 1. Coordinate systems and transformations.

The world coordinate system ($O_w - X_w Y_w Z_w$) is a three-dimensional Cartesian coordinate system, which is primarily used to define the spatial positions of the camera and detected objects. The units employed in this system are in meters. The camera coordinate system ($O_c - X_c Y_c Z_c$) is a three-dimensional Cartesian coordinate system, with the camera optical center as the origin. The optical axis is aligned with the Z_c -axis, while the X_c and Y_c -axes are parallel to the two sides of the camera surface. The image coordinate system ($o - xy$) and the pixel coordinate system ($ouv - uv$) are two-dimensional Cartesian coordinate systems. The image coordinate system is defined to lie on the image plane, with the intersection of the optical axis designated as its origin.

The x and y axes are, respectively, parallel to the X_c and Y_c -axes of the camera coordinate system, with units in millimeters. The pixel coordinate system is defined with its

origin at the top-left corner of the image. In this system, the columns and rows of pixels respectively represent the horizontal coordinate (u) and vertical coordinate (v) with units in pixels. Additionally, there is a translation relationship between the horizontal coordinate (u) and the vertical coordinate (v).

By referencing Figure 1, the world coordinate system can be transformed into the pixel-coordinate system as follows:

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} d_x^{-1} & 0 & u_0 \\ 0 & d_y^{-1} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (1)$$

where \mathbf{R} and \mathbf{t} respectively represent the rotation matrix and translation vector in the extrinsic parameter matrix. $[X_c \ Y_c \ Z_c \ 1]^T$ represent the homogeneous coordinates of any point in the camera coordinate system. $[X_w \ Y_w \ Z_w \ 1]^T$ represent the homogeneous coordinates of the corresponding point in the world coordinate system. d_x and d_y represent the physical dimensions of a single pixel on the x -axis and y -axis of the camera coordinate system, with units in millimeters. $f_x = \frac{f}{d_x}$ and $f_y = \frac{f}{d_y}$ are the normalized focal length on the x -axis and y -axis of the camera, with units in pixels. (u_0, v_0) represents the point in the pixel coordinate system corresponding to the origin of the image coordinate system.

$\begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ represent the camera's intrinsic parameters, while $\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$ represent the camera's extrinsic parameters.

Assuming the chessboard grid in the world coordinate system lies on a plane, specifically $Z_w = 0$, combining with (1), it can be deduced that the holography from the plane where the chessboard is located to the image plane is as follows:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{r}_3 \ \mathbf{t}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{r}_3] \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}, \quad (2)$$

where \mathbf{K} represents the in-camera parameter matrix; s represents the scale factor; and $\mathbf{r}_1, \mathbf{r}_2,$ and \mathbf{r}_3 are the three components of the rotation matrix \mathbf{R} ; by denoting the three-dimensional world coordinates as $\mathbf{M} = [X_w \ Y_w \ Z_w \ 1]^T$ and the two-dimensional pixel coordinates as $\mathbf{m} = [u \ v \ 1]^T$, the above can be simplified as

$$s\mathbf{m} = \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}] \mathbf{M}, \quad (3)$$

Assuming $\mathbf{H} = \lambda \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}]$ and that λ remains a constant factor, \mathbf{H} is defined as a homograph matrix. Upon column-wise partitioning of \mathbf{H} , the result is

$$\mathbf{H} = [\mathbf{h}_1 \ \mathbf{h}_2 \ \mathbf{h}_3] = \lambda \mathbf{K} [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}], \quad (4)$$

Combining Equations (3) and (4) yields

$$\begin{cases} \lambda = s^{-1} \\ \mathbf{r}_1 = \lambda^{-1} \mathbf{K}^{-1} \mathbf{h}_1 \\ \mathbf{r}_2 = \lambda^{-1} \mathbf{K}^{-1} \mathbf{h}_2 \end{cases}, \quad (5)$$

In this context, r_1 and r_2 denote two components of the rotation matrix R . Consequently, r_1 and r_2 are orthogonal, with magnitudes $|r_1| = |r_2| = 1$. According to the related properties of rotation and orthogonal matrix, the following can be inferred:

$$\begin{cases} r_1^T r_2 = 0 \\ r_1^T r_1 = r_1^T r_2 = 1 \end{cases} \quad (6)$$

Combining Equations (5) and (6) yields

$$\begin{cases} h_1^T K^{-T} K^{-1} h_2 = 0 \\ h_1^T K^{-T} K^{-1} h_1 = h_2^T K^{-T} K^{-1} h_2 \end{cases} \quad (7)$$

Using matrix K as a basis, a symmetric matrix $B = K^{-T} K^{-1}$ can be constructed:

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{f_x^2} & -\frac{1}{f_x^2 f_y} & \frac{v_0 - u_0 f_y}{f_x^2 f_y} \\ -\frac{1}{f_x^2 f_y} & \frac{1}{f_x^2 f_y^2} + \frac{1}{f_y^2} & -\frac{v_0 - u_0 f_y}{f_x^2 f_y^2} - \frac{v_0}{f_y^2} \\ \frac{v_0 - u_0 f_y}{f_x^2 f_y} & -\frac{v_0 - u_0 f_y}{f_x^2 f_y^2} - \frac{v_0}{f_y^2} & \frac{(v_0 - u_0 f_y)^2}{f_x^2 f_y^2} + \frac{v_0}{f_y^2} + 1 \end{bmatrix} \quad (8)$$

Equation (8) indicates that there are six valid elements in matrix B . Thus, these six elements are defined as a new six-dimensional vector $b = [B_{11} \ B_{12} \ B_{22} \ B_{13} \ B_{23} \ B_{33}]^T$. Additionally, the i -th column vector of the holography matrix H can be represented as $h_i = [h_{i1} \ h_{i2} \ h_{i3}]^T$. Therefore, it can be concluded as follows:

$$\begin{cases} h_i^T B h_j = V_{ij}^T b \\ V_{ij} = [h_{i1} h_{j1} \ h_{i1} h_{j2} + h_{i2} h_{j1} \ h_{i2} h_{j2} \ h_{i3} h_{j1} + h_{i1} h_{j3} \ h_{i3} h_{j2} + h_{i2} h_{j3} \ h_{i3} h_{j3}]^T \end{cases} \quad (9)$$

Combining with (7) yields

$$\begin{bmatrix} V_{12}^T \\ (V_{11}^T - V_{22}^T) \end{bmatrix} b = 0, \quad (10)$$

Utilizing the above, capturing three or more images of a chessboard grid facilitates the calculation of the symmetric matrix B . Subsequently, employing Cholesky decomposition yields the camera's intrinsic parameter matrix K [27], providing various internal camera parameters. Incorporating formula (5) and the properties of the rotation matrix $R = [r_1 \ r_2 \ r_3]$, the following can be deduced:

$$\begin{cases} \lambda = s^{-1} = (\|K^{-1} h_1\|)^{-1} = (\|K^{-1} h_2\|)^{-1} \\ r_1 = \lambda^{-1} K^{-1} h_1 \\ r_2 = \lambda^{-1} K^{-1} h_2 \\ r_3 = r_1 \times r_2 \\ t = \lambda^{-1} K^{-1} h_3 \end{cases} \quad (11)$$

Solving (11) results in the camera's extrinsic parameter matrix $\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$. Calibration of the intrinsic and extrinsic parameters of a monocular camera using the calibration tool provided by the OpenCV open-source library was implemented based on the Zhang Zhengyou method. Fifteen images of a chessboard grid taken from different angles were selected as calibration inputs. The calibration was performed using the `calibrate Camera ()` function provided by the OpenCV open-source library, followed by distortion correction using the `undist ()` function.

2.2. Color Space Transformation

In the detection of four-lane roads, it is common to encounter edge detection failures in lanes that are farther away from the current lane. When performing longitudinal positioning, it is necessary to detect stop lines in both directions of the four lanes.

To enhance the effectiveness of detecting multiple lane lines and stop lines, a combination of multiple color space transformations coupled with Sobel edge detection is used [28].

The RGB color space is converted to Lab, HLS, and Luv color spaces, respectively, and finally combined with edge detection for processing. The Lab color space comprises three channels: one for luminance and two for color. These channels are designed to optimally align with the human eye's recognition and perception of different colors. Specifically, it is a perceptually uniform and device-independent color space. Each color is characterized by three parameters: L, a, and b. Here, L represents the lightness of the pixel within the image, with a range of [0, 100]. The 'a' channel represents the color channel from red to green, with a range of [127, -128], and the 'b' channel represents the color channel from yellow to blue, with a range of [127, -128]. The process involves initially converting the RGB color space to the XYZ color space. This is followed by a subsequent conversion to the Lab color space, as illustrated in Equations (12)–(14):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}, \quad (12)$$

$$\begin{cases} L = 116f(Y/Y_n) - 16 \\ a = 500[f(X/X_n) - f(Y/Y_n)] \\ b = 200[f(Y/Y_n) - f(Z/Z_n)] \end{cases}, \quad (13)$$

$$f(t) = \begin{cases} t^{1/3}, & \text{if } t > (\frac{6}{29})^3 \\ \frac{1}{3}(\frac{29}{6})^2 t + \frac{4}{29}, & \text{otherwise} \end{cases}, \quad (14)$$

where X_n , Y_n , and Z_n are typically assumed to be 95.047, 100.0, and 108.883, respectively. Figure 2a illustrates the conversion to the Lab color space.

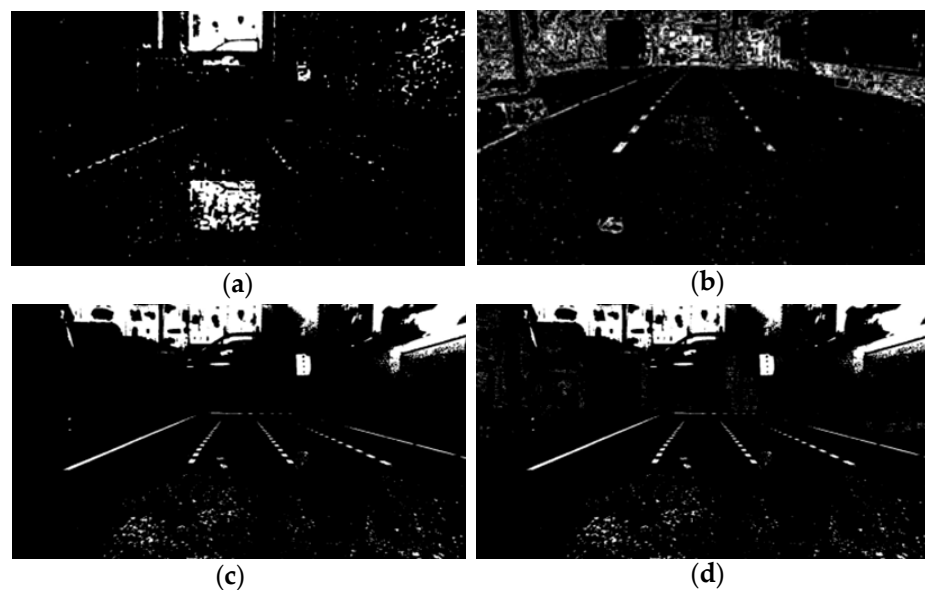


Figure 2. The result of color space transformation and combined processing. (a) Lab effect diagram; (b) HLS effect diagram; (c) Luv effect diagram; (d) Combination effect diagram.

The HLS color space comprises three components: H (hue), L (lightness), and S (saturation). The allowable ranges for H, L, and S are typically defined as [0, 179], [0, 255], and [0, 255], respectively. In the HLS color space, white is solely determined by the lightness component (L), and most of the lane lines are white, as employing HLS enhances the accuracy of lane detection. The conversion from RGB to HLS is achieved through the application of (15). The values of R, G, and B are real values within the range of [0, 1], $\max = \max\{R, G, B\}$, $\min = \min\{R, G, B\}$, so $H \in [0, 360]$, $L \in [0, 1]$, $S \in [0, 1]$. The conversion to the HLS color space is illustrated in Figure 2b.

The Luv color space exhibits visual uniformity, serving as an intermediate space derived from the RGB color space via the XYZ color space. It comprises three components: L (lightness), u (a change in chromaticity from green to red), and v (a change in chromaticity from blue to yellow). L, u, and v fall within the following intervals: the range for L is [0, 100], while that for u and v is [−100, 100]. In the Luv color space, colors are delineated into two primary components: luminance and chromaticity.

$$\left\{ \begin{array}{l} H = \begin{cases} 0^\circ & \text{if } \max = \min \\ 60^\circ \times \frac{G - B}{\max - \min} + 0^\circ & \text{if } \max = R \text{ and } G \geq B \\ 60^\circ \times \frac{G - B}{\max - \min} + 360^\circ & \text{if } \max = R \text{ and } G < B \\ 60^\circ \times \frac{B - R}{\max - \min} + 120^\circ & \text{if } \max = G \\ 60^\circ \times \frac{R - G}{\max - \min} + 240^\circ & \text{if } \max = B \end{cases} \\ L = \frac{1}{2}(\max + \min) \\ S = \begin{cases} 0 & \text{if } L = 0 \text{ or } \max = \min \\ \frac{\max - \min}{\max + \min} = \frac{\max - \min}{2L} & \text{if } 0 < L \leq \frac{1}{2} \\ \frac{\max - \min}{2 - (\max + \min)} - \frac{\max - \min}{2 - 2L} & \text{if } L > \frac{1}{2} \end{cases} \end{array} \right. , \quad (15)$$

In multi-lane detection, the colors may become blurred due to various factors, including distance, deviations in color, and the presence of distant lanes, grayish lanes, or stop lines (as depicted in Figure 3). White is a color with high brightness, whereas gray is essentially a less bright variation of white. In terms of chromaticity, white and gray are similar, but their distinction lies mainly in brightness. Utilizing the Luv color space for filtering can improve the accuracy of identifying such scenarios.

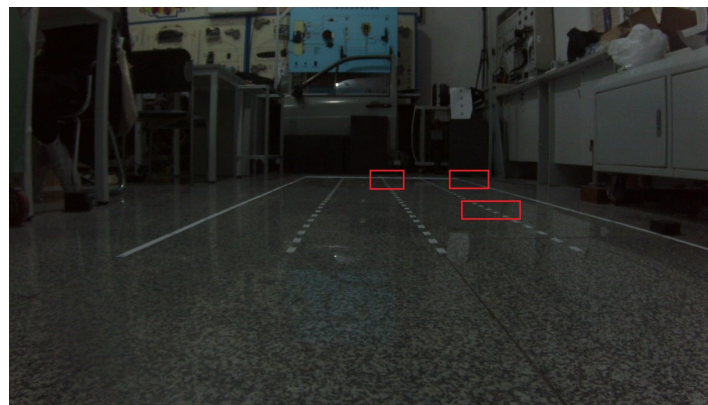


Figure 3. Visualizing color discrepancies. The lane lines and stop lines within the red-framed area appear to be gray in color.

Combining formulas (12) and (16) to perform RGB to Luv conversion.

$$\left\{ \begin{array}{l} L = \begin{cases} 116 \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 & \frac{Y}{Y_n} > \left(\frac{6}{29}\right)^3 \\ \left(\frac{29}{3}\right)^3 \frac{Y}{Y_n} & \frac{Y}{Y_n} \leq \left(\frac{6}{29}\right)^3 \end{cases} \\ u = 13L(u_1 - u_n) \\ v = 13L(v_1 - v_n) \end{array} \right. , \quad (16)$$

where $u_1 = \frac{4X}{X+15Y+3Z}$ and $v_1 = \frac{9Y}{X+15Y+3Z}$ transformed into the Luv color space as depicted in Figure 2c.

The effect of images processed in the Lab, HLS, and Luv color spaces is combined with Sobel edge detection, and this combination is applied to the processed grayscale images. The resultant combined effect is depicted in Figure 2.

2.3. Lane Detection Algorithm Design

The existing algorithms primarily focus on single-lane detection, with relatively fewer studies on detecting multi-lane lines. However, multi-lane line scenes are more common in practical traffic scenarios, making research on lane detection in multi-lane scenarios more practically significant. The experimental setting utilized in this paper featured a laboratory scene scaled proportionally, as illustrated in Figure 4.

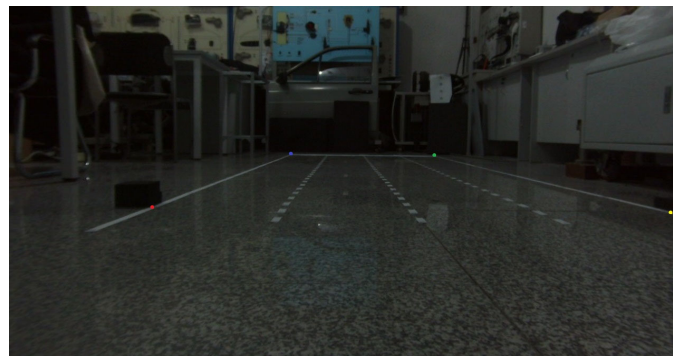


Figure 4. Isometrically scaled multi-lane line scenario.

The lane line detection method, which is based on polynomial fitting of the bird's-eye view [29], preprocesses the image. Subsequently, the road image is projected onto the bird's-eye view, which serves to make the lane line information more concise and removes interference from the surrounding environment [30]. The pixel coordinates of five lane lines on pre-selected roads were obtained through a multi-directional sliding window based on the BEV. Prior to detecting lane lines, it was necessary to utilize a histogram to determine their positions. As illustrated in Figure 5a, with an image resolution of 1280×720 , the quantity of white pixels was tallied for each column, resulting in 1280 values. These values were plotted on a pixel coordinate system, with the horizontal axis ranging from 0 to 1279, and the number of white pixels in each column representing the vertical axis. The IPM (inverse perspective mapping)-processed BEV was then combined with the histogram, which was based on the approximate horizontal coordinate range of each lane line, to identify the number of columns corresponding to the peak of the maximum number of white pixel dots. However, within the preselection zone, the detection of stop lines may cause some deviation in the location of the peak quantity relative to where the lane lines appear at the bottom of the image. To address this, the IPM without the stop lines was intercepted for the statistical histograms, as shown in Figure 5b–d.

After determining the starting positions of the lane lines, the image processed by IPM is searched using the multi-directional sliding window method and a hierarchical overlay manner. The steps are as follows:

First, a rectangular region termed a sliding window is established at the starting position of each lane line. The starting points are respectively used as the midpoints of the lower border of the sliding window, storing all the horizontal coordinates of white pixels within the square;

Second, the stored horizontal coordinates are then averaged. The column where the average is located and where the upper edge of the first sliding window is located is used as the midpoint of the lower edge of the next sliding window to continue the search;

Third, during the search process, if there are no white pixels within the region, the horizontal position of the next sliding window is not updated.

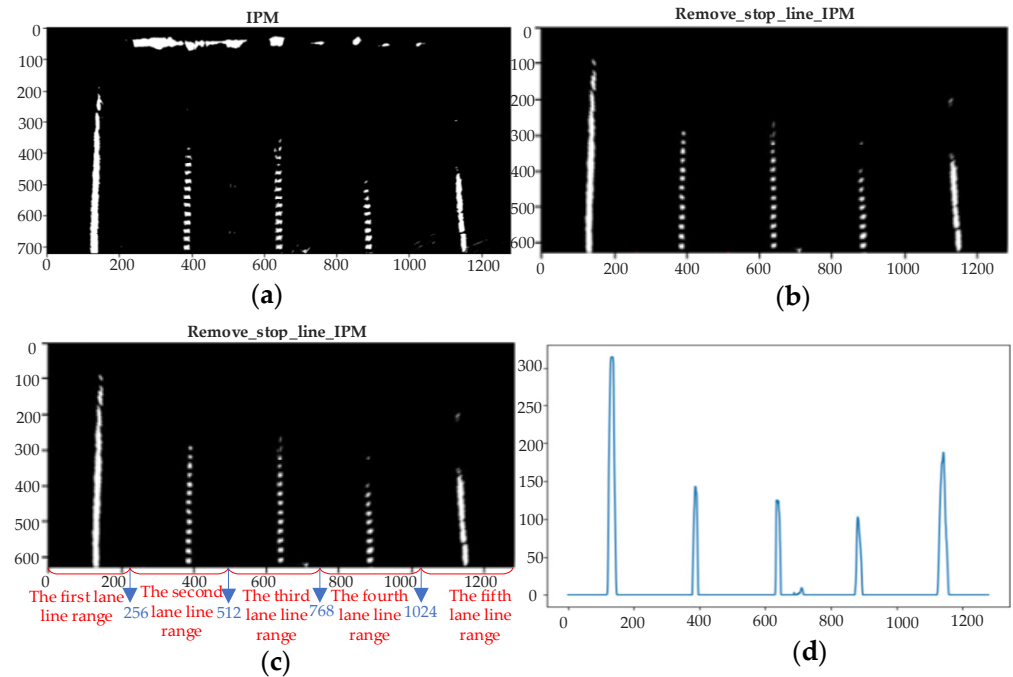


Figure 5. The positioning of the lane lines using the histogram. (a) BEV in the pixel coordinate system; (b) BEV after intercepting the stop line; (c) Positioning of lane line scope map; (d) Histogram of the number of white pixel points.

This process is repeated iteratively until the upper boundary of the sliding window reaches the top boundary of the image, at which point the search is terminated. As shown in Figure 6b, the coordinates of the pixel points of the five lane lines searched by the sliding window in the figure were recorded and are displayed respectively in red, yellow, green, blue, and purple.

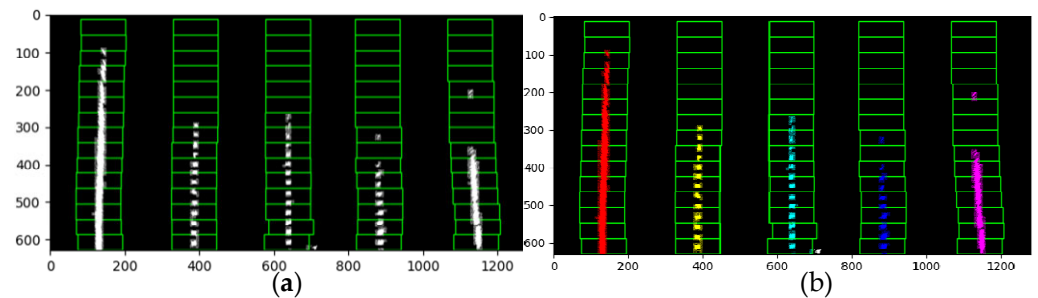


Figure 6. Multi-directional sliding window searching for lane line pixels. (a) Sliding window search procedure; (b) Coordinates of recorded lane line pixel points.

2.4. Horizontal Positioning and Markings of Multi-Lane Lines

Once a vehicle has been identified in multiple lanes, it is necessary to localize the vehicle in order to ensure stable movement in the target lane. Formula (17) illustrates that, based on the fitting of a first-order curve for each lane line, the lateral coordinate positions in the pixel coordinate system are determined as $line1x$, $line2x$, $line3x$, $line4x$, and $line5x$, with the units in pixels.

$$\begin{cases} line1x = a \times 719pixel + b \\ line2x = a \times 719pixel + b \\ line3x = a \times 719pixel + b \\ line4x = a \times 719pixel + b \\ line5x = a \times 719pixel + b \end{cases}, \tag{17}$$

It can be observed that the closer a lane line is to the autonomous vehicle, the higher the precision of recognition. Consequently, a vertical pixel value of 719 pixels was selected, allowing the determination of the corresponding horizontal pixel values for each lane line.

The method of “standard line” positioning was employed in this paper to determine the current lane of the autonomous vehicle. First, after image preprocessing (before IPM processing), in the pixel coordinate system, it is necessary to draw a red line segment three pixels wide between points $P_1 = (640, 0)$ and $P_2 = (640, 719)$. The subsequent step is the implementation of IPM processing. Thereafter, the RGB color space is transformed into the LAB color space, followed by color threshold filtering to convert it into a binary image (c). Utilizing histogram techniques, the position of the “standard line” is ascertained. The horizontal coordinate positions of the “standard line” in the pixel coordinate system can be ascertained using the “standard line” search method with unidirectional sliding window propagation and the “standard line” expression method with polynomial fitting. The unit in pixels is illustrated in (18).

$$standard_linex = a \times 719pixel + b, \tag{18}$$

Similarly, the pixel value of the vertical coordinate of the “standard line” is selected as 719 pixels, and then the pixel value of the horizontal coordinate corresponding to the “standard line” is calculated, that is, the pixel value of the horizontal coordinate corresponding to the current intelligent vehicle, as shown in Figure 7:

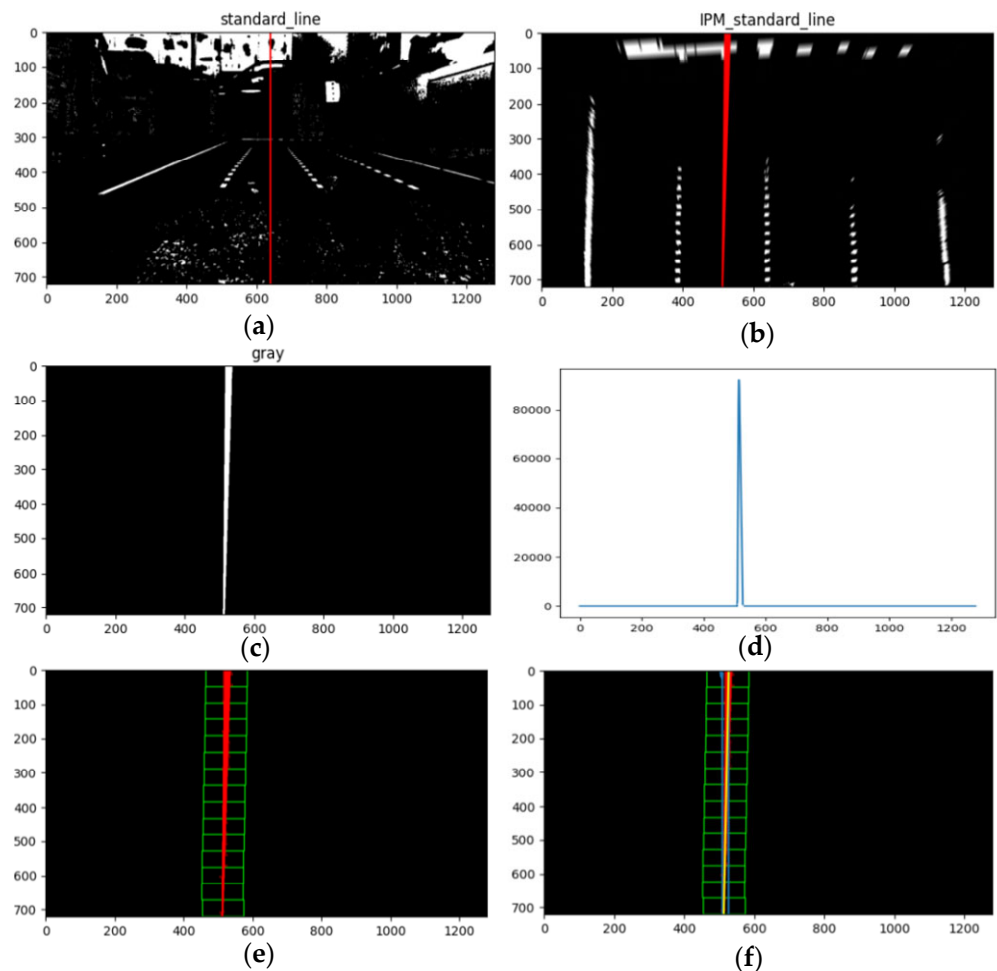


Figure 7. The positioning of the “standard line”. (a) Drawing the “standard line”; (b) IPM processing; (c) binary image; (d) “Standard line” for histogram positioning; (e) Sliding window to search for lane lines; (f) “Standard line” fitting results.

The pixel distance between the current intelligent vehicle and each lane line is determined by combining (17) and (18), as illustrated in (19). The current lane in which the intelligent vehicle is located is determined by the range of pixel values corresponding to the car's horizontal coordinate.

$$\begin{cases} distance1x = |standard_linex - line1x| \\ distance2x = |standard_linex - line2x| \\ distance3x = |standard_linex - line3x| \\ distance4x = |standard_linex - line4x| \\ distance5x = |standard_linex - line5x| \end{cases}, \quad (19)$$

where $distance1x$, $distance2x$, $distance3x$, $distance4x$, and $distance5x$ represent respectively the pixel distance between the current position of the vehicle and the first lane line with the unit in pixels. Based on this distance, the current position of the vehicle can be accurately localized.

This “standard line positioning strategy” is an innovative lane detection approach introduced in this article, which utilizes a specifically meaningful “Standard Line” in the image processing to precisely recognize the position of an autonomous vehicle within a lane. The superiority of this method is characterized by its high accuracy and robustness, enabling stable operation under diverse road conditions. By employing the pixel coordinate system of the image and leveraging histogram techniques along with polynomial fitting, the method swiftly and accurately ascertains the vehicle's lateral position relative to the lane markings.

2.5. Horizontal Positioning of Stop Lines

In addition to determining the vehicle's current lane position, it is also essential for the vehicle to accurately recognize the stop line in order to ensure the safety of the vehicle's travel [31].

An intelligent vehicle employs a two-step process to achieve longitudinal localization. First, it recognizes and detects the distance to the stop line. Second, it combines this information with the bird's-eye view obtained from lane line localization.

1. A 90° clockwise rotation of Figure 5b is performed;
2. The histogram technique is utilized to determine the location of the stop line;
3. Using the stop line search method based on one-directional sliding window growth and the stop line representation method based on polynomial fitting, the $stop_linex$ that represents the lateral coordinate position of the rotated stop line in the pixel coordinate system and the $stop_liney$ that represents the vertical coordinate position of the actual stop line in pixels are determined, as follows:

$$stop_liney = stop_linex = a \times standard_linex + b, \quad (20)$$

Extensive experimentation demonstrated that the portion of the stop line corresponding to the current lane of the intelligent vehicle exhibited the highest recognition accuracy. Consequently, the pixel value of the longitudinal coordinate of the rotated stop line was selected as the pixel value of the transverse coordinate of the $standard_linex$, and then the pixel value of the corresponding longitudinal coordinate of the actual stop line was calculated as $stop_liney$, which is the pixel value of the corresponding longitudinal coordinate of the current intelligent vehicle. The intelligent vehicle is capable of multi-lane range measurement and longitudinal and lateral localization in the current lane. Under the pixel coordinate system, the unit of the results obtained by Equations (18)–(20) is pixels. We have

$$\begin{cases} x_pix = \frac{3.6m}{256pixel} \\ y_pix = \frac{40m}{720pixel} \end{cases}, \quad (21)$$

$$\begin{cases} \text{standard_linex_rel} = x_pix \times \text{standard_linex} \\ \text{distanceix_rel} = x_pix \times \text{distanceix} \quad i = 1, 2, 3, 4, 5 \\ \text{stop_liney_rel} = y_pix \times \text{stop_liney} \end{cases} \quad (22)$$

Combined with the proportion of the numbers of vertical and horizontal pixels of the first frame to the actual distance in the middle, and the unit conversion (21), the unit of the results of ranging and localization can be converted to meters, as shown in (22).

The image processed by the initial frame's IPM shows that the conversion ratio between the longitudinal and lateral pixel distances in the pre-selected area and their actual distances can be calculated as follows: the pre-selected length along the vehicle's direction is equivalent to 40 m, which is covered by 720 pixels; the width of a single lane is equivalent to 3.6 m, which is covered by 256 pixels.

3. Multi-Lane Traffic Sign Recognition Based on YOLO

Simultaneous consideration of the driving direction of each lane is necessary for the identification of multi-lane lines on the road surface and the positioning of multiple lanes. However, this does not address the issue of whether the lanes can be replaced according to the actual plan of the route. In the driving process of the intelligent vehicle, the target detection algorithm based on YOLO is employed to identify multi-lane traffic signage. This is then matched with the relevant lanes through improvement of the detection code, thereby enhancing the intelligence of driverless vehicles [32,33].

3.1. Fabrication of a Multi-Lane Traffic Signage Dataset

The incorporation of multi-lane line recognition requires access to a dataset related to four-lane traffic signs. However, the publicly accessible domestic datasets are not comprehensive. Consequently, this paper presents the construction of an original dataset for experimental purposes.

Image collection is the initial step in the process of creating a dataset. However, mere images are not sufficient for model training. Therefore, annotation information for the images is also required. In the YOLO model, the data output format includes five parameters necessary for describing the information in the picture: the horizontal and vertical coordinates of the center point, the width w and height h of the target box, and the category of the target box. In this chapter on four-lane traffic signage detection and recognition, we will consider not only the overall signage categories but also the statistics of directional signs for each lane. Combining the category labeling naming convention of the TT100K dataset, the dataset for this chapter was ultimately divided into nine categories. The specific details are shown in Table 1

The chosen categories for the nine traffic signs studied in this chapter can be broadly categorized into two main groups: *Four_lane* signs, which represent four-lane traffic signage, and signs indicating possible turns for each lane. Given road conditions with four lanes, the selection of samples aimed for comprehensive representation. If the methods tested in this paper can successfully recognize these signs and accurately match them with the corresponding ground lanes, these methods can be easily extended to recognize other types of multi-lane traffic signs, as well as transitions between different multi-lanes in future applications.










In this chapter, the traffic sign samples used as data sources mainly came from three channels:

Firstly, the selection of annotated images for labelling in this chapter was derived from three publicly available Chinese traffic sign datasets;

Secondly, standard images of traffic signs collected from the Internet served as reference standards, while sample images used by manufacturers to produce specified traffic signs were employed as a basis for comparison;

Third, the images in (1) and (2) were subject to data enhancement to generate more training data.

Table 1. The category of traffic signs in the dataset.

Serial Number	Traffic Sign	Symbol Meaning	Category	Serial Number	Traffic Sign	Symbol Meaning	Category
1		Left turn	I10	6		Turnaround	I16
2		Right turn	I12	7		Left turn or turnaround	I17
3		Straight forward	I13	8		Straight forward or turnaround	I18
4		Straight forward or right turn	I14	9		Four lane plate traffic signage	<i>Four_lane</i>
5		Straight forward or left turn	I15				

During the training of the YOLO network model, some data augmentations were applied, such as random cropping, scaling, color jittering, and mosaic processing. However, these enhancements did not alter the distribution of the number of traffic signs during training. In order to enhance the quality of the network model training results and the diversity of the training data, especially for categories with relatively few samples, this chapter adopted the following four methods of data augmentation:

1. The first method was to augment the sample count of a specific category by duplicating and pasting fewer traffic sign images in different environments. This was achieved by copying the original image into other images and then resizing them. The images were then pasted into the corresponding position or near the original sign. During the augmentation process, it is important to avoid overlapping with other signs, which is typically achieved by keeping the number of signs augmented in each image below five. Additionally, the *IOU* between the signs in the new environment images was calculated after augmentation, as shown in (23). If $IOU > 0.1$, it is necessary to select alternative positions for pasting.

$$IOU = \frac{A \cap B}{A \cup B} = \frac{S_{AB}}{S_A + S_B - S_{AB}}, \quad (23)$$

where A and B are two traffic signs whose areas are the area of the intersection of A and B, respectively.

2. Random scaling was utilized. This process involves scaling the image up or down by a specified ratio, which alters the original image's resolution and generates new images. This step enhanced the model's generalization performance during training by increasing the diversity of the training data;
3. Random rotation was conducted. In real road scenarios, four-lane traffic signs are typically fixed at the roadside and extend a certain distance from the edge of the road. As autonomous vehicles traverse different lanes, images captured by cameras exhibit varying degrees of tilt. To ensure the comprehensiveness of the dataset, the

original images were randomly rotated by different angles to the right or left, thereby generating new images;

4. Gaussian noise was introduced. Gaussian noise is a type of noise characterized by its probability density function, which follows the Gaussian distribution, also known as the normal distribution. The addition of Gaussian noise to images facilitates the learning of more image features by neural network models. The images resulting from the addition of Gaussian noise were obtained by sampling a random number matrix obtained using a Gaussian distribution and then adding the RGB pixels of the original image and the random number matrix containing a Gaussian distribution.

XML annotation files were automatically parsed from the dataset using a Python program, and the data were divided into training and testing sets in a 9:1 ratio. This resulted in a final training set of 1080 images and a testing set of 118 images.

3.2. YOLO Modeling Training

PyTorch, a dynamic imperative programming framework, was selected as the optimal choice. The training environment was configured as shown in Table 2. The model's hyperparameters included algorithm learning epochs, initial learning rate, minimum learning rate, batch size, momentum, and decay. After multiple training sessions with varying numbers of epochs, the optimal number of algorithmic learning epochs was determined to be 300. The initial learning rate determines how quickly the weights are updated. The learning rate of the dynamic transformation is typically set during training based on the number of training epochs. The batch size refers to the number of training samples in a batch, and the network updates the parameters in real-time with the batches of samples being trained. Momentum represents the momentum parameter, influencing the speed at which the gradient descends to the optimal value. The term "decay" refers to the weight decay regularization term, which decreases the parameters after each learning step by a constant proportion, in order to prevent overfitting when the model's parameters are overly complex. The values of the parameters are presented in Table 3.

Table 2. The YOLO training environment.

Computer Operating System	Deep Learning Framework	Development Languages and Environments	Central Processing Unit	Graphic Processing Unit
Ubuntu18.04	PyTorch11.0	Python3.7/PyCharm	Intel Core i5 9600K	NVIDIA GeForce GTX 960m

Table 3. The parameter settings for the model hyperparameters.

Parameter Type	Parameter Value	Parameter Type	Parameter Value
Algorithm learning epoch	300	Batch size	16
Initial learning rate	0.0001	Momentum	0.9
Minimum learning rate	0.000001	Decay	0.0005

3.3. Model Training Results and Analysis

Figures 8 and 9 illustrate the evolution of the confidence loss, coordinate loss, classification loss, and total loss throughout the experimental process. After training the four-lane traffic sign dataset for 300 epochs following the settings mentioned earlier, convergence was achieved, and the training was stopped. At the 300th epoch, the confidence loss was around 0.63%, the coordinate loss was 5.4%, and the classification loss was approximately 1.35%, leading to a total loss of about 7.46%.

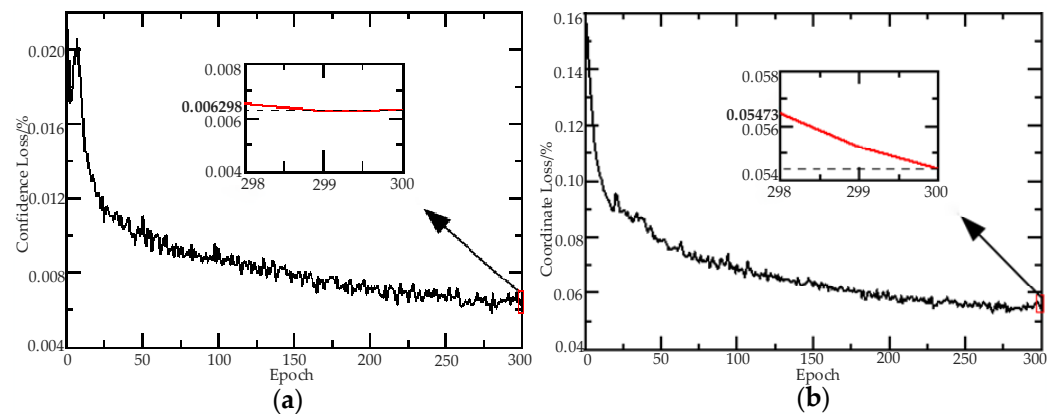


Figure 8. The confidence loss and coordinate loss curve. (a) Confidence loss; (b) Coordinate loss.

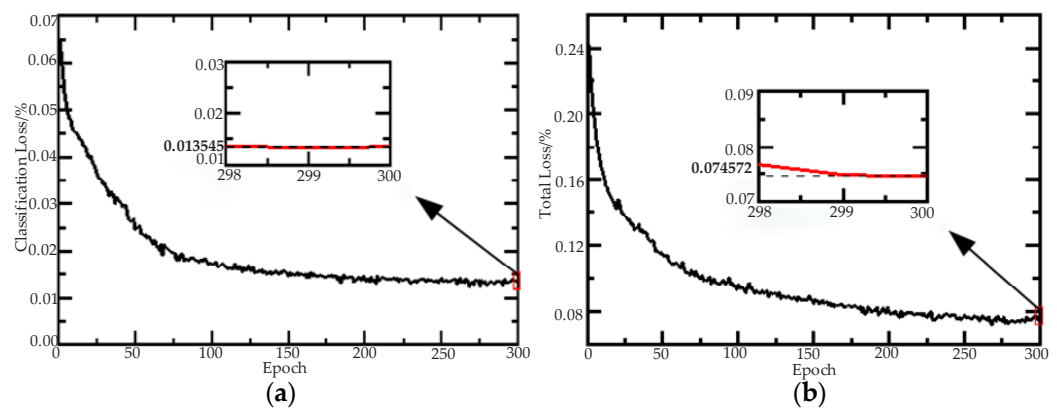


Figure 9. Classification loss and total loss curves. (a) Classification loss; (b) Total loss.

The training results were experimentally evaluated on the test set of the four-lane traffic sign dataset. The precision, recall, and mean average precision (mAP) results are depicted in Figure 10. Upon reaching convergence of the loss function, the average precision of the nine trained traffic signs was found to be 96.9%, with an average recall of 86.5% and a mean average precision of 87.3%.

Consequently, the optimal model parameters, trained on the YOLO network model and the self-made four-lane traffic sign dataset, exhibited excellent detection performance for the signs on four-lane traffic signs.

3.4. Insufficient Light Image Enhancement Based on SSR Algorithm under V Channel

A vehicle's journey can be divided into three categories based on the intensity of the ambient lighting: normal scenes, backlit scenes, and dimly lit scenes. Backlit scenes typically occur when the vehicle is driving against the light, resulting in higher illumination on target objects. Dimly lit scenes are commonly encountered in environments with poor lighting conditions, such as overcast weather or at dusk. The enhancement of contrast, edge details, and color vibrancy within the image can result in a clearer and brighter effect, which in turn facilitates the subsequent detection of multi-lane traffic signs.

HSV (hue, saturation, value) is a color space created by A. R. Smith in 1978 based on the perceptual characteristics of color. It is also known as the hexcone model [34]. The geometric model of the HSV color space is illustrated in Figure 11.

The parameters of the color model are hue (H), saturation (S), and value (V). Among these, V represents the brightness level of the color. For illuminant colors, the luminance value is related to the luminance of the light source. For object colors, the luminance value

is related to the transmittance or reflectance of the object. In this section, the input RGB color image is converted to the HSV color space. The conversion relationship is as follows:

$$V = \max\{R, G, B\}, \tag{24}$$

where R, G, and B are real numbers between [0, 1] and V range from 0% (black) to 100% (white).

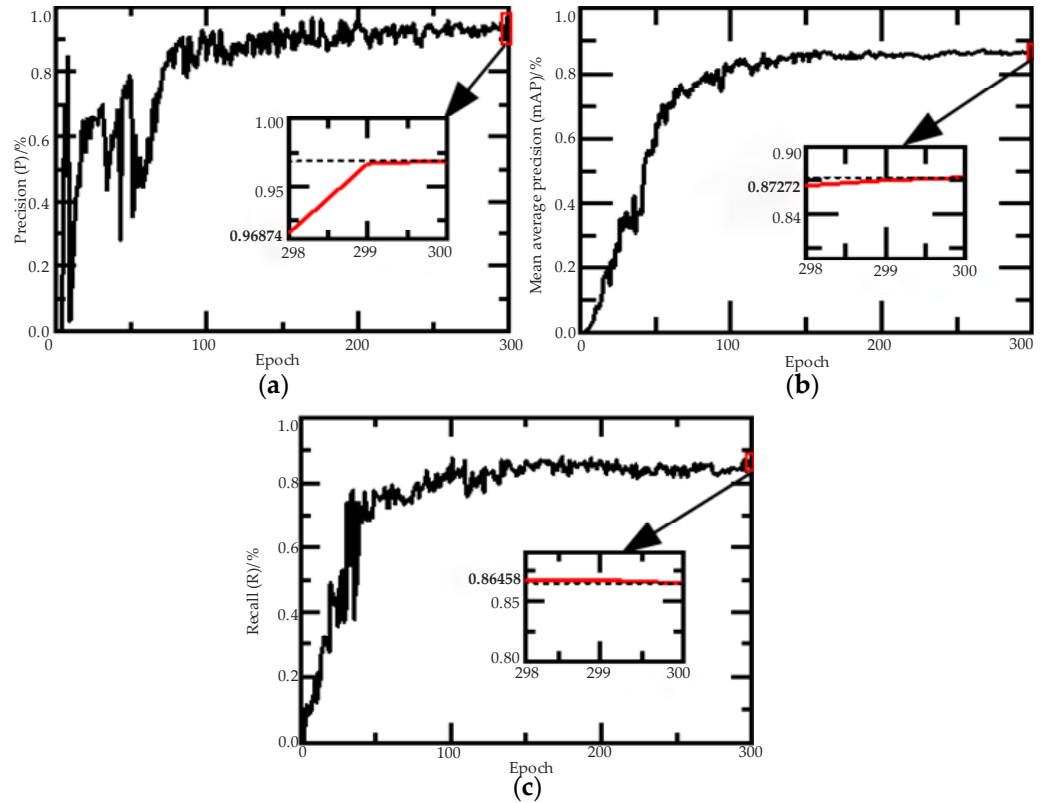


Figure 10. The precision, mean average precision, and recall on the test set. (a) Precision; (b) Mean average precision; (c) Recall.

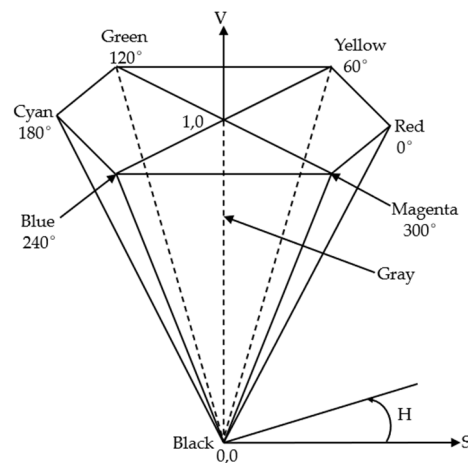


Figure 11. The geometric model of the HSV color space.

If the measurement indicates that the lighting conditions are within the normal range, the detection and recognition process will proceed directly. Otherwise, the original RGB image will be enhanced using the SSR algorithm, and the detection and recognition of multi-lane traffic signs will be performed on the enhanced image.

The SSR algorithm first convolutes the values of the R, G, and B channels in the original image with the Gaussian function to obtain the intensity estimate of the incident light source. It then converts this estimate into a logarithmic domain and removes the convoluted intensity estimate from the original image. Finally, it calculates the reflected component as the image processed by the SSR algorithm. The algorithm is detailed as follows:

$$r_i(x, y) = \log\left(\frac{I_i(x, y)}{L_i(x, y)}\right) = \log(I_i(x, y)) - \log(I_i(x, y) \times G(x, y)), \quad (25)$$

where $L_i(x, y)$ is the quantity of illumination intensity, $I_i(x, y)$ represents the original image, $G(x, y)$ is the Gaussian surround function, and $r_i(x, y)$ is the reflectance component of the i -th channel. And the expression for $G(x, y)$ is

$$G(x, y) = \frac{1}{2\pi\varepsilon^2} e^{-\frac{x^2+y^2}{2\varepsilon^2}} \quad (26)$$

where ε represents the Gaussian envelope scale. Its value reflects the clarity of the detailed parts in an image. The smaller the value, the higher the clarity of the details, but the fidelity of the image's color deteriorates. Conversely, the clarity of the detailed parts becomes worse, but the fidelity of the image's color improves, enhancing the visual effect.

The MSR algorithm, based on the SSR algorithm, uses multiple scales of Gaussian envelope functions to extract components of the illumination intensity [35,36]. It can handle images with multiple illumination components. The algorithm is as shown in (27):

$$r_i(x, y) = \sum_{j=1}^n \omega_j (\log(I_i(x, y)) - \log(I_i(x, y) \times F_j(x, y))), \quad (27)$$

where n represents the number of scales, ω_j is the weighting coefficient for the j -th term, and $F_j(x, y)$ is the j -th Gaussian envelope function. The expression for ω_j is shown below:

$$\sum_{j=1}^n \omega_j = 1, \quad (28)$$

The MSRCR algorithm, which is based on the MSR algorithm, adds a color restoration process. It enhances the reflection component within the color space, thereby preserving the original color information of the image [37]. Formulas (29) and (30) are as follows:

$$r_i(x, y) = e_i(x, y) \times \left(\sum_{k=1}^N \omega_k (\log(I_i(x, y)) - \log(I_i(x, y) \times F_k(x, y)))\right), \quad (29)$$

$$e_i(x, y) = \eta (\log(\varphi I_i(x, y)) - \log(\sum_{i\{r,g,b\}} I_i(x, y))), \quad (30)$$

where $e_i(x, y)$ represents the color restoration coefficient for the i -th channel, φ is the nonlinear adjustment coefficient, and η is a constant. The brightness histogram of the image to be detected is divided into three frequency bands, as shown in Table 4:

Table 4. Segmentation of the image brightness histogram into bands.

Frequency Bands	Range
Low frequency	[0, 85]
Medium frequency	(85, 170]
High frequency	(170, 255]

Statistical analysis of the proportion of pixels in the three frequency bands relative to the entire image being tested yields the following proportions: L, M, and H. If $L + H > 80\%$ and simultaneously satisfies both $L > H$ and $H > M$, then the current scene is a backlit scenario. If the backlit condition is not met and $L > 60\%$, then the current scene is a dimly lit scenario [38]. The SSR algorithm was employed to enhance images in non-normal scenarios. Figure 12 illustrates the enhancement effects of the SSR algorithm on two types

of non-normal illumination images. It can be observed that the images exhibit greater vibrancy in color and clarity in contour.

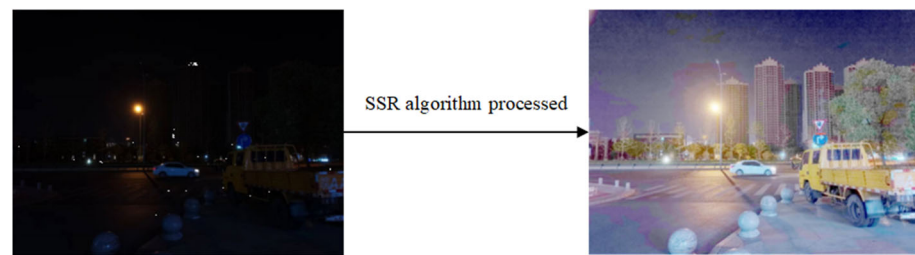


Figure 12. Effect of the SSR algorithm under non-normal conditions.

The SSR algorithm assesses the illumination level by measuring the V channel within the HSV color space, and when it detects inadequate lighting, it utilizes an enhancement mechanism to boost the image’s contrast, edge details, and color liveliness. This enables the clear identification of traffic signs in both backlit and dimly lit environments. Not only does this approach elevate the detection rate of traffic signs, but it also safeguards the security and dependability of autonomous driving systems through immediate processing, sustaining efficient performance regardless of suboptimal visual conditions.

4. Matching and Joint Experimental Verification of Multi-Lane Traffic Signage and Ground Multi-Lane

4.1. Lane and Traffic Sign Matching Algorithm

The precisely lane and traffic sign matching is crucial for autonomous driving [39]. In this section, the recognition results for the four-lane traffic signboard are orderly marked and outputted in a manner that aligns with the four lanes on the ground. Each lane must be sequentially labeled from left to right. In the YOLO model, the output of the types and related information of traffic signs in each frame of the image is unordered, which makes it impossible to match with the already marked ground lanes. Consequently, it was imperative to implement targeted modifications to the detection code within YOLO. During the detection and recognition process, the left upper corner horizontal coordinate pixel values of the four-lane traffic sign and each traffic sign prediction box are sorted, and the predicted boxes and related information of various traffic signs on the traffic sign board are outputted in sequence, with each lane on the traffic sign board being numbered. Concurrently, the current lane position information of the autonomous vehicle from Chapter 2 is utilized to provide feedback on the traffic signboard. The algorithmic process is depicted in Figure 13, where the location of “car” represents the current lane occupied by the autonomous vehicle.

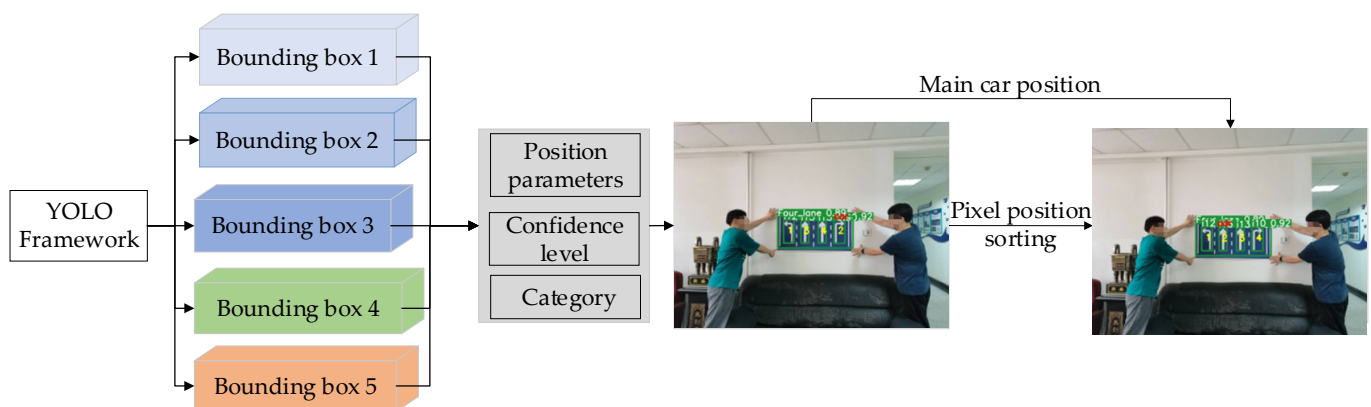


Figure 13. Lane line matching algorithm based on YOLO.

4.2. Joint Experimental Validation

To verify the accuracy of the proposed method and to assess the impact of the algorithm under various scenarios, a track was utilized to simulate lane lines while considering the conditions of a school playground. A handcrafted KT board was used as a four-lane traffic signboard. The experimental vehicle, equipped with a monocular camera, functioned as an intelligent car. Figure 14 illustrates this setup. Furthermore, the environment and equipment in the experimental images were scaled down proportionally. Using the ratio of the width of the running track to the width of domestic lanes as a benchmark, a monocular camera was set up on the experimental car, and a four-lane traffic signboard was created to a relative scale, as illustrated in Figure 14.

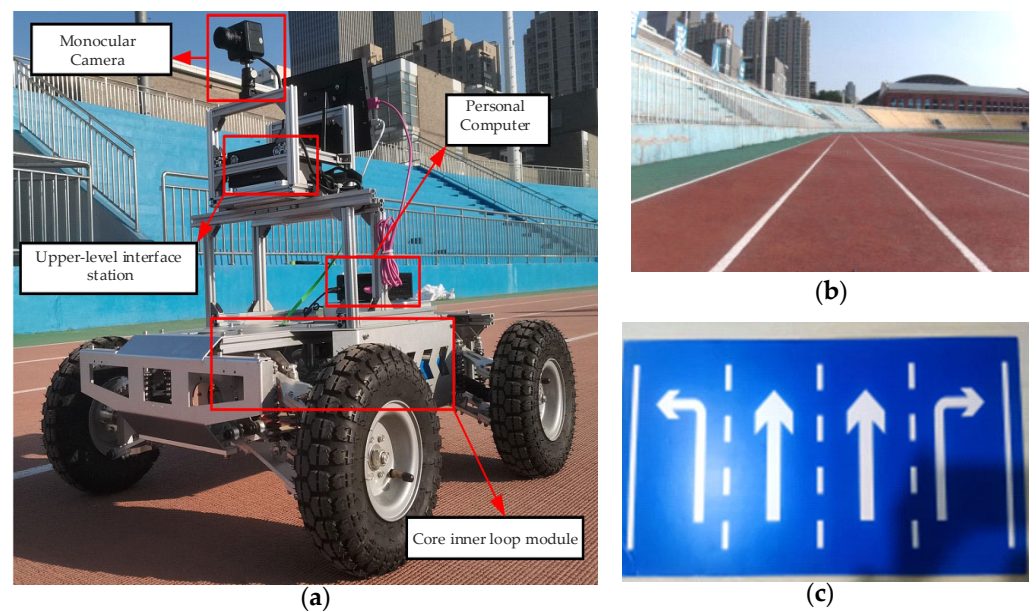


Figure 14. Experimental setup. (a) Experimental vehicle; (b) runway simulation markings; (c) handcrafted KT board traffic sign.

Step one involved detecting the multi-lane lines. The algorithm for multi-lane lines was used to detect and identify the multi-lane lines in the image, as shown in Figure 15.



Figure 15. Result image of the four-lane detection.

Step two involved the preprocessing of each frame of video. Prior to the detection of traffic signs in each frame, a statistical analysis of the brightness histogram was conducted using the V channel within the HSV color space. This analysis evaluated the proportion of pixels across various frequency bands, thereby enabling an assessment to be made of the lighting conditions in each frame. As illustrated in Figure 16, any frames that did not meet the criteria for standard lighting conditions were subjected to SSR exposure adjustment.

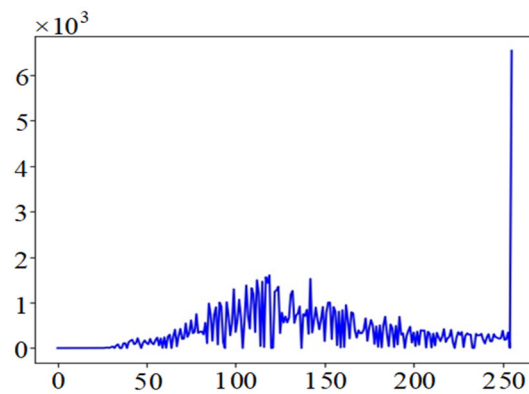


Figure 16. Brightness histogram in the V channel.

The third step involved the detection and recognition of four-lane traffic signboards. The optimal model parameters previously trained with the YOLO algorithm were applied to detect and recognize traffic signboards in each preprocessed frame. The detection and recognition results are depicted in Figure 17.



Figure 17. Four-lane traffic sign detection result image.

The final step was the joint detection and recognition. The recognition information on the traffic signboards was matched to each lane. Although the fusion algorithm is somewhat complex, the processing speed of each frame could be maintained within 25 ms by calculating the FPS, which is essentially close to real-time processing. The recognition result is shown in Figure 18. It can be observed that, when applied to the same frame, the traditional vision and deep learning methods for object detection were effectively integrated. The lanes on the traffic signboards were matched with the lanes on the ground, resulting in the desired outcome and ensuring the accuracy and real-time nature of the detection.

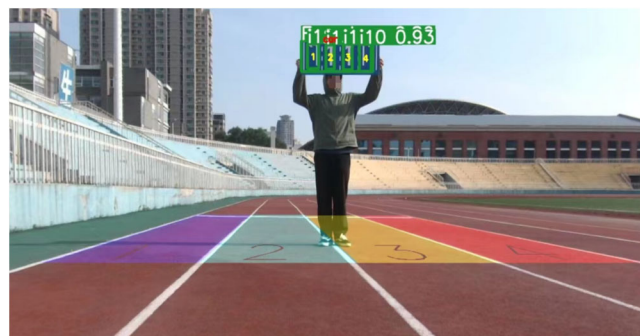


Figure 18. Joint detection and recognition result image.

5. Conclusions

This study employed computer vision techniques to identify and classify multi-lane markings. The process started with image pre-processing, followed by inverse perspective mapping, which transformed the scene into a top-down binary view of the lanes. Subsequently, a multi-directional sliding window strategy was employed to identify key lane pixels, which were subsequently curve-fitted using polynomial regression. To identify the “standard line”, which is crucial for the vehicle’s lane positioning, a unidirectional sliding window was employed. Concurrently, a YOLO network was trained on an in-house dataset to detect and classify traffic signs. The illumination of each frame was quantified using the V channel of the HSV color model. Integration of these methods with weighted file integration enhanced the robustness of traffic sign detection. Subsequently, the results from lane and sign detection processes were combined to enrich the road condition dataset. Traffic sign detection was presented systematically, with the outcomes displayed from left to right, mirroring the lane detection results. This systematic approach matched multi-lane signs to actual road markings, providing the vehicle with detailed road information for navigation. The efficacy of the visual module was validated through a collaborative testing process involving a monocular camera setup specifically designed for visual tasks. The SSR algorithm has high computational complexity and demands a large amount of training data. Additionally, its recognition performance in low-light environments needs improvement.

Author Contributions: Conceptualization, Methodology, K.X. and Z.L.; software, Z.H.; validation, formal analysis, investigation, resources, J.H.; data curation, Z.L.; writing—original draft preparation, Z.W. (Zhongnan Wang); writing—review and editing, Z.W. (Zijian Wang); visualization, Z.H.; supervision, Z.L.; project administration, K.X.; funding acquisition, Z.L. and K.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Industrial Foundation Remanufacturing and High-Quality Manufacturing Development Project of the Ministry of Industry and Information Technology, grant number (2340STCZB1929) and the Fundamental Research Funds for the Central Universities, grant number (N2403009).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Swathi, M.; Suresh, K.V. Automatic Traffic Sign Detection and Recognition: A Review. In Proceedings of the 2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies, Chennai, India, 16–18 February 2017; pp. 1–6.
2. Liang, Z.; Zhao, J.; Liu, B.; Wang, Y.; Ding, Z. Velocity-Based Path Following Control for Autonomous Vehicles to Avoid Exceeding Road Friction Limits Using Sliding Mode Method. *IEEE Trans. Intell. Transp. Syst.* **2019**, *23*, 1947–1958.
3. Ahmed, N.; Anwar, A.; Eckelmann, S. Lane Marking Detection Techniques for Autonomous Driving. In Proceedings of the 16th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, Fukuoka, Japan, 28–30 October 2021; pp. 217–226.
4. Tu, C.; Van Wyk, B.J.; Hamam, Y. Vehicle Position Monitoring Using Hough Transform. *IERI Procedia* **2013**, *4*, 316–322. [[CrossRef](#)]
5. Wei, W.; Dong, X.; Shen, Y. Research on a two value Generalized Hough transform method of identification. In Proceedings of the 2011 International Conference on Computer Science and Network Technology, Harbin, China, 24–26 December 2011; pp. 278–281.
6. Li, H.Y.; Sima, C.; Dai, J.F. Delving into the Devils of Bird’s-Eye-View Perception: A Review, Evaluation and Recipe. *IEEE Comput. Soc.* **2024**, *46*, 2151–2170. [[CrossRef](#)] [[PubMed](#)]
7. Bhupathi, K.C.; Ferdowsi, H. An Augmented Sliding Window Technique to Improve Detection of Curved Lanes in Autonomous Vehicles. In Proceedings of the 2020 International Conference on Electro Information Technology, Chicago, IL, USA, 31 July–1 August 2020; pp. 522–527.
8. Zhang, Q.; Liu, J.; Jiang, X. Lane Detection Algorithm in Curves Based on Multi-Sensor Fusion. *Sensors* **2023**, *23*, 5751. [[CrossRef](#)] [[PubMed](#)]

9. Liang, Z.; Wang, Z.; Zhao, J.; Ma, X. Fast Finite-Time Path-Following Control for Autonomous Vehicle via Complete Model-Free Approach. *IEEE Trans. Ind. Inf.* **2023**, *19*, 2838–2846. [[CrossRef](#)]
10. Wang, Z.; Wu, Y.; Niu, Q. Multi-Sensor Fusion in Automated Driving: A Survey. *IEEE Access* **2020**, *8*, 2847–2868. [[CrossRef](#)]
11. Rahman, Z.; Morris, B.T. LVLane: Deep Learning for Lane Detection and Classification in Challenging Conditions. In Proceedings of the 2023 IEEE 26th International Conference on Intelligent Transportation Systems, Bilbao, Spain, 24–28 September 2023; pp. 3901–3907.
12. Yan, F.; Nie, M.; Cai, X.Y. ONCE-3DLanes: Building Monocular 3D Lane Detection. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 July 2022; pp. 17122–17131.
13. Zheng, Z.; Zhang, X.; Mou, Y.; Gao, X.; Li, C.; Huang, G.; Pun, C.-M.; Yuan, X. PVALane: Prior-Guided 3D Lane Detection with View-Agnostic Feature Alignment. In Proceedings of the 38th AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 7597–7604.
14. Rongqiang, Q.; Zhang, B.; Yue, Y. Traffic Sign Detection by Template Matching Based on Multi-Level Chain Code Histogram. In Proceedings of the 2015 12th International Conference on Fuzzy Systems and Knowledge Discovery, Zhangjiajie, China, 15–17 August 2015; pp. 2400–2404.
15. Pandey, P.; Kulkarni, R. Traffic Sign Detection Using Template Matching Technique. In Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation, Pune, India, 15–17 August 2018; pp. 1–6.
16. Gan, Y.; Li, G.; Togo, R.; Maeda, K. Zero-Shot Traffic Sign Recognition Based on Midlevel Feature Matching. *Sensors* **2023**, *23*, 9607. [[CrossRef](#)] [[PubMed](#)]
17. Cao, J.; Song, C.; Peng, S. Improved Traffic Sign Detection and Recognition Algorithm for Intelligent Vehicles. *Sensors* **2019**, *19*, 4021. [[CrossRef](#)] [[PubMed](#)]
18. Xie, G.; Xu, Z.; Lin, Z.; Liao, X. GRFS-YOLOv8: An Efficient Traffic Sign Detection Algorithm Based on Multiscale Features and Enhanced Path Aggregation. *Signal Image Video Process.* **2024**, 1–16. [[CrossRef](#)]
19. Yalamanchili, S.; Kodepogu, K.; Manjeti, V.B. Optimizing Traffic Sign Detection and Recognition by Using Deep Learning. *Int. J. Transp. Dev. Integr.* **2024**, *8*, 131–139. [[CrossRef](#)]
20. Korshunova, K.P. A Convolutional Fuzzy Neural Network for Image Classification. In Proceedings of the 2018 3rd Russian-Pacific Conference on Computer Technology and Applications, Vladivostok, Russia, 18–25 August 2018; pp. 1–4.
21. Wang, Y.; Shen, D.; Teoh, E.K. Lane Detection Using Spline Model. *Pattern Recognit. Lett.* **2000**, *21*, 677–689. [[CrossRef](#)]
22. Yoo, J.H.; Lee, S.-W.; Park, S.-K. A Robust Lane Detection Method Based on Vanishing Point Estimation Using the Relevance of Line Segments. *IEEE Trans. Intell. Transport. Syst.* **2017**, *18*, 3254–3266. [[CrossRef](#)]
23. Chen, Z.; Liu, Q.; Lian, C. PointLaneNet: Efficient End-to-End CNNs for Accurate Real-Time Lane Detection. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium, Paris, France, 9–12 June 2019; pp. 2563–2568.
24. Haris, M.; Glowacz, A. Lane Line Detection Based on Object Feature Distillation. *Electronics* **2021**, *10*, 1102. [[CrossRef](#)]
25. Du, X.; Tan, K.K.; Ko Htet, K.K. Vision-Based Lane Line Detection for Autonomous Vehicle Navigation and Guidance. In Proceedings of the 2015 10th Asian Control Conference, Kota Kinabalu, Malaysia, 31 May–3 June 2015; pp. 1–5.
26. Bow, S.-T. *Pattern Recognition and Image Preprocessing*, 2nd ed.; CRC Press: Boca Raton, FL, USA, 2002; pp. 271–399.
27. Loehlin, J.C. The Cholesky Approach: A Cautionary Note. *Behav. Genet.* **1996**, *26*, 65–69. [[CrossRef](#)]
28. Vikram Mutneja, D. Methods of Image Edge Detection: A Review. *J. Elec. Electron. Syst.* **2015**, *4*, 1000150. [[CrossRef](#)]
29. Bhupathi, K.C.; Ferdowsi, H. Sharp Curve Detection of Autonomous Vehicles using DBSCAN and Augmented Sliding Window Techniques. *Int. J. ITS Res.* **2022**, *20*, 651–671. [[CrossRef](#)]
30. Abbas, S.A.; Zisserman, A. A Geometric Approach to Obtain a Bird’s Eye View from an Image. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop, Seoul, Republic of Korea, 27–28 October 2019; pp. 4095–4104.
31. Marita, T.; Negru, M.; Danescu, R. Stop-Line Detection and Localization Method for Intersection Scenarios. In Proceedings of the 2011 IEEE 7th International Conference on Intelligent Computer Communication and Processing, Cluj-Napoca, Romania, 25–27 August 2011; pp. 293–298.
32. Jiang, P.; Ergu, D.; Liu, F. A Review of Yolo Algorithm Developments. *Procedia Comput. Sci.* **2022**, *199*, 1066–1073. [[CrossRef](#)]
33. Du, J. Understanding of Object Detection Based on CNN Family and YOLO. *J. Phys. Conf. Ser.* **2018**, *1004*, 012029. [[CrossRef](#)]
34. Sural, S.; Gang, Q.; Pramanik, S. Segmentation and Histogram Generation Using the HSV Color Space for Image Retrieval. In Proceedings of the Proceedings. International Conference on Image Processing, Rochester, NY, USA, 22–25 September 2002; p. II.
35. Mario, D.G.; Alberto, J.R.S.; Francisco, J.G.F. Chromaticity Improvement in Images with Poor Lighting Using the Multiscale-Retinex MSR Algorithm. In Proceedings of the 2016 9th International Kharkiv Symposium on Physics and Engineering of Microwaves, Millimeter and Submillimeter Waves, Kharkiv, Ukraine, 20–24 June 2016; pp. 1–4.
36. Sun, B.; Tao, W.; Chen, W. Luminance Based MSR for Color Image Enhancement. In Proceedings of the 2008 Congress on Image and Signal Processing, Sanya, China, 27–30 May 2008; pp. 358–362.
37. Wang, J.; He, N.; Lu, K. A New Single Image Dehazing Method with MSRCR Algorithm. In Proceedings of the 7th International Conference on Internet Multimedia Computing and Service, Zhangjiajie, China, 19 August 2015; pp. 1–4.

38. Lee, C.; Moon, J.-H. Robust Lane Detection and Tracking for Real-Time Applications. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 4043–4048. [[CrossRef](#)]
39. Liang, Z.; Shen, M.; Li, Z.; Yang, J. Model-Free Output Feedback Path Following Control for Autonomous Vehicle With Prescribed Performance Independent of Initial Conditions. *IEEE-ASME Trans. Mechatron.* **2024**, *29*, 1076–1087. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.