A Robust Lane Detection and Tracking Method based on vanishing point for Multi-environment Situation

Guimin Zhou[a], Dongmin Zhang*[b], Futang Zhu[b] and Dongsheng Yang[b]
[a] Shien-Ming Wu School of Intelligent Engineering, South China University of Technology: Guangzhou, Guangzhou 51400, CN; [b] Automotive New Technology Research Institute, BYD (China): Shenzhen, Shenzhen 518118, CN, *Corresponding author's e-mail: Zhang.dongmin@163.com

ABSTRACT

Lane detection and tracking technologies play a crucial role in advancing autonomous driving and advanced driver assistance systems (ADAS). However, the existing methods struggle to achieve accurate detection across diverse environments. In this study, we proposed an adaptive lane detection and tracking algorithm based on the vanishing point. Initially, the edges of the lane lines were extracted using the ROI and 2D-FIR filter. Next, the vanishing point of the image is detected using Gabor filters, and adaptive inverse perspective transformation is achieved through coordinate conversion. Eventually, the detection of lane lines through the dynamic sliding window is attained by introducing the spatio-temporal sequence module, while tracking is performed with the adaptive Kalman filter by leveraging the lane width constraint. The method was tested on our dataset, which covers seven different environments and comprises over 4644 frames. The experimental results demonstrate that the accuracy of the detection is 92.9% and the center offset is close to 5.4 pixels.

Keywords: Lane detection, lane tracking, sliding window, vanishing point, Kalman filter

1. INTRODUCTION

The rapid development of intelligent driving technology assists drivers in driving more comfortably and safely, among which precise lane detection is an important prerequisite for advanced driver assistance systems (ADAS). Lanes serve as critical road markers that directly influence vehicle trajectories and safe distances[1-2]. They are essential for autonomous driving technology and ADAS, which require precise lane detection to determine the position of the vehicle. Consequently, numerous scholars have conducted extensive research in the field of lane detection, primarily employing two methodologies: deep learning and image processing-based methods.

1.1 Deep learning-based methods

These methods primarily implement lane detection by constructing models and training them on large datasets. In the early stages, deep learning methods predominantly utilized anchor-based models, where lane detection was achieved by defining the geometry and location of lane lines through anchors. For instance, TABELINI et al.[3] proposed an anchor-based lane detection model that uses anchors in feature pooling to aggregate global information. This approach enabled accurate real-time lane detection, even when lane lines were obscured. In CLRNet[4], the authors similarly utilized anchors to define lane lines, aiming to fully exploit both high and low-level features in lane detection. This ensures that the model refines lane features during downsampling without losing high-level features, which enables detection despite the loss of lane information. Unlike anchor-based models, keypoint-based models define lane lines by using the keypoints of lanes. In RClane[5], the authors proposed a keypoint-based model that utilizes a relay chain prediction module to capture keypoints on lanes by predicting the foreground and background of segmented images. In GANet[6], parallel detection of lane keypoints and offsets was performed through a deep learning model, enhancing both detection speed and accuracy. This model achieved high detection accuracy and FPS rate in the CULane[7] and Tusimple[8] datasets. In addition to these approaches, other approaches also deserve attention. Pan et.al proposed saptial CNN(SCNN)[9], which transforms traditional layer convolution to slice convolution within feature maps through spatial information, thereby improving the generalization of the model. Furthermore, there is also a model that regards lane detection as a time series, namely the RVLD[10]. This method is used for detecting lane lines in videos.

International Conference on Computer Vision and Image Processing (CVIP 2024)
edited by Xin Xu, Zhenghao Shi, Proc. of SPIE Vol. 13521, 135210U
 · 0277-786X · doi: 10.1117/12.3058090

Despite the notable performance of deep-learning methods in rapid lane detection, they also have several limitations. 1) Collecting and labeling large datasets requires substantial human and material resources. 2) The black-box nature of deep learning models results in a lack of interpretability, making it challenging to accurately assess and improve algorithms when adjusting the model parameters. In contrast, image processing-based methods do not require collecting a large amount of data for pre-training, and they can adjust the algorithm parameters specifically when facing different environments. Thus, image processing-based approaches still hold significant research value for lane detection.

1.2 Image processing-based methods

These approaches primarily utilize color and edge features of images to detect lane lines, employing techniques such as color space transformation, texture detection and image shape transformation. In [11-15] , the Hough Transform was applied to convert image features into parameter space, followed by a voting mechanism to accurately identify lanes. For example, EM P et al.[14] used a Region of Interest (ROI) with a 2D-FIR filter to detect edge features in images and utilized the Hough Transform to identify straight lanes. ANDREI M-A et al.[13] proposed using a probabilistic Hough Transform in place of the standard version and suggested employing a parallelogram-shaped ROI rather than a trapezoidal one. GAIKWAD V et al.[15] enhanced ROI contrast using PSLSF, partitioned it into two sub-regions, and used the Hough Transform for lane detection in each sub-region. These methods are effective for detecting straight lanes under various lighting conditions. However, due to the Hough Transform's limitations in curve detection, they cannot detect curved lane lines. In[16-20] , inverse perspective mapping (IPM) and sliding window methods were employed, enabling the detection of both straight and curved lane lines. ZHANG Q et al. [18] achieved detection of highly curved lane lines by predicting the center of the sliding window based on steering wheel angle. MA N et al.[19] employed ROI and IPM techniques to transform lane markings into two parallel straight lines, subsequently achieving lane detection through histogram analysis and sliding window methods. However, when there are changes in the vehicle's posture or camera position, IPM may introduce transformation distortions, which can interfere with lane detection and hinder accuracy. Therefore, FENG Y[20] proposed an Adaptive Edge-based Lane Detection and Tracking method, which eliminates distortion caused by IPM through median filtering and dynamically adjusts the number of sliding windows to properly calibrate the Kalman filter for tracking. Nevertheless, this sliding window detection method does not work under extensive noise interference.

In general, image processing-based lane detection methods perform well under normal conditions. However, during actual road driving in diverse environments, lane lines often encounter interference from various noise sources, such as directional arrows, text and debris. This noise can be mistakenly detected as lane information. In addition, variations in the road slope and vehicle speed during driving can alter the vehicle's posture or camera position, resulting in coordinate transformation distortions, which affect the accuracy of perspective transformation-based lane detection methods. Given these complex scenarios, further research is essential for improving the robustness and accuracy of lane detection. Thus, this study introduces a vanishing point-based method for lane detection and tracking, which is capable of identifying and tracking lanes in diverse environments. The proposed approach holds significant potential for applications in Advanced Driver Assistance Systems (ADAS), including Lane Departure Warning Systems (LDWS) and Lane Keeping Assist Systems (LKAS). The main contributions of this study are as follows:

1) An adaptive vanishing point-based inverse perspective transformation was utilized to eliminate distortions caused by changes in vehicle posture and camera deviation, thereby enhancing the robustness of lane detection.

2) A dynamic sliding window method is introduced for lane detection, where the starting point of the sliding window is redefined based on spatio-temporal information. This reduces the interference of detection noise on lane detection and improves the detection accuracy.

3) Integrating an adaptive Kalman filter based on lane width confidence for lane tracking. This helps to reduce detection disturbances and enhance detection precision by relying on the continuity of the video frames.

4) Create a diverse dataset under varying environmental conditions and define an evaluation metric to assess the deviation of the lane detection center. This enables a more accurate assessment of the algorithm performance in lane departure systems.

The remainder of this paper is organized as follows. Section 2 describes the proposed lane detection and tracking methods. Section 3 describes the experimental setup and the construction of the dataset, followed by the experimental results. Finally, Section 4 concludes the paper.

2. PROPOSED METHOD

In this section, we outline the implementation details of our proposed method, and a flow chart is illustrated in Figure 1. Step 1 involves preprocessing the input RGB image, including color space transformation to grayscale and edge extraction using a 2D-FIR Filter. Step 2 focuses on detecting the vanishing point using Gabor filters and a voting mechanism. Step 3 calculates the azimuth of the camera based on the vanishing point for adaptive inverse perspective transformation. In Step 4, lane detection is performed using a dynamic sliding window approach, followed by quadratic curve fitting of the detected lane pixels. Finally, Step 5 adapts the Kalman filter based on lane width confidence, leveraging the continuity of the video frames for lane tracking.

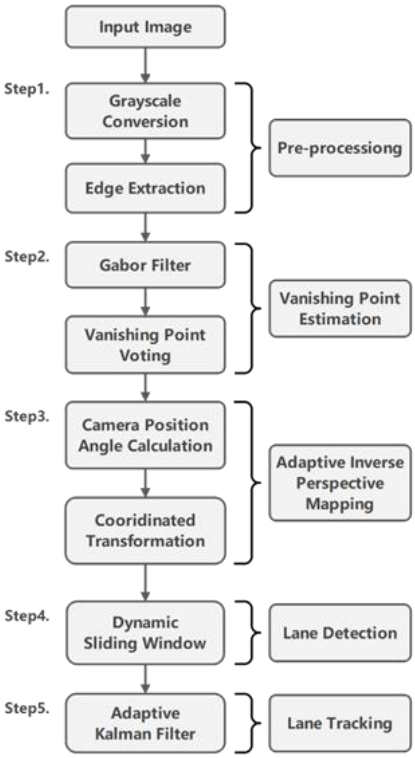


Figure1. Flow chart of the proposed method.

2.1 Preprocessing

Grayscale conversion of images. In the initial phase of the algorithm, RGB images were used as the input. To mitigate the computational complexity and processing time of the algorithm, we convert the RGB images into grayscale images. The grayscale transformation equation shown in Eq. (1). This equation takes the matrices of the red, green, and blue channels of the RGB image as the input and yields the corresponding grayscale image as the output. As depicted in Figure 2, where Figure 2(a) represents an input RGB image, and Figure 2(b) illustrates the resulting grayscale image obtained through the function computation.

$$Gray\_IMG = \begin{bmatrix} 0.299 & 0.587 & 0.114 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

R - Red color channel of image.
G - Green color channel of image.
B - Blue color channel of image.

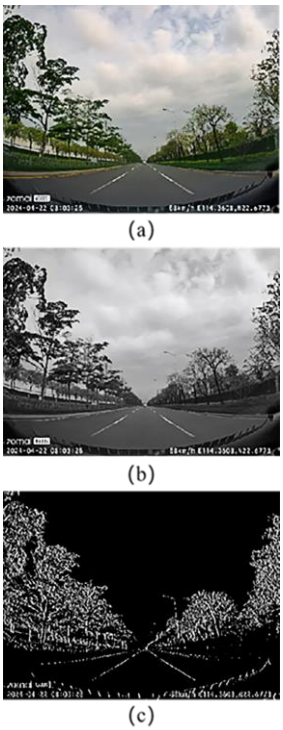


Figure2. Illustration of image edge detection. (a) Original input image, (b) grayscale of Original input image, (c) edge detection of Original input image.

Image Edge Detection. In the image, our focus is on the edges of lanes; however, the image also contains an amount of other interference. To obtain the edges of lanes, the edge extraction operations is employed. This process, as illustrated in the Figure 3(a), consists of three parts: the 2D-FIR filter, saturation processing, and adaptive threshold segmentation[14].

First, we applied the 2D-FIR filter to the grayscale image to remove noise and highlight edges. This filter, initially proposed by TERRELL, has been validated in both spatial and frequency domains and has demonstrated excellent outcomes in land satellite image processing[21]. It emphasizes differences between pixels through convolution with a kernel and performs image smoothing to achieve noise reduction. The computational formula for the convolution kernel is as follows:

$$C(i,\ j) = \sum_{m=0}^{(Ma-1)} \sum_{n=0}^{(Na-1)} A(m, n) * H(i-m,\ j-n) \tag{2}$$

$$A = \begin{bmatrix} 1, & 0, & -1 \end{bmatrix} \tag{3}$$

where $0 \le i \le Ma\text{-}1$ and $0 \le j \le Na\text{-}1$ represent the output with dimensions as the input; $H = (i, j)$ represents the grayscale i mage pixel at point (i, j); $A=(m, n)$ is the kernel, and the convolution kernel is [1, 0, -1], as shown in Eq. (3) and Figure 3(b); * denotes the convolution operation; and $C=(i, j)$ is the resulting pixel image after filtering, with image edges filled by repli cate padding.

After the 2D-FIR filter, a saturation processing module was added to limit the grayscale values of the image to the preset range of 0 to 1 to ensure stability for subsequent processing. Adaptive threshold segmentation was introduced to conduct the image binarization. It automatically selects the optimal threshold based on the grayscale histogram of the image and divides the image into the foreground and background. The final extracted edge image is shown in Figure 2 (c).

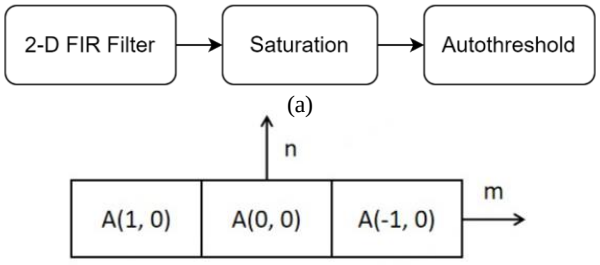


Figure 3. (a) Flow chart of edge detection. (b) Discrete spatial coordinates of the 2D-Fir filter kernel.

2.2 Vanishing point detection

In lane detection, the vanishing point refers to the intersection point of parallel lines in the perspective image, providing crucial information about the camera viewpoint and geometric relationship with the road. Detecting the vanishing point in an image is essential for implementing inverse perspective transformation. Our method employs Gabor filters and a voting mechanism[22] to detect the vanishing point in an image.

Texture Image Detection with Gabor Filters. Gabor filters were utilized to predict the texture orientation at each pixel of the grayscale image. The kernel of the Gabor filters can be represented by Eq. (4).

$$g_{\omega,\phi}(x,y)=\frac{\omega}{\sqrt{2\pi}\varepsilon}e^{\frac{-\omega^2}{8\varepsilon^2}\left(4a^2+b^2\right)}\left(e^{ia\omega}-e^{-\varepsilon^2/2}\right) \tag{4}$$

where ω and ϕ represent the dimensions and orientation of the filters ($\omega=1$, $\phi=\frac{36}{\varepsilon}$; ϕ is chosen to be from 0° to 175° with an angle interval of 5°); $a=x\cos\phi+y\sin\phi$, $b=-x\sin\phi+y\cos\phi$; is a constant ($\varepsilon=2.2$); *(x, y)* represents the coordinates of the image.

To obtain the dominant orientation $\theta(\mathrm{z})$, the grayscale image must convolve with the kernel of the Gabor filters. The convolution equation is as follows:

$$G_{\omega,\phi}=I(\mathrm{z})*g_{\omega,\phi}(\mathrm{z}) \tag{5}$$

where *I(z)* represent the grayscale value of the image pixel, and * denotes the convolution operation.

Next, we computed the average norm value of direction ϕ using Eq. (6) to obtain the multidimensional norm average value $R_{\phi}(z)$ of $G_{\omega,\phi}$. Finally, the maximum of $R_{\phi}(z)$ was determined using Eq. (7), and we obtained the texture image, as shown in Fig.5(b).

$$R_{\phi}(\mathrm{z})=average\left\{\mathrm{Re}^2\left(G_{\omega,\phi}\right)+\mathrm{Im}^2\left(G_{\omega,\phi}\right)\right\} \tag{6}$$

where $Re\left(G_{\omega,\phi}\right)$ represents the real part of $\mathrm{G}_{\omega,\phi}$ and $Im(G_{\omega,\phi})$ is the imaginary part.

$$\theta\ (\mathrm{z})=\arg\max(R_{\theta}(z)) \tag{7}$$

Voting mechanism for vanishing point estimation. In this section, we introduce a method for obtaining the vanishing point of an image using a texture image voting score mechanism. Before voting, screen out the candidate voting points of the image, the confidence of each texture was calculated and the 36-direction texture values after the filtering processing were sorted. Confidence is calculated as follows:

$$Conf(z)=1-\frac{average\left\{R_5(z),\cdots,R_{15}(z)\right\}}{R(z)} \tag{8}$$

To reduce computational costs, when the confidence of all pixels is smaller than

$$\delta\left(max\ \ Conf\ (\mathrm{z})-min\ \ Conf\ (\mathrm{z})\right) \tag{9}$$

with $\delta=0.3$ are discarded; conversely, they are considered as candidate voting pixels. For each candidate voting point, as shown in Figure 4(a), we generated a vanishing point region, where G represents the quantity, $\overrightarrow{W_{\mathrm{G}}}$ denotes the vector

direction, φ is the region encoding angle ($\varphi = 5°$), and H is the image height. Subsequently, the candidate voting points are scored against the pixels in the vanishing point candidate region using Eq. (10). The resulting voting space map of the image was obtained, with the highest score point representing the vanishing point of the image, indicated by the red point $V(u_0, v_0)$ in Figure 4(c). The VP detection shown in the original image is illustrated in Figure 4(d).

$$Vote(G, P)\frac{e^{-\mu/\varepsilon}}{1+d(G,P)} \tag{10}$$

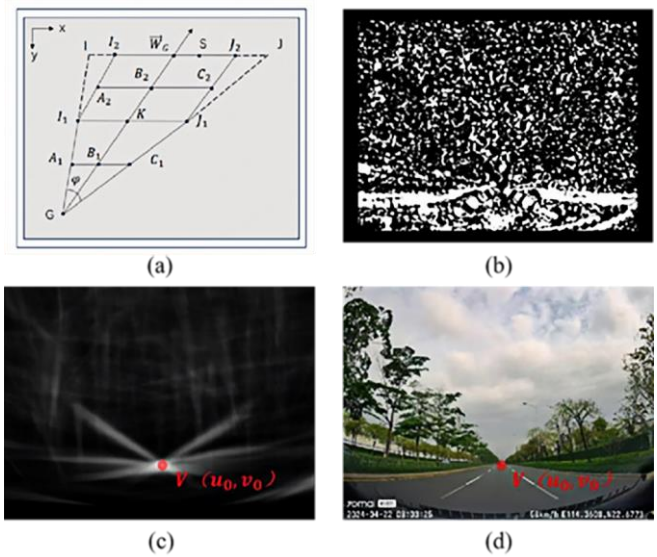


Figure 4. Illustration of vanishing point detection. (a) VP candidate region, (b) texture mapping, (c) voting mapping, (d) detection of VP shown in the original image.

2.3 Adapting inverse perspective mapping(AIPM)

In perspective transformation, a common method is four-point fixed calibration, where four calibration points are chosen in the original image and their coordinates after transformation are determined. However, as illustrated in Figure 5, road slope variations can cause deviations in the fixed-perspective transformation, leading to distortion and impacting detection accuracy. To address this problem, we proposed an adaptive perspective transformation method[23-24] in this section. It removes unnecessary noise from the image in the region of interest (ROI) and utilizes the vanishing point to calculate the camera's azimuth and coordinate transformation of the camera.

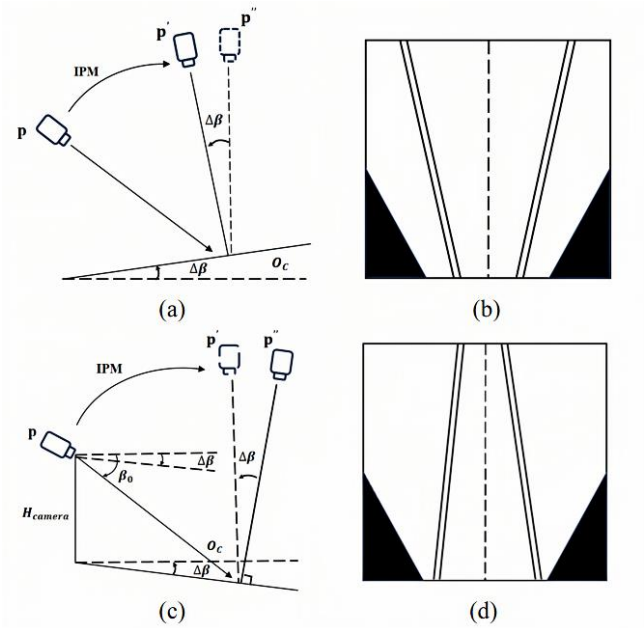


Figure 5. IPM distortion caused by road slope variation. (a)-(b) IPM distortion caused by road slope increase. (c)-(d) IPM distortion caused by road slope decrease.

Region of interest (ROI). To reduce interference from noise outside the lanes, we employed a Region of Interest (ROI) to eliminate unnecessary noise from the images. In lane detection tasks, lanes typically appear in the lower region of the image. Therefore, as illustrated in Figure 6(a), we selected a fixed rectangular region in the image as the region of interest to extract relevant lane line information efficiently. The ROI results are shown in Figure 6(b).

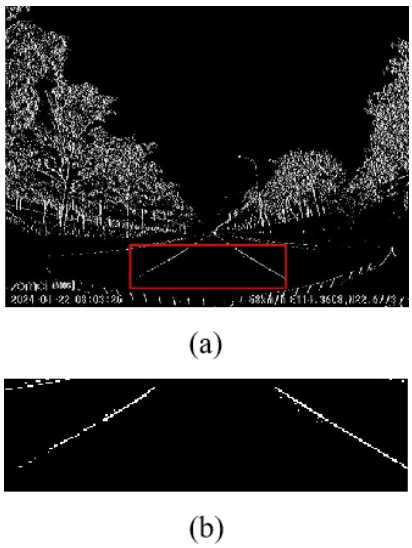


Figure 6. (a) Selection of the region of interest in the edge image. (b)Image of the region of interest (ROI).

Calculation of camera pose angles. As depicted in Figure 5(a) and (c), the camera's orientation β is determined by the combination of road slope $\Delta\beta$ and the camera mounting angle β_0, that is $\beta = \Delta\beta + \beta_0$. It can be calculated using a geometric derivation based on the vanishing point $V(u_0, v_0)$ detected in the image[24], as described in Eq. (11).

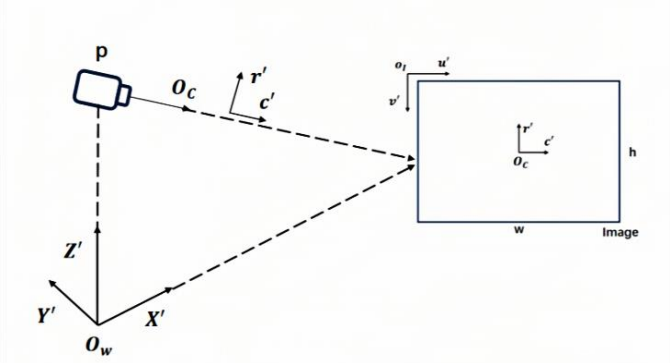
$$\beta = \tan^{-1}\left(\tan(\alpha_r)\left(1-\frac{2v_0}{H}\right)\right) \tag{11}$$

Coordinate transformation. The relationship between the camera, image and world coordinates is shown in Figure 7(a). In the inverse perspective transformation (IPM) model, the image of the camera is primarily transformed from the world coordinate to the camera coordinate through the pinhole imaging principle, and then transformed to the image coordinate through translation and rotation operations. Therefore, inspired by the need for an IPM, we employed coordinate transformation methods to achieve the AIPM.

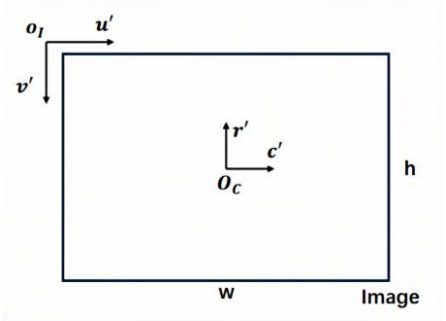
As illustrated in Figure 7(b), we can transform the image and camera coordinates through translation and rotation. The transformation formula is as follows：

$$\begin{cases} \mathrm{u}(c) = \dfrac{w}{2} - \sigma_1 c \\ v(r) = \dfrac{H}{2} - \sigma_2 r \end{cases} \tag{12}$$

where σ_1 and σ_2 represent the horizontal and vertical distances in world coordinates per pixel respectively, (u, v) denotes the coordinates of the point in the image, and (c, r) represents the coordinates of the point in the camera coordinates.



(a)



(b)

Figure 7. (a) Relationship between the camera, image and world coordinate. $Z^{'}O_{W}Y^{'}X^{'}$ is the world coordinate, $r^{'}O_{c}c^{'}$ is the camera coordinate and $u^{'}O_{I}v^{'}$ is the image coordinate. (b) Coordinate the relationship between the camera coordinate $r^{'}O_{c}c^{'}$ and image coordinate $u^{'}O_{I}v^{'}$.

Furthermore, due to the positional deviation of camera relative to the world reference frame, there exists a geometric relationship between the camera coordinate system $\boldsymbol{r'O_Cc'}$ and the world coordinate system $Z^{'}O_{W}Y^{'}X^{'}$. As shown in the side view of the IPM model (Figure 8(a)), we can calculate the vertical yaw angle of the pixel point β_{v} and obtain the horizontal coordinate $X(v)$ in world coordinate system $Z^{'}O_{W}Y^{'}X^{'}$ using Eq. (13). Then, when it is transferred to the top view

of the IPM model (Figure 8(b)), the vertical coordinate $Y(u,v)$ in the world coordinate system $Z^{'}O_{W}Y^{'}X^{'}$ can be obtained using Eq. (14)[25].

$$\begin{cases} \beta_v = \tan^{-1}\left[\left(1-2\dfrac{v-1}{H-1}\right)\tan a_r\right] \\ X(v) = \dfrac{H_{camera}}{\tan\left(\left|\beta+\beta_v\right|\right)} \end{cases} \tag{13}$$

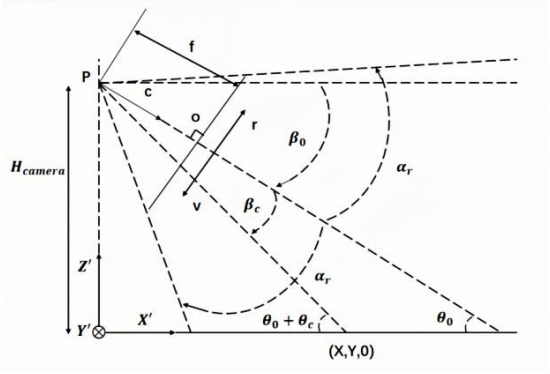
$$Y\left(\mathrm{u,v}\right) = \left(1-2\frac{u-1}{W-1}\right)\tan\left(a_c\right)X(v) \tag{14}$$

where α_r and α_c represent the vertical and horizontal azimuth angles of the camera.

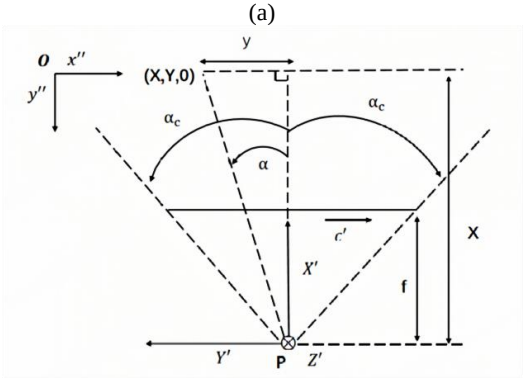
The IPM coordinates obtained must be presented in the image coordinate system. Therefore, further transformation between the world coordinate system $Z^{'}O_{W}Y^{'}X^{'}$ and the image coordinate system $x^{''}Oy^{''}$ is required. The transformation formula is as follows:

$$\begin{cases} \mathrm{x} = \dfrac{\mathrm{w}}{2} - Y\big/\sigma_1 \\ y = H - X\big/\sigma_2 \end{cases} \tag{15}$$

The final AIPM is shown in Figure 10 (a).



(a)



(b)

Figure 8. (a) Side view of IPM model. (b) Top view of IPM model.

2.4 Lane detection based on dynamic sliding window

In this study, the dynamic sliding window method was employed for lane detection. We introduce a spatio-temporal sequence module to eliminate noise interference and adjust the initial points of the sliding window. A flow chart of this method is shown in Figure 9.

Sliding window module. First, we processed the edge image by summing the pixel values along the columns to create a statistical histogram, as shown in Figure 10(b). Two distinct peak regions can be observed in the histogram, corresponding to the two lanes. To extract them, we divided the image into left and right sections and calculated the maximum value of histogram for each section. The resulting peak points, *XL* and *XR*, are used as the starting points for the first sliding window.

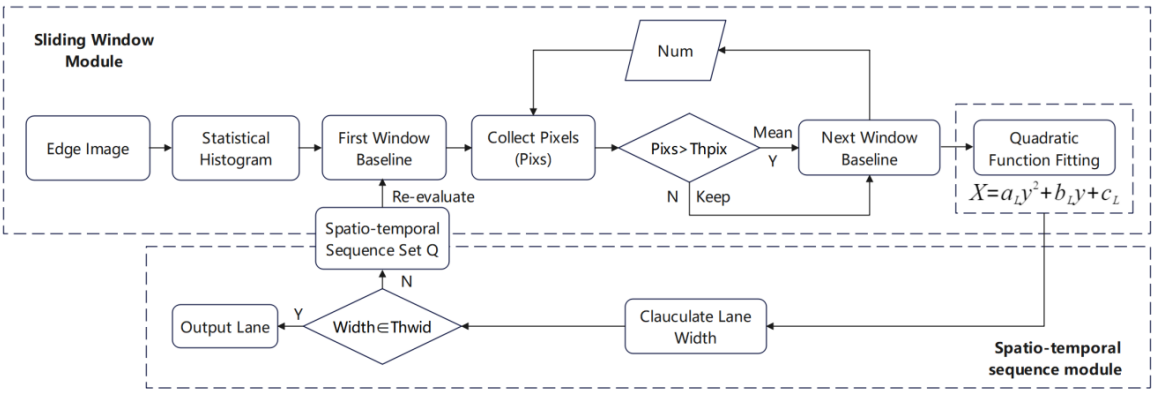


Figure 9. Flow chart of dynamic sliding window.

Next, we drew a fixed-size rectangular sliding window centered at these starting points. The width *w* of the sliding window was set to 45, and the height h was calculated using the following equation:

$$\mathrm{h} = H \diagup Num \tag{16}$$

where *H* represents the height of the image (*H* = 486) and *Num* denotes the number of sliding windows (*Num* = 8).

We collected all non-zero pixels within the rectangular window and treated them as lane pixels. To determine the reference point for the next sliding window, we established a pixel count threshold, denoted as *Thpix*, set to 13. If the number of detected pixels within the previous sliding window exceeds *Thpix*, we calculate the average of the x-coordinates of all non-zero pixels in that window and utilize it as the reference point for the current sliding window. Conversely, if the pixel count falls below *Thpix*, the reference point from the previous sliding window is retained. This sliding window operation was iteratively performed until it reached the top of the image.

Finally, considering the lanes primarily exist in the form of straight lines or curves, quadratic functions can effectively simulate the shape characteristics of the road. Therefore, we employ quadratic curves to fit the detected lane pixels. The equation of the quadratic curve is as follows:

$$x = \mathrm{a}_{\mathrm{L}} y^2 + b_L y + c_L \tag{17}$$

where *x* and *y* represent the horizontal and vertical coordinate of the lane pixel, respectively, and a_L , b_L and c_L are polynomial coefficients. The fitting was accomplished using the least squares method[26], and the fitting result is shown in Figure 10 (c).

Spatio-temporal sequence module. During lane detection using the sliding window, non-lane noises, including directional noises, texts and obstacles, are susceptible to being erroneously detected as lanes, which severely interferes with lane detection. Therefore, we introduced a lane width constraint, utilizing the detected lane width as an evaluation metric to discern central noise. In real-world scenarios, the actual width of lane lines remains consistent, typically ranging from 3 to 3.75 meters. Consequently, we defined a pixel threshold range of [120, 150] to assess the lane width completeness. For instance, in right lane detection, if the detected lane width falls within this specified threshold range, it is considered a positive detection. Subsequently, the detected starting points of the sliding window are added to a fixed-size set *Q* as follows:

$$Q = \{XR_{k-1}, \cdots XR_{k-t}\} \tag{18}$$

where Q has a fixed size t. The starting points were stored sequentially from front to back. When the set is full, the last element is discarded, and all elements are shifted back by one position to accommodate the new starting point. Here, $\boldsymbol{XR_{k-1}, \ldots, XR_{k-t}}$ represents the starting points of the previous t frames of the sorted sliding window.

In contrast, if the detected lane width exceeds the specified threshold range, it is considered noise. In such cases, we introduce a spatio-temporal sequence module to re-evaluate the starting baseline of the sliding window. Since our input images are continuously obtained from video data, they exhibit temporal continuity with minimal changes between adjacent frames. Hence, we used the collection of spatio-temporal sequence Q to re-evaluate the starting points of the sliding window for the current frame, as shown in Figure 11. The equation used is as follows:

$$XR_k = \frac{1}{t}\sum_t XR_{k-1} + \cdots + XR_{k-t} \tag{19}$$

where XR_k denotes the starting point of the right lane sliding window re-determined in the k^{th} frame. After re-determining the starting points of the sliding window, the same sliding window and curve fitting processes were carried out. The lane detection results are shown in Figure 10 (d).

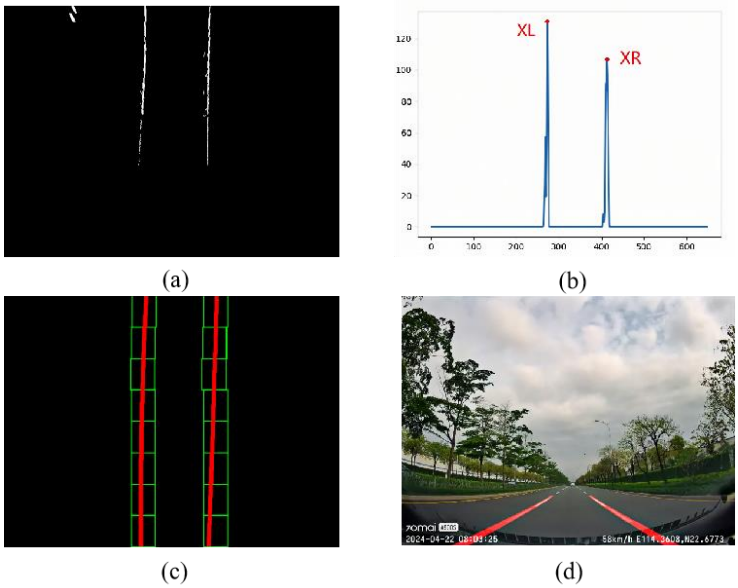


Figure 10. Illustration of lane detection using dynamic sliding window. (a)AIPM of ROI region, (b) the statistic histogram of AIPM, (c) sliding window, (d) lane detection result shown in the original image.

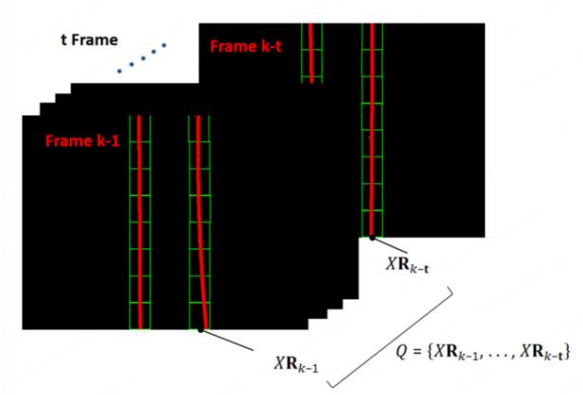


Figure 11. The collection of spatio-temporal sequence Q.

2.5 Lane Tracking Based on Adaptive Kalman Filter

The main challenge in lane detection lies in accurately detecting lanes within each individual frame. However, noise interference in the current frame can significantly impact lane detection accuracy. Given that lane detection involves processing continuous video frames, one approach to enhance accuracy is to incorporate information from previous frames using the Kalman filter. Additionally, introducing lane width confidence allows for adjusting the observation error of the Kalman filter. This adaptation enables the use of an adaptive Kalman filter, which significantly enhances the robustness of lane tracking.

Kalman Filter. The three coefficients of the quadratic function were chosen as the system state vectors for lane tracking, as shown in Eq. (20), because we fit the lanes using a quadratic function, which can be determined by the three coefficients (a_L, b_L, c_L).

$$X_L = \left(a_{L,k}, b_{L,k}, c_{L,k}, a_{R,k}, b_{R,k}, c_{R,k}\right) \tag{20}$$

The Kalman filter consists of two steps: prediction and updating. In the prediction step, the output results from the previous frame were used to predict the posterior state $\hat{x}_{k|k-1}$ and covariance matrix $P_{k|k-1}$. The equation for the calculation is as follows:

$$\begin{cases} \hat{x}_{k|k} = A\hat{x}_{k-1|k-1} \\ P_{k|k-1} = AP_{k-1|k-1}A^T + Q \end{cases} \tag{21}$$

where A is the state transition matrix, which is set as the identity matrix $I \in R^{6\times 6}$; Q is the process noise ($Q=0.5I$); $\hat{x}_{k-1|k-1}$ and $P_{k-1|k-1}$ are the prior state and covariance matrices, $\hat{x}_{k-1|k-1}$ is initialized as $\hat{x}_1$ and $P_{k-1|k-1}$ as Q.

Next, the posterior state $\hat{x}_{k|k-1}$ and $P_{k|k-1}$ covariance matrix are substituted into Eq. (22) to update the Kalman parameters.

$$\begin{cases} \hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(Z_k - M\ \hat{x}_{k|k-1}) \\ P_{k|k} = (I - K_k M)P_{k|k-1} \end{cases} \tag{22}$$

where M is the measurement matrix, Z_k is the system observation of the k^{th} frame; $\hat{x}_{k|k}$ is the posteriori system state of the k^{th} frame; $P_{k|k}$ is the posteriori covariance matrix of the k^{th} frame; and the traditional Kalman gains can be calculated by:

$$K_k = P_{k|k-1}\ M^T\left(M\ P_{k|k-1}M^T + R\right) \tag{23}$$

R is the Measurement noise.

Width conference module. In the Kalman filter, the observation noise errors are typically adjusted using empirical parameters, which may not accurately reflect the true situation of lane observation. This limitation reduces the effectiveness of the Kalman filter tracking. Therefore, considering the minimal variation in lane width between consecutive frames and the fact that lane width remains within a fixed range, we establish a width confidence metric C. This metric is based on the differences between the detected lane width and expected lane width ($d1$), and the detected lane widths between consecutive frames ($d2$). The calculation formulae are as follows:

$$\begin{cases} d_{1,k} = \left|wid_k - wid_0\right| \\ d_{1,k} = \left|wid_k - wid_{k-1}\right| \end{cases} \tag{24}$$

where wid_k and wid_{k-1} represent the width of the lane detection in the k^{th} frame and the $k\text{-}1^{th}$ frame; wid_0 is the expected the lane width.

Subsequently, substituting $d_{1,k}$ and $d_{2,k}$ into Eq. (25) to calculate the width confidence (C).

$$C = w_1 d_{1,k} + w_2 d_{2,k} \tag{25}$$

where w_1 and w_2 are confidence weights (Since $d_{1,k}$ and $d_{2,k}$ are equally important in terms of their influence on observation errors, we set the value of w_1 and w_2 to be 0.5 respectively). Finally, substituting the width confidence C into Eq. (26) to obtain the adaptive observation noise $R^{'}$.

$$R^{'} = diag(R)\, C \tag{26}$$

$diag(\cdot)$ is diagonal matrix. Introducing $R^{'}$, the adaptive Kalman gains can be denoted as:

$$K_k = P_{k|k-1} M^T (M\ P_{k|k-1}\ M^T + R^{'}) \tag{27}$$

3. EXPERIMENTS AND RESULTS

3.1 Experimental equipment and dataset

The proposed method was implemented using MATLAB and Simulink on Window 10 operating system, running on an Intel i7-13800H @2.92GHz processor with 32GB of RAM. Our test dataset was collected using a camera mounted on the windshield of a vehicle, with an image capture frequency of 30 FPS and a resolution of 2592×1944. To reduce computational resources, we resized the images to 648×486 pixels. The vehicle speed during data collection ranged from 0 to 100 km/h, and data were collected on highways and urban roads in Shenzhen, China. To assess the robustness of algorithm, we collected a dataset under various environmental and different lane width conditions, including normal, night, tunnel, rainy, crowded, noisy, and curve scenarios. The dataset comprised 4644 samples, each labeled with ground truth annotations.

3.2 Evaluation Metrics

The evaluation metrics include three official indicators adopted in the Tusimple dataset[8]: false discovery rate (*FDR*), false negative rate (*FNR*), and accuracy (*ACC*). Accuracy is defined as follows:

$$\mathrm{Acc} = \frac{\sum_{clip} C_{clip}}{\sum_{clip} S_{clip}} \tag{28}$$

where C is the number of lane points predicted correctly in the clip, and S is the total number of points in the clip. Considering the error between the predicted lane points and the ground truth, if the deviation between the predicted pixel and the ground truth was within 15 pixels, we considered the pixel to be correctly predicted. A predicted lane with an accuracy greater than 85% is considered as true positive; otherwise, it is considered as false discovery rate (*FDR*) or false negative rate (*FNR*). In addition, the *F1* score was utilized as another evaluation metric, which was calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{29}$$

$$Recall = \frac{TP}{TP + FN} \tag{30}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{31}$$

In autonomous driving, lane detection serves as a vital upstream component of the system, consistently providing crucial calculations for the relative distance offset between downstream vehicles and lanes. Consequently, the precision of lane-center detection is also a paramount evaluation metric. We define it as follow:

$$C_e = \left| \frac{X_{LB} + X_{RB}}{2} - \frac{X_{LB}^{'} + X_{RB}^{'}}{2} \right| \tag{32}$$

$$A_e = \frac{1}{N}\sum C_e \tag{33}$$

where X_{LB} and X_{RB} represent pixels at the bottom of the predicted left and right lanes closest to the vehicle, respectively. X_{LB}' and X_{RB}' represent the pixels at the bottom of the left and right ground truth, respectively, closest to the vehicle. C_e is the center distance between the predicted results and the ground truths. If it is greater than 15 pixels, the predicted lanes fail according to the actual requirements. *Ae* is the average center offset error of the true positive images, and N represents the number of true positive images.

3.3 Experimental Result of Lane Detection and Tracking

A quantitative evaluation of our method using our dataset is presented in table 1. This demonstrates that our method exhibits remarkable performance for lane detection in multiple environments. Out of 4644 frames of data, our method correctly detected 3987 frames, achieving an accuracy score of 92.9% and an *F1* score of 0.91. This indicates that the proposed method ensures high precision with few false detections. The central offset set for the evaluation of subordinate functions can also reach 5.45 pixels. Considering the influence of the lane line width and the error of the ground truth, it can meet the requirements of lane detection precision for ADAS. With the help of the dynamic sliding window, the quantitative evaluation of the noisy condition was the best among seven conditions, and its accuracy reached 98.2%.

The visual results of our multi-environment dataset are shown in Figure 12. Although lane lines are affected by the environment under multi-environment conditions, our method can still accurately detect lane lines. In poor lighting conditions, such as night, rainy and tunnel, the edge of lane lines can still be detected using the 2D-Fir filter. The difficult detection of lane line information loss can be well addressed by the adaptive Kalman filter, as shown in the curve and normal condition in Figure 12.

Condition	Frame	Correct Frame	*Acc*	*F1*	*FDR*	*FNR*	A_e
Normal	1017	965	97.0	0.95	0.05	0.03	6.14
Night	785	661	93.2	0.88	0.12	0.09	10.59
Curve	729	663	92.3	0.89	0.11	0.09	5.18
Crowd	594	343	77.5	0.76	0.24	0.21	6.04
Noisy	445	414	**98.2**	**0.97**	**0.03**	**0.02**	**2.56**
Rainy	411	323	94.4	0.93	0.07	0.05	3.73
Tunnel	663	618	97.6	0.96	0.04	0.03	3.89
Total	4644	3987	92.9	0.91	0.09	0.07	5.45

TABLE 1. Quantitative evaluation result of our method on our dataset and performance in different conditions.

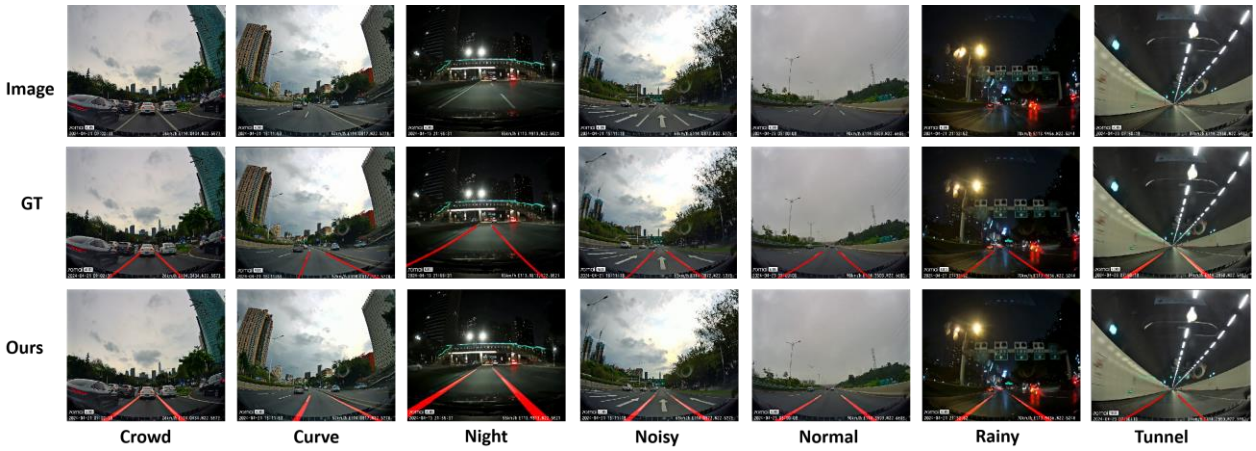


Figure 12. Visualization result of Ground Truth and our proposed method in our multi-environment dataset.

3.4 Ablation Study

To verify the different components of our method for lane detection, we conducted an ablation study on dynamic sliding window (DSW), Kalman filter (KF), adaptive Kalman filter (AKF) and adaptive IPM (AIPM). The results are presented in Table 2.

1) Dynamic sliding window: Owing to the 2D-FIR filter and sliding window, our baseline method can detect lane lines in the image with an accuracy (*Acc*) of 84.6%. However, this method tends erroneously detect lane lines under noisy conditions. As depicted in Figure 13 (a)-(b), the red rectangle is a directional arrow noise between two lane lines, which is incorrectly detected as the starting point of the sliding window and result in the final detection failure. The spatio-temporal sequence module can avoid the interference of this large-area noise and enhance the robustness of the algorithm by re-evaluating the starting point of the sliding window, as shown in Figure 13 (e)-(f). In contrast the baseline, the *F1* score of the DSW increase by 0.08 and the accuracy reached 90.4%.

2) Kalman filter: The Kalman filter is designed to improve current detection using the previous frames. In Table2, compared with the DSW in the second row, the *F1* and *Acc* scores of the Kalman filter can be increased by 0.02 and 1.2% respectively, and the average center offset error (*Ae*) can be reduced to 5.58 pixels. This implies that the Kalman filter can effectively enhance the precision of the lane detection in the current frame.

3) Adaptive Kalman filter: As shown in Figure 14(a), lane detection is disrupted by environmental factors, leading to loss of lane line information. This results in tracking failures and positional deviations of lane lines. Conversely, the adaptive Kalman filter adjusts the observation noise of the lane line using the width confidence module, ensuring that the detection results remain consistent with the previous frame, as shown in Figure 14(b). By introducing the width confidence module, the accuracy (*Acc*) increases by 0.8%, and the average center offset error (*Ae*) was reduced to 5.52 pixels.

4) Adaptive inverse perspective mapping (AIPM): Figure 15 illustrates comparisons between fixed inverse perspective mapping (IPM) and adaptive IPM (AIPM). When the vehicle drove on an uphill section, the fixed IPM caused perspective distortion, resulting in non-parallel lane lines and detection shifts. In contrast, the adaptive IPM effectively eliminated the distortion of the lane lines, reducing the average center offset error *(Ae)* by 0.07.

TABLE 2. Ablation study result.

AIPM	AKF	KF	DSW	*F1*↑	*Acc*↑	*FDR*↓	*FNR*↓	A_e ↓
N.A.	N.A.	N.A.	N.A.	0.77	84.6	0.23	0.19	5.88
N.A.	N.A.	N.A.	√	0.85	90.4	0.15	0.11	5.82
N.A.	N.A.	√	√	0.87	91.6	0.13	0.10	5.58
N.A.	√	√	√	0.88	92.4	0.12	0.09	5.52
√	√	√	√	**0.91**	**92.9**	**0.09**	**0.07**	**5.45**

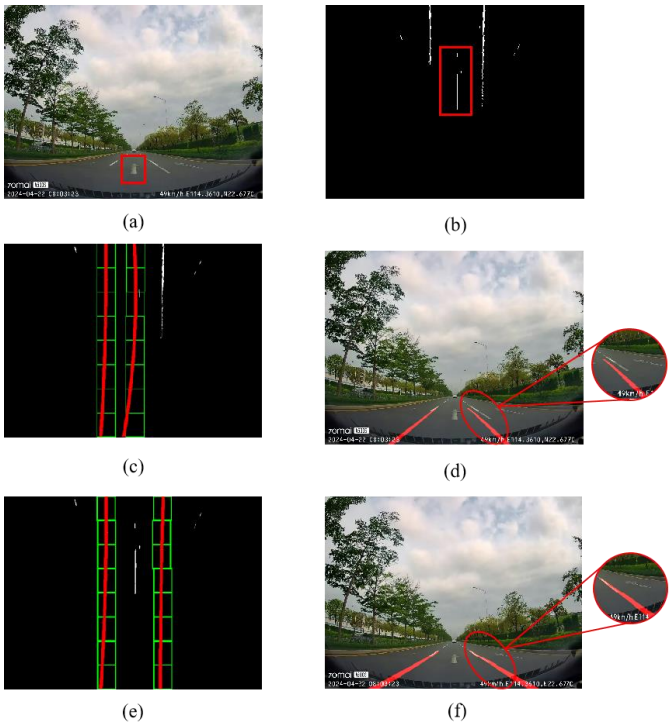


Figure 13. Comparisons of different sliding window. (a)-(b) The noise interference condition of the middle directional arrow. (c)-(d) The result of the normal sliding window. (e)-(f) The result of the dynamic sliding window with spatio-temporal sequence module.

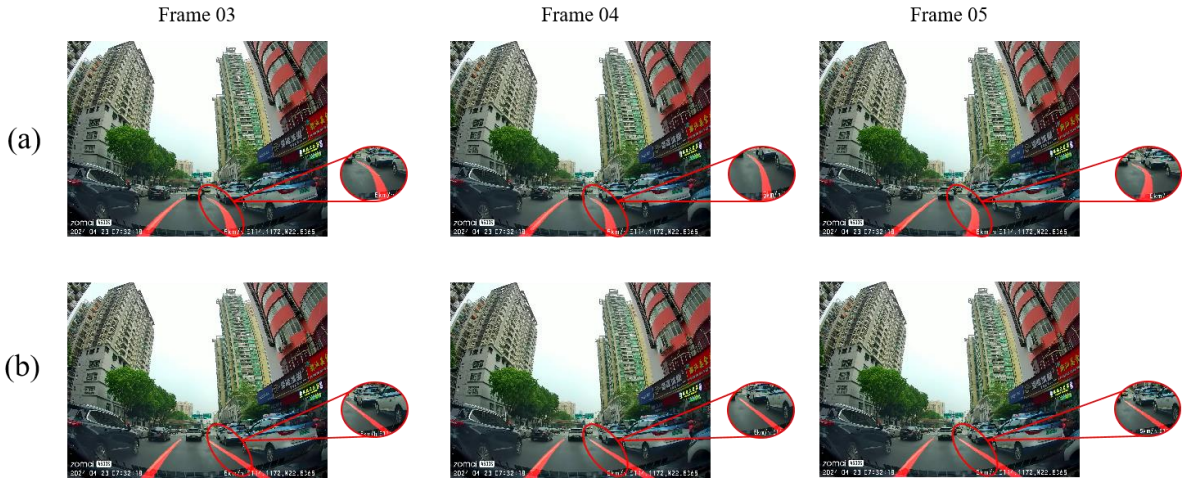


Figure 14. Comparisons of different Kalman filters. (a) The tracking result of the common Kalman filter from frame 03 to frame 05. (b) The tracking result of the adaptive Kalman filter from frame 03 to frame 05.

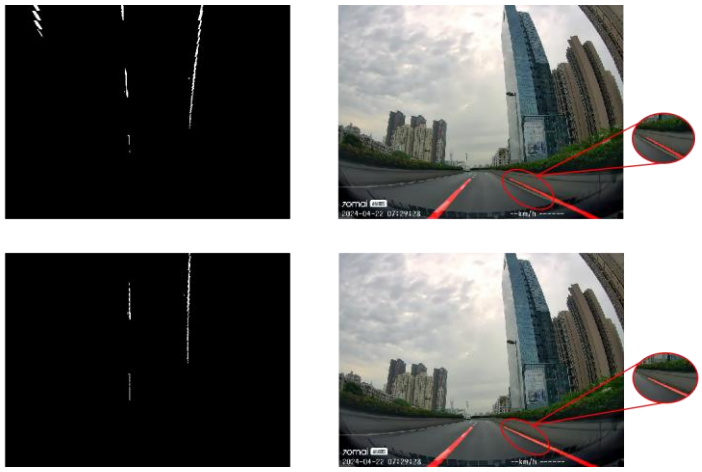


Figure 15. Comparisons between fixed IPM and adaptive IPM (AIPM). (a) The transformed edges and detection result of the fixed IPM. (b) The transformed edges and detection result of the adaptive IPM(AIPM).

3.5 Comparisons with other Algorithm

To further verify the performance of our algorithm, we tested an image-based method, HT[27], and a deep learning-based method, LaneATT [3], on our multi-environment dataset. In the HT method [27], the Region of Interest (ROI) and edge threshold parameters were adjusted to meet the requirements of the dataset. LaneATT[3] was pre-trained on the dataset for 100 epochs using ResNet34, with the learning rate and batch size set at 0.0003 and 8, respectively.

The results of these comparisons are presented in Table 3. Lane detection using HT[27] performed poorly,with an accuracy of only 83.3% and an *F1* score of 0.75. Owing to the presence of noise with linear structures to the lane lines, HT[27] struggles to accurately distinguish between them, leading to failures in lane detection under noisy conditions, as shown in the first line of Figure 16. On the contrary, LaneATT[3] get a good result in multi-environment, even under insufficient lighting conditions, as shown in the second line of Figure 16. However, the center offset error (*Ae*) of the LaneATT[3] is still approximately 2 pixels higher than that of our method. Additionally, the accuracy and *F1* score of our method demonstrate more significant improvements than those of the other algorithms. Compared to HT[27] and LaneATT[3], our method demonstrates enhanced performance in all evaluation metrics, especially in reducing the center offset error (*Ae*) by 2 to 4 pixels. This reduction underscores the potential application of our algorithm to Advanced Driver Assistance Systems (ADAS). Moreover, our method consistently maintains robust detection capabilities under various challenging conditions, accurately identifying the positions of the lane lines.

TABLE 3. Comparison of our method with other algorithms on our multi-environment dataset.

	F1↑	*Acc*↑	*FDR*↓	*FNR*↓	A_e ↓
HT[27]	0.75	83.3	0.25	0.22	10.94
LaneATT[3]	0.88	91.2	0.12	0.09	6.84
Our method	**0.91**	**92.9**	**0.09**	**0.07**	**5.45**

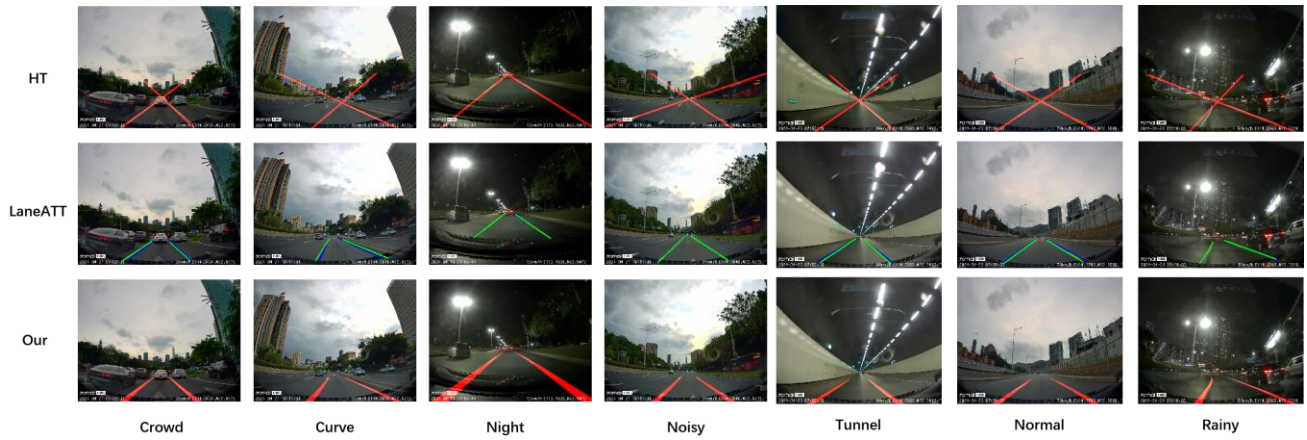


Figure 16. Visualization of different algorithms in the multi-environment dataset; in the results of LaneATT, blue lines are ground-truth, while green lines are ture -positives.

4. CONCLUSION

This paper proposes a robust lane detection and tracking method based on vanishing point for multi-environment scenarios. Gabor filters and coordinate transforms are employed to implement a vanishing-point based adaptive IPM. A spatio-temporal sequence module is designed to achieve dynamic sliding window detection of lane lines, effectively reducing road noise interference. For lane tracking, an adaptive Kalman filter is introduced, which adjusts observation noise according to lane width confidence. We constructed a multi-environment dataset and conducted extensive experiments to evaluate the performance of our method. Experimental results demonstrate that our method achieves high detection accuracy and robustness, and low center offset error across various environments. However, There are some limitations to the proposed method. Specifically, the computational complexity introduced by the convolution operations of the Gabor filters affects the real-time performance of the algorithm, which will be the focus of our future work.

REFERENCES

[1] Chetan N B, Gong J, Zhou H, et al. An overview of recent progress of lane detection for autonomous driving[C]//2019 6th International conference on dependable systems and their applications (DSA). IEEE, 2020: 341-346.
[2] Waykole S, Shiwakoti N, Stasinopoulos P. Review on lane detection and tracking algorithms of advanced driver assistance system[J]. Sustainability, 2021, 13(20): 11417.
[3] TABELINI L, BERRIEL R, PAIXAO T M, et al. Keep your eyes on the lane: Real-time attention-guided lane detection; proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, F, 2021 [C].
[4] ZHENG T, HUANG Y, LIU Y, et al. Clrnet: Cross layer refinement network for lane detection; proceedings of the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, F, 2022 [C].
[5] XU S, CAI X, ZHAO B, et al. Rclane: Relay chain prediction for lane detection; proceedings of the European Conference on Computer Vision, F, 2022 [C]. Springer.
[6] Wang J, Ma Y, Huang S, et al. A keypoint-based global association network for lane detection[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 1392-1401.
[7] Pan X, Shi J, Luo P, et al. Spatial as deep: Spatial cnn for traffic scene understanding[C]//Proceedings of the AAAI conference on artificial intelligence. 2018, 32(1).
[8] TuSimple. Tusimple lane detection benchmark, 2017. https : // github . com / TuSimple / tusimple benchmark,2017.6

[9] Pan X, Shi J, Luo P, et al. Spatial as deep: Spatial cnn for traffic scene understanding[C]//Proceedings of the AAAI conference on artificial intelligence. 2018, 32(1).
[10] Zhang X, Jiang R, Wang T, et al. Recursive neural network for video deblurring[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2020, 31(8): 3025-3036.
[11] Bisht S, Sukumar N, Sumathi P. Integration of hough transform and inter-frame clustering for road lane detection and tracking[C]//2022 IEEE International Instrumentation and Measurement Technology Conference (I2MTC). IEEE, 2022: 1-6.
[12] Javeed M A, Ghaffar M A, Ashraf M A, et al. Lane Line Detection and Object Scene Segmentation Using Otsu Thresholding and the Fast Hough Transform for Intelligent Vehicles in Complex Road Conditions[J]. Electronics, 2023, 12(5): 1079.
[13] ANDREI M-A, BOIANGIU C-A, TARBA N, et al. Robust lane detection and tracking algorithm for steering assist systems [J]. Machines, 2021, 10(1): 10.
[14] EM P P, HOSSEN J, FITRIAN I, et al. Vision-based lane departure warning framework [J]. Heliyon, 2019, 5(8).
[15] GAIKWAD V, LOKHANDE S. Lane departure identification for advanced driver assistance [J]. IEEE Transactions on Intelligent Transportation Systems, 2014, 16(2): 910-8.
[16] Samantaray S, Deotale R, Chowdhary C L. Lane detection using sliding window for intelligent ground vehicle challenge[C]//Innovative Data Communication Technologies and Application: Proceedings of ICIDCA 2020. Springer Singapore, 2021: 871-881.
[17] Ma L Y, Zhu H, Duan H. A method of multiple lane detection based on constraints of lane information[C]//2021 China Automation Congress (CAC). IEEE, 2021: 4059-4064.
[18] ZHANG Q, LIU J, JIANG X. Lane detection algorithm in curves based on multi-sensor fusion [J]. Sensors, 2023, 23(12): 5751.
[19] MA N, PANG G, SHI X, et al. An all-weather lane detection system based on simulation interaction platform [J]. IEEE Access, 2018, 8: 46121-30.
[20] FENG Y, LI J. Robust Accurate Lane Detection and Tracking for Automated Rubber-Tired Gantries in a Container Terminal [J]. IEEE Transactions on Intelligent Transportation Systems, 2023.
[21] TERRELL T, SIMPSON R. Two-dimensional FIR filter for digital image processing [J]. Journal of the Institution of Electronic and Radio Engineers, 1986, 56(3): 103-6.
[22] BUI T H, SAITOH T, NOBUYAMA E. Road area detection based on texture orientations estimation and vanishing point detection; proceedings of The SICE Annual Conference 2013, F, 2013 [C]. IEEE.
[23] ZHANG D, FANG B, YANG W, et al. Robust inverse perspective mapping based on vanishing point; proceedings of the Proceedings 2014 IEEE International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), F, 2014 [C]. IEEE.
[24] JEONG J, KIM A. Adaptive inverse perspective mapping for lane map generation with SLAM; proceedings of the 2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), F, 2016 [C]. IEEE.
[25] NIETO M, SALGADO L, JAUREGUIZAR F, et al. Stabilization of inverse perspective mapping images based on robust vanishing point estimation; proceedings of the 2007 IEEE Intelligent Vehicles Symposium, F, 2007 [C]. IEEE.
[26] JIANG L, LI J, AI W. Lane line detection optimization algorithm based on improved Hough transform and R-least squares with dual removal; proceedings of the 2019 IEEE 4th advanced information technology, electronic and automation control conference (IAEAC), F, 2019 [C]. IEEE.
[27] HUANG Q, LIU J. Practical limitations of lane detection algorithm based on Hough transform in challenging scenarios [J]. International Journal of Advanced Robotic Systems, 2021, 18(2): 17298814211008752.