LANE DETECTION FOR SELF DRIVING CARS

A major project report submitted in partial fulfillment of the requirement for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering / Information Technology

Submitted by

Yash Garg (201298)

Rohit Rakesh (201329)

Under the guidance & supervision of

Dr. Pardeep Kumar, Associate Professor (SG), SM-ACM

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology,

Waknaghat, Solan - 173234 (India)

CERTIFICATE

This is to certify that the work which is being presented in the project report titled **'LANE DETECTION FOR SELF DRIVING CARS'** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by **Yash Garg(201298)** and **Rohit Rakesh(201329)** during the period from August 2023 to May 2024 under the supervision of **Dr. Pardeep Kumar**, (Associate Professor, Department of Computer Science and Engineering, Jaypee University of Information Technology).

(Student Signature with Date)
Student Name: Yash Garg
Roll No.: 201298

(Student Signature with Date)
Student Name: Rohit Rakesh
Roll No.: 201329

The above statement made is correct to the best of our knowledge.

(Supervisor Signature with Date)
Supervisor Name: Dr. Pardeep Kumar
Designation: Associate Professor, SM-ACM
Department: CSE/IT
Dated:

Candidate's Declaration

We hereby declare that the work presented in this report entitled **'LANE DETECTION FOR SELF DRIVING CARS'** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out by **Yash Garg(201298)** and **Rohit Rakesh(201329)** over a period from August 2023 to May 2024 under the supervision of **Dr. Pardeep Kumar** (Associate Professor, Department of Computer Science & Engineering and Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature with Date)
Student Name: Yash Garg
Roll No.: 201298

(Student Signature with Date)
Student Name: Rohit Rakesh
Roll No.: 201329

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature with Date)
Supervisor Name: Dr. Pardeep Kumar
Designation: Associate Professor, SM-ACM
Department: CSE/IT
Dated:

ACKNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for His divine blessing which made it possible to complete the project work successfully.

We are grateful and wish our profound gratitude to Supervisor **Dr. Pardeep Kumar, Associate Professor**, Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of our supervisor in the field of "**LANE DETECTION FOR SELF DRIVING CARS**" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, and valuable advice have made it possible to complete this project.

The in-time facilities provided by the Computer Science department throughout the project development are also equally acknowledgeable.

Last but not the least, our sincere thanks to all our teachers and friends who have helped us straightforwardly or in a roundabout way in making this project a win.

Finally, we acknowledge with due respect the constant support and patience of our parents.

Yash Garg
201298

Rohit Rakesh
201329

TABLE OF CONTENT

Chapter-5 Results and Evaluation

Chapter-6 Conclusions and Future Scope

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In the past, self-driving vehicles have become popular. Autonomous vehicles will change how people live. Such systems demand large data sets and highly specialized computers. There is an emulator which ensures easy production of as many shots of a driving car as required by the user. The task attempted prediction and recognition of the traffic lanes utilizing only the available photographs.

When we drive, we look and go with our eyes. The road lines serve as a permanent point of reference enabling us to direct the vehicle during the drive. It is natural for us to start with the capability of detecting and recognizing the lane lines via some algorithm. This is a tough nut to crack when it comes to lane detecting problems.

Fundamentally, lane detection is a multi-features detection problem which is difficult to solve with conventional computer vision or machine learning techniques. We provide an approach for processing images using Canny edge detection, regions masking.

However, the main disadvantage about these techniques is that they may lead to erroneous results or total failure for unclear markings or no marks altogether. One of these ways to detect lanes on an unmarked road and then the other more suitable method is reviewed in thispaper. The two methods utilize strictly visionary or camera data obtained through digital imaging approach. This will be obtained in real time so as to allow the driver or autonomous vehicle to make required changes such as taking curves and remaining on the road

CHAPTER 1

INTRODUCTION

1.1 GENERAL INTRODUCTION

The race to create and market intelligent and autonomous vehicles has intensified due to advancements in knowledge and communication technologies, vehicle design, and automotive engineering. These cars have the following qualities: increased traffic efficiency, comfort for the driver, and stability. The advanced driving assistance system (ADAS) uses the lane departure warning system, lane keeping assistance system (LKAS), front collision warning system, and smart parking assistant system (SPAS) to help drivers via a human-machine interface. When the system detects that the vehicle is deviating from its lane,LKAS continually applies a small amount of counter-steering force to keep the vehicle at the center of road and on course [1]. SPAS offers parking-simplifying services. to help a motorist position themselves properly before beginning To aid a driver in reaching the proper starting position for starting to reverse at a parking place, automatic steering assistance along a predetermined path is provided [2]. An autonomous car can get where it's going without the driver having to pay attention to the road ahead of them. This study describes a route tracking approach and the creation of a lane recognition system employing methods used in intelligent and autonomous cars.

Using an automatic lane-detection system might involve anything fromshowing the user where the lanes are on a different screen to major complicated tasks such asanticipating road changes in real time to prevent accidents with different cars. A new era of transportation has begun with the introduction of autonomous vehicles, which promise safer roads, more effective traffic flow, and less of an impact on the environment. However, overcoming

difficult technical obstacles is necessary to get to a fully autonomous driving state, with lane detection serving as a keystone for both vehicle safety and navigation accuracy.
The accuracy of traditional lane detection methods is frequently compromised in a variety of environmental conditions, such as inclement weather, low light levels, or poorly marked roads.

1.2 PROBLEM STATEMENT

Obtaining enough data to feed powerful deep learning algorithms for tasks such as road detection is one of the main obstacles to edge detection in self-driving cars. In order to create the algorithms for applications involving self-driving cars, a substantial quantity of data needs to be collected and annotated. It costs money and effort to gather and categorize real-world data, and testing every possible real-world scenario—such as a fast-moving car crashing into a brick wall—is not feasible.

The problem of road lane detection and signal detection for self-driving cars is to automatically recognize lanes and traffic signals. The progress in image processing and deep learning is solely responsible for the self-driving car's capacity to identify road signs and tracks from video frames. The vehicle used in this investigation has a polynomial regression model with thresholding for lane guiding, YOLO version 1 for object recognition, and a controller to coordinate data across the systems. The proposed approaches can be used for road lane guiding, steering recommendation, object identification, and object location detection.

Real-time object identification and detection presents a significant challenge. One prominent example of a security lapse is the 2016 Tesla autopilot collision, which happened because sunlight confused the car's sensors and the system failed to detect the truck coming from the right, causing the collision. Identifying and detecting objects in real time is a crucial task. A well-known example of a security lapse is the 2016 Tesla

auto-pilot disaster. The sunobscured the car's sensors in that incident, causing the algorithm to miss a truck that was approaching from the right and causing the collision.

Lane detection is still a major bottleneck in autonomous driving, even with major advancements in the technology. This is especially true in real-world scenarios where the environment is dynamic, weather is unpredictable, and road markings vary. Under difficult circumstances, conventional lane detection systems frequently fail to reliably detect lane boundaries, which poses a risk to safety and impairs driving efficiency. The widespread adoption of autonomous vehicles and the achievement of their anticipated benefits in terms of road safety, traffic efficiency, and environmental sustainability hinge on the availability of a strong, flexible, and dependable lane detection system.

1.3 OBJECTIVES

In addition to being a reality today, self-driving car technology offers an insight into the complexity of technology to come. Many experts in different domains have worked together to create the best autonomous vehicles possible. Cars already come equipped with lane detection as standard equipment, and a great amount of real-world data is required to train. Inorder to construct fully autonomous vehicles. The system's idea is to use Python and OpenCV to build a system for identifying road lanes, with the ultimate goal of using this system to generate data for self-driving cars. Using a function to conceal unwanted objects inimages such as trees, rocks, and electrical lines, and Canny edge detection to identify lane boundaries, the algorithm will allow the model to concentrate on lane recognition. Detecting and drawing lanes using the Hough Transform is how the proposed method reliably determines lanes in real-time.

The purpose of this research is to accomplish the following goals:

1. To employ real-time lane edge detection using computer vision algorithms, such asthe clever edge detection algorithm.
2. Creating a system that could identify lanes and creating a sizable amount of datasetsin order to construct a lane detection system into self-driving cars.
3. To lessen the amount of time required to gather real-world data for creating lanedetection systems for self-driving cars
4. Create a scalable system that can be used with a variety of vehicle types and camera configurations.
5. Give detection systems the ability to manage complex road situations like merging lanes, intersections, and construction zones.

1.4 SIGNIFICANCE AND MOTIVATION

Autonomous vehicles can traverse roads safely and effectively thanks to lane detection, a critical component of their technology. In order for the car's control system to calculate itsposition and make the required corrections, it entails recognizing and following the lane markings on the road. The urgent need to improve the precision and dependability of lane detection systems in autonomous driving technology is what drives our research. Current methods frequently falter in unfavorable circumstances, endangering driving performance and safety. We hope to get around these restrictions and open the door to safer roadways and more effective traffic flow by incorporating state-of-the-art methods like deep learning and multi-sensor fusion. Our creative solution advances autonomous vehicle research technology while simultaneously enhancing driving performance. In the end, our work has the potential to completely transform transportation by making it more sustainable, safer, and available to everyone.

Significance of Lane Detection:

Safe Navigation: Lane detection allows self-driving cars to stay within their designated lanes, preventing collisions with other vehicles or obstacles. It ensures that the car adheres to trafficrules and avoids lane departure accidents.

Lane Change Assist: Lane detection facilitates lane change maneuvers, enabling the car toidentify safe opportunities to switch lanes and maintain a smooth flow of traffic.

Road Curvature Estimation: Lane detection helps determine the curvature of the road ahead, allowing the car to anticipate upcoming bends and adjust its steering accordingly.

Adaptability to Road Conditions: Advanced lane detection algorithms can adapt to variousroad conditions, including different lane marking styles, lighting conditions, and weather patterns.

Motivation for Lane Detection Research:

Increasing Safety: Lane detection is essential to increasing the safety of self- driving cars,lowering the possibility of collisions, and making the transportation system safer overall.

Improving Efficiency: Accurate lane detection enables self-driving cars to navigate roadsmore efficiently, reducing congestion and optimizing traffic flow.

Expanding Self-Driving Capabilities: Lane detection is essential for expanding the capabilities of self-driving cars, allowing them to operate in diverse environments andchallenging conditions.

Advancing Autonomous Vehicle Technology: Research in lane detection techniques drivesthe advancement of autonomous vehicle technology, paving the way for the future of transportation.

1.5 ORGANIZATION

In Chapter 1: Introduction In the introduction chapter, focus on providing an overview of the project, its significance, and the problem it aims to address. It includes background, problem statement, objectives and significance and motivation of the project work.

In Chapter 2: Literature Review Give a thorough examination of previous studies and publications that are relevant to your project in the literature review chapter. It contains critical analysis, a review of related work, and applicability to ongoing initiatives.

In Chapter 3: System Development The technical aspects of your project are covered in detail in this chapter. It includes the system architecture, which highlights the key parts and how they work together, the methodology used to design the system, and technical information about how the process was done.

In Chapter 4: Testing Examination Pay close attention to your project's testing phase. It outlines the testing procedures used as well as the standards for judging a test's performance. The results of the testing phase are shown in this chapter. Continuous testing and adjustments are made to the system to guarantee optimal performance and minimize false positives.

In Chapter 5: Result and Evaluation The project's success should be assessed and the overall outcomes should be presented in this chapter. It talks about how well the project accomplishes its goals and shows the metrics that are used to assess the system's performance with the preprocessed data.

In Chapter 6: Conclusion and Future Scope Summarize the main conclusions of the paper and suggest possible directions for future research. Describe the project's primary results and accomplishments. Provide a final assessment of the project's success. Stress how the initiative advances the field. Make suggestions for future project enhancements or extensions. Talk about potential expansions and applications for the project in various settings.

CHAPTER 2

LITERATURE SURVEY

2.1 Overview of Relevant Literature

[1] Sandeep Kumar Sathapathy and Tullimalli Sarsha Sree

We are using a software programme that we are currently developing to locate the lane lines along the road. The ability of self-driving cars to detect and maintain lanes is crucial for the development of their algorithms. The cost of creating a software pipeline to track traffic lanes using computer vision techniques will be estimated here. We'll tackle this task in two different ways. Convolutional neural networks and the Hough transform technique are these (CNN). However, lane markings could be an essential point of reference for safe driving. In order to better lane line recognition speed and accuracy, this work proposes a modified version of the Hough transform- supported lane line recognition algorithm.

After the Canny operator locates the edge of the image's region of interest, the threshold is automatically chosen for threshold processing based on edge information. To improve the recognition of lane lines by the Hough transform, three constraints derived from the angles and lane width are proposed. Rectilinear regression is therefore used to fit the appropriate lane lines.

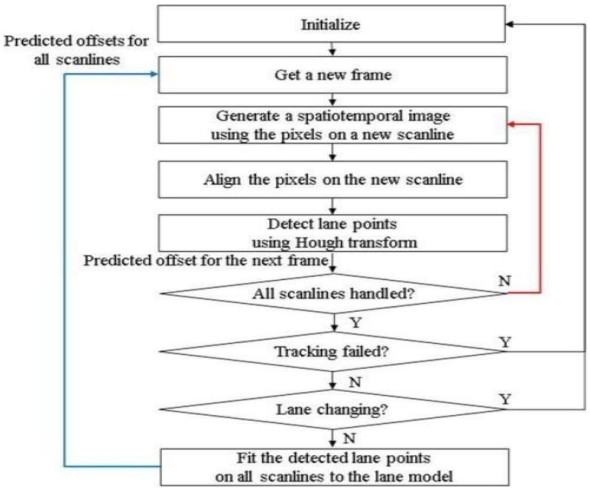

Fig 1 : Block Diagram

[2] Nidhi Lakhani, Ritika Karande, Deep Manek, Vivek Ramakrishnan

Automobiles have become increasingly popular as a means of transportation due to society's rapid development. There are an increasing variety of car types on the congested road. Lane detection is a popular topic in computer vision and machine learning, and it has been used in intelligent car systems. This is a relatively new field with applications in the business world. Using lane markers in a complex environment, the lane detection system accurately approximates the position and trajectory. Lane detection plays major role in thelane exit warning system's operation. Line detection and edge detection are the two main parts of lane detection.

The two most popular lane line detection techniques are convolution-based and the Hough transform. Lane detection is the process of identifying road lane markings and then providing the positions to an intelligent system. Intelligent vehicles in intelligent transportation systems collaborate with the infrastructure to improve traffic and foster a safer environment. Using a lane detection system can be as simple as showing the user where lanes are on an external display or as sophisticated as anticipating a lane change in real time to prevent crashes with other cars.

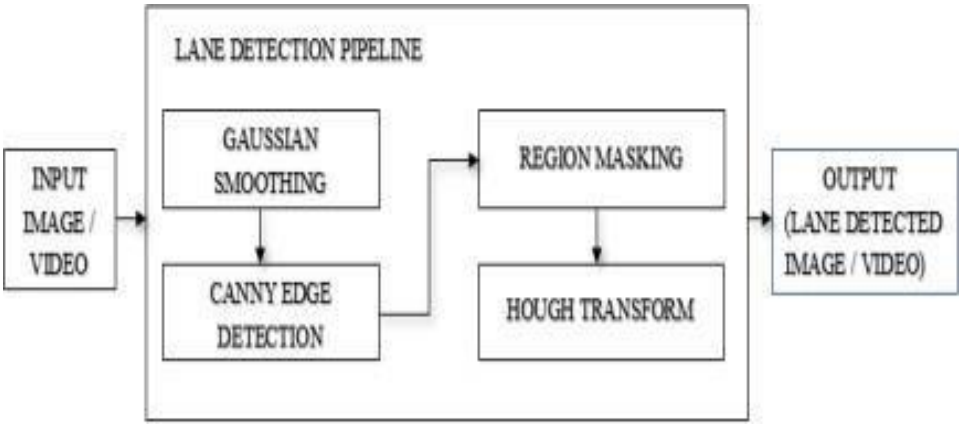

Fig 2: Block diagram

[3] Iftikhar Ahmad , Jin Ho Lee , and Soon Ki Jung

Humans make a wide range of decisions every day, many of which are impacted by the sensory information we take in from our surroundings. When it comes to driving, most of this perception is visual. Vehicles, also known as self-driving cars, are designed to be able to identify objects in their surroundings and make fast decisions about how to respond to them.

This method uses the Canny edge detection technique to suggest a lane for the real- time racing game Asphalt 8: Airborne. The game's screen is accessed through the ImageGrab module of the Python Imaging Library (PIL) programming language. Canny edge detection, a well-liked method in computational image processing, is used by OpenCV to identify the boundaries of the lanes, and a masking algorithm was used to eliminateobstacles like trees, rocks, and cables. Realistic physics, graphics, and a variety of settings—such as lane-keeping aids like sharp turns, hills, and different weather conditions—are all present in the gaming environment.

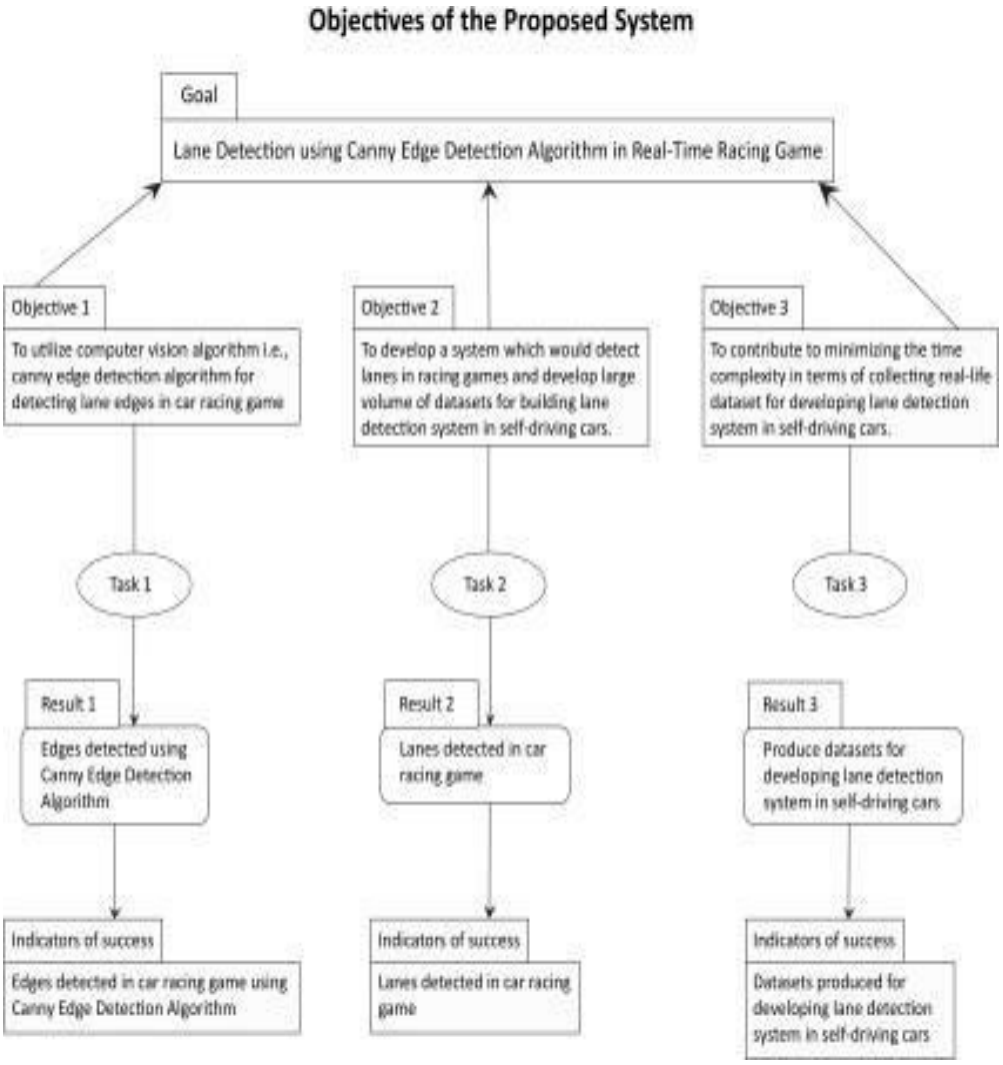

Fig 3 : Research Objective's Breakdown Structure (RBS) of our study

[4] Raja Muthalagu, Anudeep Sekhar Bolimera, Dhruv Duseja, Shaun Fernandes

People spend a lot of time in the modern world walking the streets and driving or interacting with other cars. Every year, traffic accidents claim the lives of 1.35 million people worldwide. It has been discovered by researchers that human error accounts for the majority of traffic accidents.

Self-driving car technology consists of three main parts: perception, planning, and control.

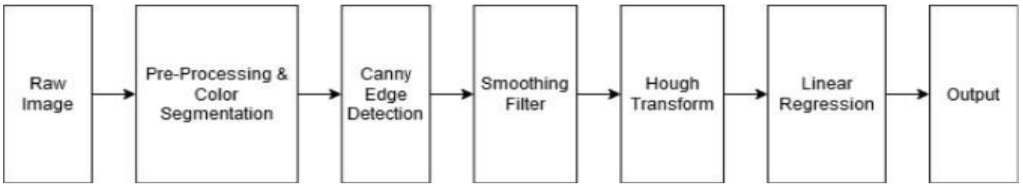

Fig 4: Minimalistic approach-based lane detection

[5] Vighnesh Devane, Ganesh Sahane, Hritish Khairmode, Gaurav Datkhile

The method suggests a useful variant of the sliding window algorithm for fitting lane curves. The benefit of this improved version is that sliding windows can be used even with uneven lane markers. Using the road borders from the binary image, this technique is applied in the second method. Next, using the geographic coordinates of the road boundaries, the curve value and the offset of the car are computed.

Making lane detection possible on roads with poor visibility or faded lane markings is the project's main objective. This research attempts to construct lane detection for an effective autonomous car using methods such as pixel summation, thresholding, image warping, andsliding window algorithm. Lastly, the most suitable application scenarios and the advantages and disadvantages of the previously mentioned techniques have been covered.

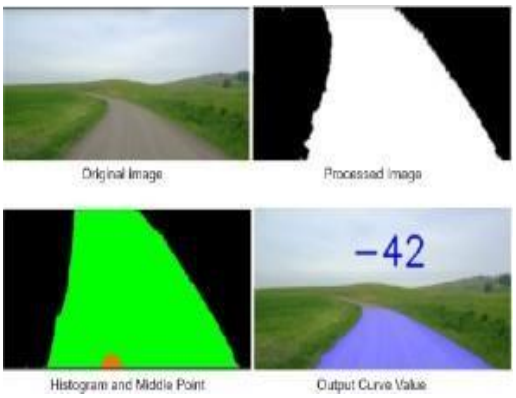

Fig 5: Pixel summation output

Fig 6: Output

[6] Jamel Baili, Mehrez Marzougui, Ameur Sboui, Samer Lahouar, Mounir Hergli

Image processing researchers have studied lane detection extensively. These studies can be categorized into two primary groups, according to a summary of the research. Lane markersin an input image from a rear-view camera are identified in the first. The front- mounted camera is utilized by the second group. Different image processing techniques, including the B-Snake technique and the Probability of Picture Shape (LOIS) algorithm, have been developed for the latter scenario. Using a feature-based methodology, these algorithms use multiple techniques to extract features from an image, including edges. The recommended approach has worked well, but it needs to be adjusted to account for badweather (snow and rain) and low light conditions (nighttime).

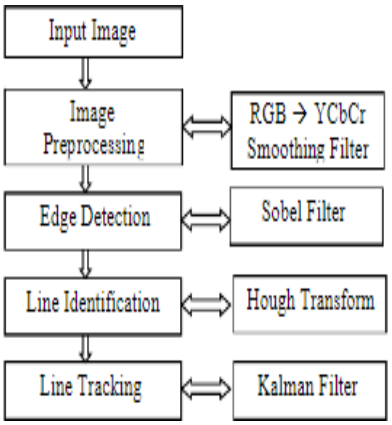

Fig 7 : Flowchart of lane boundary methodology detection using feature-based method

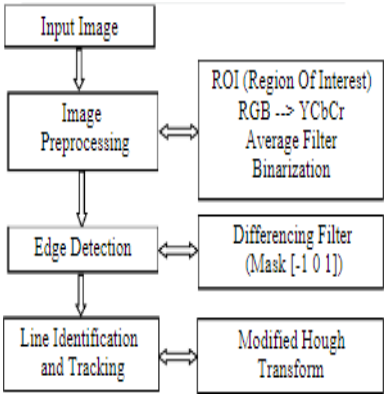

Fig 8 : Flowchart of our

[7] Yan Liu , Jingwen Wang , Yujie Li , Canlin Li

To solve the problem of lane detection in dimly lit environments, this paper suggests a lanedetection system. The primary contributions of this study are enumerated in the list below.

1. There is a dataset of hazy lane lines on this page.
2. An upgraded GAN is used to enhance the lane characteristics and boost the efficacy ofblurring lane recognition in complex road environments.

3. The suggested approach significantly improves upon the current state-of-the-art detectors by outperforming them in high speed and challenging road conditions (line curves, dirty lane lines, illumination changes, and occlusions).

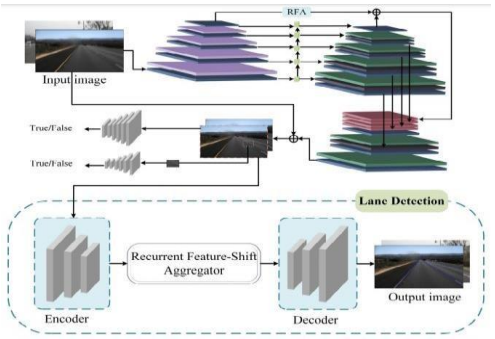

Fig 9: Architecture of Proposed Lane-GAN

[8] Zequn Qin, Huanyu Wang, and Xi Li

Row-based global image feature selection for lane detection. In other words, our approach chooses appropriate lanes for each predefined row based on the global features. Row anchors (for horizontal) sequence in pre-defined lines is our lanes modeling. Firstly, gridding is involved. This is done by breaking down the location into different cells on every row anchor.This enables a comparison between the lane identification and the cell selectionprocess over predefined row anchors.

The proposed definition states that lane detection is a row-based selection problem that requires global features to be solved. This approach helps in tackling the speed issue and visual clue problem. They also recommend using structural loss for modelling prior knowledge of lanes. Qualitative testing also supports our approach, and quantitative testing provides evidence for the structural loss.In particular, the highest accuracy and speed couldbe achieved by our model with the Resnet-34 backbone. A lightweight version of our method,called Resnet-18, was able to achieve competitive performance with 322.5 FPS even at the same resolution.

[9] Ling Ding, Huyin Zhang, Jinsheng Xiao, Cheng Shuand Shejie Lu

They fuse the discriminant loss of LaneNet, deep cavity convolution of DeepLabv1 as well asLMD. Originally, this is based on some methods of lane segmentation and tracking.The CNNlayer of feature part is replaced with void convolution in order to expand the receptive field. Because integrating into diverse network structures is easy and because instance segmentation is achieved through post-processing, the discriminant loss function is unique. Furthermore, previous articles have covered a few works on initial processing of denoising images.

After being tested on various data sets, lanes, and weather conditions, the suggested method's correctness and robustness are verified.

Particularly in the area of corner and false actual lane line recognition, the algorithm's detection accuracy is much higher despite its noticeably slower detection speed. When it comes to effective detection, on the other hand, the deep learning algorithm has none of theseproblems and can produce precise detection.

[10] Y. Wang, E. K. Teoh and D. Shen

Lane model is required for lane detection. To achieve full 3D data recovery of the 2D static image for lane modelling, some assumptions have to be made regarding the real road’s architecture. We consider the 2D lane model that assumes the two sides of the road borders tobe on a straight line on the ground plane.

To demonstrate the effect caused by general lane borders, a new lane model inspired by the B-Snake has been developed. Compared to straight and parabolic models, it is capable of representing a larger variety of lane configurations. Finding the lane's center and identifying its two ends present challenges in this situation.

[11] Gonzalez, J.P., Ozguner

This article proposes a concept for an intelligent automobile system. The algorithm uses the characteristics of the grey level histogram of the road to identify lane markers. After that, decision trees are used to analyze the connections between each lane marker, and structures delineating the lane boundaries are made. The system also generates images that can be used in the pre-processing phases of algorithms for obstacle, lane, or tracking detection. The system runs at about 30 Hz in real time.

The two recommended methods for solving the lane detection issue proved to be effective fixes. The histogram-based segmentation performed better than many conventional methods with very little computational effort. We can consider image attributes with the methodology that would be lost in other approaches because of processing constraints. The method can be adjusted for use in applications such as terrain classification and off-road navigation.We are only analyzing slightly less than half of the image's elements, so the reduction in processing time obtained and the reduction in data to be processed by the next step should both be taken into account.

2.2 Key Gaps in the Literature

S.No	Authors	Advantages	Disadvantages
1	Tullimalli Sarsha Sree and Sandeep Kumar Sathapathy	This paper proposes an enhanced Hough Transform algorithm for lane line detection, with the goal of improving accuracy and real-time performance.	Focusing on image more than video graphic
2	Nidhi Lakhani, Ritika Karande, Deep Manek, Vivek Ramakrishnan	It is one of the emerging areas that can be applied in businesses today. Lane detection system uses lane markers to reliably approximate the position and trajectory of the vehicle relative to the lane in a complex environment.	Video graphic inputs are extremely time consuming.

3	Raja Muthalagu, Anudeep Sekhar Bolimera, Dhruv Duseja, Shaun Fernandes	Results are accurate	Initially, Dropout was used in order to fight overfitting. However, the performance was much worse.
4	Vighnesh Devane, Ganesh Sahane, Hritish Khairmode	This Project is carried out with the purpose of lane detection in the poor conditioned roads.	More image focused than videos graphic
5	Jamel Baili, Mehrez Marzougui, Ameur Sboui, Samer Lahouar, Mounir Hergli	Numerous image processing algorithms have been developed, including the B- Snake algorithm and the Likelihood of Picture Shape algorithm.	It also needs to be adjusted to incorporate bad weather (rain, snow) and nighttime.
6	Yn Liu , Jinwen Wag , Yuje Li , Calin Li	Network targeting the complexity of this task in blurry situations. In this article, a blurred image is proposed.	Video graphic inputs are extremely time consuming.
7	Zequn Qin, Huanyu Wang, and Xi Li	For low level visual and high level semantic information, the other feature-aggregation method that is demonstrated is used.	The suggested formulation perceives the row-wise selections in relation to the lane detection.
8	Ling Ding, Huyin Zhang, Jinsheng Xiao, Cheng Shuand Shejie Lu	Rather than a common convolution layer, void convolution is applied to augment the receptive field.	However, the algorithm is much slower than. detection speed
9	Y. Wang, E. K. Teoh and D. Shen	Recently, a new B-Snake inspired lane model has been proposed in order to express the perspective phenomenon in parallel lines.	The image is more focused than video graph.
10	Gonzalez, J.P., Ozguner	Many conventional methods were outperformed by the histogram-based segmentation.	The high level classifier performed very well for light to medium traffic scenes but not so well in heavier traffic scenes.

Table 1: Literature Survey

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 Requirements and Analysis

Hough Transform

Subsequently, thresholding is applied using the automatically selected threshold for a particular functional area that corresponds to the edge information as detected by the Cannyoperator. Three aspects from the point of view angle and width to improve Hough transformfor the lane line detection. Therefore, rectilinear regression gives the best lane lines fit.

1. Noise Reduction

Erroneous detection is a common issue with all edge detection techniques, including noise that could make it one of the significant issues. The detector's sensitivity towards noise is minimized by smoothing the image using a five by five Gaussian filter. To achieve this, a normally distributed kernel traverses the whole image with each pixel being assigned a value to the weighted average of the surrounding ones. Five by five kernel in this instance.

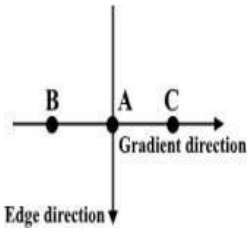

Fig :**10** 5x5 Gaussian Kernel. The asterisk(*) denotes convolution operation.

2. Intensity Gradient

An OpenCV uses the sobel kernel to determine whether the borders are horizontal,

vertical,or diagonal by using the smoothed image.

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Fig 11: Sobel kernel for calculating the primary derivative of horizontal and vertical directions

3. Non-maximum suppression

Non-maximum suppression of "thin" successfully sharpened edges. Every individual pixel value is tested to determine whether it represents the minimum locally within the calculated gradient's axis. It is placed perpendicular to the edge. Therefore, the values at the pixels of Band C will be compared with that of A in relation to A being a local maximum because the gradient is perpendicular to the edge direction. In case of local maximum value (A), the nextpoint has no non maximum suppression applied.
Otherwise, A is truncated with a pixel valueof zero.

$$\mathbf{B} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * \mathbf{A}$$

Fig 12 : Non-maximum suppression on three-point

4. Hysteresis Thresholding

The final representation after non-maximum suppression proves that there were strong

pixels.Finally, additional analysis is needed to determine whether such “weak pixels” reflect edges or noise. Taking two pre-set threshold values of minVa1 and maxVal, we choose which pixelslie in the edge category and those for removal outside the edged category. MaxVal representsthe value above which any pixel edges will be considered. Pixels with an intensity gradient between minVal and maxVal will not be called edges until such time as they link up to a pixelwhose greyscale varies more than beyond maxVal.

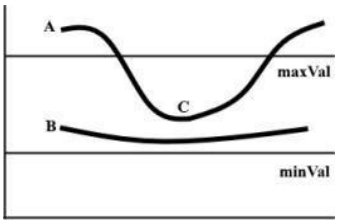

Fig 13: Hysteresis Thresholding on two lines

5. Segmenting line area

As far as the latter stages are considered, we will design a triangular mask to cut off that roadarea and eliminate undesired frames. The three coordinates are used to define the triangular mask as green circles.

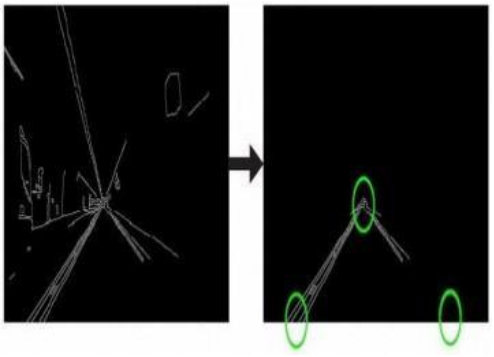

Fig 14 : The triangular mask is going to be defined by three coordinates

Probably the most popular and commonly used edge detector in the domain of computer vision and image processing is the Canny edge detector. Although it sounds trivial to understand the Canny edge detector, let's break the steps down into digestible pieces and learn the real deal. Luckily for us, OpenCV already included the Canny edge detector into us because it widely employed almost in all computer vision applications through the cv2.Canny command.

You will most likely have a call to the Canny edge detector inside in the source code of manyimage processing projects. If we want to know how far our camera is from an object, create adocument scanner, or see game boy images in an image, we can utilize the Canny edge detector, which is often taken as an important preprocessing step.

To that end, spotting edges in an image should follow the following procedures.

1. The noise is removed by using a Gaussian filter on the input image.

2. Determining the derivative of a Gaussian filter in order to derive the gradient of an image'spixels as well as determining their magnitudes along the x-axis and y- axis.

3. Considering a group of neighbors, reduce the non-max edge contributing pixel points forany curve perpendicular to the specified edge.

4. Discarding pixels with values below the low threshold and preserving ones greater than theHysteresis Thresholding gradient magnitude.

Noise Removal or Image Smoothing:

For instance, noise could be such that it never even gets close to the neighboring pixel than tolook like it. This may mean that the detection of edges will not be accurate or correct. To avoid the same, the Gauisian filter, which is ordered together with the picture and removes noise, the desired edges in the output images are not allowed.

For instance in this example, we convole the image I with a gaussian filter kernel, or G(x;,y).In this specific instance, we use the matrix [1 1 1] to maintain closeness among pixels and remove noise so as to ensure that for each individual pixel must equate with its closely surrounding pixels in the outcome.

$S=1*g(x,y) = g(x, y) * I$

Compute the filter's differential in x and y directions to obtain its gradient's magnitude. This is achieved by dividing the former quantity by I. Moreover, taking into consideration the tangential angle between x and y dimensions can help in computing image orientation.

$$\nabla S = \nabla(g * I) = (\nabla g) * I$$

$$\nabla S = \begin{bmatrix} g_x \\ g_y \end{bmatrix} * I = \begin{bmatrix} g_x * I \\ g_y * I \end{bmatrix}$$

$$\nabla g = \begin{bmatrix} \frac{\partial g}{\partial x} \\ \frac{\partial g}{\partial y} \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

Fig 15: convolution results in a gradient vector that has magnitude and direction.

Here is an illustration of how Gaussian Derivatives contribute to the edges in the finalimages.

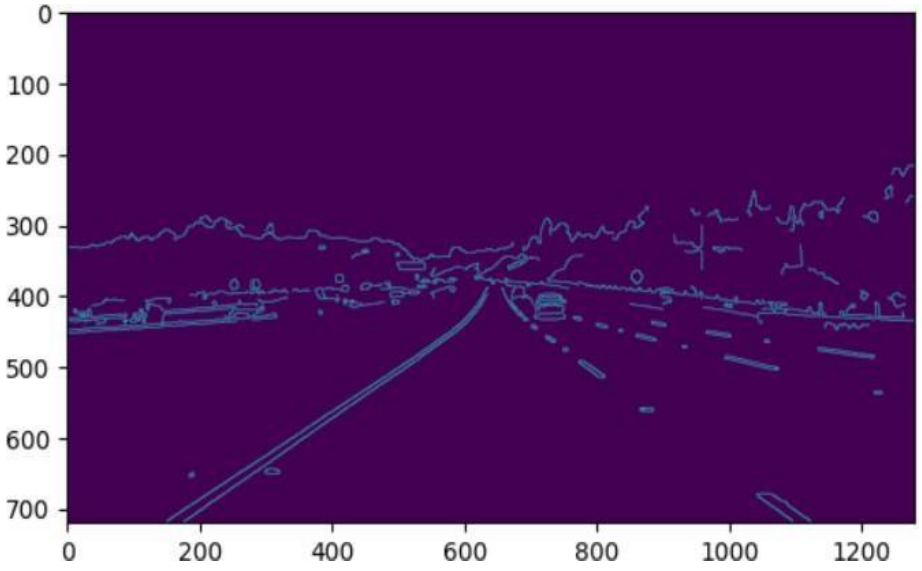

Fig 16: Edge Detection

3.2.2 Non-Max Suppression

Generally, it can be observed that only a couple points of an edge heighten edge visibility. Assuch, we can ignore features in the edge positions as these do not result in an increased visibility of these features. The task is achieved through use of the Non Maximum Suppression. Here in these points on the curve of the edge where the magnitude is the highestis denoted. This is evident in locating a maximum of and a perpendicular slice.

In the figure below, look at the edge that has three edge points. Let us assume that, point (x,y) possesses maximum gradient for an edge. Seek horizontal or even edge points with a slump which is less than (x,y).

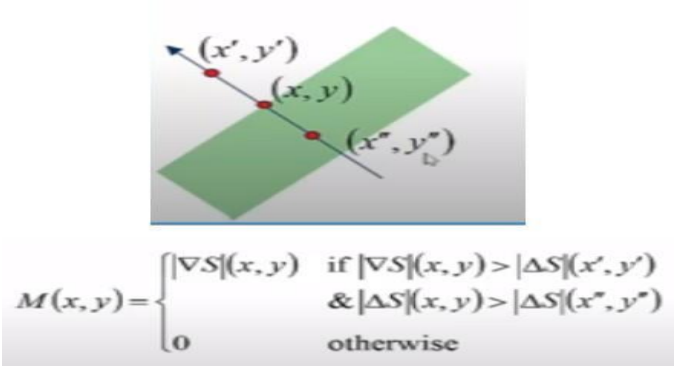

Fig 17: Non-Maxima Points along the curve

Hysteresis Thresholding

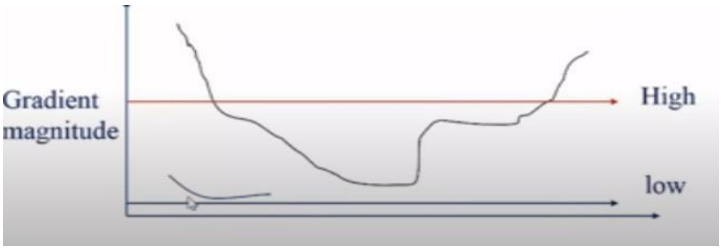

Fig 18: Hysteresis Thresholding

Hough transform is a characteristic extraction technique within the evaluation of pictures. The Hough transformation can isolate the functions of any ordinary curve, along with lines, circles, ellipses, and many more. The Hough rework can apply to the most simple of models so as to find strains that are directly in an image. The generalised Hough rework may be used in cases where an exact analytic form of capabilities is not believable. This is what makes people normally shun away from this method despite its simplicity, because it has a computational problem. Moreover, mastering-based algorithms can extract complex elements from an image or a number of images which are more suitable to the problem in question.

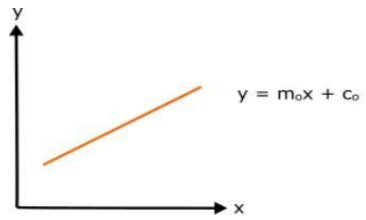

Fig 19: Image Space

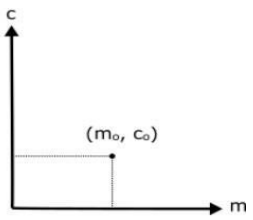

Fig 20: Hough Space

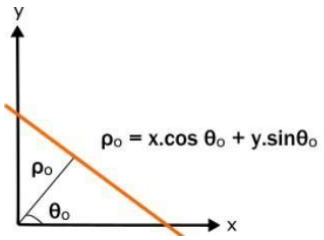

Fig 21: Image Space

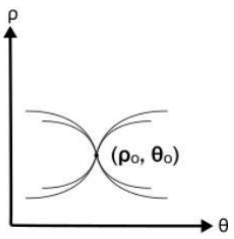

Fig 22: Hough Space

ρ is the length of the section and the angle θ made with the x-axis is the position of theline. The lines in question will be converted to points by a hough space of the form (ρ,θ).

For each parameter combination, the number of nonzero pixels all from the image input areshown for each parameter combination, and, and at (ρ,θ). The array is incremented accordingly

Intuition for line detection

The intersection of different strains in photo space corresponds to the intersection of numerous factors in 3-D space.

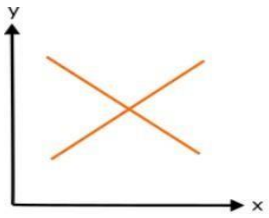

Fig 23: Image Space

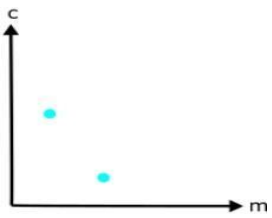

Fig 24: Hough Space

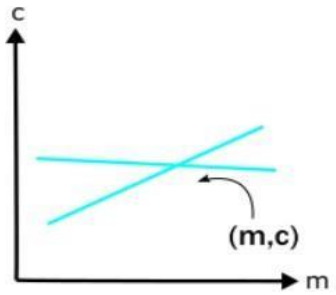

Fig 25: Hough Space

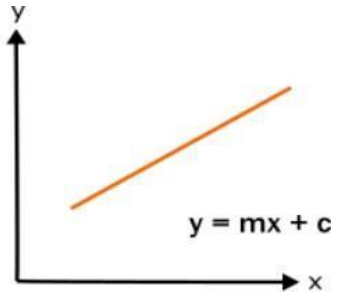

Fig 26: Image Space

When a line in an image is made up of many points that follow a similar equation, the result is a complex network of intersecting lines in a high-dimensional space.

Now, imagine a line in a picture that is slightly interrupted and has been detected as an edge. By converting this disrupted line from the image space to the hough space, we can locate the intersecting points and establish the continuous line in the image.

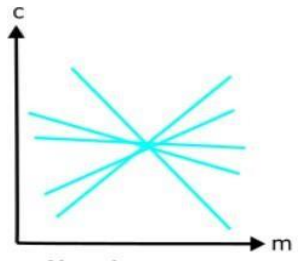

Fig 27: Hough Space

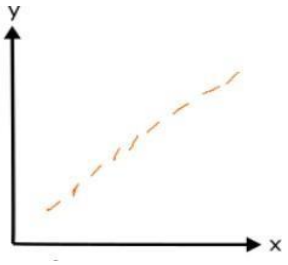

Fig 28: Image Space

Mathematical Analysis

Cartesian coordinate system: Parameters: (m,b)

Polar coordinate system: Parameters: (r,θ)

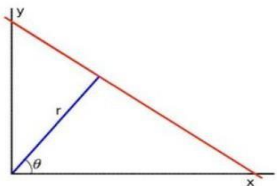

Fig 29: Hough Transformation

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right)x + \left(\frac{r}{\sin\theta}\right)$$

Fig 30: Polar System

To organize the terms, we can write the equation as r = xcosθ + ysinθ.

1. In general, for any given point (x0, y0), we can determine a family of lines passing through that specific location using the equation:

rθ = x0 · cosθ + y0 · sinθ

This means that each pair (rθ, θ) corresponds to a different line passing through (x0, y0).

2. If we graph the family of lines passing through a given point (x0, y0), we will see a sinusoidal pattern. For example, if x0 = 8 and y0 = 6, the graph will be in the θ - r plane and look something like this:

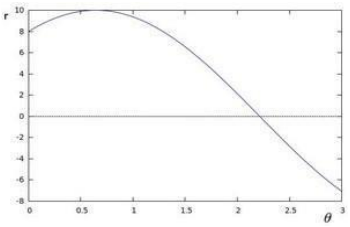

Fig 31: Sinusoidal

3. Just like that, we can use the same technique on all the points that make up that particular photo. In the coordinates system(θ-r) if the two curves of two different points come together, then both points lie along the same line.

4. Hence, in general, a line is the position where one calculates the amount of intersections of curves. The more curve intersecting points, the higher the line represented by that point. Therefore it is possible to specify a general min. number of intersections, above which a line should be visible.

5. This is implemented by the Hough Line Transform. It has a trace record of where each image line meets the curves at various points. If the number of crossings is greater than a certain limit, the line is considered as an intersection point parameters (,r). Standard and Probabilistic Hough line transformation.

3.2 Project Design and Architecture

Project Design for Image as an Input

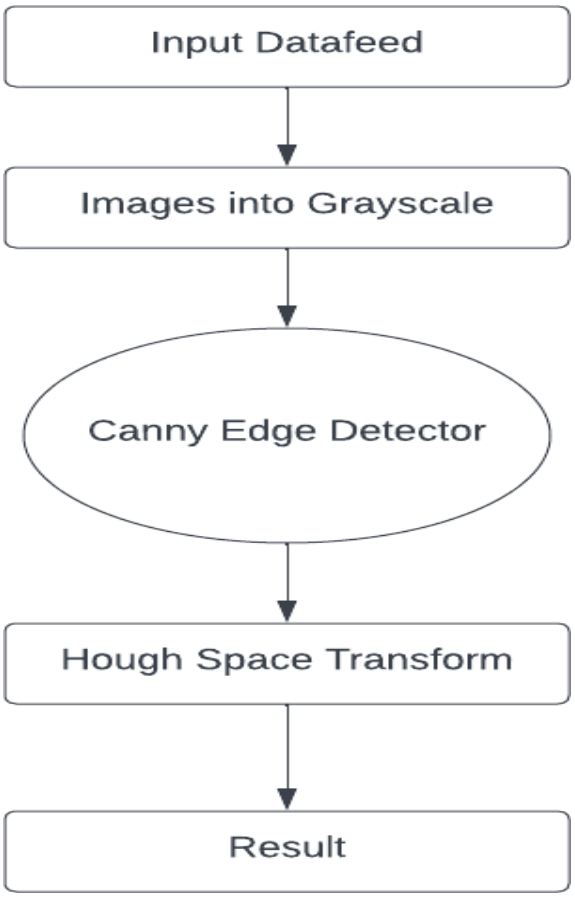

Fig: 32: Flow Diagram 1

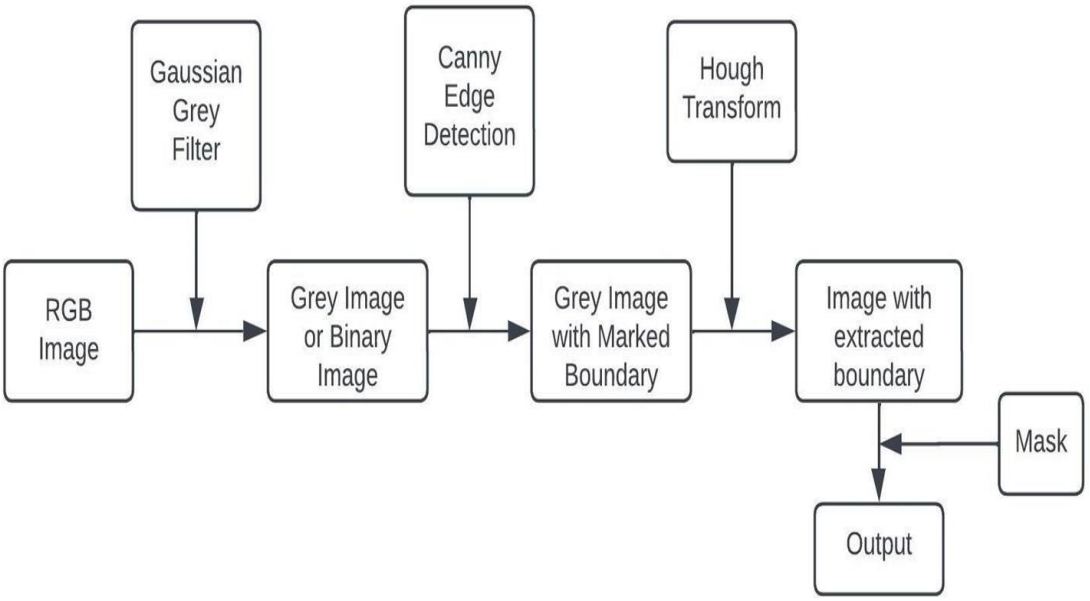

Fig: 33 Flow Diagram 2

Project Design for Video as an Input

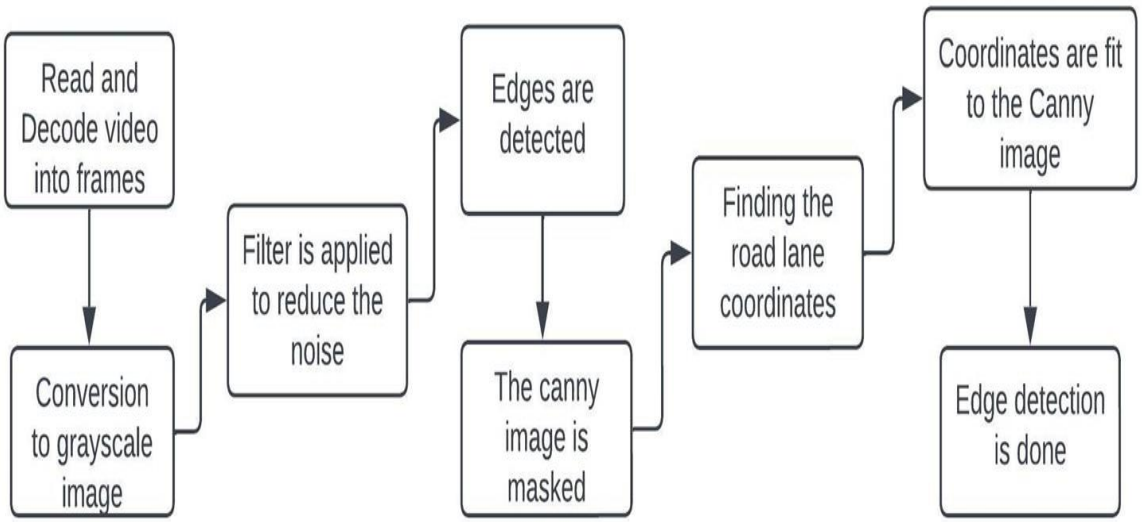

Fig: 34 Flow Diagram 3

Model Representation

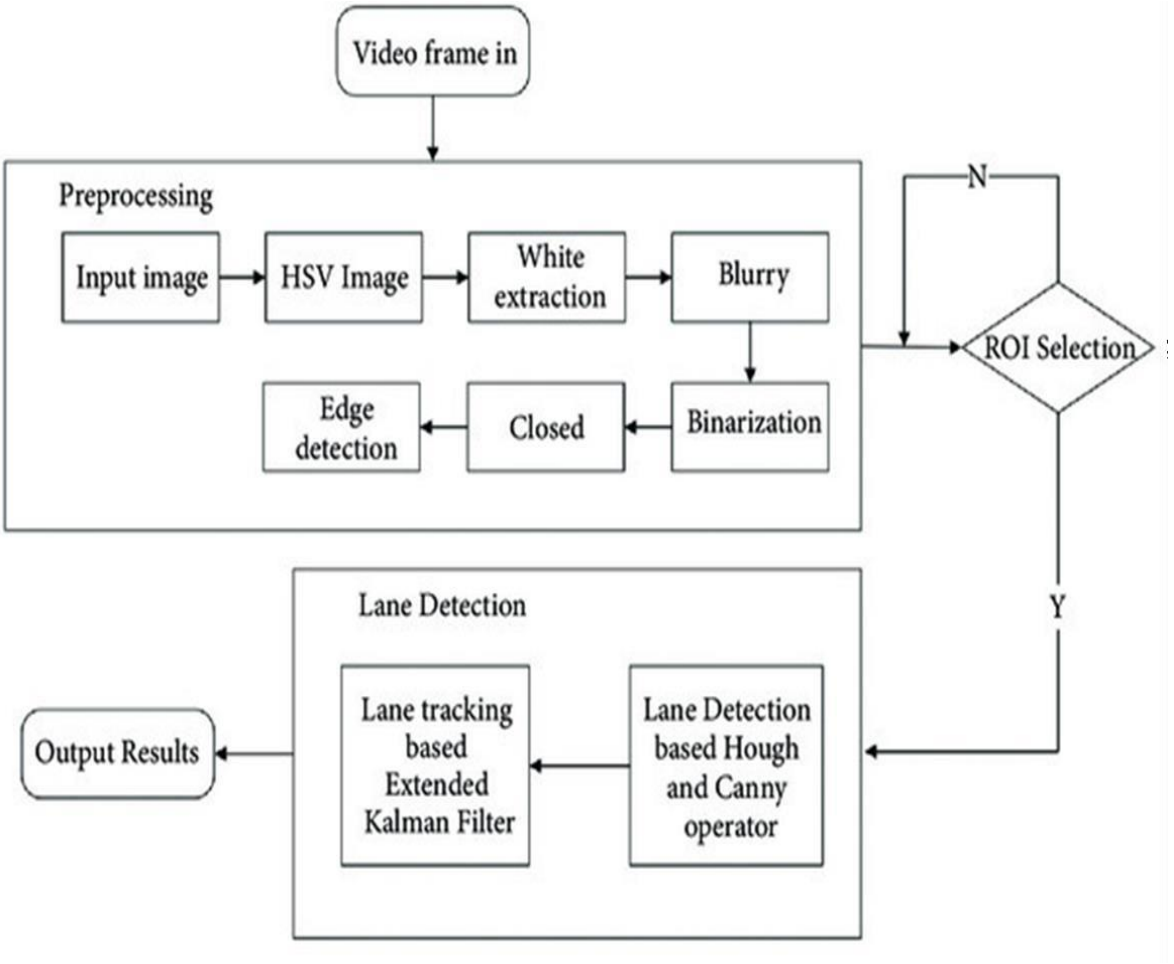

Fig: 35 Flow Diagram 4

3.3 Data Preparation

Behind the data-acquisition vehicle's glass are three strategically placed cameras that capture the human driver's steering angle from a time-stamped video. This vital piece of information is obtained through the vehicle's Controller Area Network (CAN) bus. In orderto ensure compatibility with any car model, the team has developed a method to expressthe vehicle's driving instruction as 1/r, where r represents the turning radii measured in meters. This prevents a singularity when driving straight by using 1/r instead of r.

As the turning radius becomes infinite when travelling straight, 1/r approaches 0, smoothly transitioning from easy left turns (positive values) to tight turns (negative values). The finalsteering instruction, a combination of a few select video images and training data, is represented by 1/r.

5 distinct driving films were used to create the training images:

1. In just 221 seconds, the sun shone brightly, constantly changing the lighting around us. The beginning had some exciting turns, while the final stretch posed a challenge with unexpected turns and merging lanes on the road.

2. The road lacks shoulder lines and eventually merges lanes after nearly 800 seconds on the divided highway. Along the way, one may encounter varying shadows, a green traffic signal, and direct sunlight, making for interesting driving conditions.

3. Experience a quick roundtrip journey of 3.99 seconds across the elevated section of a divided highway.

4. Prepare for a challenging training session on a two-lane road with a guardrail that may initially pose difficulties due to varying shadows, but eventually everything will go back to the usual routine.

5. Cruise through a split multi-lane motorway with moderate traffic in just 371 seconds.

Fig 36: Training dataset

3.4 Implementation

In order for the Canny Edge detector to function properly, it is necessary to transform our image into grey scale. This involves merging the Red, Green, and Blue pixels into a single channel, with each pixel's value falling within the range of 0 to 255.

```
import cv2 as cv import numpy as np
import matplotlib.pyplot as plt


from google.colab.patches import cv2_imshow from google.colab import files
files = files.upload()


img = cv.imread("img.png")
gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY) plt.imshow(gray)
plt.show()
```

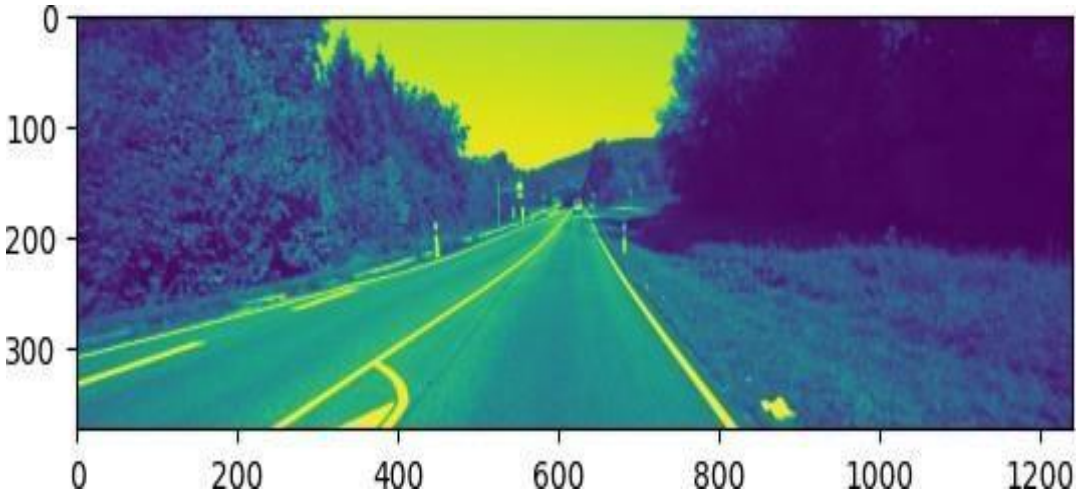

Fig 37: Preprocessed image (Grey scale)

blur = cv.GaussianBlur(gray, (5, 5), 0) plt.imshow(blur, cmap ="gray") plt.title("GaussianBlurr"), plt.xticks([]), plt.yticks([]) plt.show()

Fig 38: Gaussian Blur

edges = cv.Canny(blur, 50, 150) plt.imshow(edges)
plt.show()

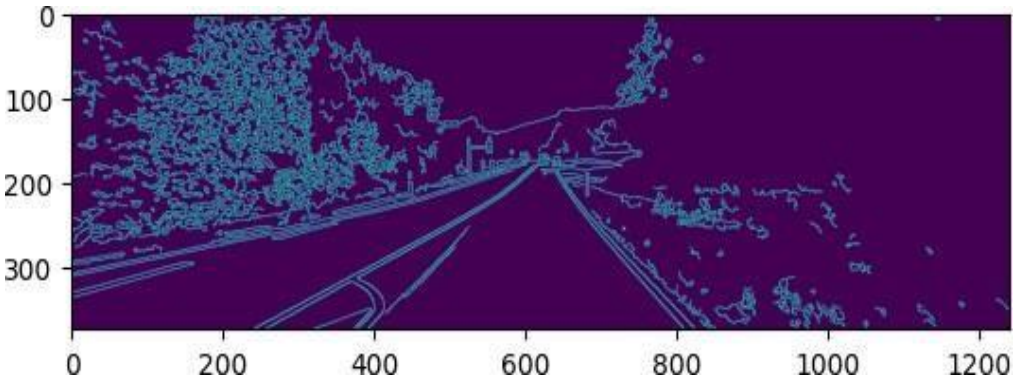

Fig 39: Canny edges image output

mask = np.zeros_like(edges) plt.imshow(mask) plt.show()

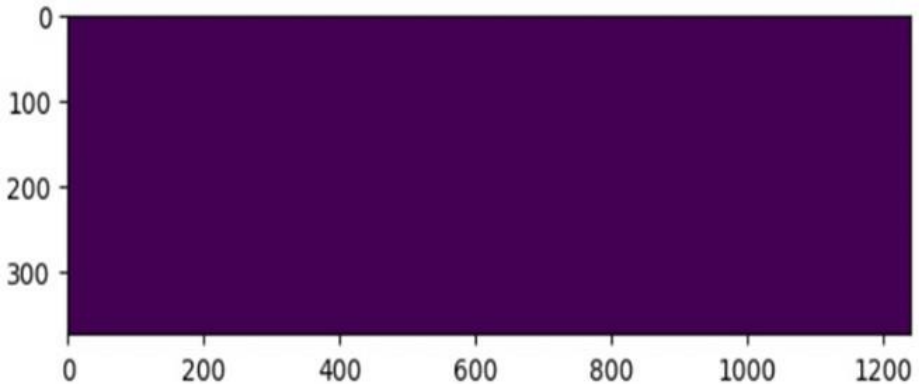

Fig: 40 Mask

height, width = img.shape[:2]
roi_vertices = [(0, height), (width/2, height/2), (width, height)] mask_color = 255
cv.fillPoly(mask, np.array([roi_vertices], dtype=np.int32), mask_color) masked_edges = cv.bitwise_and(edges, mask)

lines = cv.HoughLinesP(masked_edges, rho=6, theta=np.pi/60, threshold=160, minLineLength=40, maxLineGap=25)
plt.imshow(lines) plt.show()

line_image = np.zeros_like(img) for line in lines:
x1, y1, x2, y2 = line[0]
cv.line(line_image, (x1, y1), (x2, y2), (0, 255, 0), 5)

plt.imshow(line_image) plt.show()

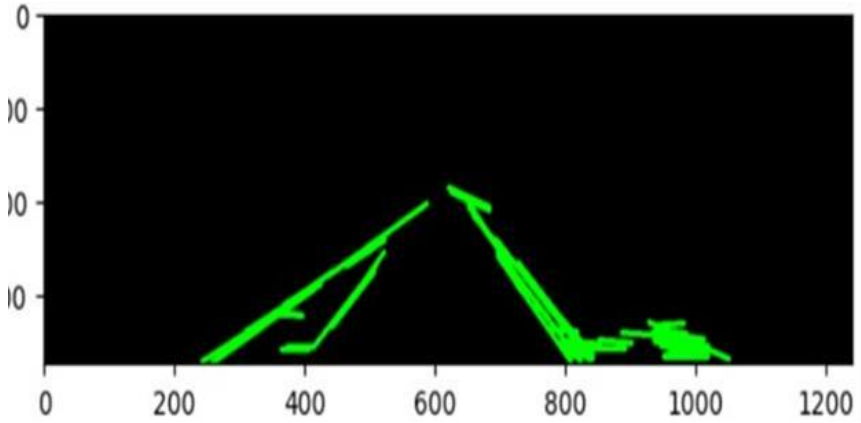

Fig: 41 Hough lines

final_image = cv.addWeighted(img, 0.8, line_image, 1, 0) plt.imshow(final_image) plt.show()

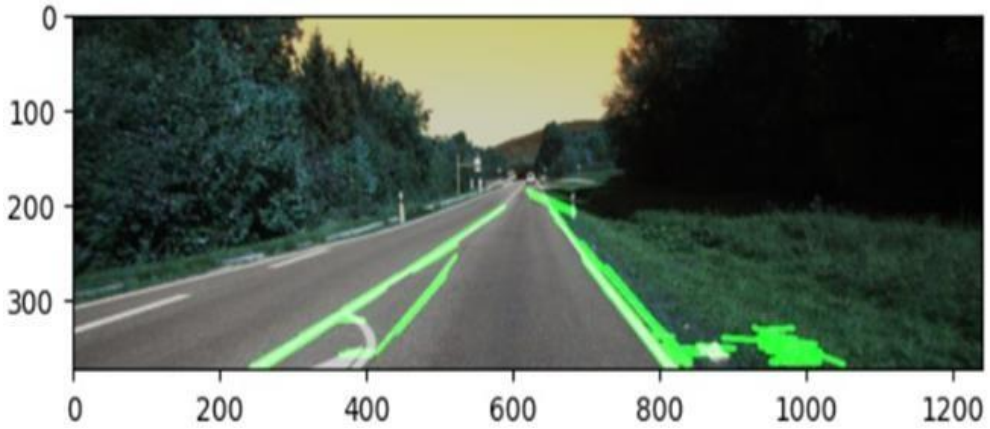

Fig: 42 Final Output 1

CODE FOR VIDEO AS AN INPUT

```
import numpy as np import cv2 as cv

def findIntersection(x1,y1,x2,y2,x3,y3,x4,y4):
    px= ( (x1*y2-y1*x2)*(x3-x4)-(x1-x2)*(x3*y4-y3*x4) ) / ( (x1-x2)*(y3-y4)-(y1-y2)*(x3-x4) )
    py= ( (x1*y2-y1*x2)*(y3-y4)-(y1-y2)*(x3*y4-y3*x4) ) / ( (x1-x2)*(y3-y4)-(y1-y2)*(x3-x4) )
return [px, py]
```

```
def draw_all_lines(img, lines, color=[255, 0, 0], thickness=7): for line in lines:
for x1, y1, x2, y2 in line:
cv.line(img, (x1, y1), (x2, y2), color, thickness)
```

```
def draw_lines(img, lines, color=[0, 0, 255], thickness=7): global
    Old_pts,result,distfromcenter
x_bottom_pos = [] x_upper_pos = [] x_bottom_neg = [] x_upper_neg = []

y_bottom = 740
y_upper = 215
```

```
slope = 0
b = 0
if lines is not None: for line in lines:
for x1,y1,x2,y2 in line:
# test and filter values to slope
if ((y2-y1)/(x2-x1)) > 0.5 and ((y2-y1)/(x2-x1)) < 0.8:

slope = ((y2-y1)/(x2-x1)) b = y1 - slope*x1

x_bottom_pos.append((y_bottom - b)/slope) x_upper_pos.append((y_upper - b)/slope)

elif ((y2-y1)/(x2-x1)) < -0.5 and ((y2-y1)/(x2-x1)) > -0.8: slope = ((y2-y1)/(x2-x1))
b = y1 - slope*x1

x_bottom_neg.append((y_bottom - b)/slope) x_upper_neg.append((y_upper - b)/slope)
```

 # To be used for transparency when drawing the road

```
overlay = img.copy()
```

```
alpha = 0.3  # Transparency factor.


try:
# Find intersection
intersect_points=findIntersection(int(np.mean(x_bottom_pos)), int(np.mean(y_bottom)),
int(np.mean(x_upper_pos)), int(np.mean(y_upper)),
int(np.mean(x_bottom_neg)), int(np.mean(y_bottom)), int(np.mean(x_upper_neg)),
int(np.mean(y_upper)))
print("Intersect :") print(intersect_points)
# a new 2d array with means
lines_mean = np.array([[int(np.mean(x_bottom_pos)), int(np.mean(y_bottom)),
int(np.mean(intersect_points[0])), int(np.mean(intersect_points[1]))],
[int(np.mean(x_bottom_neg)), int(np.mean(y_bottom)), int(np.mean(intersect_points[0])),
int(np.mean(intersect_points[1]))]])



# Draw the road path
for i in range(len(lines_mean-1)):

pt1 = (lines_mean[i, 0], lines_mean[i, 1])
pt2 = (lines_mean[i+1, 0], lines_mean[i+1, 1])
pt3 = (lines_mean[i, 2], lines_mean[i, 3])

midposX = int(abs(int(pt1[0]-pt2[0]))/2) center = int(img.shape[0])/2 distfromcenter = center-
midposX print("distfromcenteren cm :") print(distfromcenter)




# draw a triangle
vertices = np.array([pt1, pt2, pt3], np.int32) pts = vertices.reshape((-1, 1, 2))
Old_pts = pts

cv.polylines(overlay, [pts], isClosed=True, color=(255, 255, 255), thickness=5)
cv.fillPoly(overlay, [pts], color=(255, 0, 0))



except:
# draw a triangle
if Old_pts.any():
```

```
cv.polylines(overlay, [Old_pts], isClosed=True, color=(255, 255, 255), thickness=5)
cv.fillPoly(overlay, [Old_pts], color=(255, 0, 0))

   result = cv.addWeighted(overlay, alpha, img, 1 - alpha, 0) else:
pass
```

 # Video Capture

```
cap = cv.VideoCapture("project_video.mp4") global Old_pts,result,distfromcenter
Old_pts = np.array([]) distfromcenter = 0
while (cv.waitKey(10)<0):
# Capture frame-by-frame ret, frame = cap.read() result = frame
# if frame is read correctly ret is True if not ret:
print("Can't receive frame (stream end?). Exiting ...") break
```

 # Our operations on the frame starts here
 # Here goes the treatement Canny Edge Detector & Hough Line Transform

```
grayscale = cv.cvtColor(frame, cv.COLOR_BGR2GRAY) kernel_size = 5
blur = cv.GaussianBlur(grayscale, (kernel_size, kernel_size), 0) low_t = 50
high_t = 200
edges = cv.Canny(blur, low_t, high_t)

# Create a set of vertices for the mask height = frame.shape[0]
width = frame.shape[1]
   vertices = np.array([[(0,height),(5*width/10,6*height/10),(width,height)]], dtype=np.int32)


mask = np.zeros_like(edges)
cv.fillPoly(mask, vertices, color= (255,255,255))
```

```
masked_edges = cv.bitwise_and(edges, mask)

linesP = cv.HoughLinesP(masked_edges, 1, np.pi / 180, 50, None, 125, 60)
draw_lines(frame,linesP)


# Display the resulting frame
cv.imshow("Road Detection", result)

if cv.waitKey(1) == ord('q'): break
# When everything done, release the capture cap.release()
cv.destroyAllWindows()
```

Fig: 43 Final Output 2

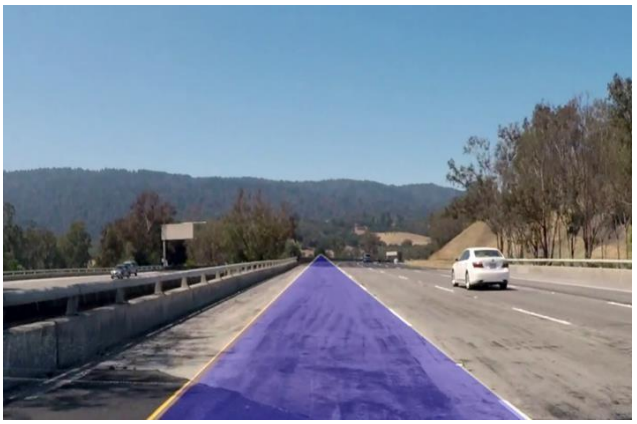

Fig:44 Final Output 3

PSEUDOCODE FOR CANNY EDGE DETECTION ALGORITHM

```
function CannyEdgeDetection(image)

#Step1: Noise reductionwith Gaussian filter smoothed_image =
        ApplyGaussianFilter(image)

# Step 2: Find image gradients gradient_magnitude, gradient_direction =
CalculateImageGradients(smoothed_image)

# Step 3: Non-maximum suppression
        thinned_edges = NonMaximumSuppression(gradient_magnitude, gradient_direction)

# Step 4: Double thresholding
edges = DoubleThresholding(thinned_edges)

# Step 5: Hysteresis thresholding
final_edges = HysteresisThresholding(edges)

  return final_edges function ApplyGaussianFilter(image) # Implement Gaussian filter

convolution here
# This function takes the image and kernel size as input
# and returns the smoothed image return smoothed_image

function CalculateImageGradients(image)
# Sobel filter can be used here for gradient calculation # This function takes the smoothed
image as input
# and returns the gradient magnitude and direction images return gradient_magnitude,
gradient_direction

function NonMaximumSuppression(gradient_magnitude, gradient_direction) # Iterate through
  each pixel and compare with neighbors based on direction # Set non-maximum pixels to
  zero
return thinned_edges
function DoubleThresholding(thinned_edges, threshold1, threshold2) # Set pixels below
   threshold1 to zero # Keep pixels above threshold2
# Set pixels between thresholds to zero only if not connected to a strong edge return edges
function

HysteresisThresholding(edges)
# Track edges by following strong edges and connecting weak ones # Suppress weak edges
not connected to strong edges
return final_edges
```

DIGITAL IMAGE PROCESSING TOOLS

1 GOOGLE COLABORATORY

One is Colaboratory, a product of Google Research also known as “Colab”. Colabs most appropriate for machine learning, data analytics, and education. With this, one can run any python code using the browser. In simple terms, Colab is an online platform which provides users with GPUs at no cost whatsoever without any registration beforehand.

What a user consumes through Colab is not always guaranteed or unlimited, while a user’s limits for consuming change sometimes. This has got to be in place if Colab has any hope of offering its resources free. Unlike other cloud platforms, Colab directs its resources towards the more active use cases. We do not allow any activity involving group-computing with malicious consequences and actions, which avoids our rules. The following are disallowed from Colab runtimes:
provide web services such as file hosting, media serving, or other non-interactive compute with Colab.
sharing files via p2p and torrent download.
such as SSH shells, remote desktops or remote UIs.

One can load colab notebooks through GitHub or store them on Google Drive. Just like sharing Google Doc and Sheets, Colab notebook too can be shared.

Code runs on a dedicated virtual machine only for your account.') The Colab service stipulates that virtual machines should not exceed a specified period before they are removed

after lying dormant for a while thus, it is emphasized on interactivity in computations. Runtimes will time out due to inactivity.

According to demand and supply patterns, the free version of Colab's notebook can be run fortwenty four hour. Colab Pro, Pro+, or Pay as you go options offer more compute availability depending on the remainder of your credits. Operational life per charge for notebooks is on average 12 hours based on usage and availability profile. If you consume all your assigned compute sessions in Pro, Pro+ or Pay as you go plan you may expect backend termination.

As long as your processing power is sufficient, colab pro + offers continuous code execution for up to 24 hours nonstop. Idle timeouts work after code has been executed. Colab provides free virtual machines having standard system memory profile which you can use freely. Memory systems that have a high profile are allowed on computers you can use under paid edition of Colab in accordance with available capacity and your computing unit level. By memory, we mean the computer's memory. However, all GPUs share the similar memory profile.

2 DIGITAL IMAGE PROCESSING

Image processing is a process whereby several approaches are applied on a picture either for enhancement, or to extract pertinent data from it. In this case, the input is a picture, while the output may be another picture, or it can also contain other features or qualities related to the original image. Image processing innovates quickly among one of them.

These visual techniques are employed by image analysts and they use different interpretational fundamentals to derive meaning out of them. Computer-based digital image editing is achieved using digital image processing technologies and their implementation in digital image processing software.

Sampling and quantization

As we construct a digital image, we have to digitise continuum information. These require two steps to complete it:

- Sampling
- Quantization

The sampled size of the image in pixels in image processing is a digital value. The digital equivalents of image functions are obtained through quantization.

The quantizations level should have been high enough for the fine shading features in the image to have been perceivable by humans. If a quantized image has too many brightness levels, it will end up having phantom outlines that were originally absent.

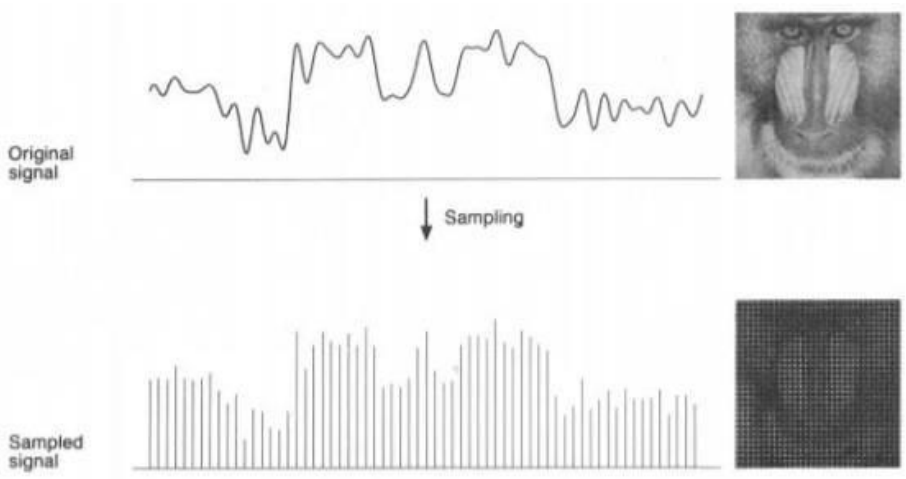

Fig 45: Sampling of an Image

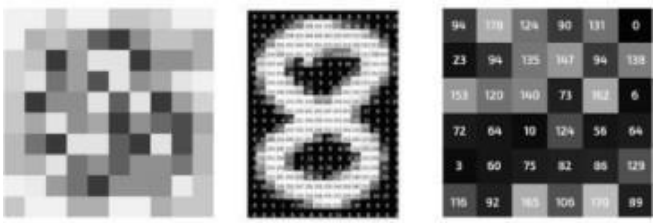

Fig 46: Quantization of an image of level 256

Resizing the Image

picture interpolation is the act of either resizing or warping one picture from a single pixel grid to another. The correction for lens distortion and rotating an image allows remapping, but the whole number of pixels requires scaling. The act of zooming adds more pixels to the picture so as to reveal more details.

Interpolation estimates the unknown values through the knowledge on values for known positions. Image interpolation involves calculating the average nearest pixel in an attempt to give an estimated value of a certain pixel. It carries out its tasks in two directions. The two major kinds of adaptive and non-adaptive interpolation techniques.

3 PYTHON FOR DIP

Several libraries exist in Python which support image processing like
OpenCV is an open library of algorithms designed for real time computer vision for image processing which is used in various applications like object detection, mobile robotics, 2D/3D feature toolkit, face and gesture recognition, human machine interface, etc.

- Use the Numpy and Scipy for image processing and manipulation.
- Sckikit is a host of numerous image processing algorithms.
- PIL (Python Imaging Library) – basic image manipulations such as thumbnails, resize, rotate, format convert.
- A 2D function of F(x,y) that describes a picture with the spatial coordinates of x and y. Brightness of an image at some location x,y is determined by its peak to trough of F. In this case, we call this object as a digital image if x, y, and amplitude are all finite. Pixels are arranged in a row-and-column patternin an array. Pixels are individual points of an image which have information on color and level of

brightness. In a three dimensional form of images X, Y and Z is transformed to spatial coordinates. The pixels are displayed such that they resemble a matrix. It is referred to as an RGB picture.

Images come in a variety of forms:

- The two dimensional RGB image is made up of these three layered sections;red, green and blue bands.
- The images are monochrome and have a variety of gray tones.
- Classic Image Processing algorithms include:

Morphological Image Processing

- Due to straight forward thresholding damaging binarised regions from noise; Morphological imaging is used to clean out noise in the binary regions from the images. It also employs opening and shuttering techniques to bring the image into focus.
- Morphological operations can be extended to grayscale pictures too. The organization features characteristic interrelationships including non-linear processes. The ordering matters too, not just the numbers on those pixels. In this method, one uses a small template (the "structuring element") which is placed in different possible locations in the picture and compared with related neighbourhood pixels. listade2

Gaussian Image Processing

- Applying a Gaussian function to a photo is called "Gaussian blur". It is usedto reduce the distracting details and the surrounding visual clutter. Thus, theblurring associated with this type mimics the visual effect of looking at an image in a transparent window. It can serve as a method of supplementing information during deep learning and improve images at different scales within the framework of computer vision.

Fourier Transform in image processing

- The Fourier transform allows for an input image to be broken up into cosineand sine components.

This tool is very useful in various applications like image filtering, imagecompression, and image reconstruction. Talking about the image, we will consider the discrete Fourier transform.

Edge Detection in image processing

- Edge detection is a process to identify the outlines of objects on photos. It works by detecting changes in the light level. This way can also help gain valuable information from the image considering that shape has most of its data found on the edge. Traditional edge detectors reveal brightness discontinuities. It can discover some levels of noise upon finding this in a picture and respond quickly when identifying grey level adjustments. The edges are also known as the local gradient maxima.

4 IMAGE PROCESSING LIBRARIES

OpenCV

- The abbreviated form is Open Source Computer Vision Library (OpenCV).This library has over 2000 optimized algorithms.
- For example converting images between color spaces such as BGR tograyscale, BGR to HSV, etc.
- Basic thresholding and adaptive thresholding in picture data application.
- Image processing tools include blurring and use of custom filters on thephotos.
- Morphological manipulations of pictures.
- Building pyramidal images.
- Foreground extraction from images using the GrabCut algorithm.
- Watershed algorithm image segmentation.

Scikit-image

- It is a free source preparation image library. It can perform many complex image operations via machine learning with only a few built- in functions.
- It is also easy for those new to Python, and uses numpy arrays. Among the operations that scikit image may perform are:
- Implement thresholding operations on the picture using the try_all_threshold() method. This study will employ seven international thresholding methods. The filters module contains this.
- Perform edge detection using sobel() as part of the filters module. To begin,we need to convert an image into grayscale since the former requires a two-dimensional grayscale picture as an input.
- You should do gaussian smoothing by using the gaussian() function.
- Rotate an image utilizing rotate() function from transform module.
- Rescale the image using the rescale() method in the transform module.
- Incorporate the binary_erosion() and binary_dilation() functions from the morphology module to undertake morphological operations.

NumPy

- The same library lets you do some basic photo manipulation exercises such as image flipping and feature extraction.

- They are called NdArrays because they consist of Numpy multidimensional arrays which are used to represent images. Colour image is a three-dimensional numpy array. The multidimensional array divides theRGB channels.

- Performing the following operations on the image could yield useful results (NumPy loads the image in a variable called test_img through the imread function).xbörd: The operations could produce valuable results after one applies them to an image using the software, NumPy (a variable referred to as test
- Flip the image such that it is upside down using np.flipud(test_img).
- Flip the picture horizontally using np.fliplr().

- Use test_img[: Inverting the image is performed by flipping it (since the array is stored as an element with name, >img_name).

3.5 Key Challenges

- **Variability in Road Conditions:**

1. Weather Conditions: Lane markings can be affected by adverse weather conditions such as rain, snow, or fog, making it challenging for the model to accurately detect lanes.

2. Day/Night Conditions: Illumination changes between day and night can impact the visibility of lane markings, requiring robust algorithms that can adapt to varyinglighting conditions.

- **Complex Road Geometries:**

Curves and Intersections: Roads are not always straight, and they often include curves and intersections. Designing a model that can accurately detect and predict lanes in complex road geometries is a significant challenge.

- **Lane Marking Variability:**

1. Faded Markings: Lane markings may be faded or partially obscured, making it difficult for the model to identify and track them accurately.

2. Temporary Markings: Construction zones or temporary road markings may confuse the lane detection system, necessitating the ability to distinguish between permanent and temporary markings.

- **Diverse Vehicle and Camera Configurations:**

1. Vehicle Positioning: The position and orientation of the cameras on different vehicle models may vary. Designing a model that can handle these

variations is crucial for general applicability.

2. Camera Calibration: Accurate calibration of the cameras is essential for precise lane detection. Changes in camera angles or misalignments can affect the model's performance.

- **Real-time Processing Requirements:**

Low Latency: Autonomous vehicles require real-time processing to make split-second decisions. Developing a deep learning model that can perform lane detection with low latency is a critical challenge.

- **Data Annotation and Collection:**

1. Large and Diverse Datasets: Training deep learning models for lane detection requires large and diverse datasets. Annotating these datasets accurately can be time-consuming and expensive.

2. Labeling Challenges: Properly annotating challenging scenarios such as ambiguous orchanging lane markings is a non-trivial task.

- **Robustness to Obstacles and Occlusions:**

1. **Obstacle Handling:** The presence of other vehicles, pedestrians, or obstacles canobstruct the view of lane markings. The model needs to be robust enough to detect lanes in the presence of such occlusions.

Dynamic Objects: Handling dynamic objects like moving vehicles that may occludethe lane markings is a challenging aspect.

- **Generalization to Different Regions and Countries:**

Global Applicability: Lane markings can vary between regions and countries. Creating a model that can generalize well across different road systems, signage, and cultural differences is a significant challenge.

CHAPTER – 4

TESTING

TESTING STRATEGY

The primary objectives of the testing strategy are to:

1. Validate the performance of Hough Transform and CNN object detection andclassification on self-driving cars scenarios.

2. Study on precision versus efficiency, as well as communication charges in varied federated training settings.

3. Verify whether the privacy-preserving approaches and data protection mechanisms areproperly functional.

Testing Approach: The testing procedures will comprise both a simulation as well as aconventional testing technique.

1. Simulation Testing

- **Simulated Data:** Use simulated self-driving car data based on realistic scenarios andobject distributions.

- **Model Evaluation:** Test the performance of Hough Transform and CNN in identification and classification of objects under ideal circumstances by applying them tosimulated data.

- **PerformanceAnalysis:** Examine the efficiency of concerning model convergence, communication load, and generalization competences.

2. Real-world Testing

- **Real-world Data:** Acquire real-world self-driving car data from different sensors such ascameras, radar, lidar. 32

- **Privacy Validation:** Inform information sharing patterns to support monitoring andenforcing of privacy preservation and data security guidelines.

Testing Metrics:

The following metrics will be used to evaluate the performance of the object detection system

- **Object Detection Accuracy:** Different values of precision, recall, and mAP in some ofthe object classes.

- **Communication Overhead:** Network traffic and round-trip time during federated training rounds.

- **Privacy Preservation:** Privacy – preserving mechanisms to protect sensitive data.

Testing Plan:

The testing plan will be divided into phases:

- **Unit Testing:** The correctness and functionality of individual components of this systemsuch as data preprocessing modules, model training pipelines, and federated learning algorithms will also be evaluated.

- **Integration Testing:** Interactions between elements, as well as consistency in datatransfer, will be ensured by carrying out integration tests.

- **System Testing:** The whole system will conduct testing in a virtual environment tovalidate its overall accuracy, performance, and private preserving ability.

- **Real-world Validation:** With the use of real-word self-driven cars, the system mayfinally be validated in the light of real-cases practice and its adequacy for real life

TEST CASES AND OUTCOMES

The test cases and outcomes are as follows:

Accuracy Testing:

- The CNN exhibits high performance in detecting and classifying objects involved inself-driving cars.

- High levels of accuracy involve great communications among participants and have aresultant effect on communication overhead.

- Accuracy peaks once there are enough communication overheads.

Privacy Validation:

- The privacy-preserving methods used within the system efficiently safeguard confidentialdetails from unauthorized exposure.

- This has a negligible effect on model accuracy preserving privacy with minimum loss inperformance.

CHAPTER – 5

RESULTS AND EVALUATION

5.1 RESULTS

When an Image is given as an Input

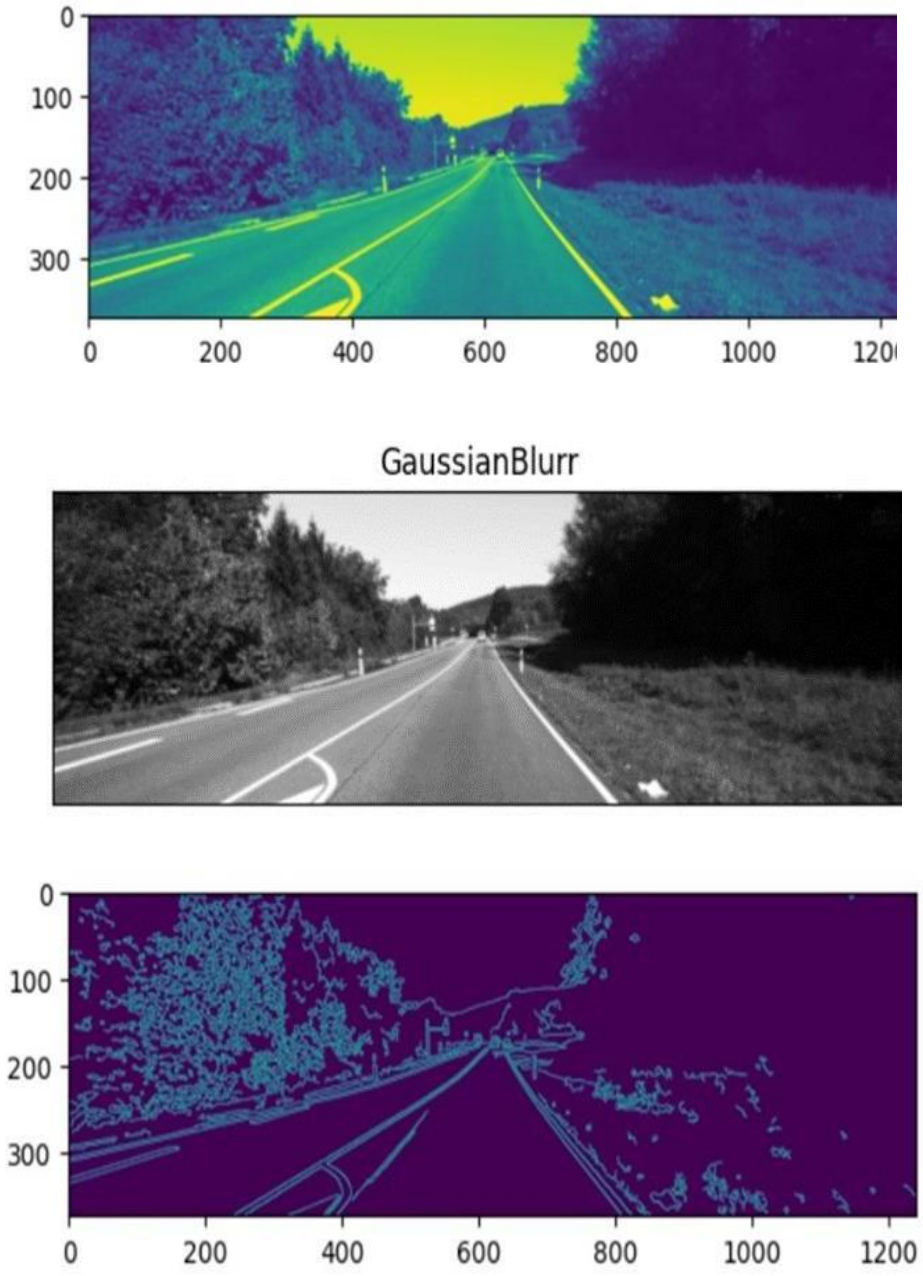

Fig: 47 Results 1

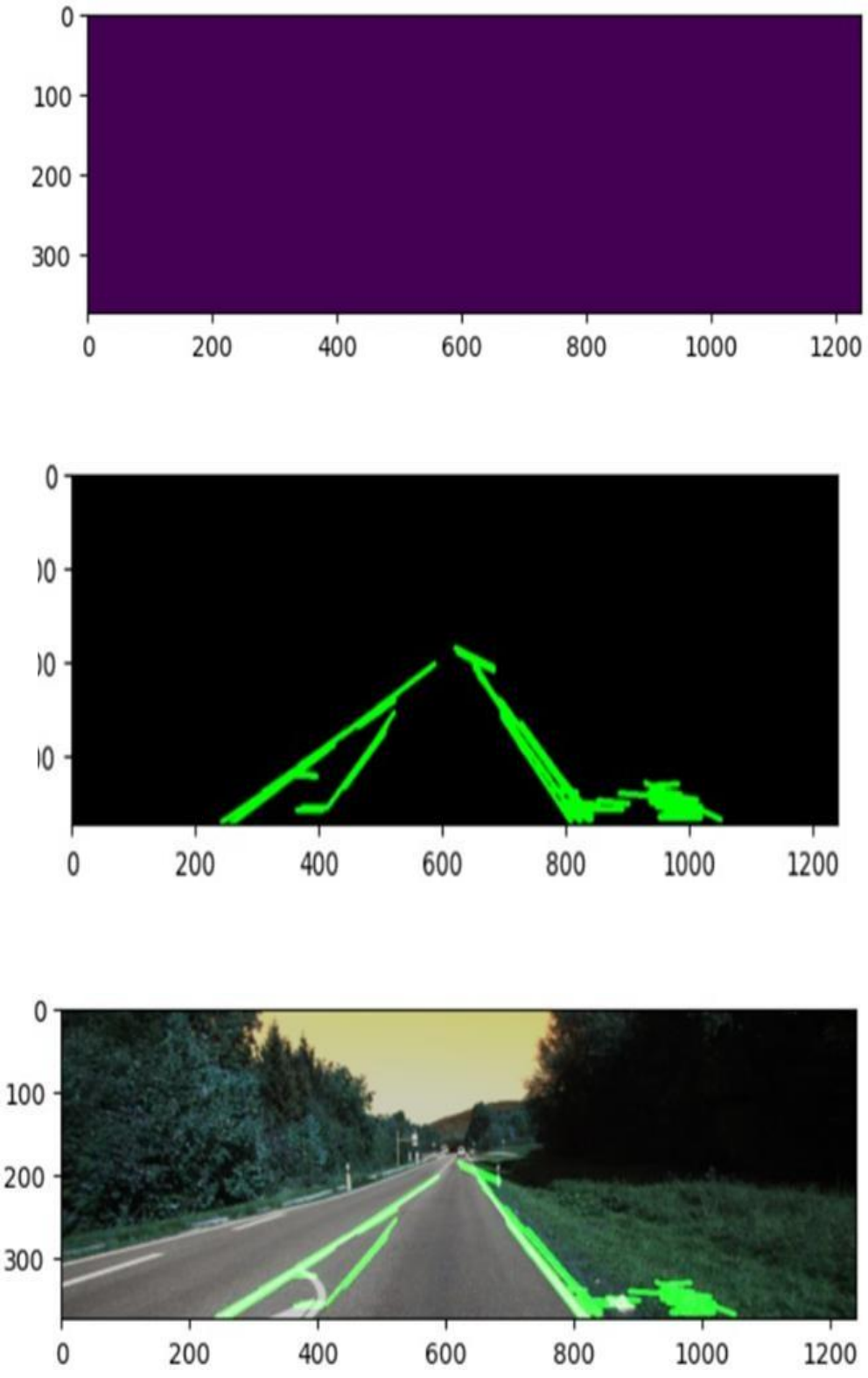

Fig: 48 Results 2

Results when Input is a Video

Fig: 49 Results 3

5.2 Comparison with existing solutions

Fig: 50 Comparison set 1

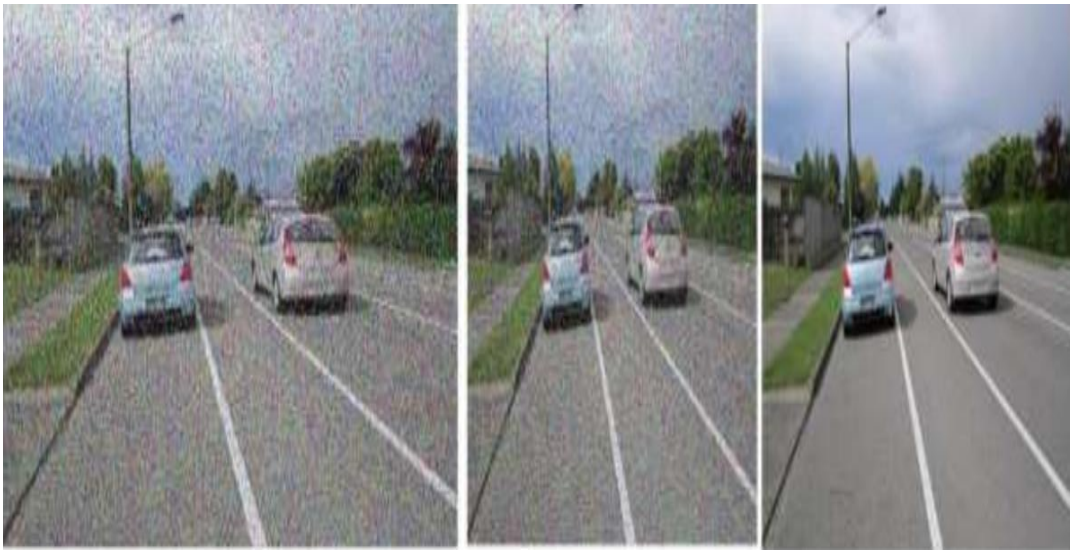

Fig: 51 Comparison set 2

CHAPTER – 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

Vision guides us in making decisions when driving. The model has marked certain elements of the road way we will use for orientation and this includes the lane markings which are our constant reference points on where to put direction of the car. Additionally, this steering is automatic. One among the first important things to do while developing a self-driving car is that it should be able to automatically detect the lane lines through an algorithm naturally. Road detection require ROI that should be adaptive.

Moving up or down a steep incline, the horizon will also not correspond to the proportions of the picture. Furthermore, this should be considered in situations featuring tight corners and crowded streets.

Extensive research, design and planning culminated with the construction of a lane detection system which would be instrumental towards the identification of lanes by autonomous driving systems whose sole purpose is to provide safe journey for the passengers in the car as well as the various categories of other road users like pedestri Clever edge detection technique is used in order that the system pre-process the image frame before it apply the Hough transform algorithm in order that it highlight the lane for the convenience of the user.

The main advantage of this approach is the ability to predict the lane layouts of the traffic flow without a trained model. It instead passively receives the video frames and actively learns of the frame boundaries that it then issues back to the user as lanes.

Due to the development of artificial intelligence and deep learning, image enhancing methods using deep convolutional neural networks are gradually

appearing on the scene. Hence this type of an approach can cater to the speed and clarity needs of processing digital images. Moreover, this shows that underground image processinghas been developing.

One can do certain things to an image so that it can be improved, or for obtaining useful information from such images. The second kind is an image-based one, in whichone may feed an image as an input and receive another photo or properties of the initialone. Currently, one of the most rapidly developing technologies is image processing.

The technique should also address ease of use and miniaturization, that is, make it easy for use to be expanded to a larger audience through its simplicity or size. Finally, image processing technology has a great societal impact and is used almost everywhere. Indeed, the area of digital image processing has many more aspects to explore and it is expected that the process will take place in a more optimistic manner for as long as human desires are growing.

6.2 Future Scope

This article presents a generic lane detecting system based on visualization, which incorporates research from multiple authors as well as test results. We only talked about the most extensively used techniques and algorithms. The importance of perception sensors, algorithms, and their integration in achieving optimal lane detectionsystem results is the key finding of this brief analysis. On the topic of trace detection, there is a lot of research going on. To reduce calculation time, cost, and improve effective perception, dual-threshold research of efficient sensor integration is necessary.The necessity of the hour is for high-security ADAS to reduce technology misuse and data theft. This model can be refreshed and tuned with more proficient numerical demonstrating, while the old style OpenCV approach is restricted and no overhaul is conceivable as the approach isn't effective. It can't give precise outcomes on the streets which do not have clear markings present on the street.

Future Explorations

Adaptive Learning Mechanisms: By introducing adaptive learning mechanisms into our lane detection system, we can improve its ability to adjust to changing environmental factors like traffic, weather, and lighting. Through constant parameter updates driven by real-time feedback, our system can become more resilient and perform better under a variety of conditions.

Multi-Agent Interaction Modeling: By integrating multi-agent interaction models, our system will be able to predict the actions of other road users, including cars, bicycles, and pedestrians. Autonomous vehicles have the ability to anticipate the movements and intentions of other agents, which allows them to navigate intricate traffic situations with greater efficiency and safety.

Cross-modal Sensor Fusion: We can improve our system's perceptual capabilities by investigating new approaches for cross-modal sensor fusion, such as merging data from cameras, LiDAR, radar, and GPS. Our system's robustness and reliability can be improved by combining data from several sensor modalities to create a more thorough understanding of the surrounding environment.

REFERENCES

[1] Tullimalli Sarsha Sree and Sandeep Kumar Sathapathy, Volume: 07 Issue: 06 | June 2020, 'Lane Line Detection using Hough Transform and Convolutional neural Networks'

[2] Nidhi Lakhani, Ritika Karande, Deep Manek, Vivek Ramakrishnan, Volume: 09 Issue: 04| Apr 2022, 'Lane Detection using Image Processing in Python'

[3] Raja Muthalagu, Anudeep Sekhar Bolimera, Dhruv Duseja, Shaun Fernandes, Volume: 22, no.4, 2021 , 'Object and Lane Detection Techniques for Autonomous Car using Machine Learning approaches'

[4] Vighnesh Devane, Ganesh Sahane, Hritish Khairmode, Gaurav Datkhile, ITM Web of Conferences 40, 03011 (2021), 'Lane Detection Techniques using Image Processing'

[5] Jamel Baili, Mehrez Marzougui, Ameur Sboui, Samer Lahouar, Mounir Hergli,2017 Second International Conference on Anti-Cyber Crimes (ICACC), 'Lane Departure Detection using Image Processing Techniques'

[6] Yan Liu , Jingwen Wang , Yujie Li , Canlin Li, Accepted: 28 April 2022 | Published: 30th April 2022 'Lane-GAN: A Robust Lane Detection Network for Driver Assistance System in High Speed and Complex Road Conditions'

[7] Nushaine Ferdinand, May 5, 2020, 'A Deep Dive into Lane Detection with Hough Transform'

[8] Zequn Qin, Huanyu Wang, and Xi Li, August 5, 2020, 'Ultra Fast Structure-aware Deep Lane Detection'

[9] Ling Ding, Huyin Zhang, Jinsheng Xiao, Cheng Shuand Shejie Lu, CMES, vol.122, no.3, pp.1039-1053, 2020, 'A Lane Detection Method Based on Semantic Segmentation'

[10] Y. Wang, E. K. Teoh and D. Shen, 22, P 269-280, 2004, 'Lane detection and tracking using B-snake, Image and Vision Computing'

[11] Qiu, D., Weng, M., Yang, H., Yu, W. and Liu, K., 2019, June. Research on Lane Line Detection Method Based on Improved Hough Transform. In 2019 Chinese Control And Decision Conference (CCDC) (pp. 5686-5690). IEEE.

[12] Wu, P.C., Chang, C.Y. and Lin, C.H., 2014. Lane-mark extraction for automobiles undercomplex conditions. Pattern Recognition, 47(8), pp.2756-2767.

[13] de Paula, M.B. and Jung, C.R., 2013, August. Real-time detection and classification of road lane markings. In 2013 XXVI Conference on Graphics, Patterns and Images (pp. 83-90). IEEE.

[14] Kunz, N. Chase. (2017) Vision-Based Control of a Full-Size Car by Lane Detection. All Graduate Theses and Dissertations. 6534. Utah State University, United States.

[15] Open-Source Autonomous Driving Dataset. (2017)

[16] Eskandarian A. (2012) Fundamentals of Driver Assistance. In: Eskandarian A.(eds) Handbook of Intelligent Vehicles. Springer, London.

[17] Zhao, Z.Q., Zheng, P., Xu, S.T. & Wu, X. (2018) Object detection with deep learning: A review, arXiv e-prints, arXiv:1807.05511.

[18] Bellis, Elizabeth, & Jim. (2008) National motor vehicle crash causation survey(NMVCCS) .(SASanalytical user's manual. No. HS-811 053.)

[19] Hernández, D.C., Hoang, V.D. and Jo, K.H., 2013, July. Vanishing point based image segmentation and clustering for omnidirectional images. In the International Conference on Intelligent Computing (pp. 541-550). Springer, Berlin, Heidelberg.

[20] de Paula, M.B. and Jung, C.R., 2013, August. Real-time detection and classification of road lane markings. In 2013 XXVI Conference on Graphics, Patterns and Images (pp. 83-90). IEEE.

[21] Nushaine Ferdinand, May 5, 2020, 'A Deep Dive into Lane Detection with HoughTransform'

[22] Zhao, Z.Q., Zheng, P., Xu, S.T. & Wu, X. (2018) Object detection with deep learning: A review, arXiv e-prints, arXiv:1807.05511.

[23] Lin, J. P., & Sun, M. T. (2018, November). A YOLO-based traffic counting system. In 2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI) (pp. 82-85). IEEE.

[24] Redmon, J. , Divvala, S. , Girshick, R. , & Farhadi, A. . (2016). You only look once:unified, real-time object detection.

[25] Bradski, G. R. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools,25(11), 120-125.

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: …….

Type of Document (Tick): **PhD Thesis** | **M.Tech Dissertation/ Report** | **B.Tech Project Report** | **Paper**

Name: _____ **Department:** _____ **Enrolment No** _____

Contact No. _____ **E-mail.** _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

_____

_____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at..... (%). Therefore, we
are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

<table>
<tr><td>Copy Received on</td><td rowspan="4">• All Preliminary Pages
• Bibliography/Images/Quotes
• 14 Words String</td><td rowspan="2"></td><td>Word Counts</td><td></td></tr>
<tr><td>Report Generated on</td><td>Character Counts</td><td></td></tr>
<tr><td rowspan="2"></td><td>Submission ID</td><td>Total Pages Scanned</td><td></td></tr>
<tr><td></td><td>File Size</td><td></td></tr>
</table>

Checked by
Name & Signature **Librarian**

……

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

Report

ORIGINALITY REPORT

19%	19%	4%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ir.juit.ac.in:8080 Internet Source	15%
2	open.uct.ac.za Internet Source	1%
3	stackoverflow.com Internet Source	<1%
4	docs.opencv.org Internet Source	<1%
5	programtalk.com Internet Source	<1%
6	www.ir.juit.ac.in:8080 Internet Source	<1%
7	github.com Internet Source	<1%
8	zenzer.net Internet Source	<1%
9	www.coursehero.com Internet Source	<1%