Real-time Lane Departure Warning System on a Lower Resource Platform

Aman Irshad[1], Anam Ahmad Khan[1], Ibrar Yunus[1] and Faisal Shafait[1]
[1]School of Electrical Engineering and Computer Science, National University of Sciences and Technology (NUST)

***Abstract*—Real-time lane detection and tracking is one of the most reliable approaches to prevent road accidents by alarming the driver of the excessive lane changes. This paper addresses the problem of correct lane detection and tracking of the current lane of a vehicle in real-time. We propose a solution that is computationally efficient and performs better than previous approaches. The proposed algorithm is based on detecting straight lines from the captured road image, marking a region of interest, filtering road marks and detecting the current lane by using the information gathered. This information is obtained by analyzing the geometric shape of the lane boundaries and the convergence point of the lane markers. To provide a feasible solution, the only sensing modality on which the algorithm depends on is the camera of an off-the-shelf mobile device. The proposed algorithm has a higher average accuracy of 96.87% when tested on the Caltech Lanes Dataset as opposed to the state-of-the-art technology for lane detection. The algorithm operates on three frames per second on a 2.26 GHz quad-core processor of a mobile device with an image resolution of 640 x 480 pixels. It is tested and verified under various visibility and road conditions.**

Keywords— Lane Departure Warning System, Lane Detection, Convergence Point, Region of Interest, Lane detection, Lane Tracking

I. Introduction

Traffic congestion in urban areas has led to a drastic increase in car accidents. Each year over 37,000 people die in road crashes and traffic accident statistics in the United States (US) shows that excessive lane changing contributes to 4% of all police reported accidents. In recent years, many research groups have focused their attention towards developing efficient lane departure and warning systems. For this purpose, many lane detection algorithms using a variety of sensing modalities for various road conditions have been proposed. These algorithms either rely on machine learning techniques or involve computations based on the geometric shape of the road and lanes. This is a challenging task, as when detecting the current lane from the viewer's point of view we deal with complex scenarios such as no-lane areas, non-lane markings, writings on the roads, neighboring objects (and their shadows) and inconsistency in lane markings. The proposed algorithm provides a novel solution for real time lane detection. The sensing modality used is a mobile camera which is mounted on the car dashboard. After an image of the road is captured, the algorithm finds all edges approximated by straight lines from this image. This is followed by calculating a region-of-interest (ROI) for both the left and right lane markings and eliminating the lines outside of it. An ROI is a region where the lanes are supposed to reside. In the third step, we calculate the approximate convergence point from the lines lying in the left and right ROI. The fourth step involves eliminating lines based on the distance from the calculated convergence point. In the final step, the current lane of the car is selected by further processing and refining the lines remaining in the left and right ROI. The current detected lanes are extended backwards to intersect with the tracking ROI and with the help of simple heuristics, the algorithm suggests whether a lane is changing or not.

The algorithm is tested on the standard Caltech Lanes Dataset as well as with some collected road videos which displayed a variety of complex scenarios suggested above. Accuracy was maintained by the algorithm in all of these cases and results were provided in real time using a mobile device.

The paper is organized as follows: Section II gives a brief overview of related work in the field of lane detection, Section III illustrates the algorithm proposed for lane detection and Section IV describes the experimental setup. The results of detection are depicted through figures and tables. Section V further discusses the result and concludes the paper.

II. Related Work

In recent years, algorithms devised for lane detection either use machine learning techniques or utilize geometric model of the road. This section will highlight the significant work done in both categories.

Mohammed Aly [1] proposed a four step lane detection process based on generating a top view of the road, filtering using selective oriented Gaussian filters and fast Random sample consensus (RANSAC) algorithm for fitting splines followed by a post image processing step. Kreucher et al. [2] proposed a simple algorithm to detect lanes using Likelihood Of Image Shape (LOIS). It relies on the temporal locality of detected lanes in a sequence of images to adjust the detection. Kalman filter is later applied to predict the position of the vehicle (and lanes) in future frames. Wanga et al. [3] proposed a BSnake based lane detection and tracking. B-Snake can form any arbitrary shape by a set of control points, hence, presents a wider range of lane structures and uses Canny Edge Detection

978-1-5386-2839-3/17/$31.00 ©2017 IEEE

and Hough Transform for providing a good initial position for the B-Snake. The overall processing time for the Bsnake method is less than four seconds when run on a Pentium three system with 128 MB RAM.

Assidiq et al. [4] came up with a vision-based approach capable of performing in real time which utilize Final High Definition (F.H.D) algorithm. The edge detector is later applied to produce an edge image by using a modified canny filter with automatic thresholding. The yielded image is then sent to a line detector which produces lane boundaries. The method is limited by the assumption that the horizon is always parallel to the X-axis and suffers from weak detection results in rain.

Li Sha Sha [5] proposed a system in which Hough Transform is used for detecting the lane boundary after the initial image pre-processing and then the final result is fitted with least squares method. Khalifa et al. [6] proposed an algorithm which also does image pre-processing and then detects lines through Hough Transform, followed by fitting the lanes to a hyperbola model. The robustness of the model was discussed with varying parameters of curvature of roads, image resolution and processing speed. However shadowing, Hough line horizon overlaying with the lane boundary points and incorrect camera orientation can highly affect the accuracy.

Bertozzi and Broggi proposed a Generic Obstacle and Lane Detection System [7], which uses a stereo vision approach to detect lanes. Initially, perspective effect from the left and right images is removed through geometric transformation. Then the system uses morphological filters to detect lane markings. The algorithm [7] has been tested on speeds up to only 80Km/s and the algorithm runs at a rate of 80 Hz but is limited by the assumptions that the road markings are visible and the road surface is always flat.

Kreucher and Lakshmanan [8] proposed LANA (lane finding in another domain) algorithm, which is based on frequency domain features. Information related to the strength and orientation of spatial edges is provided by the frequency domain features. LANA took approximately 30 seconds to search over the 400,000 possible lane shapes on a Pentium-266MHz 96MB-RAM Desktop-PC. Benligiray, et al [14] proposed fast vanishing point estimation method to detect lane boundaries. The first step of the algorithm is to extract and validate the line segments from the image using EDLines. This is followed by removing lines based on their orientation and the location form the vanishing point. The algorithm is tested on the Caltech dataset and has an average accuracy of 95.9%. In recent years, the research trend in this domain has been shifted to the machine learning techniques. Gopalan et al. proposed a learning technique [9], using pixel-hierarchy feature descriptor to model spatial context information shared by lane markings and surrounding scene. An outlier-robust boosting algorithm is used to learn relevant contextual features and kernal discriminant analysis is used to initialize the weights of samples. Finally possible variations in the road scene are earned by analyzing tracked parameters. The tracker processes 240 x 320 pixel images at 25 frames-per-second on a 4-GHz processor. Ng et al. [10] proposed a learning algorithm that uses Overfeat CNN [11] for the purpose of detecting lanes. After detection of an object, same features are used to create a bounding box via regression. Density-based Spatial Clustering of Applications with Noise (DBSCAN) is used to get semantic lane information, which later clusters line segments into lanes. However, the system uses expensive sensors such as Light Detection and Ranging (LIDAR), radar and high-precision Global Positioning System (GPS) maps. Zhang et al. [12] proposed a lane detection algorithm based on support vector machine. The steps of this algorithm include road surface extraction by using Support Vector Machine (SVM) pattern recognition, image morphology operations, transforming the image into a bird's eye view by using the relationship of image coordinate system and world coordinate system, getting the center points from the road's mid line and regressing the road shape function by using SVM. This approach is suitable for both painted and unpainted roads. Accuracy is affected by bad weather and weakness of vision sensor itself. However, the specifications of vision sensors and processing speed is not mentioned in this paper.

Shuai et al. [13], proposed a novel lane detection algorithm based on machine learning in which image is treated with a pre-processing stage to remove noise and then Haarlike features are used rather than SVM. A class function g(x) decides whether a point x belongs to a lane or not. Lastly, Fisher Discriminant Analysis is used to decide the relative importance of samples. It assumes that the lanes are present only in the lower half region of the image and is tested with images of only 240 x 320 pixels.

The main objective of this research was to develop a lane departure warning system that is computationally efficient and can give real time results on a mobile processor with off-the-shelf specifications. For this purpose, machine learning techniques are found to be processor and memory intensive. State of the art deep learning techniques give accurate and real time results for object detection [10], [18], [19], [20] but is limited for object localization. This is because, deep learning techniques usually operates on down sized lower resolution image (256 x 256 pixels) so that results could be obtained with minimum processing time. Also, they usually operate on high end GPUs but their frame rate substantially drops on mobile platforms. Therefore, we followed the traditional computer vision method using the geometrical shape of the road to detect lanes from image of size 640 x 480 pixels. Figure 1 shows the overall flow of our designed methodology.

III. METHOD

The following section explains the proposed algorithm of lane detection. Correct detection of lanes is dependent on accurately calculating the region of interest where the current lane markings from the driver's point of view will reside.

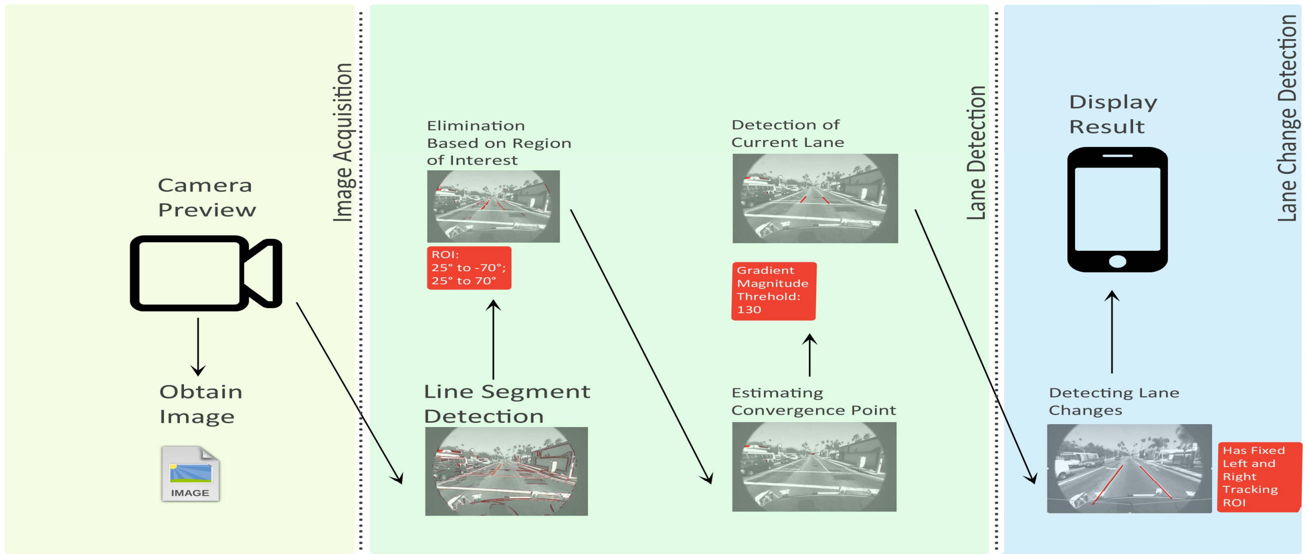


Figure 1: Overview of the deigned methodology with the proposed flow of each module resulting in the detection and tracking of the lanes.

A. Line Segment Detection

The first step of the algorithm is to detect all edges from the captured image, approximated by straight lines. For this purpose, we have used Line Segment Detector (LSD) [17]. The computational complexity of the line detector is proportional to the number of pixels in the image. On average, the detector gives one false result per image. When tested against Hough Lines, LSD gave similar results with less processing time. The result of this module is shown by Figure 2 using an image from Caltech lane dataset as demonstration of example.

Figure 2: LSD result: Red lines correspond to the detected straight lines

B. Elimination based on region of interest:

The second step of the process is to first divide the image into two halves vertically from the middle of the image and then for each side, a region is calculated where road lanes should reside. This region is known as the ROI. The ROI for the left portion of the image will constitute of all those straight lines whose angle ranges from -25° to -70°. Similarly, the ROI for the right portion of the image will consist of lines lying in the angle range of 25° to 70°. This range of angles is based on the typical orientation of the lane markings as observed from the view-point of the driver.

All lines lying outside the defined left and right ROIs are eliminated from the image frames. Before calculating the convergence point of the remaining lines, horizontal lines are removed from the image frame. This removal is based on the assumption that the road lane boundaries can never have horizontal orientation from the driver's viewing angle. The output of this module is shown in Figure 3.

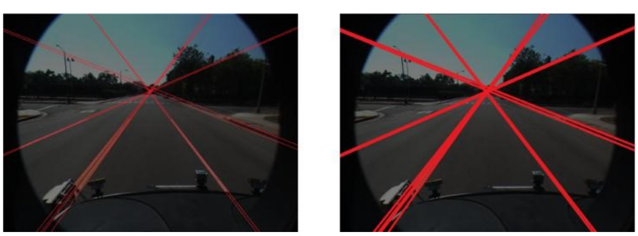
C. Estimating convergence point:

After the elimination of the lines based on their angle, remaining lines are used to estimate the position of the convergence point. The convergence point will be the point where the maximum number of lines will intersect, when reasonably extended. For this purpose, the remaining lines are extended across the whole image and assigned unit weights (or "one"). These weights are added where lines intersect. If the lines are only kept one pixel wide, then there are usually multiple intersections of maximum weight of two, hence an estimate of the convergence point cannot be found. A good estimate of the convergence point can be found, if the width of lines is increased. Hence, the lines are thickened to be seven

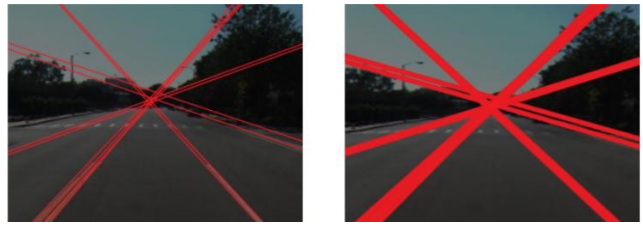
Figure 3: Resultant image after eliminating lines based on region-of-interest

pixels wide to greatly increase the probability of more than two lines intersecting at one point. The value of seven was chosen after testing the detection of correct convergence-point on 15 randomly chosen images from the Caltech Lanes Dataset. During this testing, a thicker line resulted in false location of the estimated convergence point and a thinner line resulted in multiple points of same maximum weight as seen in the Figure 4.

After calculating the convergence point, lines are thresholded based on their perpendicular distance from the convergence point. The perpendicular distance is the minimum distance from the convergence point and a line. The perpendicular distance can also be defined as the length of a line which connects the convergence point with a straight line; and makes and angle of 90° with the respective line. For our testing, we set the threshold value to 60, as this removes unnecessary lines for later stages. Figure 5 shows the lines as a result of this elimination.



(a) (b)



(c) (d)

Figure 4: Convergence point estimation with and without thickening of the lines shown in a and be respectively with a zoomed view in image c and d

(a) (b)

Figure 5: Red circle is the calculated convergence point for the sample image in a. Image b is obtained after eliminating lines based on distance from convergence point

D. Detection of current lane:

The last step of the detection algorithm deals with the selection of current lane of the car from the pool of lane markings. For this purpose, it is necessary to first remove all lines which are either detected due to a shadow boundary or other road markings. We did not use any advanced shadow detection and removal method as this would significantly affect the overall run-time, which should be kept minimal to allow real-time processing of images on mobile phone processor. In order to cater for this problem, lines are made five pixels wide and the gradient magnitude of the resulting lines is calculated. The line with the highest gradient magnitude is selected as the reference line. Gradient magnitude difference is calculated for all the lines in the ROI with the selected reference line. Lines which have gradient magnitude difference above a certain threshold are eliminated. For Caltech Lanes Dataset the threshold is 130. This threshold is selected after experimenting under all the lightning conditions which we encountered in the dataset.

After eliminating these unneeded lines, the resultant lines depict the current lane markings from the view-point of the driver. A reference line is drawn from the bottom center of the image to the convergence point. The next step is to extend all the lines from the left and right ROI to the convergence point. The line from the left ROI having minimum angle with the reference line is selected as the left current lane. Similarly, the line in the right ROI forming the minimum angle with the reference line is selected as the right current lane. The resultant image is shown in Figure 6.

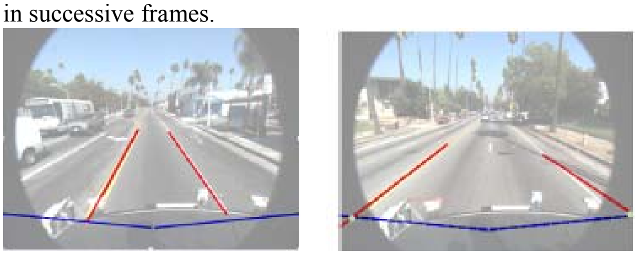
E. Detecting lane changes:

After detecting the current lane from the previous module, we studied and analyzed the behavior of the lanes to see if there is a possible lane change or not. For this purpose, we again divide the image into left and right ROI (the ROIs used for tracking are different from the ROIs used for detection). The line depicting the lower boundary of each ROI is tilted upwards from near the image boundary. The detected lanes are extended backwards and their point of intersection with the boundary of their ROI is calculated. Now, these points of intersection are analyzed with simple checks that suggest whether a lane is changing or not.

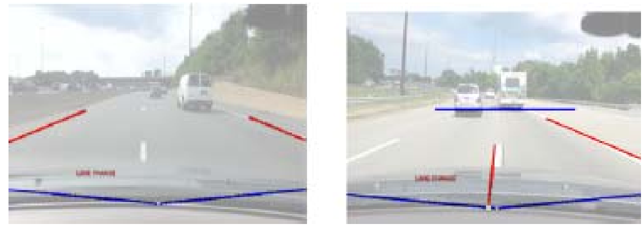
Figure 6: Red lines mark the current lane of the car

1. One scenario is that, while the lane is changing, both the right and left lanes will be present at the extreme sides of their respective ROI. This means that their point of intersection will either lie at the extreme left or extreme right side of the ROI. In some cases the point of intersection will lie outside the scope of the captured image as shown in Figure 7, a and b.
2. Another scenario is that while a lane is changing from left to right, the calculated intersection point of the left lane will lie at the extreme right side of the left ROI and at same time the point of intersection of the right lane should either be at the extreme right side of the right ROI or it lies outside both of the ROI. Similar behavior is depicted when the lane is changing from right to left as shown in Figure 7, c and d.

A lane departure behavior is only detected by the system if one of the aforementioned behaviors is repeated continuously in successive frames.



(a) (b)



(c) (d)

Figure 7: Different lane changing scenarios with blue lines at the bottom of the picture marks the boundary and green dot shows the point of intersection with ROI

IV. Experimental Setup and Result

A. Setup:

We tested our algorithm on the Caltech Lanes Dataset. This dataset constitutes of four videos, with a total of 1225 frames. It provides labeled data as a MATLAB file for each dataset. To compute the accuracy of the algorithm, we referred to the method of quantitative analysis used by Mohammed Aly [1]. As our algorithm approximates the current detected lane with straight line segments, the only difference from the above cited reference is that the splines in the labelled frames from Caltech Lanes Dataset are approximated as lines by connecting the endpoints of a spline by a straight line segment. The comparison of lanes with the ground truth is then done by finding the perpendicular distance of a detected lane from its endpoints with its corresponding labelled line. If this distance exceeds a given threshold of 30 pixels, that point is marked as a wrong detection and contributes to the overall score of wrong detections. To test the algorithm in real time environment, video is captured by mounting mobile on the center of the car dashboard. The frames of the real time video are processed sequentially and resultant image with detected lanes are displayed on the mobile. Our testing mobile device had a 2.26 GHz quad-core Qualcomm Snapdragon 800 processor.

B. Results:

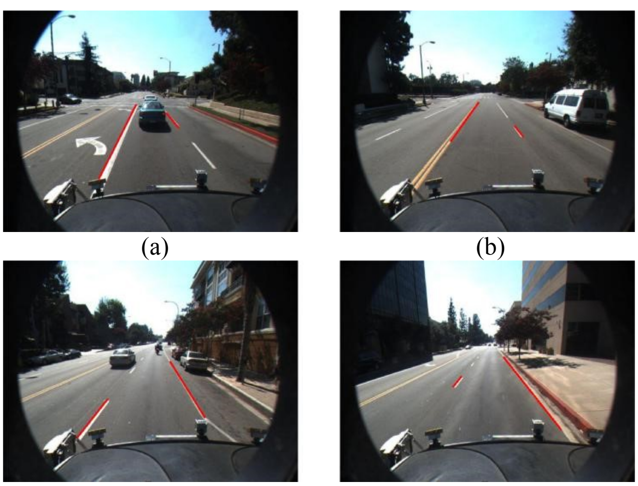
1. *Lane Detection:*

Caltech Lanes dataset constitutes of various complex road scenarios such as markings on the road, shadows cast due to nearby objects, curvatures and zebra crossings. Due to the use of a similar dataset, we have compared our results with Benligiray, et al [14] and Mohammed Aly [1]. As shown in Table 1, the average accuracy of our method is greater than the other two compared methods. By comparing the experimental setup for the algorithms under consideration, it is concluded that our proposed algorithm gives real time result on a lower resource platform (2.26 GHz quad-core Qualcomm Snapdragon 800 processor) unlike the other two methods one of which uses a 2.20 GHz CPU [14] and the other hasn't specified their platform [1].

	C1	C 2	W1	W2	Avg Accuracy
Proposed method	98.5	95.5	95.7	**97.8**	**96.9**
Benligiry et al [14]	**98.8**	**98.3**	91.4	95.3	95.9
Moham-med Aly [1]	97.2	96.1	**96.7**	95.1	96.3

TABLE I: Percentage accuracy of detection on the Caltech Lanes Dataset. Last column depicts the average accuracy. In the table C1 correspond to Cordova 1, C2 correspond to Cordova 2, W1 correspond to Washington 1 and W2 correspond to Washington 2.

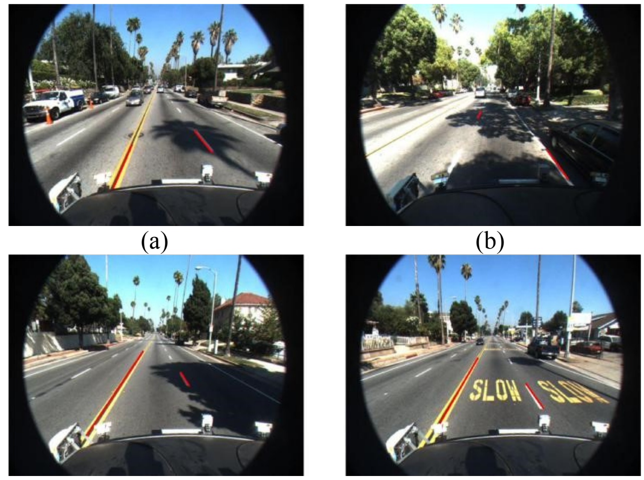
Cordova 1 images of the Caltech Lanes dataset has clear lane boundaries with some writing on the road and zebra crossings. Highest accuracy is obtained for this dataset as shown in Table 1. Figure 8 a and b shows the result of lane detection for Cordova 1 dataset. Cordova 2 images of Caltech Lanes dataset is more challenging as mostly there is no right lane boundary. Apart from lane boundaries, there are a lot of high intensity marks and lines on the road due to construction. Due to these challenges some false detections are obtained for this dataset. Figure 8 c and d shows the result of lane detection for Cordova 2 dataset.



(a) (b) (c) (d)

Figure 8: Lane detection for Cordova dataset 1 and 2

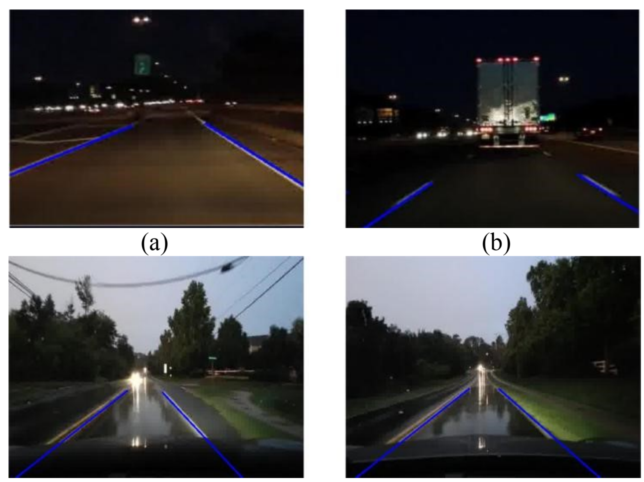
Washington 1 and Washington 2 dataset consist of shadows of trees and vehicles parked on the road. Writings on the road and nearby vehicles are also some of the challenges faced when lanes are detected for these dataset. Figure 9 shows the result of lane detection for Washington 1 and Washington 2 dataset.



(a) (b) (c) (d)

Figure 9: Lane detection for Washington dataset 1 and 2

Apart from Caltech Lanes Dataset, the proposed algorithm was tested with various road conditions such as nighttime and rain. Good accuracy was obtained when lanes and road boundaries were more distinct. Figure 10 a and b shows the results of lane detection during night and Figure 10 c and d are the outputs in rain condition.



(a) (b) (c) (d)

Figure 10: Lane detection at night and during rain

We also ran our algorithm with varying parameters; such as range of ROI selected in module 2 and gradient magnitude difference threshold selected in module 5; to study their effect on accuracy of lane detection. Results are mentioned in the above two tables along with their graph representation in Figure 11 and 12. As shown in the Table 2, results of lane detection for Cordova 2 and Washington 1 will improve drastically if we decrease the gradient magnitude difference threshold from 220 to 130. This happens due to the fact that in the above mentioned dataset there are a lot of sharp markings and shadows with high gradient magnitude. In order to detect the current lane of the car the gradient magnitude difference threshold should be small enough so that all other high gradient lines could be eliminated. Table 3 depicts that the selected range of ROI in module two gives the best result as when the range ROI is increased or decreased from the selected range, the accuracy of lane detection decreases. This happens because when the range of ROI is increased many objectionable lines with various orientation will also be selected in module two which will lead to a calculation of a wrong convergence point. If the range of ROI is decreased then many candidate lane marking with the correct orientation will not be selected in module two hence again leading to a wrong calculation of convergence point.

Gradient Magnitude	220	190	160	130
Cordova 1	97.2	97.2	98.0	98.5
Cordova 2	75.3	82.7	90.4	95.5
Washington 1	77.8	80.7	88.1	95.7
Washington 2	94.9	95.3	96.9	97.7

TABLE II: The effect of change in gradient magnitude difference on accuracy for Caltech dataset

Range of ROI	±40 TO 65	±30 TO 75	±20 to ±85
Cordova 1	92.0	98.5	80.0
Cordova 2	76.8	95.5	77.5
Washington 1	82.5	95.7	81.0
Washington 2	93.1	97.7	94.4

TABLE III: The effect of change in the size of ROI on accuracy for Caltech dataset

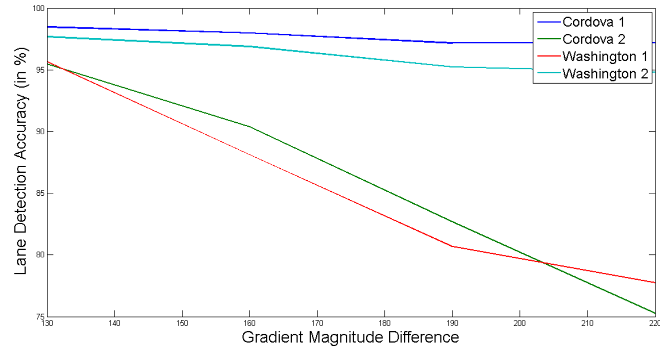


Figure 11: The effect of change in gradient magnitude difference on accuracy for Caltech dataset

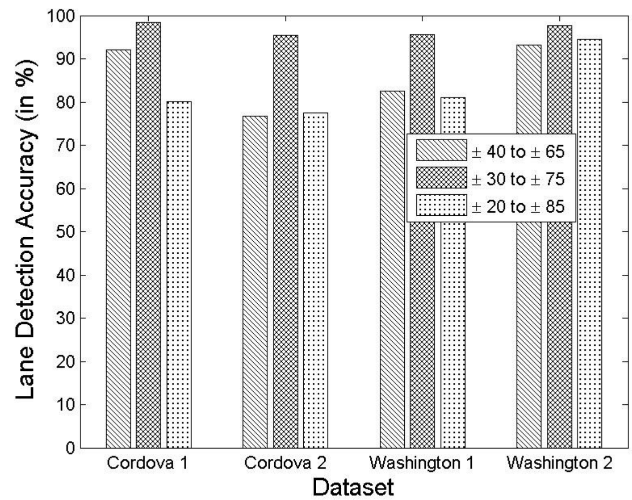


Figure 12: The effect of change in the size of ROI on accuracy for Caltech dataset

2. *Lane Tracking:*

Lane tracking algorithm gives a result of 100% accuracy with the Caltech Lanes Dataset. Table 5 shows the results when the proposed algorithm is evaluated on this dataset. Since there is only one lane changing scenario in Caltech Lanes Dataset, we compiled our own dataset that is recorded for roads under circumstances of varying light conditions and traffic complexity. Lane tracking heavily depends on correct lane detection. If the lane detection is inaccurate for consecutive frames, results of lane tracking also get affected. For example, in case of vehicle passing with a very contrasting color to the background (hence, having a high intensity gradient), wrong lanes get detected which can ultimately lead to wrong lane tracking as well. Similarly, at night-time, glare (due to lighting) sometimes leads to inaccurate lane detection which again negatively affects the lane tracking. Another problem that we have faced is determining the right value of the parameter that is going to determine where your point of intersection of lane lies with its respective region of interest because that is how we are going to declare whether a lane is changing or not. And that is done by varying the values of the parameters and test running it on the extensive dataset provided to us.

	Cordova 1	Cordova 2	Washington 1	Washington 2
Lane Tracking	100 %	100 %	100 %	100 %

TABLE IV: Results of Lane tracking on Caltech Dataset

V. CONCLUSION

The proposed method provides a reliable and robust solution to the problem of real time lane detection using a lower processor platform. When tested, it processes three frames per second and outputs correct lanes for the provided road imagery. It takes images of the road surface and then detects straight line segments from it. The detected line segments are then passed through a number of steps, with each step reducing the number of available lines based on thresholds. The end output consists of lines representing the location of the current lane. This is further passed onto the tracking module to alarm the driver of any lane change by keeping count of possible lane change scenarios for a number of past consecutive frames. The novelty of the proposed algorithm is that it gives higher accuracy as well as real time result for a lower resource platform. This is not possible though deep learning techniques as they work on down sized image in which the localized current lane of the car cannot be found as lanes in this downsized image will have very less visibility. Even if Lanes are detected through deep learning techniques, the algorithm used are processor and memory intensive and require expensive sensing modality other than mobile camera like LIDAR or GPS [10]. The average lane detection accuracy of the algorithm for Caltech Lanes Dataset is 96.87%.

REFERENCES

[1] A. Mohamed, "Real Time Detection Of Lane Markers In Urban Streets", IEEE Intelligent Vehicles Symposium, 2008.

[2] C. Kreucher, S. Lakshmanan, and K. Kluge, "A Driver Warning System Based on the LOIS Lane Detection Algorithm", IEEE International Conference on Intelligent Vehicles, 1998.

[3] Y. Wanga, E. Khwang Teoha,and D. Shenb , "Lane detection and tracking using B-Snake", Image and Vision Computing, Vol. 22, 2004.

[4] A.A.M. Assidiq, O.O. Khalifa, M.R. Islam, and S. Khan, "Real Time Lane Detection for Autonomous Vehicles", International Conference on Chemistry and Chemical Engineering (ICCCE), May 2008.

[5] L.S. Sha, "The Research of the Lane Detection Algorithm Base on Vision Sensor", Research Journal of Applied Sciences, Engineering and Technology 6(4): 642-646, 2013.

[6] O.O. Khalifa, I.M Khan, A.A.M. Assidiq, A.H Abdulla and S. Khan, "A Hyperbola-Pair Based Lane Detection System for Vehicle Guidance", Proceedings of the World Congress on Engineering and Computer Science Vol I, October 2010.

[7] M. Bertozzi and A. Broggi, "GOLD: A Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection", IEEE Transactions On Image Processing, Vol. 7, no. 1, January 1998.

[8] C. Kreucher and S. Lakshmanan, "LANA: A Lane Extraction Algorithm that Uses Frequency Domain Features", IEEE Transactions on Robotics and Automation Vol. 15, no. 2, April 1999.

[9] R. Gopalan, T. Hong, M. Shneier and R. Chellappa, "A Learning Approach Towards Detection And Tracking Of Lane Marking", IEEE Transactions On Intelligent Transportation Systems, Vol. 13, no. 3, 2012.

[10] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue, F. Mujica, A. Coates, and A. Ng, "An Empirical Evaluation of Deep Learning on Highway Driving", technical report arXiv preprint arXiv:1504.01716v3 [cs.RO], 2015.

[11] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus,and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks", technical report arXiv preprint arXiv:1312.6229, 2014.

[12] H. Zhang, D. Hou, and Z. Zhou, "A Novel Lane Detection Algorithm Based on Support Vector Machine", Progress In Electromagnetics Research Symposium, August 2005.

[13] F. Chao, X. Jing-bo and D. Shuai, "Lane Detection Based on Machine Learning Algorithm", TELKOMNIKA Indonesian Journal of Electrical Engineering, Vol. 12, no. 2, pp. 1403 - 1409, February 2014.

[14] B. Benligiray, C. Topal, C. Akinlar, "Video Based Lane Detection Using a Fast Vanishing Point Estimation Method", IEEE International Symposium on Multimedia, 2012.

[15] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting", Journal of Computer and System Sciences, Vol. 55, no. 1, pp. 119–139, August 1997.

[16] A.B. Hillel, R. Lerner, D. Levi and G. Raz, "Recent progress in road and lane detection: a survey", Machine Vision and Applications, Vol 25, no. 3, pp 727-745, April 2014.

[17] R.G.V. Gioi, J. Jakubowicz, J.M. Morel, G. Randall, "LSD: a Line Segment Detector", Image Processing On Line, March 2012.

[18] X. Sun, P. Wu and S.C.H. Hui, "Face Detection using Deep Learning: An Improved Faster RCNN Approach", technical report arXiv preprint arXiv:1701.08289v1 [cs.CV], 2017.

[19] Y. Xiang, W. Choi, Y. Lin and S. Savarese, "Subcategory-aware Convolutional Neural Networks for Object Proposals and Detection", technical report arXiv preprin arXiv:1604.04693v3 [cs.CV], 2017.

[20] J. Redmon, A. Farhadi, "YOLO9000: Better, Faster, Stronger", Conference on Computer Vision and Pattern Recognition (CVPR), 2017.