Advances in Technology Innovation, vol. 6, no. 3, 2021, pp. 137-145

Driving Assistance System with Lane Change Detection

Jia-Shing Sheu[1,*], Chun-Kang Tsai[1], Po-Tong Wang[2]

[1]Department of Computer Science, National Taipei University of Education, Taipei, Taiwan

[2]Department of Mechanical Engineering, Minghsin University of Science and Technology, Hsinchu, Taiwan

Received 04 February 2021; received in revised form 10 March 2021; accepted 12 March 2021

DOI: https://doi.org/10.46604/aiti.2021.7109

Abstract

In this study, a simple technology for a self-driving system called "driver assistance system" is developed based on embedded image identification. The system consists of a camera, a Raspberry Pi board, and OpenCV. The camera is used to capture lane images, and the image noise is overcome through color space conversion, grayscale, Otsu thresholding, binarization, erosion, and dilation. Subsequently, two horizontal lines parallel to the *X*-axis with a fixed range and interval are used to detect left and right lane lines. The intersection points between the left and right lane lines and the two horizontal lines can be obtained, and can be used to calculate the slopes of the left and right lanes. Finally, the slope change of the left and right lanes and the offset of the lane intersection are determined to detect the deviation. When the angle of lanes changes drastically, the driver receives a deviation warning. The results of this study suggest that the proposed algorithm is 1.96 times faster than the conventional algorithm.

Keywords: computer vision, embedded system, lane-shift detection, driver assistance system

1. Introduction

With the increasing use of vehicles, frequent traffic accidents occurs worldwide. To reduce the casualties caused by traffic accidents, this study proposes a lane-shift detection system that combines an embedded device and image identification. In this system, the embedded device and a camera module are used to capture lane images. The captured images are processed through grayscale, binarization, erosion, and dilation in order to filter image noise and achieve lane-line labeling. When an irregular deviation from a lane line is identified, the driver is automatically warned of the deviation.

Real-time lane detection using embedded devices is often impossible because of the high-performance hardware required by current lane detection algorithms. In the proposed algorithm, a substantial reduction in performance is required for lane detection to maintain the lane line detection in conventional algorithms. In addition, traditional algorithms depend on the use of masks or perspective transformation [1] to obtain a region of interest [2], but these prior manual adjustments and lane slope restrictions are not applicable to any road.

To overcome the shortcomings of traditional algorithms, this study proposes a lane detection algorithm that can greatly reduce lane-image-processing time and does not depend on masks and perspective transformation. The proposed intersection detection algorithm can not only run smoothly on embedded devices, but can also be adapted to lane lines of different shapes. Because the intersection detection algorithm does not use the region of interest algorithm, it can avoid mismatches between the region of interest and the lane area.

* Corresponding author. E-mail address: jiashing@tea.ntue.edu.tw

Tel.: +886-9-38397255; Fax: +886-2-27375457

2. Literature Review and Methodology

A lane departure detection system is an alarm system that helps drivers avoid unintentionally switching lanes. Driver assistance systems, which debuted in 2000, reduce the number of vehicle crashes by helping drivers preemptively avoid accidents [3]. Many new vehicles are equipped with such a system.

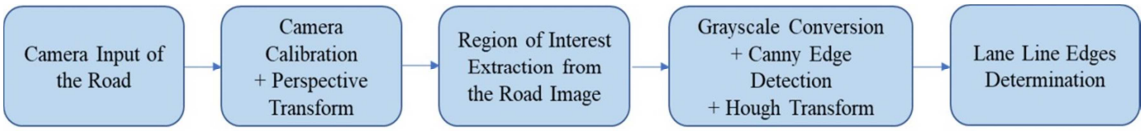

Fig. 1 Traditional lane detection algorithm

Fig. 1 presents the steps involved in the most current lane detection algorithms [4-7]. In this study, a new lane detection algorithm is proposed to improve the execution efficiency of traditional lane detection algorithms. Image processing in traditional algorithms can be roughly divided into five steps: perspective transformation, region of interest extraction, grayscale conversion [8], Canny edge detection [9], and Hough transformation [10].

In the proposed method, grayscale is retained; perspective transformation, region of interest extraction, Canny edge detection, and Hough transformation are removed. In addition, image binarization, image morphological erosion, expansion operation, and the algorithm for horizontal line intersection detection are added to the method of detecting lane lines. This method is not limited by the region of interest of the traditional algorithm. The new algorithm can be used to retain a large amount of the lane image information, adapt to different lane images, and provide rapid lane detection real-time data to determine whether the vehicle is deviating.

The first step of image processing is grayscale conversion. An original image captured from a camera module [11] is in the *RGB* color space, and such colored images lead to poor performance of direct calculations and consequent faulty lane line determinations. The grayscale equation (Eq. (1)) can be used to transform an *RGB* color space into a *YCbCr* [12] color space. As a result, the computational amount is reduced from 24 bits per pixel to 8 bits per pixel. This step is employed to improve the calculation efficiency and accuracy of lane detection. Then, the *Y* brightness obtained in the *YCbCr* color space can be considered the grayscale result.

$$Y = 0.299R + 0.578G + 0.114B \tag{1}$$

The second step is Canny edge detection, a compound edge detection algorithm that integrates a Gaussian filter [13], which is used to determine the intensity gradient of images, nonmaximum suppression [14], double threshold, and edge tracking through hysteresis for practicing edge detection. Gaussian filtering can be effectively used to reduce the noise generated by grayscale images, and its formula is as follows:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}} \tag{2}$$

where sigma (σ) of approximately 0.68-0.95 is suitable (normal distribution) and is user-defined. The commonly used 3 × 3 Gaussian filter matrix is as follows:

$$k = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{3}$$

The next step is to find the intensity gradient of images. Lane lines can be accurately detected by using Canny edge detection to select the Sobel [15] operator as a kernel for calculating the intensity gradient of images, given by the following equation:

$$G_x=\begin{bmatrix}1 & 0 & -1\\ 2 & 0 & -2\\ 1 & 0 & -1\end{bmatrix},\ G_y=\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix},\ G=\sqrt{G_x^{\,2}+G_y^{\,2}},\ \theta=\tan^{-1}\left(\frac{G_x}{G_y}\right) \tag{4}$$

Non-maximum suppression is divided into three angles, $\theta = 0°$, 45°, and 135°. The algorithm is employed to determine a point with the highest gradient change in each angle and remaining the points in the same direction. Figs. 2 and 3 present the example of nonmaximum suppression.

$$\begin{bmatrix}0.5 & 0.9 & 1\\ 0 & 0.5 & 0.7\\ 0.1 & 0.2 & 0.3\end{bmatrix}\rightarrow\begin{bmatrix}0.5 & 0.9 & 1\\ 0 & 0 & 0.7\\ 0 & 0.2 & 0.3\end{bmatrix}$$

Fig. 2 Nonmaximum suppression

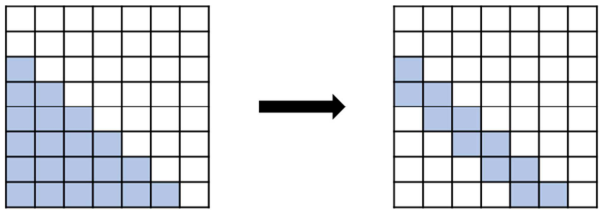

Fig. 3 Matrix with nonmaximum suppression

To suppress the noise and color changes of images, most algorithms use double thresholds for filtering, which can be achieved with high and low thresholds. The nonmaximum suppression algorithm is as follows: First, let x be the gradient intensity value of an edge pixel.

(a) When x is higher than the high threshold value, x is marked as a strong edge pixel.

(b) When x is less than the high threshold and greater than the low threshold, x is marked as a weak edge pixel.

(c) When x is lower than the low threshold value, x is suppressed.

The last step of Canny edge detection is edge tracking using hysteresis. However, weak edge pixels are controversial because they can be extracted from real edges and noise/color changes. To determine whether weak edge pixels should be judged in the real edge, we use eight adjacent pixels around the weak edge. According to the judgment method, if the gradient intensity value of an adjacent pixel is higher than that of the strong edge pixel, the weak edge pixel is considered the real edge. Otherwise, it is suppressed.

Traditional image processing uses the Hough transform to detect lines. Hough transform is an algorithm used for finding straight lines in images. The algorithm uses a point-slope formula to convert the original pixel points of a Cartesian coordinate system into the polar-coordinate system and a voting mechanism to find the required (γ, θ).

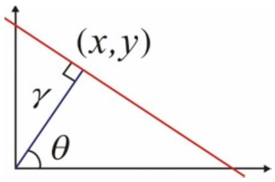

Fig. 4 Hough transform

In Fig. 4, γ is the shortest straight-line distance between the origin and line, and θ is the angle between the shortest line and the X-axis. The Hough transform is an algorithm used to determine the γ and θ coordinates, and after the determination of these two values, it can be used to create a straight line on the image. The normal representation of a line given by the following equation is used in this study:

$$\gamma = x\cos\theta + y\sin\theta \tag{5}$$

where θ is the angle between the normal straight line, and the X-axis and γ is the normal form of the length. This method of computation can solve an infinite number of problems by using the characteristics of a triangle function despite being limited to a certain range of angles.

3. System Structure

This study employs image processing technology to realize a lane deviation warning system. Continuous images captured by the embedded camera are used as input. After image processing, it is possible to judge whether the lane is offset by the change of lane line angle. In the case of deviation, the system warns drivers. Fig. 5 illustrates the system architecture obtained by using ICAM Definition method 0 (IDEF0) [16]. IDEF0 is a part of the IDEF family of modeling languages used in software engineering, and is built on the functional modeling language of Structured Analysis and Design Technique (SADT). This study is divided into three parts, namely image processing (Fig. 6), deviation judgment (Fig. 7), and notification to drivers.

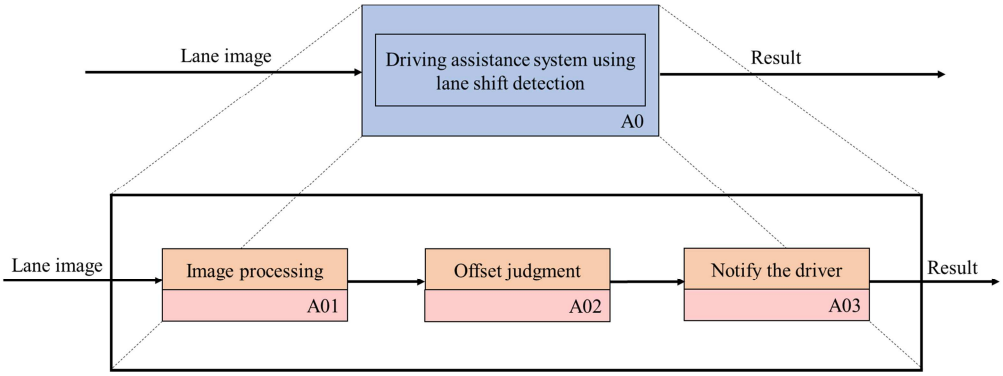

Fig. 5 System structure

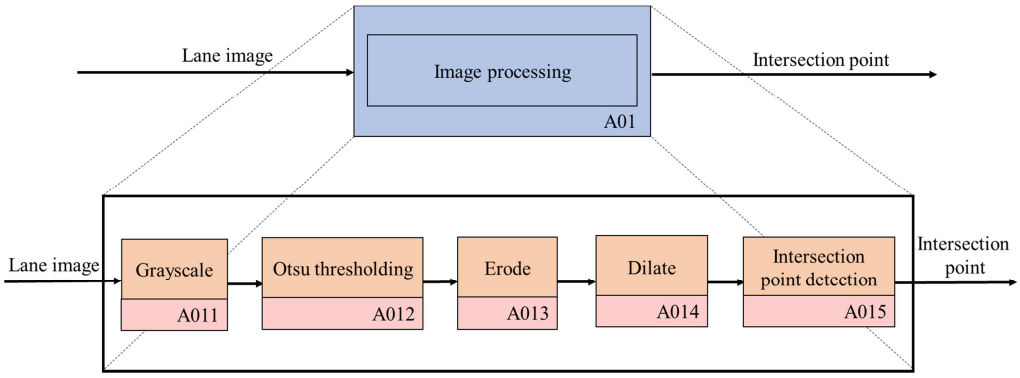

Fig. 6 Image processing

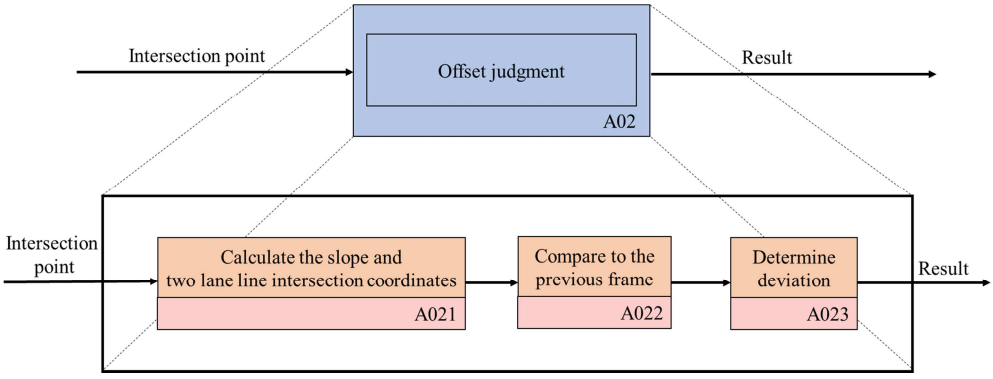

Fig. 7 Offset judgment

3.1. *Part 1*: *Image processing*

(1) Grayscale: The *RGB* color space of the lane image captured by the camera is converted to a *YCbCr* color space by using Eq. (1).

(2) Otsu thresholding [17]: Otsu is used to perform automatic image thresholding. The purpose of this algorithm is to return a threshold that separates pixels into two classes, the foreground and the background. This threshold is determined by minimizing intraclass intensity variance or by maximizing interclass variance. In this study's proposed system, Otsu thresholding is used to separate the lane line from the background.

(3) Erode [18]: By using an erosion morphology algorithm, the majority of the noise in binary black-and-white lane maps can be filtered out, and the main lane lines can be retained. In this part, we use a 5×5 rectangle matrix to erode the lane map.

(4) Dilate [19]: Lane lines narrowed using erosion are expanded back to their original width to facilitate the next step in lane detection. In this step, we use the 5×5 rectangle matrix to dilate the lane map. By using dilate (AND gate) characteristics, the operation time is reduced by half.

(5) Intersection point detection: Two horizontal lines parallel to the X-axis with a fixed range and interval are used for intersection detection. These two horizontal lines are used to determine four intersection coordinates between the lane line and two horizontal lines. This study simplifies the intersection detection method. Given a preset range between *y1* and *y2*, as the basic Y-coordinate of the two horizontal lines, the overlapping parts of the two horizontal lines and the lane line are used to perform a grouping calculation, and the four possible coordinates (*Ly1*, *y1*), (*Ry1*, *y1*), (*Ly2*, *y2*), and (*Ry2*, *y2*) can be obtained. The four coordinates are used to extend the lane line, find the intersection of the lanes, and present the slope of the left and right lanes, as shown in Fig. 8.

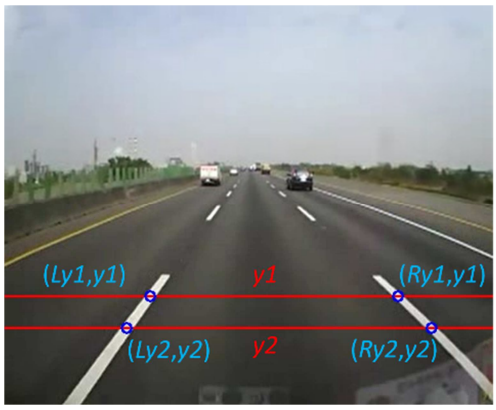

Fig. 8 Intersection point detection

3.2. *Part 2*: *Offset judgment*

(1) Calculation of the slope and two lane line intersection coordinates: The coordinates of the four intersection points obtained from part 1 are used to calculate the slope of the left and right lane lines The intersection points of the extended lane lines and the left and right lane lines are marked [20].

(2) Comparison with a previous frame: The slope of the left and right lane lines and the intersection coordinates of the extended left and right lane lines can be compared with the data in previous frames to determine whether the vehicle has positional offset.

(3) Deviation determination: To prevent obtaining unclear lane lines or noise interference, the offset data of the previous 15 frames are stored in buffer and subsequently compared with the current data to increase the judgment accuracy.

3.3. *Part 3*: *Notify the driver*

(1) When abnormal deviations in slope and intersection coordinates occur, a warning is sent to alert the driver.

4. Experiment Results

The experimental equipment consists of a Raspberry Pi board and a camera for still image processors, as listed in Table 1. The experimental system is also shown in Fig. 9.

Table 1 Experimental equipment

Components	Specification
Operating System	Raspbian
Central Processing Unit	ARM Cortex-A72
Random Access Memory	4 GB (LPDDR4)
Camera	8MP Raspberry Pi Camera Module (v2)

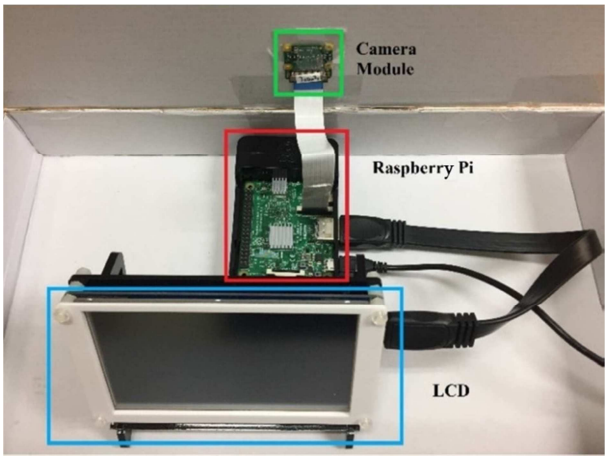

Fig. 9 Raspberry Pi combined with the camera module and a 7-inch touch display

Raspberry Pi is a popular embedded system developed by Raspberry Pi Foundation in the United Kingdom. The processor used in the experiment is a Broadcom BCM2711 SoC with a 1.5-GHz 64-bit quad-core ARM Cortex-A72 processor. Raspberry Pi has a 40-pin General Purpose Input-Output (GPIO) connector that enables users to connect peripherals such as LCD1602 modules and sensors through Inter-Integrated Circuit (I²C), Universal Asynchronous Receiver-Transmitter (UART), and Serial Peripheral Interface (SPI).

To reduce operation time and maintain accuracy, the 8-MP Raspberry Pi camera module (v2) is used to continuously capture images [21-22] with 640 × 480 pixel resolution (Fig. 10), and the *RGB* image is returned to the Raspberry Pi by using Eq. (1) for grayscale pretreatment. Matrix data are obtained from OpenCV to store the grayscale images (Fig. 11) in the memory.

Fig. 10 *RGB* image

Fig. 11 Grayscale image

After the grayscale conversion, image binarization is performed to accurately separate the lane lines from the asphalt. In this step, the Otsu algorithm is used to determine an adaptive threshold, and the threshold is used as a boundary, as shown in Fig. 12. The pixel values which are lower and higher than this threshold value are classified as black (lightness $Y = 0$) and white

(lightness Y = 255), respectively. After binarization, black and white images are subjected to the erosion morphology for filtering white impurities remaining in the binarization image and increasing the accuracy of lane line detection. In the erosion, 5×5 rectangular erosion elements are used, and Fig. 13 presents the erosion results.

Fig. 12 Otsu thresholding image

Fig. 13 Erosion image

After erosion calculation, the lane line shape is no longer true. Therefore, using an expansion operation in typology is necessary to restore the eroded lane line to obtain an appearance consistent with the actual situation. Moreover, in the expansion operation, 5×5 rectangular elements are used. Fig. 14 presents the results.

In intersection detection, two horizontal lines parallel to the X-axis are used to determine the intersection with the lane line. The method involves scanning for specific Y coordinates in the image, recording the X coordinates of white points, and dividing the points into "left" and "right" groups based on their X coordinates. In this manner, two specific Y coordinates are selected for scanning simultaneously, and the intersection of horizontal lines 1 and 2 with the left and right lane lines is obtained, which leads to four intersection points. In this step, the vehicle movement causes the Y coordinates of lane line to not exhibit a fixed relationship. Approximately 10 groups of horizontal lines with different Y coordinates are detected simultaneously during calculation to improve the detection accuracy of intersection points. Fig. 15 illustrates the detection of intersection points by using two red horizontal lines, and the blue circle represents the coordinates of four intersection points.

Fig. 14 Dilation image

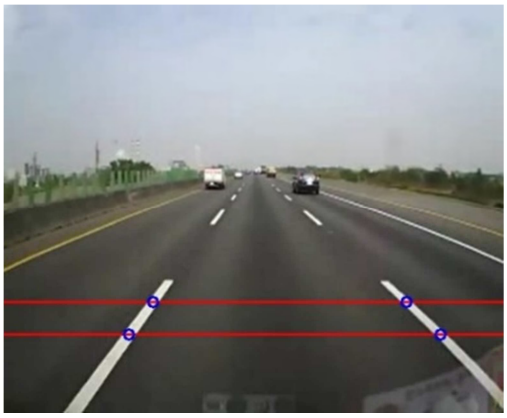

Fig. 15 Intersection point detection

The first step of the migration judgment is performed to calculate the point of intersection of the slope and the two lines, as shown in Fig. 16. By connecting the intersection points of left and right groups and extending them, the intersection coordinates of the slope of left and right lane lines and the lane lines of both sides are obtained. The intersection coordinates of left and right lane lines and the lane slope are stored in buffer, and the next frame of the image is captured using the camera. In the practical method of deviation judgment, the intersection coordinates of left and right lane lines and the lane slope of this frame with the historical data of the previous 15 frames stored in buffer are compared. When abnormal deviations in slope and intersection coordinates occur, a warning is sent to alert the driver of lane deviation.

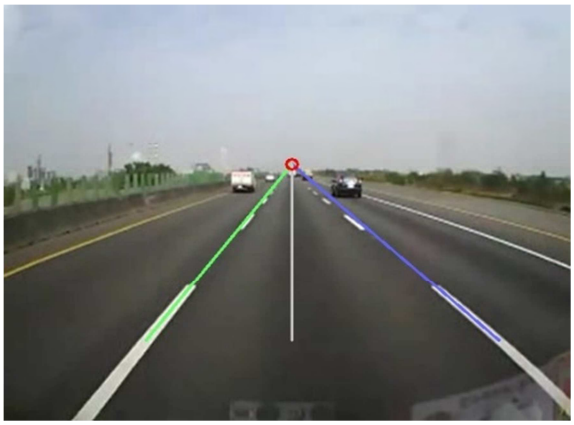

Fig. 16. Intersection point of slope and two lines

Table 2 compares the conventional algorithm [23] with the intersection detection algorithm proposed in this paper. The terms of comparison include the time spent on each step of image processing for each frame and accuracy.

Table 2 Experimental results

Procedure / Algorithm	Traditional lane detection	Intersection point detection
Grayscale	0.005 sec	0.005 sec
Canny edge detection	0.018 sec	-
Hough transform	0.023 sec	-
Otsu thresholding	-	0.005 sec
Erode	-	0.007 sec
Dilate	-	0.004 sec
Intersection point detection	0.001 sec	0.003 sec
Sum (sec)/frame	0.047 sec/frame	0.024 sec/frame
Accuracy	0.981	0.924

**Accuracy = (Number of detected/actual number of lanes)*

5. Conclusions

The intersection detection algorithms proposed in this study achieved a higher testing efficiency than the traditional lane detection algorithms did, and can be used to determine whether a vehicle is in the middle of a lane. The test results revealed that the proposed algorithm was 1.96 times faster than the conventional algorithm at a speed of 90 km/h on a highway. Moreover, the system can determine whether a vehicle is turning, as well as detect if the external environment causes the slope of the lane to have an irregularly large change in angle and send a warning to the driver. However, because lane detection depends on white pixels for group judgment after binarization, judgment errors can occur when white road signs are located on the lanes or when white vehicles are under strong light. These situations may require additional lane clustering to reduce the likelihood of judgment errors.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] F. Bonin-Font, A. Burguera, A. Ortiz, and G. Oliver, “Concurrent Visual Navigation and Localization Using Inverse Perspective Transformation,” IET Electronics Letters, vol. 48, no. 5, pp. 264-266, March 2012.

[2] L. Zhang and K. Yang, “Region-of-Interest Extraction Based on Frequency Domain Analysis and Salient Region Detection for Remote Sensing Image,” IEEE Geoscience and Remote Sensing Letters, vol. 11, no. 5, pp. 916-920, May 2014.

[3] J. S. Sheu, C. H. Chang, T. L. Huang, and H. J. Lin, "Lane Departure Warning System Using Front-View and Two Mirror-View Cameras," Scientia Iranica, vol. 22, no. 6, pp. 2126-2133, January 2015.

[4] D. C. Andrade, F. Bueno, F. R. Franco, R. A. Silva, J. H. Z. Neme, E. Margraf, et al., "A Novel Strategy for Road Lane Detection and Tracking Based on a Vehicle's Forward Monocular Camera," IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 4, pp. 1497-1507, April 2019.

[5] S. Jung, J. Youn, and S. Sull, "Efficient Lane Detection Based on Spatiotemporal Images," IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 1, pp. 289-295, August 2015.

[6] H. Wang, Y. Wang, X. Zhao, G. Wang, H. Huang, and J. Zhang, "Lane Detection of Curving Road for Structural Highway with Straight-Curve Model on Vision," IEEE Transactions on Vehicular Technology, vol. 68, no. 6, pp. 5321-5330, June 2019.

[7] C. Lee and J. H. Moon, "Robust Lane Detection and Tracking for Real-Time Applications," IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 12, pp. 4043-4048, February 2018.

[8] S. M. Borodkin, A. M. Borodkin, and I. B. Muchnik, "Optimal Requantization of Deep Grayscale Images and Lloyd–Max Quantization," IEEE Transactions on Image Processing, vol. 15, no. 2, pp. 445-448, February 2006.

[9] J. Lee, H. Tang, and J. Park, "Energy Efficient Canny Edge Detector for Advanced Mobile Vision Applications," IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 4, pp. 1037-1046, December 2016.

[10] R. K. Satzoda, S. Sathyanarayana, T. Srikanthan, and S. Sathyanarayana, "Hierarchical Additive Hough Transform for Lane Detection," IEEE Embedded Systems Letters, vol. 2, no. 2, pp. 23-26, July 2010.

[11] A. Gupta and A. Choudhary, "A Framework for Camera-Based Real-Time Lane and Road Surface Marking Detection and Recognition," IEEE Transactions on Intelligent Vehicles, vol. 3, no. 4, pp. 476-485, December 2018.

[12] S. Lee, Y. Kwak, Y. J. Kim, S. Park, and J. Kim, "Contrast-Preserved Chroma Enhancement Technique Using YCbCr Color Space," IEEE Transactions on Consumer Electronics, vol. 58, no. 2, pp. 641-645, May 2012.

[13] C. C. Lee and W. L. Hwang, "Mixture of Gaussian Blur Kernel Representation for Blind Image Restoration," IEEE Transactions on Computational Imaging, vol. 3, no. 4, pp. 783-797, December 2017.

[14] S. Jiang, T. Xu, J. Li, B. Huang, J. Guo, and Z. Bian, "IdentifyNet for Non-Maximum Suppression," IEEE Access, vol. 7, pp. 148245-148253, September 2019.

[15] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, "Design of an Image Edge Detection Filter Using the Sobel Operator," IEEE Journal of Solid-State Circuits, vol. 23, no. 2, pp. 358-367, April 1998.

[16] J. S. Sheu and C. Y. Han, "Combining Cloud Computing and Artificial Intelligence Scene Recognition in Real-Time Environment Image Planning Walkable Area," Advances in Technology Innovation, vol. 5, no. 1, pp. 10-17, January 2020.

[17] Q. Chen, L. Zhao, J. Lu, G. Kuang, N. Wang, and Y. Jiang, "Modified Two-Dimensional Otsu Image Segmentation Algorithm and Fast Realization," IET Image Processing, vol. 6, no. 4, pp. 426-433, June 2012.

[18] J. Y. Gil and R. Kimmel, "Efficient Dilation, Erosion, Opening, and Closing Algorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 12, pp. 1606-1617, December 2002.

[19] D. Nadadur and R. M. Haralick, "Recursive Binary Dilation and Erosion Using Digital Line Structuring Elements in Arbitrary Orientations," IEEE Transactions on Image Processing, vol. 9, no. 5, pp. 749-759, February 2000.

[20] M. Hua, M. K. Lau, J. Pei, and K. Wu, "Continuous K-Means Monitoring with Low Reporting Cost in Sensor Networks," IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 12, pp. 1679-1691, December 2009.

[21] S. Fernandes, D. Duseja, and R. Muthalagu, "Application of Image Processing Techniques for Autonomous Cars," Proceedings of Engineering and Technology Innovation, vol.17, pp. 1-12, January 2021.

[22] J. S. Sheu and W. H. Tsai, "Implementation of a Following Wheel Robot Featuring Stereoscopic Vision," Multimedia Tools and Applications, vol. 76, pp 25161-25177, January 2017.

[23] J. H. Yoo, S. W. Lee, S. K. Park, and D. H. Kim, "A Robust Lane Detection Method Based on Vanishing Point Estimation Using the Relevance of Line Segments," IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 12, pp. 3254-3266, December 2017.