IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 December 2023, accepted 6 January 2024, date of publication 16 January 2024, date of current version 24 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3354975

RESEARCH ARTICLE

A Robust Lane Detection Algorithm Adaptable to Challenging Weather Conditions

I-CHEN SANG AND WILLIAM R. NORRIS, (Member, IEEE)
Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana–Champaign, Urbana, IL 61820, USA
Corresponding author: I-Chen Sang (ichens2@illinois.edu)

This work was supported by the gift grant of TechnipFMC plc through the University of Illinois at Urbana-Champaign.

ABSTRACT Driver assistance systems and autonomous vehicle navigation have become important topics in vehicular technology. Among all of the functions, lane detection is one of the most important. A variety of approaches have been proposed in this field. While learning-based methods achieve impressive accuracy in detecting complex lane markings under clear daylight conditions, adapting these models to diverse weather conditions remains a challenge. On the other hand, geometric-based approaches require parameter tuning for different scenarios but require fewer computational resources. A self-tuned algorithm with high generalizability across diverse weather conditions is proposed in this paper. The algorithm integrates fuzzy logic-based adaptive functions with edge identification and line detection modules, enabling image adjustments in response to challenging weather conditions. The proposed tracking function utilizes previous detection results to fine-tune the selected Range of Interest (ROI), optimizing both accuracy and processing time. By incorporating these adaptive features into common geometric-based frameworks, the algorithm achieves higher detection rates compared to previous studies during challenging weather conditions. Furthermore, the proposed work exhibits better generalizability and significantly shorter processing time when compared to state-of-the-art learning-based models, as demonstrated through extensive testing on multiple datasets.

INDEX TERMS Adaptive, fuzzy logic, image processing, lane detection.

I. INTRODUCTION

Driver assistance and autonomous driving systems have become important topics in vehicular technology. To reduce human error and thereby enhance the safety of driving, a variety of functions have been developed for driver assistance systems. The most common applications include Lane Keep Assistance (LKA), Autonomous Emergency Braking (AEB), and Autonomous Cruise Control (ACC) systems. The reliability of these functions has continuously improved as research advances.

Lane detection is regarded as an essential technology because it is a building block for most functions in driver assistance systems and autonomous driving systems. Several different approaches have been developed in this area. One of them involves collecting the position of the lane along a large number of roads, and creating a lane map with respect to the GPS signal [1]. Though the lane map data can be downsized by approximating the lanes with curves, a considerable amount of data collection work is still required. Apart from lane maps, proximity sensors have been used to detect lanes. For example, LiDAR sensors could be leveraged to construct a 3D map of the scene. By differentiating the ray intensity reflected from the road and the lane lines, the position of the lines were identified [2], [3].

Another common approach used camera sensors of various wavebands, and is known as the vision-only approach. As all lane marks were designed for human eyes, cameras have been adopted by a majority of lane detection algorithms. Other than vision-only approaches, [4] proved that the fusion of LiDAR and camera could further improve the detection rate.

Though many algorithms have been proposed, complex weather conditions can greatly influence the detection results from LiDAR and cameras [4], [5]. The light during day

The associate editor coordinating the review of this manuscript and approving it for publication was Tai Fei.

and night conditions vary widely, usually requiring different parameters or even different algorithms. During rainy and snowy weather conditions, raindrops or snowflakes on the windshield reduce visibility. Furthermore, the fast-moving wipers pose additional challenges for camera-based algorithms. The water accumulating on the ground changes the reflected intensity of LiDAR rays.

The key finding from this study is a lane detection algorithm adaptable to varying weather effects. With an adaptively tuned edge detection and ROI selection module, the algorithm demonstrated exceptional generalizability over data collected under diverse weather conditions. It was shown that under identical initial parameters, the proposed algorithm could tune the parameters adaptively under all weather conditions. The findings were verified experimentally under a geometrical detection framework. Results demonstrated that the lane detection rate improved significantly under challenging weather compared to previous approaches. Furthermore, the adaptability of this work was also shown to outperform learning-based methods.

The remainder of this study describes the method, demonstrates results and provides a conclusion. All of the theoretical derivations and experimental setup are illustrated in detail in section III. The features of all datasets included in this work and the experimental setup are included in section IV. The analysis of the lane detection errors under various weather conditions is presented in section V. The conclusion of the proposed study and several potential future research topics are proposed in section VI and VII.

II. BACKGROUND

Lane detection algorithms can be broadly categorized into geometric-based (model-based) and learning-based approaches. Geometrical approaches leverage intrinsic lane characteristics such as parallelism and color [6]. Though manual parameter-tuning is usually required, geometrical approaches are more robust over various lighting and weather conditions. On the other hand, learning-based algorithms rely on data collected under different road and weather conditions. However, the availability of training datasets that cover diverse weather conditions remains limited. Well-known datasets like Tusimple [7], LLAMAS [8], and Caltech [9] mainly consist of images collected under clear weather conditions. However, the size of data in challenging weather is still limited [10]. Consequently, developing robust learning-based methods for different weather conditions becomes more challenging.

To tackle this challenge, researchers have proposed models trained on daytime data and tested on other weather conditions [11], [12], [13]. Additionally, efforts have been made to generate datasets with complex weather conditions, aiming to increase the diversity of available data. For instance, [14] and [15] successfully synthesized foggy and blurry images from data originally collected under clear daytime conditions. Similar attempts have been made to transform daytime images into nighttime representations [16]. However, rainy and snowy weather conditions have yet to be adequately addressed. The presence of raindrops or snowflakes on windshields, with their dynamic flow while the vehicle is in motion, presents additional challenges for both approaches.

FIGURE 1. The flow diagram of a typical lane detection algorithm.

Given the difficulties faced in the dataset of challenging weather conditions, efforts have been made to improve the generalizability of learning-based methods. The study in [15] created images with blurred lane lines in order to enhance the performance of algorithms under complicated weather conditions. The work in [17] adopted self-attention distillation (SAD) to improve the algorithms' accuracy in the scenes where only limited visual cues were present.

In addition to learning-based methods, non-learning-based lane detection algorithms were also proposed in the community as they are not limited by the size and abundance of datasets. For example, [18] utilized EDLines [19] along with a temporal tracking function to detect and connect the lanes in the images. References [20] and [21] incorporated a range of interest (ROI) selection strategy to raise the detection rate. In addition, [22] applied particle filter in a temporal-spatial framework performed lane detection task in multiple datasets. A general flow diagram of lane detection algorithms is shown in Figure. 1.

A. CONTRAST ADJUSTMENT

In most previous studies, typical RGB cameras were adopted for lane detection. Since lane lines are usually bright in color compared to the asphalt road, the images were usually translated into grey-level representations. A typical translation formula considering the luminance is $Graylevel = 0.3 \times Red + 0.59 \times Green + 0.11 \times Blue$.

It is more difficult to detect yellow lane lines than the typical white lines. In [23], the YUV color representation was leveraged to increase the detection rate of yellow and blue lines. The transformation from RGB channels to YUV color space used in [23] is written as

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

In [18], it was shown that the YUV color space was capable of enhancing the detection rate of the yellow and blue lane lines in typical daylight. However, this function had to be switched off manually under certain conditions, such as during nighttime operation. A robust algorithm automatically enabling/disabling the YUV channel under the proper situation could have improved the overall lane detection rate.

B. ROI SELECTION STRATEGY

The Range of Interest (ROI) selection strategy is a critical building block in lane detection algorithms. Proper ROI

selection can increase the performance of lane detection significantly. Rectangular ROIs were adopted in a majority of previously proposed algorithms. The top (sky) and the bottom (hood) of the image were usually cropped. Rectangular ROIs were used in [6] and [24].

Typically, lane lines are parallel to one another. According to the optical property of cameras, they coincide at the vanishing point near the center of the image. Triangular ROIs were commonly used in lane detection algorithms [20], [21], [25], [26], [27]. The advantage of triangular ROIs is that the information from adjacent lanes can be filtered out, thereby increasing the accuracy of the center lane. However, as the vanishing point changes with the angle of the camera, so when a curved road is encountered, a fixed triangular ROI could fail to detect the lines.

In [18], a unique Λ–shaped ROI was adopted. With one vanishing point at the center and four adaptively calculated points at the bottom of the image, an Λ-shaped ROI could be fully defined. In adopting this Λ-shaped ROI, the system was able to further avoid the distraction of lane marks such as arrows and speed limits painted on the ground between lane lines. However, more calculation was required for all anchor points of this setup, and the smaller ROI selection limited the tolerance of detection errors. When the ROI did not include the lane lines, it significantly increased the possibility of a detection failure.

Given the variety of ROI selection strategies, it is important to design one that can adapt to the road and the field of view of the cameras.

C. LINE DETECTION

The lane lines are usually composed of elements with a similar color or gradient value [19]. However, lane lines can be painted in a range of colors and their appearances may also vary with illumination and shadows. Even with the gradient calculating edge detection method, the threshold value where the edges/non-edges are differentiated requires tuning. Studies were proposed on the tuning strategy of the threshold value. For example, [20] used an equation to adjust the threshold during rainy weather. However, after implementation, it was found that the equation had difficulty working at night. Therefore, a more robust and automated method to determine a proper threshold under varying weather and illumination conditions is needed.

III. METHOD

The developed algorithm presented in this study demonstrates precise lane detection through a vision-only approach. The primary contribution of this approach lies in the adaptation of an error-based control scheme to the field of image processing. In contrast to learning-based methods that determine parameters based on overall detection error, the proposed framework defined ''observable parameters'' and utilized them to adjust the corresponding ''target parameters'' accordingly.

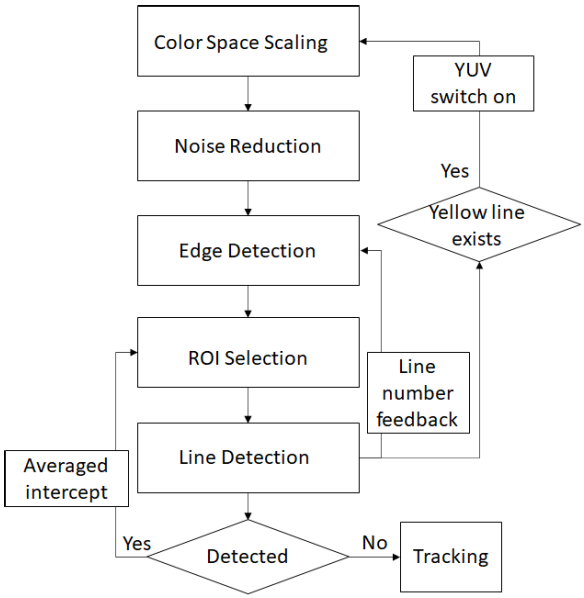


FIGURE 2. The flow diagram of the proposed algorithm.

Unlike approaches that rely on pre-trained parameters or select parameters under discrete conditions, the proposed method iteratively tuned the ''target parameters'' based on the deviation of ''observable parameters'' in each frame. This iterative process optimized overall accuracy and enhanced the algorithm's adaptability across diverse lighting and weather conditions.

Analogous to control theories, the amount of adjustment increased with the deviation of observable parameters, making the tuning process interpretable. In this work, a fuzzy inference system (FIS) was employed to determine the extent to which target parameters should be adjusted. Leveraging the probabilistic properties of fuzzy logic, the algorithm was tuned continuously, ensuring it is not unduly affected by sudden environmental changes.

The proposed framework comprised several modules, including contrast adjustment, noise reduction, ROI selection, line detection, and a tracking function to enhance algorithm stability. Figure. 2 illustrates the flowchart depicting the algorithm's sequence of operations.

The proposed method processed the input images in sequence from the database. After contrast adjustment, the image underwent a noise reduction process. Followed by the Canny edge detection function, a proper ROI was selected from the image, where line detection was done within the ROI. The detected line number and line color were used in adapting the edge detection and contrast adjustment function. In the case where no line was detected, the tracking function was enabled. When a valid result was obtained, its position is used to refine the ROI for improved outcomes.

A. COLOR SPACE SCALING

The color space scaling module was responsible for transforming the image from RGB to YUV or Grayscale color

FIGURE 3. At 3/4 of the image, the detected lines were sampled along the white lines shown in the figure.

space. The switching mode was based on comparing the current image and the previous detection results. With the default setting, the module transformed the images from RGB to Grayscale. If yellow lines were detected in the images, the transformation from RGB to YUV color space was leveraged instead.

According to [23], the difference between the U channel and the V channel is the indication of a yellow element. The threshold of a yellow pixel is defined as

$$U - V < -15 \tag{2}$$

To identify a yellow line, a horizontal line was plotted at 3/4 of the image height, centering at the detected lane line. 11 points with a separation of 1% image width were sampled along the horizontal line. An example is shown in Figure. 3. If any of the points met the requirement of Equation. 2, the line was regarded as a yellow line. If a yellow line had been detected on the left/right-hand side of the previous image, the corresponding side of the current image was transformed into the YUV color space. This approach successfully increased the contrast between the yellow lane lines and the asphalt ground (Figure. 4(b)). However, the performance of the YUV color space deteriorated under yellow light sources [23]. The YUV transforming module was therefore switched off when more than 15% of the whole image was defined as yellow pixels.

This approach could successfully detect the yellow lines, switch on the YUV color space representation, and identify the lines better than in grayscale images. By processing the lines on the left and right-hand sides independently, the image could be interpreted and detected correctly even when the color of the lines differed from each other. Additionally, the colors of the lines were checked every 30 frames in order to maintain the robustness of the contrast adjustment module. An example of successful lane detection with the aid of the automatic YUV switch is shown in Figure.5.

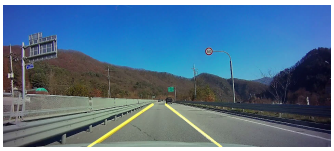
B. NOISE REDUCTION

The noise reduction technique used in this algorithm is a Bilateral filter [28]. The filter is capable of eliminating the noise in the image while preserving the edges, thereby assisting the lane line detection task.

(a) (b)

FIGURE 4. An example of a single sided YUV color space transformation with (a) the image with only grayscale transformation (b) the left hand side of the image in the YUV color space.

(a)



(b)

FIGURE 5. An example of the detection result of a frame before (a) and after (b) applying YUV channel.

FIGURE 6. The image before (left) and after (right) the denoise function done by the bilateral filter. It can be found that the noise in the images were greatly reduced.

The Bilateral filter uses the convolution of a kernel with the image. The value in the kernel is represented as

$$K_{x_i}(x) = \frac{1}{W} exp\left(\frac{||x_i - x||^2}{\sigma_s}\right) exp\left(\frac{|I(x_i) - I(x)|^2}{\sigma_I}\right) \tag{3}$$

where W is the normalization factor of the kernel, the first exponential is the Gaussian smoothing function of space, and the second exponential is the Gaussian smoothing function of intensity. In the equation above, x_i is the element in the kernel while x is the element in the image.

The parameters used in this work were determined experimentally. A range of parameters were tested using the datasets, and the most accurate option was selected. In this filter module, a kernel size of 7×7, with the variance of 25 in intensity and 50 in space was adopted. The image before and after the filtering process is shown in Figure 6. It was shown that the noise in the image was suppressed while the edges of the lane lines were still clear.

C. FUZZY-ADAPTIVE EDGE DETECTION

In the proposed algorithm, a Canny edge detection algorithm [29] was used to identify the points with larger gradient values. With two independent threshold values, the Canny edge detector outperformed the other algorithms with its higher degree of flexibility. When the gradient value was larger than the higher threshold, this pixel was regarded as an edge. When it was smaller than the lower threshold, the pixel was regarded as a non-edge point. And when the gradient value of the pixel was between the two thresholds, it was identified as an edge only if it was neighboring other edge points. The flexibility provided by the two thresholds contributed to the algorithm's adaptability over varying weather conditions.

In order to adapt to all scenarios, a fuzzy logic-based tuning method is proposed in this study. Fuzzy inference systems are also known as expert systems because they integrate the decision-making process of experts [30], [31]. Their ability to handle uncertainty gives the system more flexibility and tolerance toward image noise. Though an Adaptive Neuro-Fuzzy Inference System (ANFIS) was adopted in image segmentation [32], fuzzy inference systems have not been used to tune parameters.

Fuzzy systems are commonly used in control applications. The relative control commands are applied to systems to reduce errors based on required performance criteria. This work adopted a similar approach for the parameter-tuning process of image processing. With the empirical observations of the edge detector, the threshold value required tuning to maintain adequate edge information while lowering the computation cost. The fuzzy system rule base was derived from adjusting the threshold value based on the observed information.

The Canny thresholds adjustments were made recursively. The thresholds depended on the number of detected lines from the previous frame. An ''observable parameter'' (the detected line number) was used to tune the ''target parameter'' (the Canny thresholds). When the detected line number was higher than expected in the previous frame, the Canny thresholds were raised, discarding the edge pixels with lower gradient values. On the other hand, when the edges of the lines blurred and therefore reduced the detected lines, the thresholds were tuned lower to consider more pixels.

The detailed tuning process involved in the proposed module adopted a Mamdani fuzzy inference system (FIS). The Canny threshold started with an initial value of 1 in all experiments. The tuning process of the ''high threshold'' in the Canny edge detector was performed with the FIS. Both the input and the output linguistic values were determined experimentally. The detected line number was categorized into five different conditions according to the range of threshold values observed in the experiments. Specifically, the threshold variation was observed from 20 to 40 in one scene. Therefore, to converge within 30 frames (1 second), the increment was designed to be ±1.5/frame. As a result,

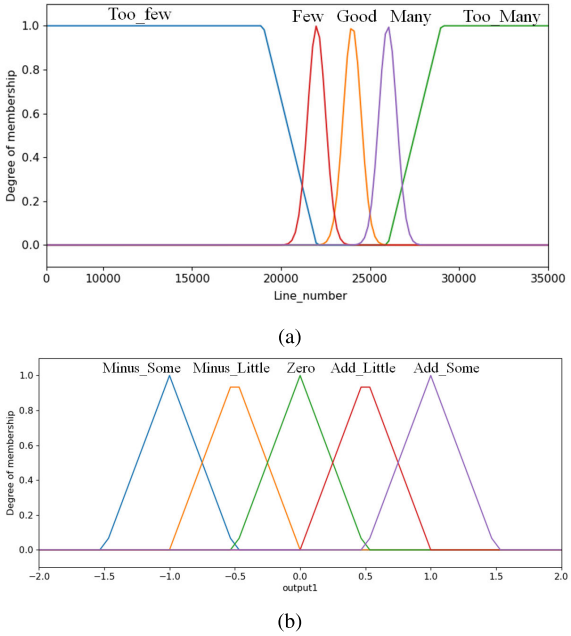


FIGURE 7. (a) The input linguistic variable "Line number" (b) The output linguistic variable "Canny threshold", Linguistic values and membership functions.

TABLE 1. The rule base of the fuzzy edge detection module.

Detected line number	Action to Canny threshold
Too few	Minus some
Little few	Minus a little
Good	Zero
Little many	Add a little
Too many	Add some

5 rules were designed in the proposed FIS. The case ''Zero'' adjusted the threshold value for $-0.5 \sim 0.5$. The cases ''Add/Minus Little'' adjusted the threshold value for $\pm 0 \sim 0.5$. The cases ''Add/Minus Some'' adjusted threshold value for $\pm 0.5 \sim 1.5$.

The input linguistic variables that mapped the input into five categories are shown in Figure. 7(a). Following the concepts of control algorithms, the shapes of the membership functions were comparable to the design adopted in [33]. Using the detected line number, the FIS rule base was defined in Table, 1. The Canny threshold was determined using the detected line number in the Line detection module. When the detected line number was higher/lower/similar to the expected value, a corresponding adjustment was applied to the Canny threshold. Finally, the decision made according to the rule base was defuzzified using the linguistic variables shown in Figure. 7(b) to generate a quantitative action and apply it to the system. The fuzzification and defuzzification method is the Center of Area (CoA) method. After tuning the ''high threshold'', the ''low threshold'' was calculated by dividing the high threshold by 3.

$$Threshold_{Low} = \frac{Threshold_{High}}{3} \tag{4}$$

The introduction of the fuzzy logic-based technique demonstrated that the algorithm could successfully adapt to different weather conditions and further improve the detection rate.

D. ROI SELECTION

The ROI (Range of Interest) is an effective approach to exclude non-lane objects from the images. In order to optimize the processing time and system robustness, a triangular ROI was adopted in this study. Ideally, the triangular ROI should be based on the bottom of the image, with the tip (X_{ROI}, Y_{ROI}) located at the vanishing point. However, in practice, the tip of the ROI needs to be adjusted adaptively according to the changing road directions.

1) CALCULATION OF X_{ROI}

The default position of the X_{ROI} was set at the center of the image. When the vehicle was going through a turn in the road, one of the lines might be cropped from the ROI. As a result, the X_{ROI} was adjusted 5% towards the left when the lane line on the left-hand side disappeared, and vice versa. This setting was shown to overcome the failed frames when the vehicle was turning and also worked well on all selected datasets in this work.

2) CALCULATION OF Y_{ROI}

Though the typical vanishing point is at the center of the image, setting the Y_{ROI} at 1/2 of the image height might crop part of the lane when either the road or the camera is not horizontal. The default position of the y-axis intercept was set at 2/3 of the image height to ensure the complete lane lines were shown in the ROI.

Based on image projection principles, the parallel road lines intersect at the vanishing point of images. However, though usually centered on the horizontal axis, different camera mounting angles and the field of view change the height of the vanishing point. A mechanism was introduced in this algorithm to trim the ROI according to the camera configuration. When the previous frame was successfully detected, e.g. detection of both the lines on the left and right-hand side, the y-axis intercept was lowered to 1.1 times the vanishing point height. When either of the lanes was not detected, the Y_{ROI} was replaced by the averaged vanishing point height in the previous 30 detected frames.

The methodology mentioned above was used to derive the X_{ROI} and Y_{ROI}. The ROI was able to adapt to varying camera angles, effectively cut out unwanted objects, and leave only the lane in view. This approach successfully reduced the difficulties encountered in the line detection process.

E. LINE DETECTION

The line detection was performed using the Hough Transform [34] and some geometrical properties. The distance resolution of the Hough transform was set as 1 pixel, while the angular resolution was 1 degree. The accumulator threshold of the Hough Transform was set to 5, which meant that only line candidates that went through at least 3 edge points were considered. This value could be fixed since the threshold of the Canny edge detector was adaptive. Specifically, if only very few lines had intersected over 3 edge points, the threshold of the Canny edge detector decreased, allowing more edge points to be considered.

The geometric properties of the lines were used to filter out the lines that did not apply to normal road conditions. As in [20], only the lines with angles between 25° and 65°, or 110° and 155° were considered. In addition, the coordinates where the lines intercepted the bottom of the image were used to identify reasonable detection results. For the lines with angles between 25° and 65°, only the candidates that intercepted the bottom on the left-hand side were considered. For lines with angles between 110° and 155°, the intercepts on the right-hand side of the image were used. When the detected line angle was between 25 and 65, it was regarded as the line located on the left-hand side of the lane. In this case, only those lines intercepted with the bottom of the image at the left half of the image were considered valid candidates.

Using the geometric criteria, the candidates could be effectively identified. Based on the Hough accumulator, the three highest accumulated candidates on each side were chosen. Their Hough Transform parameters, ρ and θ, were averaged and recorded.

The averaged ρs and θs were used to visualize the detection results. The detected lines were plotted accordingly on the images. An example of the detection result visualization is shown in Figure 5.

F. TRACKING

A tracking function was also essential for the algorithm. Some of the previous studies used Kalman filter to predict the line position when the detection failed. However, when driving in the lane, the lines tend to stay in similar positions. When one of the frames had a deviated detection result, the Kalman filter takes the false velocity into account and predicts a result that was even farther from the lanes. In this case, a simpler solution was shown to benefit the detection rate more. When the algorithm failed to detect the left/right line in this frame, the ρ and θ from the previous frame were adopted.

IV. EXPERIMENT

To assess the accuracy and robustness of the proposed algorithm, comprehensive experiments were conducted on three distinct datasets. In these evaluations, the algorithm was compared against several prominent methods in the field. This section includes detailed information on the datasets and the experimental setups.

A. DATASETS

The DSDLDE dataset [35] offers a diverse range of weather conditions, making it an ideal choice for assessing the

robustness of the proposed algorithm. The dataset comprises 50 video clips, including 27 captured during the daytime and 23 during the nighttime. Each video has a resolution of 1080 × 1920 with three color channels (Red, Green, and Blue).

The dataset presents significant challenges due to its extensive variation in weather and road conditions. It encompasses a wide range of rain intensities, from drizzling to pouring, providing a comprehensive representation of different precipitation scenarios. Moreover, the dataset includes diverse scenes such as tunnels, highways, and urban areas, further adding to its complexity. The inconsistent camera setup makes the dataset more challenging. The camera was positioned inside the car, resulting in varying viewing angles across the different videos. The hood of the vehicle may take 1/10 to 1/3 of the image. Additionally, objects and their reflection on the windshield are visible in some of the clips (Figure. 9h). To eliminate the effect caused by objects in the vehicle, everything below the windshield was cropped from the images.

To demonstrate the generalizability of the proposed work, the algorithm was tested on the popular Caltech dataset [9] and Tusimple dataset [7] as well. The Caltech and Tusimple datasets include only clear to cloudy daytime images with more complicated lane markings. The traffic scene of the three datasets also highly differ from one another. While DSDLDE dataset contains various scenes, such as highway, urban, cave, and country, the Caltech dataset only contains urban scenes, and the Tusimple dataset contains only highway.

B. EXPERIMENTAL SETUPS

The experiment was performed independently between different videos. The frames in the videos were processed in sequence by the algorithm, simulating the scenario of a computer processing the images captured by the onboard camera. The validation of the algorithm was performed by processing the images in sequence using a PC with an Intel core i7, 3.2GHz CPU with 16GB RAM, and a GTX 1660 GPU with 16GB memory. Each clip was processed independently with identical initial conditions. When conducting the comparison study, the proposed algorithm was executed on the CPU. And the compared algorithm (CLRNet) [36], which was trained with the training set of the Tusimple dataset, and tested on the DSDLDE dataset using GPU.

The detection algorithm was programmed in Python 3.8 under Windows 10 operating system and included the fuzzylab, OpenCV2, and numpy libraries. A minor edit was made in the fuzzylab package to realize the designed shape of the fuzzy membership functions. The revised version is available at https://github.com/ichensang/fuzzylab. The training and testing of CLRNet was conducted according to the software and package requirements listed on the official github of CLRNet.

During the evaluation of the proposed algorithm, there was no need for additional resizing or cropping, except for the area below the windshield. In contrast, to optimize the performance of CLRNet on the test set, an additional resizing process for validation due to the significant difference between the camera views in the training and test sets. All images in the DSDLDE dataset were consequently cropped and resized to align with the geometry of the Tusimple dataset.

To ensure a valid and fair comparison, the verification process and the exclusion of samples were conducted in accordance with the methodologies employed in previous works published on each respective dataset. By adhering to these established rules, the results of our method were effectively compared with those reported in the literature. The experiments on the DSDLDE and Caltech datasets followed [18], [20], and [21]. The detection accuracy was calculated for every video independently. The formula for the detection accuracy is

$$Detection\ Accuracy = \frac{TP}{TP + FN} \tag{5}$$

where TP is true positive and FN stands for false negative.

In the experiments of DSDLDE dataset and Caltech dataset, a stricter frame-based metric was adopted. Instead of calculating the detection accuracy of individual lane marks, only the frames with visible lines on both sides were considered effective cases. Following previous studies, no false positive or true negative cases were included in the calculation. As verified, this method was comparable to the official quantitative metrics provided by the Tusimple dataset. With the provided ground truth and evaluation code, the proposed algorithm achieved 86.94% accuracy in the Tusimple dataset, while the abovementioned method measured 87.05%.

The result showed that the proposed algorithm could successfully detect most of the frames in the dataset. Figure 8 shows several examples of successfully detected frames.

V. RESULTS

Using the experimental setup and methodology previously discussed, a series of experiments were conducted. In the following subsections, the accuracy of lane detection on multiple datasets, the generalizability of the model, the contribution of each module, and the processing efficiency will be discussed sequentially.

A. LANE DETECTION ACCURACY

The proposed algorithm was evaluated on DSDLDE, Caltech, and Tusimple datasets. The detection result was compared with both geometric-based and learning-based methods. The results of the comparison are shown in Table. 2-7.

The performance of the proposed model on the DSDLDE dataset is presented in Table. 2 and 3. The detection accuracy from all six weather conditions is provided accordingly. The proposed algorithm demonstrably achieved high levels of accuracy (>97%) in daytime cases, while nighttime results

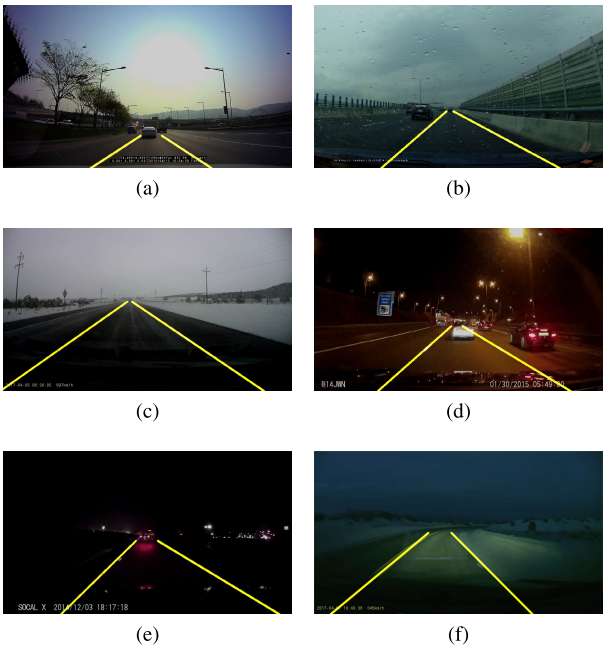


FIGURE 8. Successful detection results in (a)Clear days (b) Rainy days (c) Snowy days (d) Clear nights (e) Rainy nights (f) Snowy nights.

TABLE 2. The comparison between the proposed method and several previous studies. The unit of all numbers is percentage.

	Lee [18]	Ahmed [21] [20]	Proposed
Day Clear (5)	99.3	-	**99.5**
Day Rain (5)	96.9	92-97	**98.6**
Day Snow (3)	93	-	**97.9**
Night Clear (5)	97.7	93-99	**99.3**
Night Rain (5)	94.0	94.1	**94.6**
Night Snow (2)	92.2	-	**92.3**

ranged from 92% to 99%. Among all the weather conditions, the snowy night scene was the most challenging.

The algorithms this work compared with on the DSDLDE dataset are [18], [20], and [21]. In [18], a variety of videos were selected from each weather condition to calculate the average. In this research, an identical sampling approach was taken to achieve a more effective comparison as presented in Table. 2. The number of video clips sampled from each weather condition is shown in parentheses in Table 2. Although a simpler ROI design was adopted, the proposed algorithm provided a greater detection accuracy under all weather conditions as compared to [18].

In [20] and [21], although attempts to tackle the challenging weather conditions were made, the parameters only worked for rainy and nighttime clips respectively.

From the results in Table 2, the proposed algorithm works on all challenging weather conditions with the same initial parameters. The adaptive algorithm attained the highest detection rate among all the previous studies performed on the same dataset.

In addition to geometric-based methods, the proposed method was also compared with state-of-the-art learning-based methods. In this study, a comparison study with CLRNet [36] was conducted on the DSDLDE dataset. The

TABLE 3. The result of the comparison study between the proposed work and CLRNet [36]. (Unit:%).

	Typical		Complicated	
	CLRNet*	Proposed	CLRNet*	Proposed
DayClear	**99.9**	99.88	93.16	**99.26**
DayRain	**99.86**	96.57	49.23	**96.00**
DaySnow	**99.86**	97.54	86.93	**98.05**
NightClear	98.17	**99.27**	98.98	**99.80**
NightRain	95.59	**96.51**	58.27	**91.93**
NightSnow	**99.96**	90.95	56.37	**93.88**

*: Requires image resizing and cropping

overall accuracy of the proposed algorithm was 97.28% while CLRNet only reached 85.80%. The detailed experimental result is shown in Table. 3.

The DSDLDE dataset comprises a wide range of weather conditions and scenes, which can be categorized into "typical scenes" and "complicated scenes" to facilitate a clearer comparison. The typical scenes consist of regular traffic scenarios with clear windshields. The generalizability of lane detection methods is an important indicator of the performance. According to the result in Table. 3, the proposed algorithm showed better or comparable accuracy with CRLNet in the typical scenes. However, CLRNet's accuracy significantly dropped significantly in DayRain/NightRain/NightSnow cases to around 50%, while the proposed method still maintains accuracies that are higher than 90%. Some examples are shown in Fig. 9.

The proposed approach outperformed the learning-based method due to the invariant geometric properties of the lines during challenging weather conditions. While the color and clarity of the images changed with illumination or precipitation under extreme weather, the information processed by the neural network changed significantly. However, since the geometry of lanes remained the same, geometric methods outperformed learning-based methods as long as the algorithms could tackle varying image clarity.

On the other hand, both the proposed work and CLRNet were evaluated using the Tusimple dataset. Trained on the Tusimple dataset, CLRNet achieved a 96.8% accuracy in the test set of Tusimple. The proposed method showed a lower (87.0%) accuracy in this dataset as the focus was set on the robustness over different weather conditions instead of various lane configurations. However, when considering the performance drop between datasets, CLRNet exhibited a significant decrease from 96.8% on typical scenes in the Tusimple dataset to 50% on the non-typical scenes in the DSDLDE dataset. In contrast, the proposed method experienced a relatively smaller decrease from 99% on the DSDLDE dataset to 87% on the Tusimple dataset. Moreover, due to the substantial difference in the field of view between the DSDLDE and Tusimple datasets, significant cropping and resizing of images were required to make CLRNet work on the DSDLDE dataset. This observation highlights the better generalizability of the proposed method across diverse datasets, further emphasizing its practical applicability.

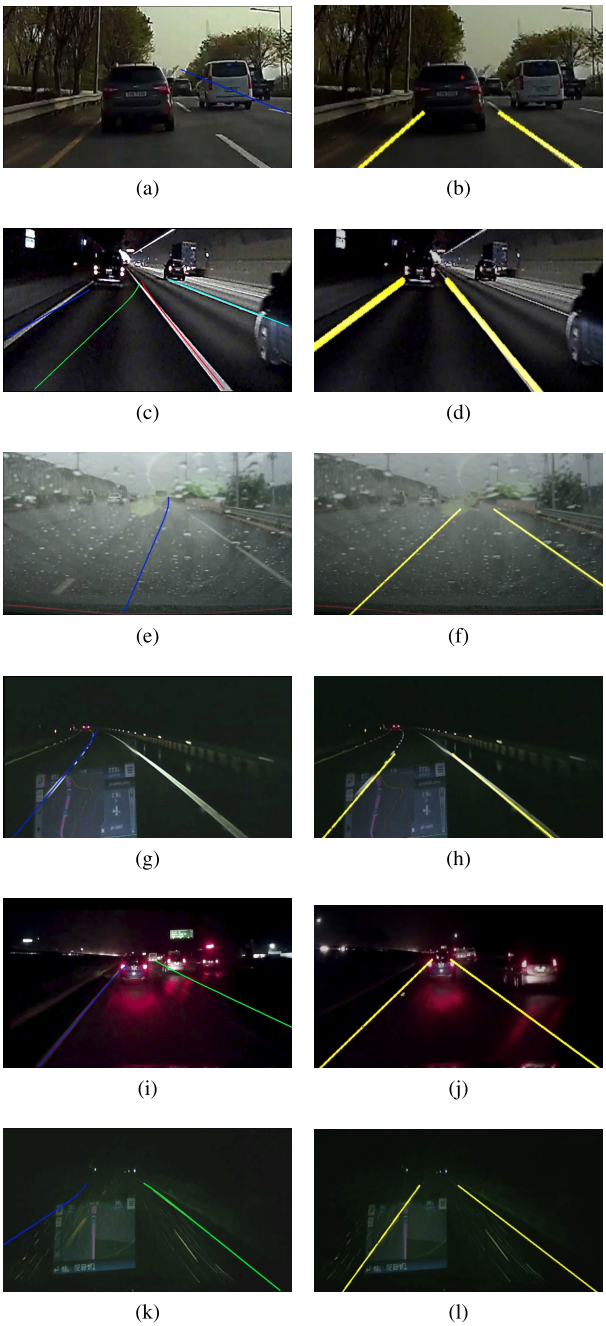


FIGURE 9. Detection results compared between the CLRNet (a)(c)(e)(g)(i), and the proposed method (b)(d)(f)(h)(j) in challenging scenarios.

The generalizability of the proposed algorithm was further tested with the Caltech dataset. The lane detection accuracy of the proposed method and previously published literature are shown in Table. 4. The result has shown that the proposed algorithm is able to generalize multiple datasets with different weather conditions and image configurations.

The proposed algorithm showed improved generalizability compared to previous work. Though the proposed method did not outperform most studies in regular (clear day weather) scenes, it demonstrated significant improvements in complicated scenes with various weather effects, including raindrops, snowflakes, and moving wipers on the windshield.

TABLE 4. The lane detection result on the Caltech dataset. (Unit:%).

	Proposed	[37]	[38]	[39]	[40]
Cordova1	99.1	98.0	93.2	98.5	**99.4**
Cordova2	97.3	**98.5**	93.6	95.2	97.8
Washington1	**99.6**	97.0	93.8	97.5	92
Washington2	93.5	99.1	98.3	98.1	**99.3**

TABLE 5. The result of the ablation study.

	Accuracy (%)	Time/frame (ms)
Fundamental	52	86
Fundamental + 1	52	91
Fundamental + 2	66	37
Fundamental + 4	77	31
Fundamental + 2 + 3	71	37
Fundamental + 2 + 3 + 4	87	13
Fundamental + 1 + 2 + 3 + 4	87	16

B. ABLATION STUDY

An ablation study was conducted to show the contribution of four key modules. This experiment was done on the Tusimple dataset because of the availability of an official evaluation metric. The evaluated modules are labeled below:

1) Yellow line detector
2) Triangular ROI
3) Adaptively adjusted triangular ROI
4) Fuzzy-tuned Canny edge detection threshold

When excluding all modules listed above, the algorithm was implemented without an ROI, a fixed Canny edge detection threshold, and without all yellow-line-related calculations. This fundamental configuration was defined as the baseline of this work. The result of the ablation study is shown in Table. 5.

As the Tusimple dataset contained very few images with yellow lane lines, the inclusion of the yellow line detector (1) added 5 ms in the processing time while having minimal improvement in accuracy. However, the triangular ROI module (2) significantly improved the detection results. By cropping unnecessary backgrounds, the triangular ROI improved the accuracy by more than 10%, significantly reducing processing time. Notably, even without an ROI selection module, the fuzzy-tuned Canny threshold module (4) alone could tune the threshold adaptively and diminish the unnecessary information in the images. The results demonstrated a significant improvement in accuracy (25%) and a reduction in processing time (55 ms).

The effect of the adaptively adjusted ROI (3) was shown by comparing cases "Fundamental + 2" and "Fundamental + 2 + 3." When including the ROI averaging mechanism previously mentioned, the size of the ROI was further reduced, raising the detection accuracy by 5%.

Based on the results presented above, it is evident that functions 2, 3, and 4 have made substantial improvements to the detection performance. Additionally, these improvements have had a positive impact on the processing speed,

TABLE 6. The processing time of a single frame compared to the proposed method and several previous studies.

	Processed resolution	Processing time (ms)
Lee [18]	300-400 pixels in height	7.4
Ahmed [21] [20]	540 × 960	26.11 / 30.24
Proposed	540 × 960	16.5

as indicated in the last column of the table. The utilization of a triangular ROI, even with a fixed pivot, effectively narrows down the search region for lane marks, resulting in enhanced accuracy. Furthermore, the adaptive pivot of the ROI, which averages previous detection results and tightens the ROI, further contributes to improved performance.

The fuzzy-tuned threshold of the Canny edge detector plays a crucial role in controlling the number of detected edge pixels. By dynamically adjusting the threshold, the algorithm maintains a reasonable number of edge pixels in the image, thereby minimizing the risk of mis-detecting noise pixels as lane marks during the Hough Transform process.

The experimental data also clearly demonstrates an improvement in processing time. The processing time of the Hough Transform is directly affected by the number of edge pixels present in the images. Without the triangular ROI, the processing time significantly increased due to the large number of detected edge pixels. The fuzzy-tuned Canny edge detector helps optimize the processing time by incorporating feedback from the Hough Transform and reducing the number of edge pixels to an essential level.

In the context of the yellow line detector, its impact was not particularly pronounced in this experiment, as the Tusimple dataset predominantly contains white lane marks. However, we have included an example below where the yellow line detector successfully assists in lane detection where the fundamental structure alone fails (Fig. 5).

C. PROCESSING TIME

An analysis was conducted on the processing time of the algorithm. A comparison was performed between the proposed algorithm and the referenced work. The adopted resolution and the processing time of all the algorithms are shown in Table. 6.

The proposed method also demonstrated a considerable improvement in processing speed compared to CLRNet. The results are presented in the table below, highlighting the significant advantages of the proposed approach over learning-based methods. Unlike CLRNet, which necessitates a GPU for execution, the proposed method can efficiently operate on a CPU, further underscoring its practicality and accessibility. The system does not require a GPU, so the processing speed of the proposed algorithm has the potential of being applied to real-time systems with minimal hardware requirement issues.

As a result of integrating a fuzzy logic-based parameter tuner with a geometric-based lane detection framework, this study has demonstrated comparable accuracy to previous work while exhibiting improved generalizability across various datasets. The algorithm's generalizability is primarily attributed to its robust and adaptive edge detection and ROI localization methods. Due to its modular design, the proposed approach can seamlessly integrate with various lane detection frameworks, showcasing its versatility and applicability in broader contexts.

TABLE 7. The processing time comparison between the proposed work and CLRNet on the Tusimple dataset.

Method \ Metric	System	Time/frame (ms)
CLRNet [36]	GeForce GTX1660/ 16G	22
Proposed	Intel i7-8700 CPU/3.2GHz/16G	16

VI. CONCLUSION

In this study, an algorithm equipped with a fuzzy logic-based adaptive module was introduced. The key findings from this work include an adaptive ROI selection module, an adaptive threshold-tuning module, and a color channel selection mechanism. The proposed approach demonstrated exceptional resilience in tackling diverse and challenging weather conditions, employing consistent initial parameters throughout. Additionally, the algorithm exhibited a higher level of generalizability over learning-based frameworks and outperformed previous geometric-based studies in terms of accuracy. There are some limitations in using the proposed geometric-based method. Specifically, learning-based methods still outperform geometric methods in dealing with multiple and curved lines. Integrating the proposed parameter-tuning method with state-of-the-art learning-based methods is a promising alternative.

VII. FUTURE WORK

There is substantial future potential for this algorithm. As mentioned above, the proposed approach not only functions as an independent lane detection solution but can be seamlessly integrated with other learning-based methods. The fusion between geometric-based methods and learning-based methods to further enhance the accuracy in lane detection is very promising, serving as an effective and accurate pre-processing module. Leveraging the algorithm's generalizability, future research could explore its application in alleviating the manual labor associated with labeling datasets containing challenging weather conditions. Additionally, drawing inspiration from the field of control systems, there is scope for transferring these concepts to the computer vision domain for adaptive-tuning applications.

REFERENCES

[1] S. Kim, M. Lee, M. Sunwoo, and K. Jo, "Probabilistic smoothing based generation of a reliable lane geometry map with uncertainty of a lane detector," *IEEE Access*, vol. 8, pp. 170322–170335, 2020.
[2] Q. Li, L. Chen, M. Li, S.-L. Shaw, and A. Nüchter, "A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios," *IEEE Trans. Veh. Technol.*, vol. 63, no. 2, pp. 540–555, Feb. 2014.
[3] Z. Chen, J. Zhang, and D. Tao, "Progressive LiDAR adaptation for road detection," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 3, pp. 693–702, May 2019.

[4] C. Rose, J. Britt, J. Allen, and D. Bevly, "An integrated vehicle navigation system utilizing lane-detection and lateral position estimation systems in difficult environments for GPS," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 6, pp. 2615–2629, Dec. 2014.

[5] M. Aldibaja, N. Suganuma, and K. Yoneda, "Improving localization accuracy for autonomous driving in snow-rain environments," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Dec. 2016, pp. 212–217.

[6] S. Luo, X. Zhang, J. Hu, and J. Xu, "Multiple lane detection via combining complementary structural constraints," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 12, pp. 7597–7606, Dec. 2021.

[7] *Tusimple Benchmark*. Accessed: Jun. 6, 2023. [Online]. Available: https://github.com/TuSimple/tusimple-benchmark/issues/3

[8] K. Behrendt and R. Soussan, "Unsupervised labeled lane markers using maps," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 832–839.

[9] M. Aly, "Real time detection of lane markers in urban streets," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2008, pp. 7–12.

[10] X. Li, S. Zhang, X. Chen, Y. Wang, Z. Fan, X. Pang, and J. Hu, "Robustness of visual perception system in progressive challenging weather scenarios," *Eng. Appl. Artif. Intell.*, vol. 119, Mar. 2023, Art. no. 105740.

[11] M. M. Yusuf, T. Karim, and A. F. M. S. Saif, "A robust method for lane detection under adverse weather and illumination conditions using convolutional neural network," in *Proc. Int. Conf. Comput. Advancements*, Jan. 2020, pp. 1–8.

[12] Y. Dong, S. Patil, B. van Arem, and H. Farah, "A hybrid spatial–temporal deep learning architecture for lane detection," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 38, no. 1, pp. 67–86, Jan. 2023.

[13] Y. Lee, J. Lee, Y. Hong, Y. Ko, and M. Jeon, "Unconstrained road marking recognition with generative adversarial networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 1414–1419.

[14] X. Nie, Z. Xu, W. Zhang, X. Dong, N. Liu, and Y. Chen, "Foggy lane dataset synthesized from monocular images for lane detection algorithms," *Sensors*, vol. 22, no. 14, p. 5210, Jul. 2022.

[15] Y. Liu, J. Wang, Y. Li, C. Li, and W. Zhang, "Lane-GAN: A robust lane detection network for driver assistance system in high speed and complex road conditions," *Micromachines*, vol. 13, no. 5, p. 716, Apr. 2022.

[16] T. Liu, Z. Chen, Y. Yang, Z. Wu, and H. Li, "Lane detection in low-light conditions using an efficient data enhancement: Light conditions style transfer," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1394–1399.

[17] R. Zhang, Y. Wu, W. Gou, and J. Chen, "RS-lane: A robust lane detection method based on ResNeSt and self-attention distillation for challenging traffic situations," *J. Adv. Transp.*, vol. 2021, pp. 1–12, Aug. 2021.

[18] C. Lee and J.-H. Moon, "Robust lane detection and tracking for real-time applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 4043–4048, Dec. 2018.

[19] C. Akinlar and C. Topal, "EDLines: A real-time line segment detector with a false detection control," *Pattern Recognit. Lett.*, vol. 32, no. 13, pp. 1633–1642, Oct. 2011.

[20] S. Sultana and B. Ahmed, "Lane detection and tracking under rainy weather challenges," in *Proc. IEEE Region 10 Symp. (TENSYMP)*, Aug. 2021, pp. 1–6.

[21] S. Sultana and B. Ahmed, "Robust nighttime road lane line detection using bilateral filter and SAGC under challenging conditions," in *Proc. IEEE 13th Int. Conf. Comput. Res. Develop. (ICCRD)*, Jan. 2021, pp. 137–143.

[22] S. Chen, L. Huang, H. Chen, and J. Bai, "Multi-lane detection and tracking using temporal-spatial model and particle filtering," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2227–2245, Mar. 2022.

[23] Q. Lin, Y. Han, and H. Hahn, "Real-time lane departure detection based on extended edge-linking algorithm," in *Proc. 2nd Int. Conf. Comput. Res. Develop.*, May 2010, pp. 725–730.

[24] A. S. Hanuman and G. P. Kumar, "Robust and real-time multi-lane and single lane detection in Indian highway scenarios," in *Proc. E3S Web Conf.*, vol. 309, 2021, pp. 839–846.

[25] R. Kanjee, A. K. Bachoo, and J. Carroll, "Vision-based adaptive cruise control using pattern matching," in *Proc. 6th Robot. Mechatronics Conf. (RobMech)*, Oct. 2013, pp. 93–98.

[26] M.-A. Andrei, C.-A. Boiangiu, N. Tarbă, and M.-L. Voncilă, "Robust lane detection and tracking algorithm for steering assist systems," *Machines*, vol. 10, no. 1, p. 10, Dec. 2021.

[27] Y. Y. Ye, X. L. Hao, and H. J. Chen, "Lane detection method based on lane structural analysis and CNNs," *IET Intell. Transp. Syst.*, vol. 12, no. 6, pp. 513–520, Aug. 2018.

[28] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, Jan. 1998, pp. 839–846.

[29] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[30] S.-H. Liao, "Expert system methodologies and applications—A decade review from 1995 to 2004," *Expert Syst. Appl.*, vol. 28, no. 1, pp. 93–103, Jan. 2005.

[31] L. R. Medsker, *Hybrid Intelligent Systems*. Cham, Switzerland: Springer, 2012.

[32] A. Küçükmanisa, O. Akbulut, and O. Urhan, "Robust and real-time lane detection filter based on adaptive neuro-fuzzy inference system," *IET Image Process.*, vol. 13, no. 7, pp. 1181–1190, May 2019.

[33] I.-C. Sang and W. R. Norris, "An autonomous underwater vehicle simulation with fuzzy sensor fusion for pipeline inspection," *IEEE Sensors J.*, vol. 23, no. 8, pp. 8941–8951, Apr. 2023.

[34] P. V. Hough, "Method and means for recognizing complex patterns," U.S. Patent 3 069 654, Dec. 18, 1962.

[35] *DSDLDE v.0.9: Video Clips for Lane Marking Detection*. Accessed: Feb. 15, 2022. [Online]. Available: https://drive.google.com/file/d/1315Ry7isciL-3nRvU5SCXM-4meR2MyI/view?usp=sharing

[36] T. Zheng, Y. Huang, Y. Liu, W. Tang, Z. Yang, D. Cai, and X. He, "CLRNet: Cross layer refinement network for lane detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 888–897.

[37] A. Gupta and A. Choudhary, "A framework for camera-based real-time lane and road surface marking detection and recognition," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 4, pp. 476–485, Dec. 2018.

[38] M. N. Rahaman, M. S. Biswas, S. Chaki, Md. M. Hossain, S. Ahmed, and M. Biswas, "Lane detection for autonomous vehicle management: PHT approach," in *Proc. 24th Int. Conf. Comput. Inf. Technol. (ICCIT)*, Dec. 2021, pp. 1–6.

[39] S. Wang, Y. Wang, and Y. Li, "Robust lane detection method based on dynamic region of interest," *IEEE Sensors J.*, early access, May 2023, doi: 10.1109/JSEN.2023.3274194.

[40] H. Zhou and X. Song, "Lane detection algorithm based on Haar feature based coupled cascade classifier," in *Proc. IEEE Asia–Pacific Conf. Image Process., Electron. Comput. (IPEC)*, Apr. 2021, pp. 286–291.

I-CHEN SANG received the bachelor's and master's degrees in physics from National Tsing Hua University, in 2014 and 2016, respectively. She is currently pursuing the Ph.D. degree with the Systems Engineering Program, University of Illinois at Urbana–Champaign. She has five years of experience in the optoelectrical industry as a Systems Engineer. She is a Research Assistant with the Autonomous and Unmanned Vehicle Systems Laboratory (AUVSL), founded by Prof. William R. Norris. Her research interests include the navigation of autonomous vehicles, sensor fusion, and image processing using fuzzy logic.

WILLIAM R. NORRIS (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in systems engineering from the University of Illinois at Urbana–Champaign, in 1996, 1997, and 2001, respectively, and the M.B.A. degree from the Fuqua School of Business, Duke University, in 2007. He has over 23 years of industry experience with autonomous systems. He is currently a Clinical Associate Professor with the Industrial and Enterprise Systems Engineering Department, University of Illinois at Urbana–Champaign, the Director of the Autonomous and Unmanned Vehicle System Laboratory (AUVSL), as well as the Founding Director of the Center for Autonomous Construction and Manufacturing at Scale (CACMS).