
VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

## School of Computer Science and Engineering

### J Component report

**Programme : B.Tech (CSE)**

**Course Title : Image Processing**

**Course Code : CSE4019**

**Slot : E1**

# Title: Path Lane Detection of Self Driving Cars

**Team Members: Megha Singh 19BCE1203**

**Chinmayee Das 19BCE1204**

**Parul Mudaliar 19BCE1531**

**Faculty:** Geetha S **Sign:**

**Date:**

# DECLARATION

I hereby declare that the report titled “PATH LANE DETECTION OF SELF DRIVING CARS” submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of prof. Geetha S **,**School of Computer Science and Engineering, Vellore Institute of Technology, Chennai.

Signature of the Candidate

## CERTIFICATE

Certified that this project report entitled "**PATH LANE DETECTION OF SELF DRIVING CARS"** is a bonafide work of Megha Singh 19BCE1203 Chinmayee Das 19BCE1204 Parul Mudaliar 19BCE1531 and they carried out the Project work under my supervision and guidance for CSE4019- Image Processing.

**Dr.Geetha S**

SCOPE, VIT Chennai

## ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr.Geetha S,** Assistant Professor(Senior), School of Computer Science and Engineering, for her consistent encouragement and valuable guidance offered to using apleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Ganesan R,** Dean of School of Electronics Engineering, VIT Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our Head of the Department **Dr. Nithyanandam. P** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institute

## ABSTRACT

In our project, “Path Lane detection of self-driving vehicles”, our primary focus lies in designing an algorithm for allowing lane detection in Indian roads. Initially, we define the region of interest (ROI) and crop the image based on the same to focus on the necessary information. We have color thresholding to detect the color of the road and lane. Canny edge detection algorithm and Hough transformation can be used to detect lanes on the road. Additionally, focusing on lane detection for Indian roads, we plan to implement image enhancement for gathering data in darker conditions using histogram equalization method on the hue component of HSV color space.

# CONTENTS


Declaration i
Certificate ii
Acknowledgement iii
Abstract iv

1 Introduction 1
1.1 Objective and goal of the project .....1
1.2 Problem Statement.....2
1.3 Motivation .....3
1.4 Challenges .....5

2 Literature Survey 6

3 Requirements Specification 7
3.1 Hardware Requirements .....8
3.2 Software Requirements .....9

4 System Design 10

5 Implementation of System

6 Results & Discussion

7 Conclusion and Future Work

8 References

Appendix <Sample code, snapshot etc.>

# 1. Introduction

## 1.1 Objective and goal of the project

With this project we aim to provide

- a feasible and economical solution of the path lane detection technology that caters to all the needs as per the Indian roads.
- To present an all-round solution that can make autonomous cars fit for our roads.

## 1.2 Problem Statement

Lane detection in Indian roads, constitute a major challenge due to the presence of obstacles and uneven road conditions; one can encounter during travel. Even main roads frequently have only two lanes, with poor visibility and inadequate warning markers. Moreover, due to the variance in structure and uniformity of the path, lane detection algorithms require additional assistance to ensure accuracy and safety to make self-driving cars, a reality in India.

So, we hope to present a solution of a better lane detection technique that can actually make self-driving cars somewhat fit for roads in India. An algorithm that can detect thelanes when they aren't finely visible at night and in case of uneven and blurred lanes

## 1.3 Motivation

A self-driving vehicle is a car or truck that can sense its environment and control its movements without human intervention. To start off, self-driving cars can be a source of life-saving technology. Roughly 1.3 million people die in car accidents around the

world each year. 90% of car accidents are caused by human error. Self-driving cars will eliminate the key source of human error as they will not make any errors of judgment. According to research, self-driving cars could save 29,447 lives annually by reducing traffic accidents.

Self-driving cars are aimed at increasing safety of transportation such as reducing the instances of impaired driving, reducing the number of crashes and fatalities caused due to human error. A major challenge of self-driving vehicles includes lane detection for secure and safe travel.

So, we hope with this project we hope to present a better lane detection technique that can actually make self-driving cars fit for roads in India.

### 1.4 Challenges

- ❏ India's infrastructure in its present state is not suitable for self-driving cars. Theroads are overflowing with traffic especially in the metros.
- ❏ There is inconsistent signage and markings on roads which make deciphering the signs hard for sensors.
- ❏ The occasional presence of animals on the roads.
- ❏ Potholes and lack of proper roads.
- ❏ Delay in image processing and lack of real time communication

# 2. Literature Survey

### 2.1 Lane detection using image processing

In the paper, “Lane detection techniques using Image Processing”, the main problem addressed is the absence of markings on the road, which may lead to inaccuracy in the lane detection algorithm. The approaches described in the paper include Thresholding, Warping and Region of Interest (ROI). Further processing involves Pixel Summation (Histogram). The approach discussed follows thresholding, which is used to detect the color of the road, converting the image into a binary image. This is followed by warping which is used to change the perspective of the image to get a top view of the path. The RoI is identified and the image is cropped. Finally, to find the curve of the path, summation of pixels is done.

https://www.itm-conferences.org/articles/itmconf/pdf/2021/05/itmconf_icacc2021_03011.pdf

### 2.2 A Much Advanced and Efficient Lane Detection Algorithm for Intelligent Highway Safety

In the paper, “A Much Advanced and Efficient Lane Detection Algorithm for Intelligent Highway Safety”, by Prof. Sachin Sharma and Dr. D. J. Shah, Region of Interest based lane detection algorithm is described. The problems identified include the presence of multiple lanes on a road, faded lanes which makes identification a difficult task. Multiple lanes would also lead to false warnings in a Lane Departure warning system in Advanced driver assistance systems. The paper further describes the method of RoI (Region of Interest) to solve the multiple lane problem by segmenting into left and right RoIs. Further proceeding use of Hough transform is seen, which ensures identification of lane markings for the region of interest.

https://airccj.org/CSCP/vol3/csit3106.pdf

### 2.3 Recent progress in road and lane detection: A survey

In the paper, "Recent progress in road and lane detection: A survey", by Aharon Bar Hillel, Ronen Lerner, Dan Levi, Guy Raz, the paper describes the development achieved in the field of Lane detection. The paper describes various systems identified for providing accurate lane detection methodologies. A few mentioned features are: -Lane departure warning, this system is expected to issue warnings for near lane departure events. Adaptive Cruise Control (ACC) : the system is designed such that it follows the nearest vehicle in the host lane with safe headway distance. Lane centring, aims to keep the vehicle between the lane at all times, lane change assist, turn assist etc.

https://www.researchgate.net/publication/257334033_Recent_progress_in_road_and_lane_detection_A_survey

### 2.4 Lane Keeping Assist with Lane Detection

A lane keeping assist (LKA) system is a control system that aids a driver in maintaining safe travel within a marked lane of a highway. The LKA system detects when the vehicle deviates from a lane and automatically adjusts the steering to restore proper travel inside the lane without additional input from the driver.

Lane Following Control with Sensor Fusion and Lane Detection - MATLAB & Simulink

### 2.5 Realtime Road Boundary Detection and Vehicle Detection for Indian Roads

The technique is based on modified road boundary detection which first segments the road area based on color segmentation and Hough transform is applied to find out the near vertical lines. Even in the absence of prominent lanes in the road, the segmentation line itself acts as a boundary line. Further optical flow based vehicle detection is integrated with the system.

https://research.ijais.org/volume5/number4/ijais12-450894.pdf

| EXISTING SYSTEMS | FEATURES OF EXISTING SYSTEMS | FEATURES OF PROPOSED SYSTEM |
|---|---|---|
| Lane Following Control with Sensor Fusion and Lane Detection | Both lane detection and surrounding cars are considered. The lane following system synthesizes data from vision and radar detections, estimates the lane center and lead car distance, and calculates the longitudinal acceleration and steering angle of the ego vehicle. | We are making it based on purely software vision in order to overcome any delay in the feedback mechanism involved in the existing system. |
| Realtime Road Boundary Detection and Vehicle Detection for Indian Roads | In the absence of prominent lanes in the road, the segmentation line itself acts as a boundary line. Further optical flow-based vehicle detection is integrated with the system. | We are using a series of techniques to get a better view of the road lanes and avoid any obstacle on the road. |
| Lane Detection Techniques using Image Processing | The paper views two approaches for lane detection. The first one | The proposed system aims to describe an approach to deal with the presence of |

| | | |
|---|---|---|
| | being thresholding, in which color thresholding is done in order to detect the color of the road instead of detecting the lane markings. The second one, instead of using the entire road pixels, only the edges of the path are considered. | obstacles in order to avoid the possibilities of damage in a self-driving vehicle for Indian roads. Even though, improved lane detection approach is considered, in the paper, the approach for dealing with presence of hurdles in the middle of the roads, is left out of focus. |
| A Much Advanced and Efficient Lane Detection Algorithm for Intelligent Highway Safety | The problem more focussed in this paper, is the presence of multiple lanes in the scenario. The approach for the same is, the idea of dividing the region of interest into two parts, before progressing with other image processing techniques. | The proposed system uses the region of interest approach, focusing on changing the perspective of the region, to advantage the image processing methods and provide higher accuracy in lane detection. |
| Recent progress in road and lane detection: A survey | The paper describes the recent developments in lane detection algorithms and real-time applications of the same. It describes the development of features including lane | The proposed system adds the clearance of lighting in areas of poor light conditions, for improved lane detection. |

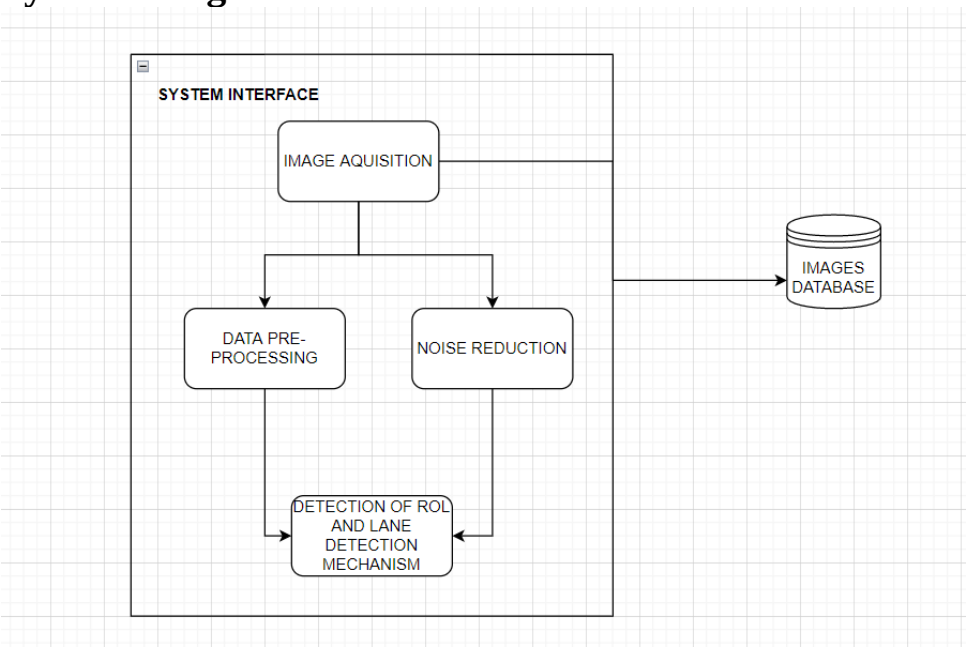
| | | |
|---|---|---|
| | departure warning, adaptive cruise control, lane keeping, lane centring, lane change assist, turn assist etc. | |

# 3 Requirements Specification

## 3.2 Software Requirements

1. Jupyter notebook
2. python
3. numpy
4. matplotlib
5. opencv
6. python3
7. sklearn.cluster
8. collections

# 4 System Design



## HOUGH TRANSFORM

The Hough Transform is a global method for finding straight lines (functions) hidden in larger amounts of other data. It is an important technique in image processing. For detecting lines in images, the image is first binarized using some form of thresholding and then the positive instances catalogued in an example's dataset.

## CANNY EDGE ALGORITHM

A Canny edge detector is a multi-step algorithm to detect the edges for any input image. It involves the below-mentioned steps to be followed while detecting edges ofan image.

1. Removal of noise in input image using a Gaussian filter.

2. Computing the derivative of Gaussian filter to calculate the gradient of image pixels to obtain magnitude along x and y dimension.

3. Considering a group of neighbors for any curve in a direction perpendicular to the given edge, suppress the non-max edge contributor pixel points.

4. Lastly, use the Hysteresis Thresholding method to preserve the pixels higher than the gradient magnitude and neglect the ones lower than the low threshold value.

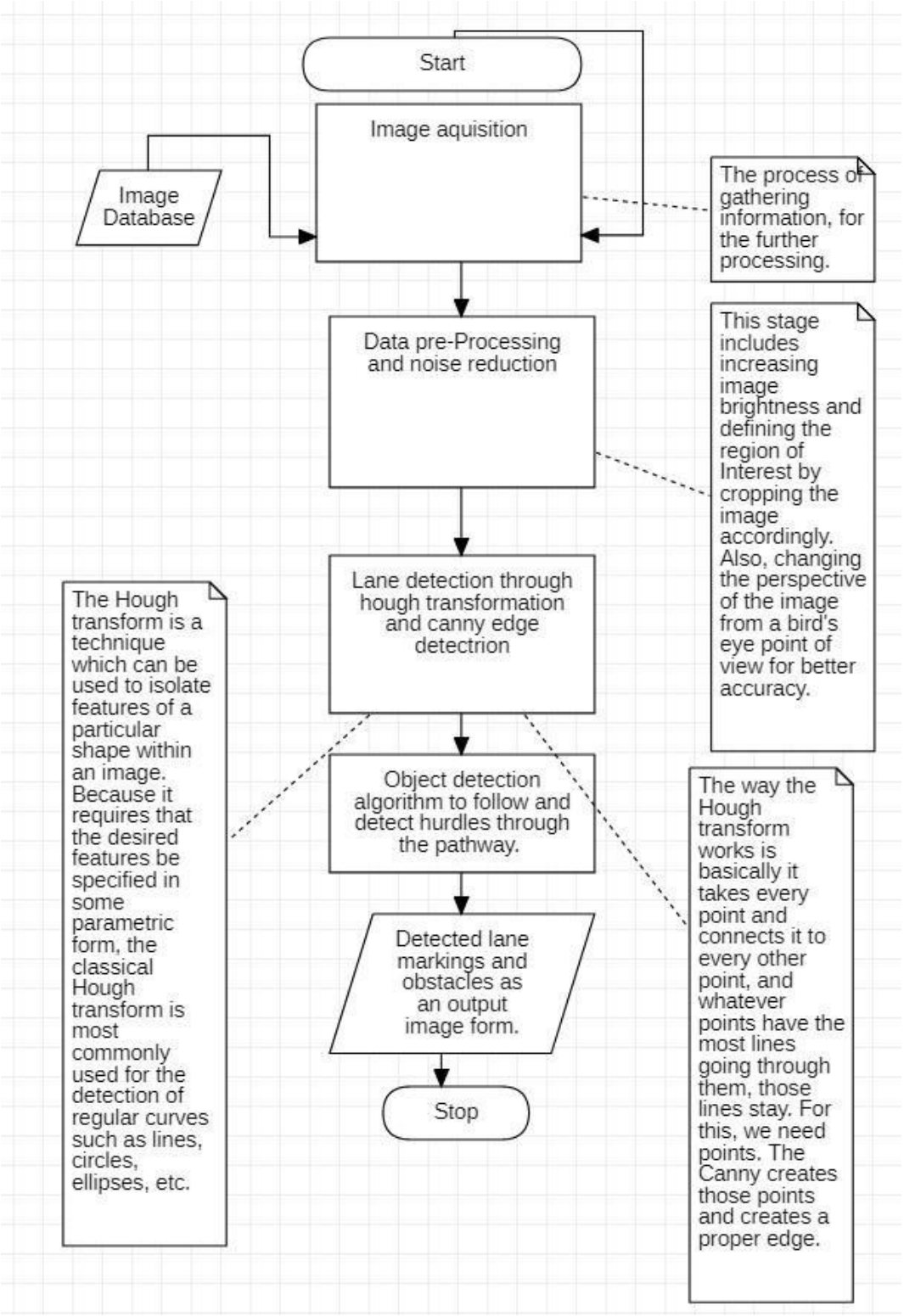
## CLUSTERING ALGORITHM

Clustering is a method of unsupervised learning and is a common technique for statistical data analysis used in many fields. In Data Science, we can use clustering analysis to gain some valuable insights from our data by seeing what groups the data points fall into when we apply a clustering algorithm.

## COLOR THRESHOLDING

If you want to filter out any pixel that is not the color of the lanes**.** Lane lines are either white or yellow on roads. Color Thresholding is a perfect solution to discard objects of other colors.

# 4 Implementation of System



### 4.1 Method

- Apply gamma correction to deal with night or very less brightness in the image.
- Define our region of interest.
- Reduce the noise and make data smoother.
- Apply Color thresholding to identify road lanes clearly.
- Apply Canny Edge Detection algorithm for finding edges.
- Apply Hough transform to make lines on our data.
- Apply Clustering algorithm to draw clear line.

## 5.2 MODULES

### 5.2.1 Data smoothing and enhancement

On Indian road due to absence of street light, night videos are very dark which makes road lines almost invisible. The purpose of a colour model is to facilitate the specification of colours in some standard way. Histogram Equalization is an image enhancement technique used to improve the dynamic range of an image and thereby improving its perceptibility. We're planning for Histogram Equalization on Hue channel of HSV color space, so as to increase the brightness of the image and improve the visual quality of the image.

Also, the data can be noisy, the solution is bilateral filter algorithm that reduces the noise and preserves edges, crucial for our algorithm to work. This is done after the definition of RoI.

### 5.2.2 Definition of RoI

We crop the image with Region of Interest mask in order to get rid of objects at the top and at the sides of the roads. We don't want them to influence the outcome, since we only care about the road. We know that the lanes will be located in the lower half of the image, usually in a trapezoid covering the bottom corners and the centre.

### 5.2.3 Color Thresholding

In context with Indian, we're going to apply color thresholding to detect the color of the road and differentiate it from the rest.

### 5.2.4 Canny edge detection and hough transformation

These methods are used for edge detection, Hough transform allows us to easily extract all the lines passing through each of our edge points. It's basically a brute force algorithm that tries all possible lines intersecting marked points in the image.

### 5.2.5 Clustering algorithm

The output of the Hough Transform algorithm can consist of too many lines (usually over 100). If we draw all of them we would get a really messy result. So, we plan to plot the lines' parameters and get an idea if formation of clusters are seen, then we can red.

# 6. Results and Discussion

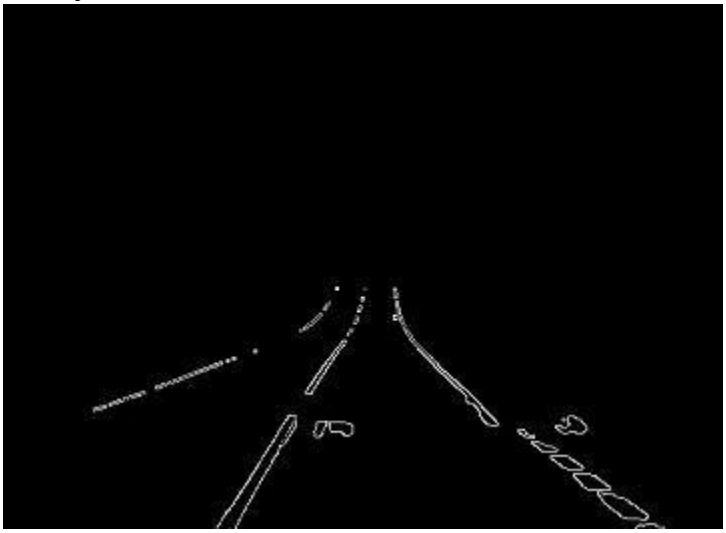
Original image

ROI

Bilateral

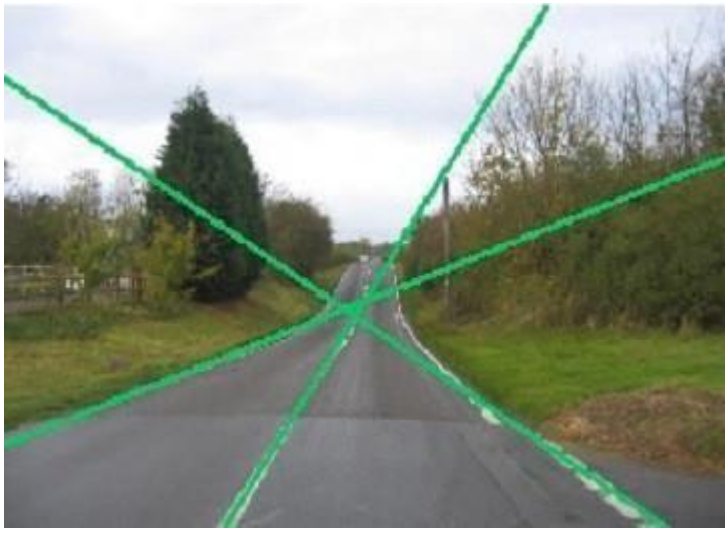
Color

Canny



Hough lines

K means clustering

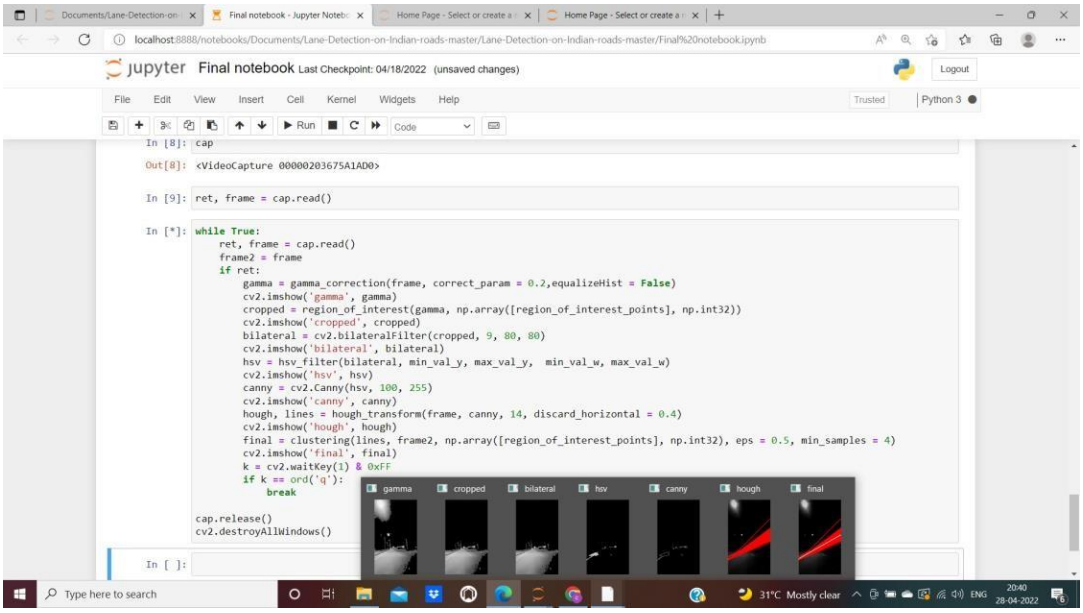


Gamma correction

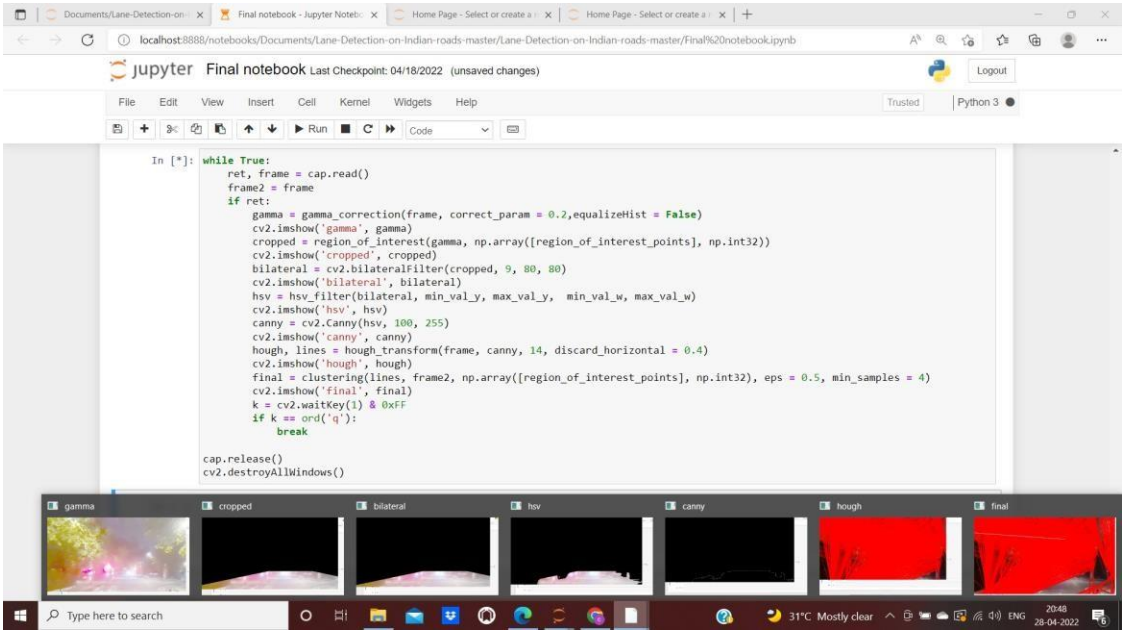
Gamma correction

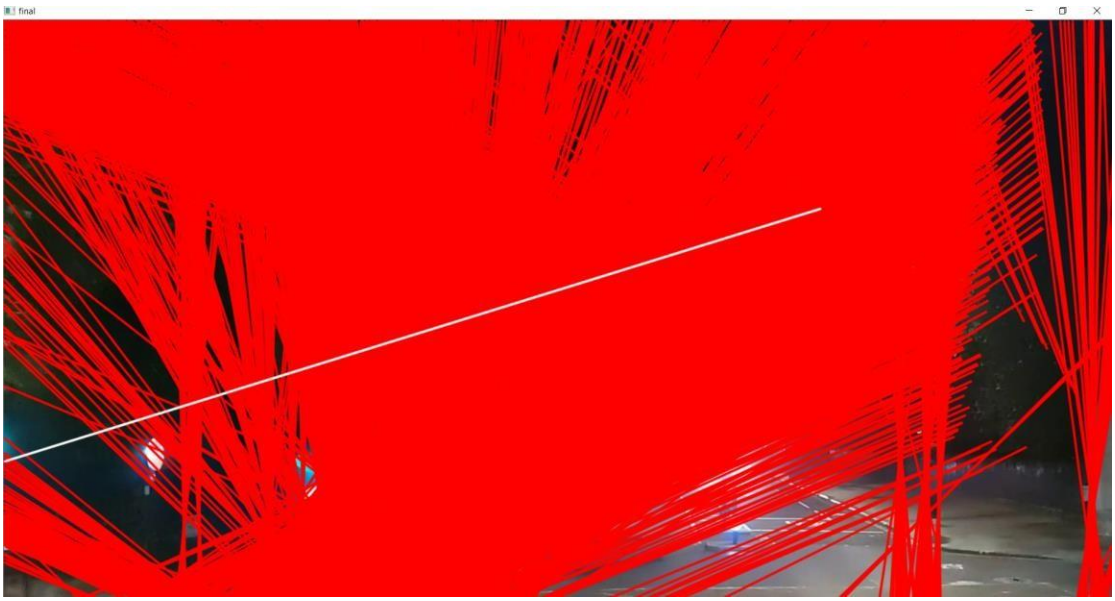
Outcome at night

One of the lane is correctly detected



One of the lanes is correctly detected but a huge cluster of red lines.




## 7. Conclusion and Future Work

The result changes with different videos depending on the speed of the recording and the angle of the recording taken.As of now the lane that is detected overlaps with some red lines from the previous step, we hope to make our algorithm efficient enough to get the perfect output all the time.

- Going ahead we have planned to use Particle filter (Cascade Particle filter), Kalman filter and recursive Bayes etc.
- **Road Lane detection using Deep Learning:**We will be using the existing model and will train with some other data.We hope deep learning can help us in detecting the lanes and also able to change the lane as video frames changing the lane.We will be using Neural networks more specifically Convolution neural.The architecture will be comprising of few convolutional layers followed by deconvolutional layers along with Max Pooling ,Upsampling and Dropout.

# 8. REFERENCES


- **Road Lane line detection - Computer Vision Project in Python**-https://r.search.yahoo.com/_ylt=Awrx355n0mpiJygATwO7HAx.;_ylu=Y29sbwNzZzMEcG9zAzQEdnRpZAMEc2VjA3Ny/RV=2/RE=1651196647/RO=10/RU=https%3a%2f%2fdata-flair.training%2fblogs%2froad-lane-line-detection%2f/RK=2/RS=7IiU2JzJ.3A8y7fT3XxorhsrzR4-
- **Lane and Vehicle Detection in Simulink Using Deep** Learning-https://in.mathworks.com/help/deeplearning/ug/lane-vehicle-detection-simulink-using-predict-block.html
- **End-to-End Deep Learning of Lane Detection**-https://arxiv.org/abs/2102.04738

- **Realtime Road Boundary Detection and Vehicle Detection for Indian Roads**-https://research.ijais.org/volume5/number4/ijais12-450894.pdf

# APPENDIX

**Region of Interest**

```
def region_of_interest(img, vertices):
    # Define a blank matrix that matches the image height/width.
    mask = np.zeros_like(img)

    # Retrieve the number of color channels of the image.
    #channel_count = img.shape[2]

    # color used to fill polygon
    match_mask_color = 255

    # Fill the polygon with white
    cv2.fillPoly(mask, vertices, (255,255,255))

    # Returning the image only where mask pixels match
    masked_image = cv2.bitwise_and(img, mask)

    return masked_image
```

**Gamma correction**

```
def gamma_correction(RGBimage, correct_param = 0.35,equalizeHist = False):
    red = RGBimage[:,:,2]
    green = RGBimage[:,:,1]
    blue = RGBimage[:,:,0]

    red = red/255.0
```

```
    red = cv2.pow(red, correct_param)
    red = np.uint8(red*255)
    if equalizeHist:
        red = cv2.equalizeHist(red)

    green = green/255.0
    green = cv2.pow(green, correct_param)
    green = np.uint8(green*255)
    if equalizeHist:
        green = cv2.equalizeHist(green)


    blue = blue/255.0
    blue = cv2.pow(blue, correct_param)
    blue = np.uint8(blue*255)
    if equalizeHist:
        blue = cv2.equalizeHist(blue)


    output = cv2.merge((blue,green,red))
    return output
```

### HSV Filter

```
def hsv_filter(image, min_val_y, max_val_y, min_val_w, max_val_w):
        hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    mask_yellow = cv2.inRange(hsv, min_val_y, max_val_y)
    mask_white = cv2.inRange(hsv, min_val_w, max_val_w)
    mask = cv2.bitwise_or(mask_yellow, mask_white)
    img_filtered = cv2.bitwise_and(image, image, mask=mask)

    return img_filtered
```

### Hough transform

```
def hough_transform(original, gray_img, threshold, discard_horizontal = 0.4):
        lines = cv2.HoughLines(gray_img, 0.5, np.pi / 360, threshold)
    image_lines = original
```

```
lines_ok = [] #list of parameters of lines that we want to take into account (not horizontal)

if lines is not None:
    for i in range(0, len(lines)):
        rho = lines[i][0][0]
        theta = lines[i][0][1]
        #discard horizontal lines
        m = -math.cos(theta)/(math.sin(theta)+1e-10) #adding some small value to avoid dividing by 0
        if abs(m) < discard_horizontal:
            continue
        else:
            a = math.cos(theta)
            b = math.sin(theta)
            x0 = a * rho
            y0 = b * rho
            pt1 = (int(x0 + 1000*(-b)), int(y0 + 1000*(a)))
            pt2 = (int(x0 - 1000*(-b)), int(y0 - 1000*(a)))
            cv2.line(image_lines, pt1, pt2, (0,0,255), 2, cv2.LINE_AA)
            lines_ok.append([rho,theta])

lines_ok = np.array(lines_ok)

return image_lines, lines_ok
```