XXXX, et. al. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 10, Issue 7, (Series-VII) July 2020, pp. 36-45

RESEARCH ARTICLE OPEN ACCESS

Path Detection Based on Bird Eye View with Feature Extraction

Gummadi Sai Dheeraj[1], K.B.V. Lakshmi[2]
SRM institute of Science and Technology, Student of Electronics and communication engineering, Guntur, Andhra Pradesh, India
SRM institute of Science and Technology, Student of Electronics and communication engineering, Guntur, Andhra Pradesh, India

ABSTRACT:
Path detection is one of the challenging problems. It has attracted the attention of computer vision for several years.Essentially path detection problem that has become a real challenge for computer vision.Although many Deep Learning techniques are used for path detection they are mainly used for classification rather than feature design.But modern Deep Learning methods can identify the features and succeeded in feature detection.However, these methods are not been fully implemented in efficiency and accuracy of lane detection.Inthis Paper we propose a new method to solve it.We introduce a new method of preprocessing and Bird eye view of the path.
The main goal is to use the HSL(hue,saturation and lightness) colour transformation to extract the white features and odd preliminary edge feature detection in the preprocessing stage and select bird eye view region based on preprocessed image.The new preprocessing method is used to detect the lane.Many papers have used certain standard datasets and not released their source code publicly.We used public dataset which everyone can play around and people can see the code in this github link https://github.com/GSaiDheeraj/Path-Detection

Date of Submission: 18-07-2020 Date of Acceptance: 02-08-2020

I. INTRODUCTION

In Recent times autonomous vehicles have received much attention.Fully autonomous cars are the main focus of computer vision nowadays, both at an academic and industrial level.The goal in each case is to attain a full understanding of the environment around the car. Camera based lane detection is one of the important step towards the environmental perception because it allows the car to properly place itself within the road lanes

At the core Autonomous vehicles perform Three tasks Perception,Decision,Action.Perception is nothing but vision and understanding the situation,Decision is nothing but taking decision whether to move left or right or move straight and finally Action is implementing the decisions.

With the rapid development of society, automobiles have became one in all the transportation tools for people to travel.In the narrow road there are more and more vehicles of all kinds[1].As more and more vehicles are driving on the road,the number of victims of car accidents is increasing each year.How to drive safely under the condition of diverse vehicles and narrow roads has become the main focus of attention.Advanced driver assistance systems(ADAS) which include lane departure warning[3],Lane keeping assist and Adaptive Cruise Control can help people analyze the present driving environment and supply appropriate feedback for the safe driving or alert the driving force in dangerous situations.This kind of auxiliary driving system is predicted to become more and more perfect[5].

After investigation,in a complex traffic environment where vehicles are numerous and speed is simply too fast, the probability of accidents is far greater than usual[1].In such a high traffic situation,road colour extraction and texture detection also road boundary detection and lane making are main perceptual clues of human driving.
Lane Detection plays a vital role in the LDW system.Path detection is split into two major components
1)Edge Detection
2)Line detection

There are some false edge detections even after applying filters.Wang proposed a canny edge detection algorithm for better accuracy in detecting edges.This algorithm provides better understanding of the complicated environment.In 2014 the algorithm was updated that it can even deal with the noisy environments[13].Sobel and Canny are the two foremost used algorithms for detecting edges.

Line Detection is one of the most important in lane detection,We have two methods for detecting lines, Niu used a modified Hough Transform to

extract the segments of the lane and used density based spatial application noise clustering algorithm for clustering[14].Another one is Mammeri used progressive probabilistic Hough Transform combined with maximum stable extreme area to detect lane lines and used Kalman filter to achieve continuous tracking of lane lines[15].These algorithms may be good but they can not detect the curves and can't work in the different light conditions and while occlusions.Most of the methods are usually tested on USA data sets only where it can't be generalize each country needs its own method to detect.

In this paper we propose a Path detection method that is suitable for almost all kinds of traffic situations.First we preprocess the image so that it'll be easy and clear for the computer to understand and we apply warp perspective on the region we are interested in the preprocessed images.First in the preprocessing stage, we undistort the image and then we apply warp perspective and later we apply canny filter and the colour filter to the undistorted image.Compared to the other preprocessing methods only do fundamental operations such as Thresh,Blurring, gradient and so on.In this paper,experiments show that the proposed method is significantly better than the prevailing preprocessing method in lane detection

II. OVERVIEW OF THE PROPOSED SYSTEM

This paper presents an advanced Path detection technology to improve the accuracy of real time lane detection[16].The lane detection module is splitted into two steps :

1)Image Processing
2)Matching of line lane detection

Figure 1 shows the overall process of the proposed system in a simple manner so that understanding and implementing paper will become easy for others.The First step is to read the frames in the video stream.The second step is to enter into the image processing.The difference between other papers and this paper in preprocessing is we undistort the image and apply warp perspective before detecting edges and applying colour filter

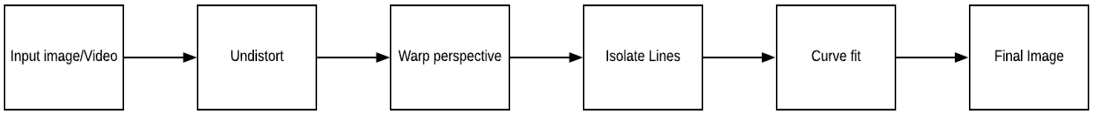


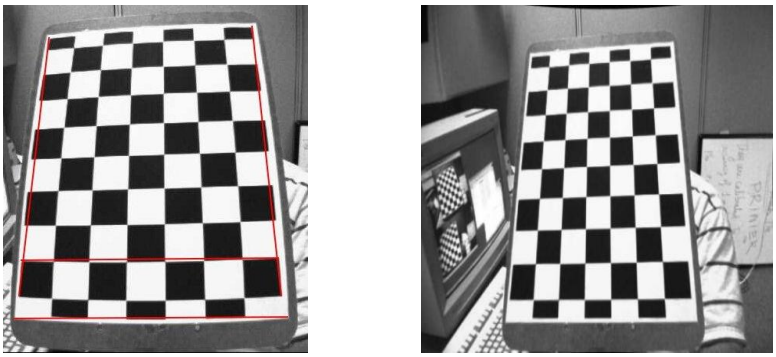
Figure 1

In the next step we fit the curve through obtained lines and Finally we merge the preprocessed image and original image to get the final image

III. PROPOSED METHODS

In this paper based on previous papers, we firstly undistort the image as camera distorts the image to fit large amount data into small frame,So we first undistort it to get clear image for further preprocessing.Then we apply warp perspective to view the road environment from top.Now we extract the edge features through canny filter and colour features through histogram.As high speed section in the traffic is accident prone area, the high speed road mostly straight line lane and sometimes it may be curved lane also[18].Inorder to induce high recognition rate,We successively stick with colour detection and edge detection to the lane.This paper combines colour feature extraction and edge feature extraction and this experiment proves that the accuracy of lane detection is extremely improved

Our contribution in this paper is to do lot of work in the preprocessing stage.We proposed to perform colour transform of HSL, So we can extract the different colours of lanes like as we said before this paper will be generalised for almost all road conditions,for that we perform colour transform of HSL to extract both yellow and white lane lines and then we perform preprocessing steps conventionally in a sequence.



(a) *(b)*

Figure 2
(a)Distorted image capture by the camera with curved lines before calibration lines (b) undistorted image with straight lines after calibration

This paper doesn't include any hough transform or Kalman filter which most of the papers used while detecting the paths, instead we use histograms to detect the lane lines and fit second order polynomial function to the lane lines we obtained

IV. EXPERIMENTS

4.1.Preprocessing

The first we undistort the images,the raw camera contains a little bit of distortion so we are going to clear that out.Cameras uses lenses that distort the image so it can fit more of the surroundings into smaller frame

In the image (a) we can see a typical type of distortion called Barrel distortion, where it looks like those straight lines bowed on the outside of the image.If we take this image and do transformation warping to get birds eye view the lines should be straight but there are bowed towards outside So we have to perform some distortion corrections

Opencv has number of functions to do this,We use findChessboardCorners() function which finds the corners of the image and pass it into calibrateCamera() function which takes image points found in the image and it's going to return a set of distortion coefficients which transforms the image

There is a function called undistort() which takes the original image and the two matrices that we found previously and returns the undistorted image.

If A is the camera matrix

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

fx=(double)dsize.width/src.cols
fy=(double)dsize.height/src.rows
fx,fy = inverse mapping of 2D images
gx,gy = forward mapping
Distortion coefficients are represented by (k1,k2,p1,p2,[k3,[k4,k5,k6]])
Then the distortion is corrected by

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Still if there are any Tangential distortions they are corrected by

$$x_{corrected} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$

$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

Figure 3
Undistortion under different conditions and different data

4.2.Warping Images

Warping images means seeing images from top which is "Bird Eye View".We need this to fit the curves and finding lane lines,it'll be easier to find those lane lines if we can look down on the road instead of looking out on it.To do this we need four source points from the original image.we are going take those points and warp them into straight lines.

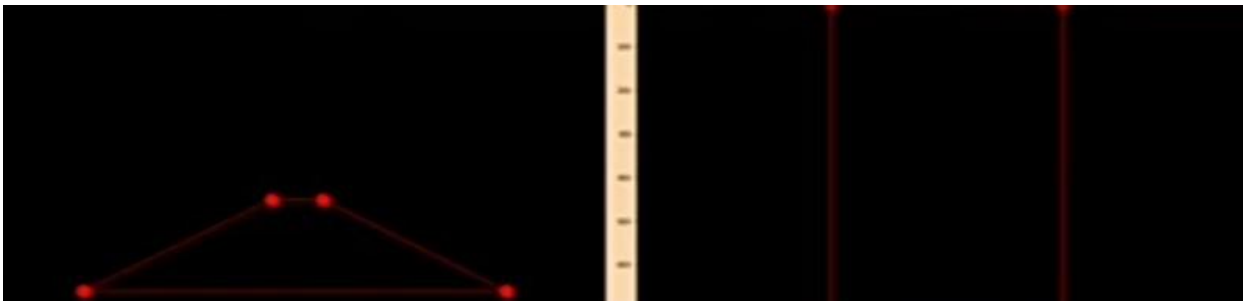


Figure 4
Four warping Source points that looks like going to infinity but when we see it from above those are straight lines

To do this we will take the help of this getPerspective() function which returns a matrix
$\mathrm{dst}(i) = (x'_i, y'_i), \mathrm{src}(i) = (x_i, y_i), i = 0, 1, 2, 3$
Then we take output of that matrix and apply the warpPerspective(img,M,img_size) function and it returns the image like below

Figure 5
warp perspective under different conditions,curves and different data

When you see from the top of the image you notice that it gets little more pixelated,obviously we can't create resolution that didn't exist in the original image but there should be enough fidelity there to isolate those pixels from the surrounding image

4.3.Finding Lane Lines

There are two approaches that we are going to take one's colour selection and another one's edge detection.So for colour selection first we are going to look at what an image is inside the computer.Taking a look at first ten pixels of the image each pixel will have set of three values that is RGB(Red,Green,Blue) values,these values vary from 0-255 in most images just because there are 8-bit images $2^8$is 256 so it ranges from 0-255.If all pixel values are 255 it shows white image and if all pixel values are 0 we will see black image.If we have a list of 255,0,0 then we will see red colour image.

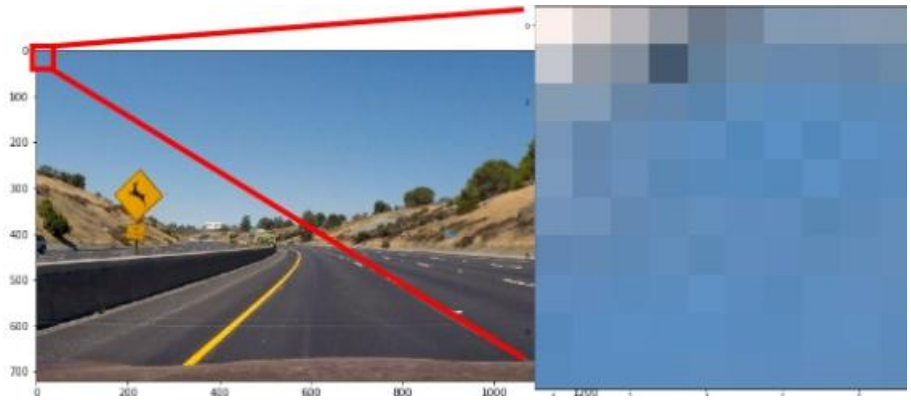


Figure 6

Now,there are many other colour spaces than Red,Green,Blue.HSL(hue,saturation,light) help us to isolate these colour spaces,so it just takes RGB colours and then turns them into other colour representations of that same image.

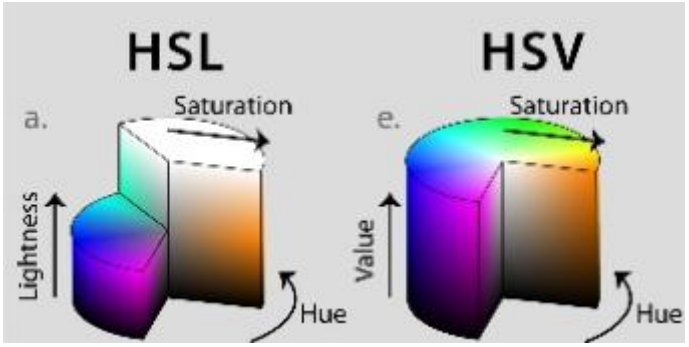


Figure 7

Opencv makes it very easy to convert colours between these,we just use cvtColor (RGB,cv2.COLOR_RGB2HLS) function and pass in RGB colour or any other colour as long as we have correct conversion coefficient which RGB2HLS.So it outputs HLS(hue,light,saturation) image which looks a little different but its really the same image,it's just different representation of that same image

From here we can isolate each layer of that image,so there's Red layer,Green layer,Blue layer

Red channel

Figure 8

blue channel

We can see that the yellow line can be just visible in Red and Green Channels but not in Blue channel Because the combination of red and blue gives yellow.

Now getting into Edge detection,Opencv hasa couple of functions that will detect edges one is Sobel and another one is Canny.All this Soble operator is doing is taking the derivative of one of colour channels,So if we pass in the red channel if we have red meeting white it's going to detecta line where those two meet,it's just calculate the derivative across that line so the value of that derivative is what this plot is

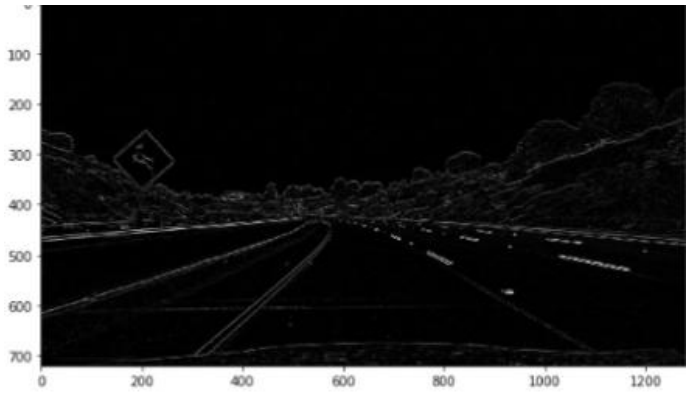


Figure 9

It calculates derivatives in x-direction , y-direction and we get Sobel magnitude as the sum of those two squares.This would be pretty useful on occasions where our colour detection system might not work so well, so it's a good substitute.

We have another one called canny edge detection which just uses the same soble system but applies a couple nice little tricks throughs out pixels that aren't part of the edge and then also thresholds the image to try to connect the edges that weren't connected before so we are going to use both of these to produce our final guess where the lane lines are.We look at this BInary Thresholding,where ever the red channel saying there's white pixel we give high value for the red channel.So if we just apply a threshold to this red channel image and write that thresholding to new image we are going to get an image where everything is black except the lane lines.So here we use the thresholding between 200 - 255 which indicates wider on that gray scale image but we can adjust these thresholds to try to capture more or less of the lane lines and then once we do that to all the colour channels and then add in the edge detection as just a series like a linear addition so if there is any white pixel on any of these binary thresholding images we just add it to the final image.

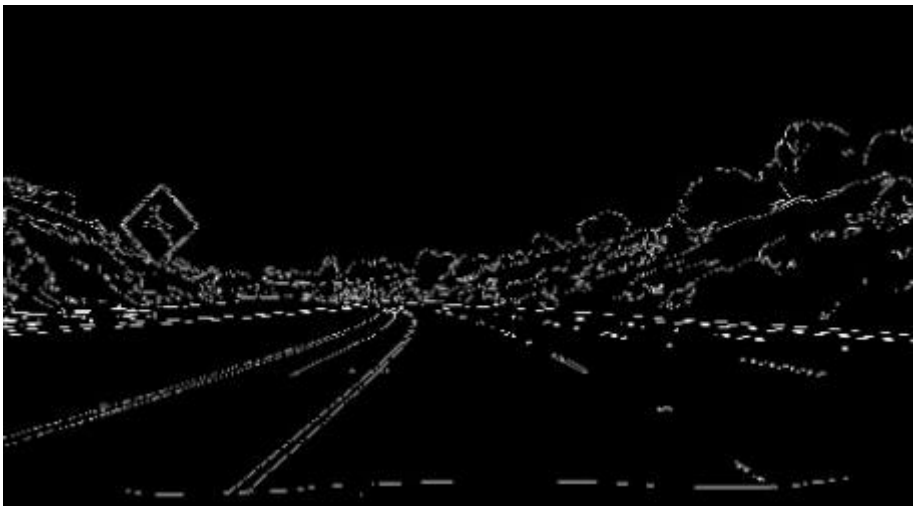


Figure 10

4.4.Curve Fitting:

Now we have the binary image,we can just run a some algorithm to find the left and right lane lines and then we can apply curve fit to those lines.So there are two methods to fit the curve,first method can be applied on any image and method two need previous fit but runs quite bit faster.We try to use method two.

So for the first method we take a histogram of the bottom half of the image and wherever that histogram peaks that's where we think that the line starts, so the y value here is just sum of the pixels at that x location and the two maximums are where those lines start.

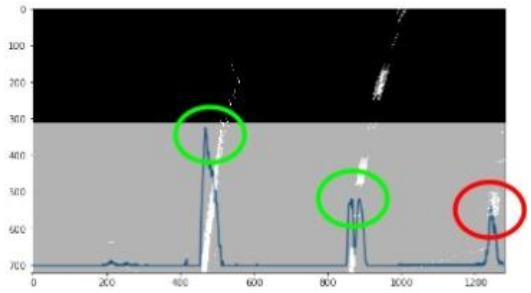


Figure 11
HIstogram of the image

Red circle in the Figure is a good indicator of how close it can be between a false value and a real value,so here the lane thats to the right of our image is detecting that line two so throwing out some of these extraneous points would be probably pretty useful for this algorithm but anyway we have our left start and right start we are going to draw a box around those starting points and any of those starting points and any of those white pixels from the binary image that fall in that box we just put them in a big list of x and y values and then next one we add a box on top of that and do the same thing as move up the line.We will move the boxes based on the average of the previous line,so if average start moving left we will know that next box needs to move left that's how these boxes start falling in that line in Figure

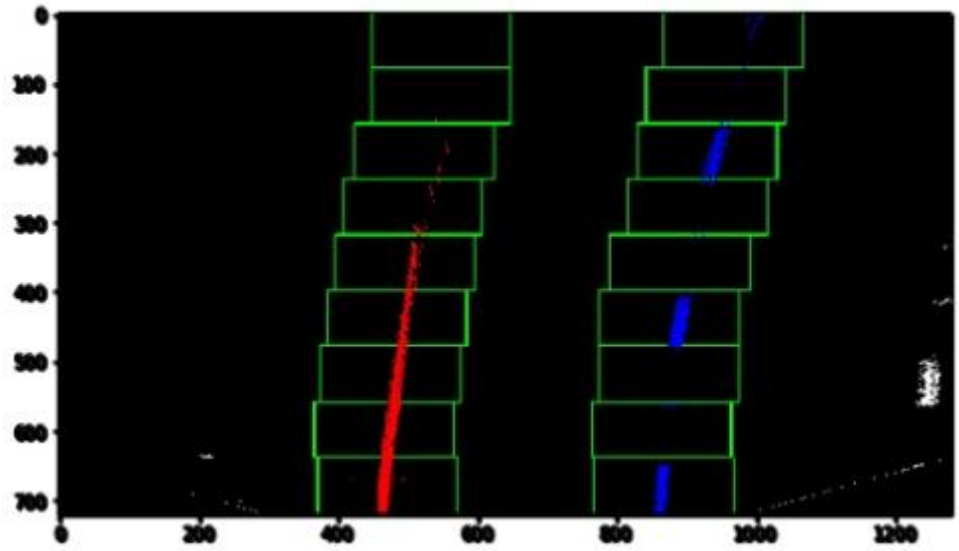


Figure 12

So at the end of this we have a big list of all the Y values and all the X values of the pixels that were in that line and we run numpy second order polyfit function to those pixels

leftfit = np.polyfit(lefty,leftx,2)
rightfit = np.polyfit(righty,rightx,2)

2 indicates second order polynomial $Ax^2 + Bx + C$
fit = [A,B,c]

This gives us the curve fit in pixel space,we can measure the distance by pixel to meter conversion by knowing that in United States the lanes are may be three meters wide.It may different for different countries.We can do that conversion and then for the Y direction you know that there each of the dashed line is ten feet long on the right and there's a set gap it gives us the good conversion so we can go from pixel measurements to meter measurements

Looking into method two it just relies on first method,we take that previous fit and move it to the left and right by a hundred pixels and that creates this green boundary and any pixels that lie in that boundary we throw those into a list and then run that same polyfit system and that produces these yellow lines which they seem look pretty good

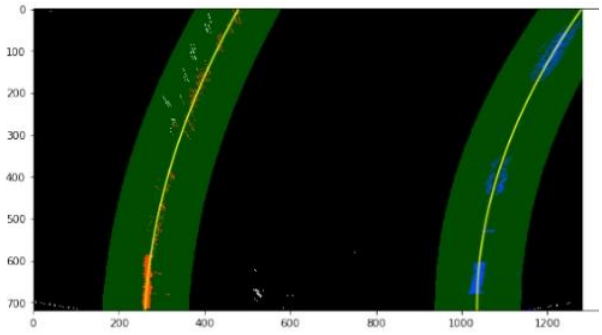


Figure 13

4.5.Final Image:

Final Step is creating a nice image,we measure some radius of curvature.This helps the car to guess how tight the turn needs to be and how much the steering wheel needs to put in to turn at a curve.

Radius of curvature = $[1 + (dy/dx)^2]^{3/2}/|d^2y/dx^2|$

We just use that fit those fits we found previously and calculate radius of curvature also useful is the distance to center of the lane,so if the car starts veering to left if we keep track of that we can have simple controller so that car will add a little bit of right steering wheel angle to bring it back to the center of the lane and it's fairly straightforward to get this value if we know where the X location of the right lane and left lane are and you assume that the cameras on the centerline of the car then the centerline of car is at the center of the image,so that's how we got those values.Obviously if it's not mounted on the center then there just need to be some offset

So getting that nice yellow picture we just take the two curve fits we found and plot those on the image using polylines() function and then fill in the space between those lines with the yellow fill with the function fillpoly().But this is in the warped space so this is from the birds eye view which we can't really overlay on that final image it'll kind of look a little goofy but it's pretty easy to go from that warped image to the original image if you just swap those source and destination points,so we swap those apply the function undistort() function and this will output that nice overlay Figure

Figure 14
Path detection under different light conditions and on different data

We just use addWeighted() function which adds a transparently weighted layer to that image,So finally that's how we got that nice picture and then to run this on a series of images we just used a couple of tools for that but basically applied this pipeline to all the images in a video and saves it.

V. RESULT AND DISCUSSION

Most of the research scholars who worked on Lane detection with opencv directly select the region of interest and apply Hough transform or Kalman Filter which they can only work for detecting straight lanes,they fail when it comes to turnings.

In this paper a new method is introduced that without using the hough transform we detect the lanes and curves more efficiently than previously proposed methods.

To improve the accuracy of path detection, we used the correct detection rate to evaluate the performance of our proposed method on the public data set we used.In order to see the excellence of the proposed method we have taken the test set with 100,500,1000,1500

Images and tested the accuracy of the method in the figure .We can see the proposed method was giving the highest correct detection rate compared to other proposed methods.

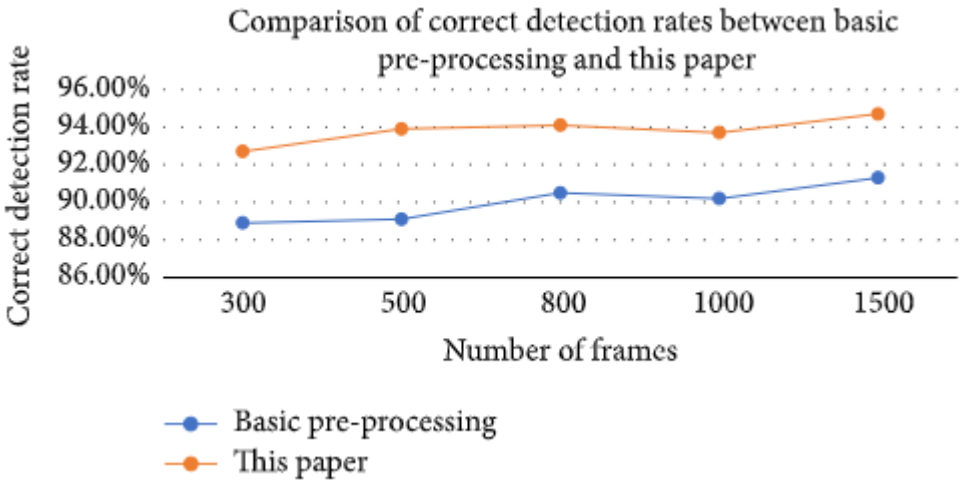


Figure 15

VI. CONCLUSION:

In this paper, we proposed a new path detection preprocessing and Bird eye view methods to design a path detecting system.The main idea is to apply warp perspective before applying the edge feature extraction and colour feature extraction to improve lane detection accuracy. We also introduced Histogram based lane detection to find the left and right lanes.Compared with applying edge detection before applying warp perspective on original image it reduced the nonlane parameters and improved the accuracy of the path detection

REFERENCES:

[1]. D. Pomerleau, "RALPH: rapidly adapting lateral position handler," in Proceedings of the Intelligent Vehicles '95. Symposium, pp. 506–511, Detroit, MI, USA, 2003.

[2]. J. Navarro, J. Deniel, E. Yousf, C. Jallais, M. Bueno, and A. Fort, "Influence of lane departure warnings onset and reliability on car drivers' behaviors," Applied Ergonomics, vol. 59, pp. 123–131, 2017.

[3]. P. N. Bhujbal and S. P. Narote, "Lane departure warning system based on Hough transform and Euclidean distance," in Proceedings of the 3rd International Conference on Image Information Processing, ICIIP 2015, pp. 370–373, India, December 2015.

[4]. V. Gaikwad and S. Lokhande, "Lane Departure Identification for Advanced Driver Assistance," IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 2, pp. 910–918, 2015.

[5]. H. Zhu, K.-V. Yuen, L. Mihaylova, and H. Leung, "Overview of Environment Perception for Intelligent Vehicles," IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 10, pp. 2584–2601, 2017.

[6]. F. Yuan, Z. Fang, S. Wu, Y. Yang, and Y. Fang, "Real-time image smoke detection using staircase searching-based dual threshold AdaBoost and dynamic analysis," IET Image Processing, vol. 9, no. 10, pp. 849–856, 2015.

[7]. P.-C.Wu, C.-Y. Chang, and C. H. Lin, "Lane-mark extraction for automobiles under complex conditions," Pattern Recognition, vol. 47, no. 8, pp. 2756–2767, 2014.

[8]. M.-C. Chuang, J.-N. Hwang, and K. Williams, "A feature learning and object recognition framework for underwater fish images," IEEE Transactions on Image Processing, vol. 25, no. 4, pp. 1862–1872, 2016.

[9]. Y. Saito, M. Itoh, and T. Inagaki, "Driver Assistance System with a Dual Control Scheme: Effectiveness of Identifying Driver Drowsiness and Preventing Lane Departure Accidents," IEEE Transactions on Human-Machine Systems, vol. 46, no. 5, pp. 660–671, 2016.

[10]. Q. Lin, Y. Han, and H. Hahn, "Real-Time Lane Departure Detection Based on Extended Edge-Linking Algorithm," in Proceedings of the 2010 Second International Conference on Computer Research and Development, pp. 725–730, Kuala Lumpur, Malaysia, May 2010

[11]. C. Mu and X. Ma, "Lane detection based on object segmentation and piecewise fitting," TELKOMNIKA Indonesian Journal of Electrical Engineering, vol. 12, no. 5, pp. 3491–3500, 2014.

[12]. J.-G. Wang, C.-J. Lin, and S.-M. Chen, "Applying fuzzy method to vision-based lane detection and departure warning system," Expert Systems with Applications, vol. 37, no. 1, pp. 113–126, 2010.
[13]. S. Srivastava, M. Lumb, and R. Singal, "Improved lane detection using hybrid median filter and modified hough transform," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 1, pp. 30–37, 2014.
[14]. J. Piao and H. Shin, "Robust hypothesis generation method using binary blob analysis for multi-lane detection," IET Image Processing, vol. 11, no. 12, pp. 1210–1218, 2017.
[15]. J. Niu, J. Lu, M. Xu, P. Lv, and X. Zhao, "Robust Lane Detection using Two-stage Feature Extraction with Curve Fitting," Pattern Recognition, vol. 59, pp. 225–233, 2015.
[16]. J. Son, H. Yoo, S. Kim, and K. Sohn, "Real-time illumination invariant lane detection for lane departure warning system," Expert Systems with Applications, vol. 42, no. 4, pp. 1816–1824, 2015.
[17]. A. Mammeri, A. Boukerche, and Z. Tang, "A real-time lane marking localization, tracking and communication system," Computer Communications, vol. 73, pp. 132–143, 2016.
[18]. C. J. Chen, B. Wu, W. H. Lin, C. C. Kao, and Y. H. Chen, "Mobile lane departure warning system in," in Proceedings of the 2009 IEEE 13th International Symposium on Consumer Electronics, pp. 1–5, 2009.
[19]. J. W. Lee, C. D. Kee, and U. K. Yi, "A new approach for lane departure identification," in Proceedings of the IEEE IV2003 Intelligent Vehicles Symposium, pp. 100–105, 2003.
[20]. J. W. Lee and U. K. Yi, "A lane-departure identification based on LBPE, Hough transform, and linear regression," Computer Vision and Image Understanding, vol. 99, no. 3, pp. 359–383, 2005.

XXXXXX, et. al. "Path Detection Based on Bird Eye View with Feature Extraction." *International Journal of Engineering Research and Applications (IJERA)*, vol.10 (07), 2020, pp 36-45.