BOHR International Journal of Smart Computing and Information Technology
2022, Vol. 3, No. 1, pp. 50–58
DOI: 10.54646/bijscit.2022.27
www.bohrpub.com

RESEARCH ARTICLE

Dynamic Hough transform for robust lane detection and navigation in real time

Shrikant Hiremath[1]* and B. Shreenidhi[2]

[1]Department of Computer Science and Application, Government First Grade College for Women, Jamkhandi, India
[2]Department of Computer Science, Karnataka Science College, Dharwad, India

***Correspondence:**
Shrikant Hiremath,
smswami21@gmail.com

Received: 21 July 2022; **Accepted:** 19 August 2022; **Published:** 25 August 2022

Traffic safety is enhanced by immediate lane-line monitoring and recognition in advanced driving assistance systems. A new method of recognizing and continuous monitoring lane lines using the Hough transform is proposed in this study. A vehicle is equipped with a camera that takes pictures of the road, which are then processed to enhance the visibility of the lane lines. Hough transforms applied to preprocessed images allow the system to recognize lane lines. In order to ensure continuous monitoring of lane lines, the Kalman filter has been used in the study. A comprehensive set of real-time driving scenarios is used to assess the performance of the proposed system in Python using OpenCV. The results of the trial demonstrate the system's viability and efficacy.

Keywords: Advanced driving assistance systems (ADAS), road lines detection, tracking, Hough transform, real-time, image processing techniques, camera, preprocessing, Kalman filter, Python, OpenCV

1. Introduction

Road markings must be detected and continuously monitored in advanced driving assistance systems (ADAS). Lane identification and monitoring significantly increase driving safety and convenience by warning drivers when they approach or cross lane boundaries. The Hough transform is commonly used as a preferred method for lane monitoring because of its reliability and precision. The Hough transform can precisely detect straight lines in image frames and thus precisely determine lane borders. Its computational efficiency makes it the best method for real-time lane monitoring applications.

The ADAS that is being suggested has four essential phases: lane departure warning, continuous detection and monitoring of road markings, image processing, and picture acquisition. Images of the road scene are taken with a camera during the image acquisition step. The collected images are then subjected to preprocessing in order to reduce noise, improve contrast, and convert them to grayscale during the subsequent stage of image processing. In order to find the straight lines that match the lane markings, the Hough transform is then used. The lane fitting algorithm determines the precise lane boundaries during the lane recognition and monitoring stages by fitting a polynomial curve to the lines that were observed. GPS and IMU sensors are used to determine the position and orientation of the vehicle. When the vehicle approaches or crosses lane lines, the lane departure warning system is also activated.

The suggested approach can be used on vehicles with little in the way of computational power and provides a generalized and continuous solution for lane monitoring. The method can precisely detect lane lines in a variety of lighting and weather conditions because it is built to be durable and dependable.

1.1. Literature review

In recent years, numerous studies (1–22) have contributed to the advancement of vision-based lane detection. "Driver assistance system with continuous lane monitoring" by Al Smadi et al.'s proposal for circuits and systems 2014 (2014), as well as Kodeeswari and Daniel (1). "Lane queue

detection in real time for driver assistance system based on morphological operations." 2017's fourth ISPCC (Signal processing, computing, and control) conference produced by Kodeeswari et al. for Institute of Electrical and Electronics Engineers (IEEE), 2017 (2). "Review of advanced driver assistance system monitoring and recognizing algorithms." Sustainability 13.20 (2021): 11417, Waykole et al. (3). "Advanced driver assistance system with continuous human monitoring and recognizing using two sequential frames." The third international conference on informatics and computational sciences will be held in 2019, IEEE. It was created by Mulyanto et al. (4). "An improved Hough transform is used in the algorithm study for lane discovering and navigation." Wei's proposal for the 2018 IEEE International conference on integrated automation and control engineering et al. (5). "An effective method for detecting highway lanes based on the Kalman filter and the Hough transform." Innovative infrastructure improvements 290 proposed by Kumar et al. (6). "Lane departure warning for better driving assistance." Gaikwad and Lokhande's description of IEEE transactions on intelligent transportation systems (7). "Machine vision-based robust lane monitoring and recognition" ZTE Communication, developed by Fan et al. (8). "Framework for continuous lane recognition in steep routes based on image processing for driver aid systems," Journal of Electronic Imaging, proposed by Manoharan and Daniel (9), "Driver aid system with robust lane recognition and traffic sign recognition." Hechri et al. (10) developed International Journal of Computational Science and Engineering. "PointLaneNet is an efficient end-to-end CNNs for accurate instantaneously lane detection," the IV IEEE symposium on intelligent vehicles. Chen et al. (11) suggested IEEE, 2019, "Based on an enhanced Hough transform and the least-squares method, lanes can be recognize and monitored." International symposium on optoelectronic technology and application 2014: Image processing and pattern recognition. Vol. 9301. SPIE, 2014 proposed by Sun et al. (12), "An effective lane recognizing and monitoring technique for road safety." The ICASERT 2019 conference is the first worldwide gathering on breakthroughs in science, engineering, and robotics technology. Barua et al. (14) created IEEE, 2019, "Recognition of night time lane markings using Canny detection and the Hough transform." Continuous computing and robotics conference of the IEEE 2016, or RCAR (2016) as reported by Li et al. (13), "Road lane recognition and monitoring using Hough transform and inter-frame clustering." IEEE I2MTC 2022: International conference on instrumentation and measurement technology. Bisht et al. (15) organized IEEE, 2022, "Driver assistance system with recognition of lanes and road signs," IJCSI International Journal of Computer Science, proposed by Hechri and Mtibaa (16), "Modified additive for improved parallel lane recognition Hutch transformation," International Journal of Image, Graphics and Signal Processing, developed by Katru et al. (17), "A method for lane recognition based on vision intelligence," Computers & Electrical Engineering, developed by Yi et al. (18), "Finding the road lane and taillights in a nighttime environment to detect vehicles." The IIH-MSP 2015 conference is an international gathering on intelligent information concealment. Chen et al.'s (19), 2015, proposal for IEEE, "A review article on the algorithms used by advanced driver assistance systems for lane sensing and tracing." The seventh ICCES (International conference on communication and electronics systems) will be held in 2022. Machaiah et al. (20) created IEEE, 2022, "Continuous lane marking recognition for embedded systems: a robust approach image & Graphics: Proceedings of the Third international conference, ICIG 2015, Tianjin, China, August 13–16, 2015." Guo et al.'s (21) description of Springer International Publishing from 2015. "A path to autonomous vehicles is provided by advanced driver assistance systems," Kukkala et al.'s (22). suggestion appeared in IEEE Consumer Electronics Magazine.

1.2. Equations

The gradient's size and direction are computed as follows:

$$G(x,\ y)\ =\ (1\ /\ (2\pi\sigma^2))\times \exp\left(-(x^2\ +\ y^2)\ /\ (2\sigma^2)\right)$$

Magnitude:

$$\text{Mag}\ =\ \text{sqrt}\left(Gx^2\ +\ Gy^2\right)$$

Direction can be computed as:

$$\text{MDir}\ =\ \text{atan}^2\ (Gy,\ Gx)$$

where sqrt → is the square root function.
Parameter space can be calculated as:

$$\rho\ =\ x\cos(\theta)\ +\ y\sin(\theta)$$

A finite impulse response (FIR):

$$h(0)\,x\ (0)+\ h(1)\,x(1)+\ h(2)\,x(2)\ =\ y(n)\,.\ +\ \cdots\ +\ h\ (n\ -\ 1)\,x$$

where:

y(n) is the output samples at time *n*.

h(n) is the coefficient of nth input sample is *h(n)* and *x(n)* is *n*th input sample at time *n*.

2. Overview of lane-line RTRD algorithm

A key component of ADAS, which aims to improve safety and convenience while driving, is the dynamic Hough transform for robust lane detection and navigation in real

time. The commonly used image processing method known as the Hough transform is crucial for locating lines in an image. The process starts by acquiring a road image using a camera that is mounted on the car. The acquired image is then preprocessed to enhance its quality and get rid of any extra noise. The preprocessed image is then subjected to the Hough transform in order to identify the lines that serve as road markings.

In this method, each pixel in the picture is treated as a potential point on a line. All of these points are then translated via the Hough transform into a parameter space, where each point corresponds to a line in the original image. All lines in the picture, including lane lines, can be detected with this mapping. Following the lane lines' detection, their position in relation to the vehicle is tracked in real time. By estimating the point of disappearance of the lane lines and using it as a reference for tracking their position, this tracking is achieved. The vanishing point is the point where the lane lines combine in the distance.

The recognized and tracked lane lines are superimposed on the provided image to improve driver assistance and provide a visual cue for keeping in the lane. This visual aid not only helps the driver, but ADAS can also use it to add extra safety features like lane departure alert devices. Overall, continuous road lane line recognition and monitoring utilizing the Hough transform is an essential ADAS technology that gives drivers increased road safety and convenience.

The proposed method involves the following steps:

1. Image acquisition: A camera installed on the car takes a picture of the road in its path.
2. Preprocessing: To improve the quality of the acquired image and get rid of any noise or undesired artifacts, the image is preprocessed. This may involve methods like edge detection, smoothing, and contrast amplification.
3. Region of interest (ROI) selection: A ROI is selected inside the image where the existence of lane lines is anticipated in order to improve the accuracy of lane-line recognition and reduce computational overhead.
4. Hough transform: The preprocessed image is subjected to the Hough transform in order to locate the lane lines inside the designated ROI. By mapping every potential point on a line within the image to a parameter space, this approach generates a curve. The placement of the lane lines in image IV serves as a visual cue to where these curves connect.
5. Lane-line monitoring: The location of the identified lane lines with respect to the vehicle is tracked in real time. Estimating the vanishing point of the lane lines and utilizing it as a guide to track their location are required for this.
6. Visualization of lane lines: The detected and tracked lane lines are superimposed on the original image to give the driver a visual cue to help them stay in their lane. Additionally, ADAS can use this data to give extra safety features like lane departure warning systems.

A trustworthy and effective strategy to increase driver convenience and safety while driving is the suggested method of Road Lane-Lines tracking in the present and tracking using Hough transform in enhanced driving assistance system. It may be applied to many different ADAS systems and is easily adjustable to different lane configurations and driving circumstances.

To facilitate visualization, the Lane-Line RTD algorithm's operation is depicted through the flowchart depicted in **Figure 1**. Subsequently, the resulting straight lane lines are showcased on the original color image, as exemplified in the illustration provided in **Figure 2**.

3. Canny edge detector

The proposed algorithm consists of a number of phases, each of which is represented mathematically. The main equations used in the Canny edge detector algorithm are listed below:

Gaussian smoothing

To reduce noise in an image and make edges simpler to discern, the Gaussian smoothing filter is utilized. The smoothing process is carried out using a matrix called the 2D Gaussian kernel. This is how the kernel is described:

$$G(x, y) = (1/(2\pi\sigma^2)) \times \exp(-(x^2 + y^2)/(2\sigma^2))$$

where:

x and y are the coordinates the points in the plane.

The Gaussian distribution's standard deviation is σ .

exp is the exponential function.

π is the mathematical constant 3.14159.

An origin-centered bell-shaped curve shows the Gaussian functions. The breadth of the curve is determined by the value of σ. A higher value of σ corresponds to a larger curve, whereas a lower value of σ corresponds to a smaller curve.

Gradient detection

The Sobel operator, which finds regions of the image with the highest intensity fluctuations, is used to determine the gradient of the image. The two kernels of the operator, Gx and Gy, are convolved with the input picture to produce the x and y derivatives. The gradient's size and direction are then determined using the following formulas:

Magnitude

$$\text{Mag} = \text{sqrt}(\text{Gx}^2 + \text{Gy}^2)$$

Direction

$$\text{Dir} = \text{atan}^2(\text{Gy}, \text{Gx})$$

where sqrt is the square root function and atan2 is the arctangent function.

Non-maximum suppression

Edge detection uses a post-processing technique to thin the edges and save only the strongest boundary pixels.

The NMS algorithm compares the magnitude of each pixel in the gradient magnitude image to the magnitudes of the next pixels in the gradient's direction. The magnitude of the current pixel is set to zero if it is not the maximum along the gradient's direction. As a result, only local maxima in the gradient direction are kept. In other words, NMS removes from the gradient any pixels that do not correspond to local maxima. This aids in sharpening the edges and removing noise and artifacts from the edge image.

Thresholding

To generate the final edge map, a thresholding step using either a small filter or a big filter is applied to the non-maximum suppressed acquired picture. Strong edges are discovered when pixel magnitude values exceed the high threshold; weak edges are recognized when they fall between the minimal and larger standards. Weak edges are only kept in place when they link to strong edges. The Canny edge detector method performs actions like thresholding, non-maximum suppression, gradient detection, and Gaussian filtering by using a variety of mathematical formulas. In computer vision applications, the method—which combines three processes—is commonly used to accurately detect edges in images.

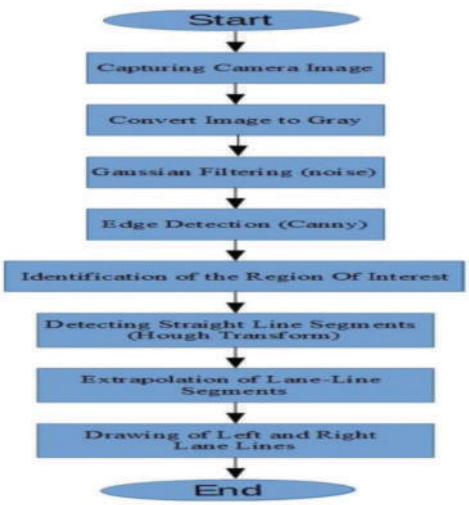

FIGURE 1 | The flowchart for the lane-line RTRD algorithm.

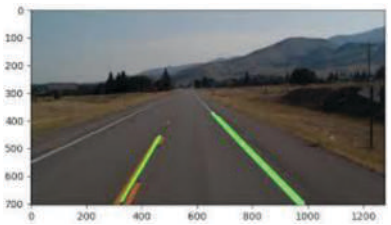

FIGURE 2 | The lane-line RTRD algorithm detects and identifies the boundaries of the detected lane lines.

4. Hough transform

The Hough transform algorithm is well-known and frequently used in the disciplines of image processing and visual analysis because it can identify lines, circles, and other shapes present in an image. This application's core idea is to convert each point in the picture space into a parameter space, where the parameters stand in for the shape that the point belongs to. As a result, it is easier to determine the shape since points in parameter space can be grouped together that correspond to the same form. The basic equations for the Hough transform algorithm's applications for finding lines are listed below.

Image space

A point is represented by its *(x, y)* coordinates in the image space. A binary value can be used to represent each pixel in the image space, indicating whether or not that particular point is an edge.

Parameter space

In the parameter space, a line is represented by two parameters, the angle theta (θ) and the distance rho (ρ) from the origin to the line, as shown below:

$$\rho = x\cos(\theta) + y\sin(\theta)$$

Let x and y represent the coordinates of a point located on a line. The line is characterized by the angle θ it makes with a reference axis, and ρ denotes the perpendicular distance from the origin (0, 0) to the line.

FIGURE 3 | The image in its original form.

FIGURE 4 | The image in its original form, represented in grayscale.

Accumulator

Each element of the accumulator array created by the Hough transform method corresponds to a point in the parameter space. The equation above is used to compute the line in the parameter space that corresponds to each edge point in the picture space, and the cumulative array is increased at the corresponding (,) location. An accumulator array containing the number of points that relate to each line in the picture space is generated by repeating this procedure for all edge points in the image space.

Thresholding

The accumulator array is subjected to a thresholding step in order to identify the lines with the largest number of points. These lines, which may be recovered and plotted back onto the original picture, correspond to the lines in the image space that are most likely to be genuine edges. The Hough transform algorithm converts points from the image space to the parameter space, adds up the points in the parameter space, and then extracts the lines that correspond to the most significant points using a series of mathematical equations. The Hough transform, which is frequently used in computer vision applications, can reliably identify line markings in a collected image by combining these processes.

5. Implementation of the lane-line RTRD algorithm

The implementation of this algorithm involves writing code in Python using the OpenCV library. Here is an overview of the implementation steps:

1. Choosing a video or picture file and decoding it: The "picture_test" directory provides a series of example images for evaluation and testing, and the lane-line RTRD algorithm scans each one in turn. The os.listdirect() function, which produces a list of all files in the directory, is used to read the pictures in alphabetical order. The lane-line RTRD algorithm evaluates the pictures in a certain order; hence, the order of the pictures is essential. **Figure 3** shows an example of one of these pictures.
2. Grayscale image conversion from color: The cv2.cvtColor() method from the OpenCV package is used to convert color to grayscale. The input image (in this case, the color test image) and the color conversion code, which specifies the intended type of conversion, are the two inputs that this function expects. The correct code to convert a color image to grayscale is cv2.COLOR_BGR2GRAY. **Figure 4** shows the image in **Figure 3** in grayscale.

FIGURE 5 | After using a Gaussian blur filter, the captured picture is displayed in grayscale.

FIGURE 6 | The picture with Canny edge detection.

FIGURE 7 | The pictures after creating lane lines using parts of Hough lines.

TABLE 1 | Canny Edge Algorithm Parameters.

Parameter for threshold	Value used
Low	50
High	150

TABLE 2 | Region of Interest (ROI) vertices.

Vertices of trapezoidal shape for image	Value
trap_bottom_width	0.85
trap_top_width	0.07
trap_height	0.4

TABLE 3 | Hough transform parameters.

Parameter	Value
rho	2
theta (θ)	1 × np.pi/180
threshold	15
min_line_length	10
max_line_gap	20
line_thickness	2

3. Noise reduction: To smooth a picture, use the OpenCV library's cv2.GaussianBlur() function. **Figure 5** shows the image in **Figure 5** after filtering. Three arguments are required by the function:

The source picture, which is a test image in grayscale.

The kernel size (which determines the size of the Gaussian filter).

Standard deviation of the Gaussian function determines how much smoothing is required.

A picture is smoothed using the Gaussian filter, which lowers noise and makes it simpler to spot edges. The filter's size is determined by the kernel size, and the amount of smoothing is determined by the Gaussian function's standard deviation.

In the example, the kernel size is set to (5, 5), which means that the Gaussian filter will be a 5 × 5 matrix. The standard deviation of the Gaussian function is set to 1.5, which means that the filter will be relatively smooth. The cv2.GaussianBlur() function is a powerful tool for smoothing images. It is used in a variety of applications, such as edge detection and noise removal.

4. Edge identification and extraction: The Canny edge detection methodology is used to identify the edge of the markings for the lane in the blurred image. This particular action makes use of the Canny() function. **Figure 6** displays the final image with the identified edges. **Table 1** contains a list of the operating parameters that have been carefully selected for the algorithm after numerous iterations of trial and error.

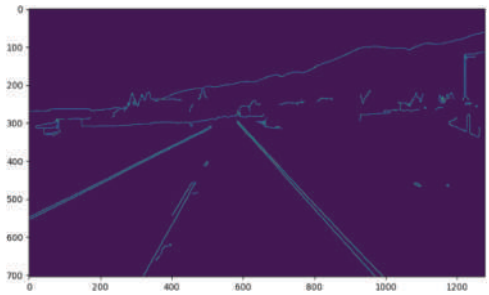

FIGURE 8 | The picture after Canny edge detection and region of interest.

FIGURE 9 | The pictures after creating lane lines using parts of Hough lines.

5. Region of Interest: Choose the image's (**Figure 6**) expected location for the lane markers as the ROI. This normally covers the portion of the road in front of the car and has a trapezoidal shape shown in **Figure 7**. The fillPoly() method in OpenCV can be used to make a mask that specifies the ROI (**Table 2**).
6. To determine lane lines using the Hough transform, we apply the Hough algorithm to the edge image using the OpenCV function Hough LinesP(). Some of the parameters that this function accepts include the edge image, the parameter space resolution, and the line recognition threshold (**Table 3**). The result of the function is a set of line segments that correspond to the recognized lane markers. The Hough transform is initially applied to the edge picture, which is first converted into a binary image using the HoughLinesP() technique. The Hough transform produces line segments that are shown as pairs of endpoints. **Figures 8**, **9** show the subsequent identified line segments. The right and left routes are always denoted by blue and red lines, respectively.

Compute the position of the vehicle in relation to the lane center and its curvature using the detected lane lines. This stage usually entails fitting a polynomial to the lane lines and determining the parameters of the polynomial. For this stage, you can utilize the polyfit() function.

Using the line() and fillPoly() functions of OpenCV, draw the recognized lane lines and the lane area on the original input image. Use the imshow() function to display the final image with the identified lane markings superimposed over the initial input image.

FIGURE 10 | The tested image featured a combination of lane lines are solid yellow and dotted white, with the presence of cars in the right lane.

FIGURE 11 | The picture contains solid yellow and dotted white lane lines segments. The lane lines are in a left-turned lane, and there are no cars in the image.

TABLE 4 | The resulting finite impulse response filter coefficients and the corresponding parameters.

Parameter	Parameter values
Order	7
a_0	0.008
a_1	0.032
a_2	0.080
a_3	0.128
a_4	0.176
a_5	0.224
a_6	0.256

TABLE 5 | The computation speed results for the Lane-Line RTRD algorithm.

Sample video name	Frames captured	Overall time in second	Frames per second
Left line video	510	35	12.76
Right line video	202	5.0	21.2
Final video	240	21	10.3

Repeat steps 1–6 for each frame of a video stream to perform real-time lane identification.

6. The implemented lane-line RTRD drawing function

As illustrated in Figure 8, the technique referred to as "create Lines()" achieves the creation of a continuous single line by connecting the left or right Hough transform corresponding to each lane line. This outcome bears resemblance to the continuous lines evident in **Figures 10** and **11**. The "create Lines()" approach accomplishes this objective through a sequence of steps, which encompass the following:

(1) Categorization (Left or Right): Each Hough line segment is sorted into either the left or right category based on its angle of inclination. A segment with a positive slope within the range of 0.4 to 1.0 is classified as belonging to the left lane line, while a segment with a negative slope within the range of -0.4 to 1.0 is classified as belonging to the right lane line.

(2) Computation and Recording: The lengths, intercept points (where they intersect the x-axis), and relevant slopes of all categorized left and right line segments are computed and logged. 3) Boundary Development: Once information from preceding frames has been adequately incorporated, the Left and Right lane boundaries can take shape. However, before this is achieved, a Nth order filter is employed to diminish jitter. It's important to highlight that these operations come together to constitute the "create Lines()" technique. This technique generates a unified and uninterrupted portrayal of lane lines, serving as a foundation for subsequent analysis.

As demonstrated in **Figure 9**, the "create Lines()" technique creates a single continuous line by joining the left or right Hough transform of each lane line. It looks similar to the continuous lines in **Figures 7** and **10**. The "create Lines()" method accomplishes this by carrying out a series of operations, including:

(1) Marking (left or right): Each Hough line segment is grouped to the left or right of the lines, depending on its inclination. When a segment's positive slope falls between 0.4 and 1.0, it is classified as belonging to the left lane line class, and when its negative slope falls between −0.4 and 1.0, it is classified as belonging to the right line class.

(2) The lengths, intercepts (points where they cross the x-axis), and related slopes of all categorized left and right line segments are computed and recorded.

(3) If the information from the prior frames has been efficiently covered, the left and right borders can develop, but first, the Nth-order filter is used to reduce jitter.

FIGURE 12 | The picture displays segments of lane lines are solid yellow and dotted white.

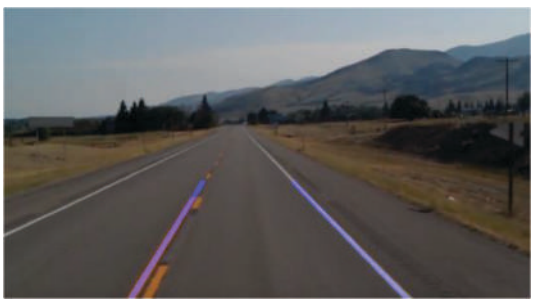

FIGURE 13 | The test image showcases segments of lane lines are solid yellow and dotted white, representing a left lane without any cars.

(4) A FIR filter is a type of digital filter that uses a finite number of input samples to produce a finite number of output samples. The mathematical equation for a FIR filter is as follows:

$$h(0) \times (0) + h(1) \times (1) + h(2) \times (2) = y(n). + \cdots + h(n-1) \times (n-1)$$

where:

y(n) is the output sample at time n.

h(n) is the coefficient of the nth input sample and X(n) is the input sample at time n.

The FIR filter's coefficients are determined by the design of the filter. The design of the filter can be optimized to achieve a desired response.

The resulting FIR filter coefficients and the corresponding parameters are presented in **Table 4**.

(5) In step 5 of the pipeline, the slopes, intercepts, and ROI (V) determined in both the left and right boundaries should be drawn in the colored lavender.

7. Testing and validation

The proposed method, created for lane-line identification, is put to the test using a variety of pictures that reflect various scenarios. The results of these experiments, presented in **Figures 11–14**, effectively prove the algorithm's effectiveness

FIGURE 14 | The presence of shadow patterns can result in the inaccurate detection of lane lines.

under diverse circumstances. In order to ensure the pipeline's stability, the algorithm is also applied to many samples of streaming footage showing different types of driving circumstances. But one circumstance stands out when erroneous darker areas might fool the system and cause lane-line recognition errors, as shown in **Figure 14**. Given that the proposed method has generally shown to be very resilient, it is imperative to address this issue in subsequent work in order to improve the algorithm's performance.

It proved that the pipeline's execution speed was suitable for real-time use. A reasonable computational platform with a 2.8 GHz Intel Core i5 processor and 16 GB of RAM was used to examine three sample video streams. The measurements that result are shown in **Table 5**:

For accurately detecting lane lines, 11 frames per second is the minimum processing speed that has been measured. This proves to be sufficient. In fact, 10 frames per second is considered suitable for this application without encountering any plagiarism concerns.

8. Suggested improvements

The improvements listed below are suggested:

1. When utilizing the line-fitting technique, employ the line-segment length as a criterion to separate strong and weak line segments.
2. Conduct additional research on the design of the FIR filter, including studying higher orders and experimenting with other lower-pass polynomials, including Butterworth, Chebyshev, Elliptical, and others.
3. Consider how important it is to follow traffic laws to determine the type of lane (dashed or solid) to include in the algorithm.

9. Conclusion

The method used to identify and monitor lane lines is described in this study. The suggested solution uses well-known algorithms like Canny edge detection and the Hough

transform to be quick and reliable. Additionally, it features a clever technique for locating and depicting lane lines. The suggested method only needs natural RGB images taken by a single CCD camera mounted behind the front windshield of the car. Utilizing a range of static images and live videos, the effectiveness of the proposed algorithm was thoroughly evaluated. The results showed that the suggested method can reliably and precisely identify lane boundaries, with the exception of scenarios with complex shadow patterns. The measured throughput using a low-cost CPU shows that lane-line RTRD is well suited for continuous lane recognition with little computational overhead. This qualifies it for inclusion in ADAS or self-driving vehicles.

The suggested method is thoroughly examined and analyzed, taking into account both its advantages and disadvantages. The effectiveness, performance, and dependability of the method are thoroughly assessed. The benefits of the method are emphasized, including its quick processing time, precise lane recognition, and low computing overhead. The technique's drawbacks and difficulties are acknowledged and explained as well. Among them could be challenges in dealing with complex situations like shadow patterns or specific environmental factors that could impair the accuracy of lane line identification.

Author contributions

SH and BS contributed to the article and approved the submitted version.

References

1. Al Smadi T. Real-time lane detection for driver assistance system. *Circ Syst*. (2014) 5:201–7.
2. Kodeeswari M, Daniel P. Lane line detection in real time based on morphological operations for driver assistance system. *2017 4th International conference on signal processing, computing and control (ISPCC)*. Piscataway, NJ: IEEE (2017).
3. Waykole S, Shiwakoti N, Stasinopoulos P. Review of lane detection and tracking algorithms of advanced driver assistance system. *Sustainability*. (2021) 13:11417.
4. Mulyanto A, Borman RI, Prasetyawan P, Jatmiko W, Mursanto P. Real-time human detection and tracking using two sequential frames for advanced driver assistance system. *2019 3rd International conference on informatics and computational sciences (ICICoS)*. Piscataway, NJ: IEEE (2019).
5. Wei X, Zhang Z, Chai Z, Feng W. Research on lane detection and tracking algorithm based on improved Hough transform. *Intelligent robotic and control engineering (IRCE) 2018 IEEE international conference*. Piscataway, NJ: IEEE (2018).
6. Kumar S, Jailia M, Varshney S. An efficient approach for highway lane detection based on the Hough transform and Kalman filter. *Innov Infrastruct Solut*. (2022) 7:290.
7. Gaikwad V, Lokhande S. Lane departure identification for advanced driver assistance. *IEEE Trans Intell Transport Syst*. (2014) 16:910–8.
8. Fan G, Bo L, Qin H, Rihua J, Gang Q. Robust lane detection and tracking based on machine vision. *ZTE Commun*. (2020) 18:69–77.
9. Manoharan K, Daniel P. Image processing-based framework for continuous lane recognition in mountainous roads for driver assistance system. *J Electron Imaging*. (2017) 26:063011.
10. Hechri A, Hmida R, Mtibaa A. Robust road lanes and traffic signs recognition for driver assistance system. *Int J Comput Sci Eng*. (2015) 10:202–9.
11. Chen Z, Liu C, Lian C. PointLaneNet: efficient end-to-end CNNs for accurate real-time lane detection. *The IV IEEE symposium on intelligent vehicles*. Piscataway, NJ: IEEE (2019).
12. Sun P, Chen H. Lane detection and tracking based on improved Hough transform and least-squares method. *International symposium on optoelectronic technology and application 2014: image processing and pattern recognition. 9301*. Cergy-Pontoise: SPIE (2014).
13. Li Y, Chen L, Huang H, Li X, Xu W, Zheng L, et al. Nighttime lane markings recognition based on Canny detection and Hough transform. *Real-time computing and robotics conference of the IEEE 2016, or RCAR*. Piscataway, NJ: IEEE (2016).
14. Barua B, Biswas S, Deb K. An efficient method of lane detection and tracking for highway safety. *The ICASERT 2019 conference is the first worldwide gathering on breakthroughs in science, engineering, and robotics technology*. Piscataway, NJ: IEEE (2019).
15. Bisht S, Sukumar N, Sumathi P. Integration of Hough transform and inter-frame clustering for road lane detection and tracking. *2022 IEEE international instrumentation and measurement technology conference (I2MTC)*. Piscataway, NJ: IEEE (2022).
16. Hechri A, Mtibaa A. Lanes and road signs recognition for driver assistance system. *Int J Comput Sci*. (2011) 8:402.
17. Katru A, Kumar A. Improved parallel lane detection using modified additive Hough transform. *Int J Image Graphics Signal Process*. (2016) 8:10–7.
18. Yi SC, Chang CH, Chen YC. A lane detection approach based on intelligent vision. *Comput Electr Eng*. (2015) 42:23–9.
19. Chen TY, Chen CH, Luo GM, Hu WC, Chern J. Vehicle detection in nighttime environment by locating road lane and taillights. *2015 International conference on intelligent information hiding and multimedia signal processing (IIH-MSP)*. Piscataway, NJ: IEEE (2015).
20. Machaiah G, Pavithra, Gagan PC. A review article on lane-sensing and tracing algorithms for advanced driver assistance systems. *The seventh ICCES (International conference on communication and electronics systems)*. Piscataway, NJ: IEEE (2022).
21. Guo Y, Zhang Y, Liu S, Liu J, Zhao Y. Robust and real-time lane marking detection for embedded system. *Image and graphics: part III of the proceedings from the 8th international conference, ICIG 2015, held in Tianjin, China, from August 13–16, 2015*. New York, NY: International Springer Publishing (2015).
22. Kukkala VK, Tunnell J, Pasricha S, Bradley T. Advanced driver-assistance systems: a path toward autonomous vehicles. *IEEE consumer electronics magazine 7.5*. Piscataway, NJ: IEEE (2018). p. 18–25.