Driving Assistance using Lane Detection and Object Recognition

Shreejit Gajanan Deshmukh
MS in Robotics, MIE
Northeastern University
Boston, Massachussets, USA
deshmukh.shre@northeastern.edu

Venkata Sai Advaith Kandiraju MS in Robotics, MIE
Northeastern University
Boston, Massachussets, USA
deshmukh.shre@northeastern.edu

***Abstract*— In this report, we present a comprehensive driving assistance system that combines lane detection and object recognition to improve road safety. The system employs the state-of-the-art YOLOv8 model for object detection, specifically targeting vehicles within the field of view. For lane detection, a series of image processing techniques, including Canny edge detection, Hough transforms, and masking, are utilized to identify and draw the left and right lanes. The proposed system distinguishes between vehicles on the left, right, and ahead, issuing alerts to the driver when a vehicle is deemed too close. The results of both lane detection and object recognition are combined and visualized in real-time video output, showcasing the system's potential as a valuable driving aid. This work demonstrates the effectiveness of combining advanced deep learning techniques with traditional computer vision algorithms to enhance the driving experience and promote safety on the road.**

I. Introduction (*Heading 1*)

With the increasing number of vehicles on the road, driving safety has become a critical concern in modern transportation. As a result, there is a growing interest in developing advanced driving assistance systems (ADAS) that can intelligently analyze and react to the environment, supporting drivers in making informed decisions and promoting safer driving habits. These systems have the potential to significantly reduce the number of accidents and fatalities on the road, ultimately saving lives and resources.

One of the essential aspects of ADAS is the ability to accurately detect and monitor both the vehicle's lane and surrounding objects, such as other vehicles. In this paper, we propose a comprehensive driving assistance system that combines lane detection and object recognition to enhance driving safety. Our system utilizes computer vision algorithms and deep learning techniques to process video input in real-time, providing valuable information to the driver.

For object recognition, we employ the state-of-the-art YOLOv8 model, a deep learning-based approach that has demonstrated impressive performance in detecting and recognizing objects within a given scene. The model is specifically trained to identify vehicles, providing crucial information on their location and proximity to the driver's vehicle.

Simultaneously, the system uses a series of image processing techniques to detect and draw the left and right lanes of the road. These techniques include grayscale conversion, Gaussian blur, Canny edge detection, Hough transforms, and masking. By combining these approaches, our system can accurately detect the lanes, helping drivers maintain their position on the road and avoid unintended lane departures.

The proposed system processes video input to determine the position of surrounding vehicles relative to the driver's vehicle. It then issues alerts when a vehicle is deemed too close, encouraging the driver to maintain a safe distance. By combining the results of both lane detection and object recognition, our system provides real-time visual feedback, enabling drivers to react more effectively to potential hazards.

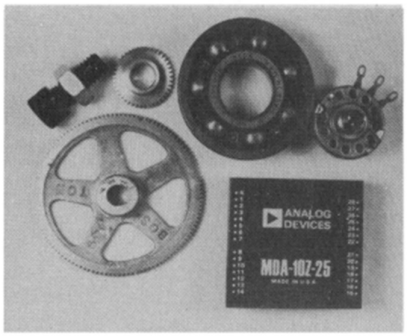
In summary, this work demonstrates the potential of combining advanced deep learning techniques with traditional computer vision algorithms to create a powerful driving assistance system. The proposed approach aims to improve the driving experience and promote safer road practices, contributing to a reduction in accidents and fatalities.

II. Related Work

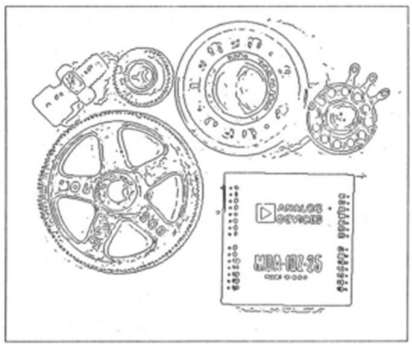
A. *Canny Edge Detection* ***[1]***

The first operation done on our frames is to extract edges. Hence, we use the Canny edge detector. It is a widely used image processing technique for identifying edges in an image, which is a crucial step in many computer vision applications, including lane detection. The algorithm operates in multiple stages to produce accurate and continuous edge information. First, the input image is smoothed using a Gaussian filter to eliminate noise. Next, the gradient magnitude and direction are calculated for each pixel, typically using the Sobel, Prewitt, or Scharr operators. This step highlights regions with rapid intensity changes, which likely correspond to edges.

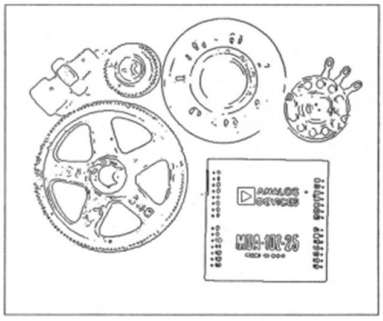
To ensure that the detected edges are thin, the algorithm employs non-maximum suppression, which removes pixels that are not local maxima in the gradient direction. The result is a binary image with potential edges. Finally, the Canny algorithm applies a double thresholding technique to differentiate between strong, weak, and non-edges as shown in *Fig. 1*. Strong edges are those with a gradient magnitude higher than the high threshold, while weak edges have a magnitude between the high and low thresholds. Non-edges have magnitudes below the low threshold. Weak edges are only preserved if they are connected to strong edges, effectively suppressing isolated noise while preserving continuous edges.



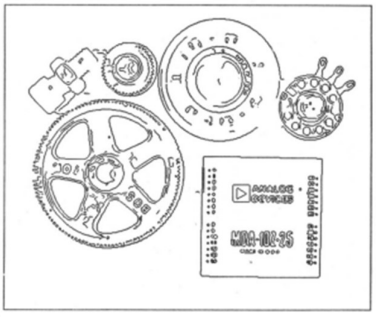
(a)



(b)



(c)



(d)

Fig. 1 – Multi thresholding technique for finding edges.

a. – Original Image

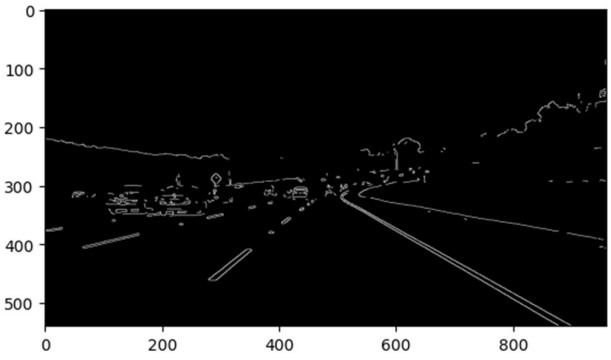
b. – Image thresholded at T

c. – Image thresholded at 2T

d. – Image thresholded with hysteresis using b and c.

In our system, the Canny edge detector is used to identify the boundaries of the lanes. By extracting these edges, the subsequent stages of the lane detection pipeline, such as Hough transforms and masking, can operate on a simplified representation of the image, improving the efficiency and accuracy of the overall system. Below is the applied Canny filter on our lane frame.

(a)



(b)

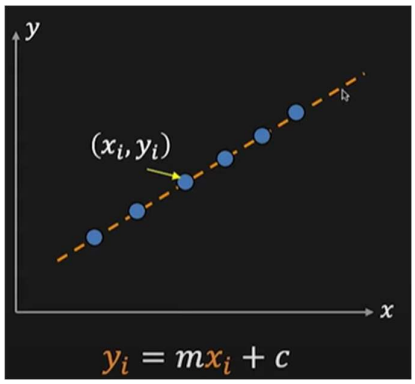
Fig. 2 – Canny Edge detection on Lane frame.

a. Original frame

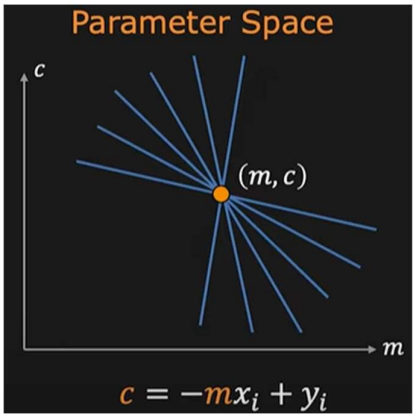
b. Edges extracted from Canny Edge detector.

B. Hough Transform ***[2]***

Hough Transform is a widely used technique in image processing and computer vision for detecting shapes such as lines, circles, or ellipses in an image. It works particularly well in cases where the image contains noise or partial occlusions. The algorithm operates by transforming the image from the Cartesian space (x, y) to the parameter space, usually represented by the polar coordinates (ρ, θ) for lines. Each point in the Cartesian space corresponds to a sinusoidal curve in the parameter space, and the intersection of these curves indicates the presence of a shape in the original image.



(a)



(b)

Fig. 3 – Hough transformation visualization from x, y space to slope-intercept space (Similarly can be transformed to radius, theta space.

a. X, Y space of lines

b. Slope and intercept space for the same points

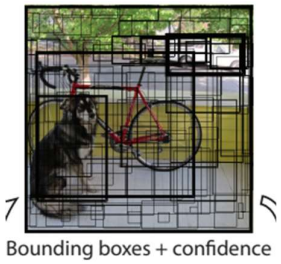
In our system, Hough Transform is used to identify lane lines in the image after edge detection using the Canny algorithm. The Hough Transform is applied to the binary edge image, which highlights the potential lane boundaries. By accumulating votes for each point in the parameter space, the algorithm can identify lines that meet a specified threshold of votes. This threshold is crucial for distinguishing actual lane lines from random noise in the image.

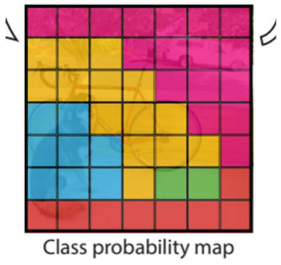
Once the lines are identified, the code further processes the results by calculating the slopes and segregating them into left and right lanes based on the sign of the slope. By averaging the slopes and intercepts of these lines, the algorithm can approximate a single representative line for each lane. The resulting lines are then drawn on the original image, providing visual feedback to the driver and assisting them in maintaining their position on the road. In summary, the Hough Transform plays a crucial role in the lane detection pipeline by converting the edge information obtained from the Canny algorithm into meaningful lines representing the lanes in the scene.

*C. YOLOv8n **[3]***

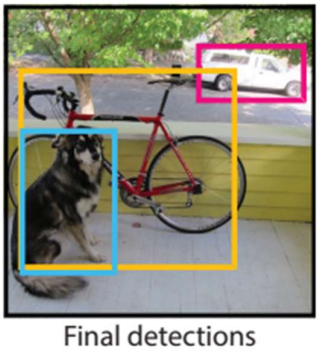
YOLOv8n (You Only Look Once version 8) is a state-of-the-art deep learning-based object detection model that can identify and locate objects in an image with high accuracy and speed. The YOLO algorithm divides an input image into a grid, with each cell responsible for predicting multiple bounding boxes and class probabilities. The model uses anchor boxes and learns to adjust offsets to predict the final bounding box coordinates, making it highly efficient and suitable for real-time applications.

(a)





(b)



(c)

Fig. 4 – YOLO method of working from a -> c

a. S x S grids to then find bounding boxes and offsets

b. All the BB set on each grid

c. Most probable detections from grids to get classified.

In our system, YOLOv8n is employed for object recognition and driving assistance. The primary focus is on detecting and tracking vehicles in the scene. By analyzing the position, size, and proximity of the vehicles, the algorithm provides relevant information to the driver, such as the number of cars on the left, right, and ahead, as well as any nearby vehicles that could pose a potential hazard. This information is crucial for maintaining a safe driving distance and assisting the driver in making informed decisions on the road.

In summary, YOLOv8n plays a vital role in the driving assistance system presented in the code by efficiently detecting and tracking vehicles in real-time. When combined with the lane detection pipeline that uses Canny edge detection and Hough Transform, the algorithm provides a comprehensive solution for driver assistance, enhancing safety and situational awareness on the road.

There are over 80 classes of objects in the YOLO v8 nano model which we are using for recognizing vehicles in the frame.

Below is the screenshot of cars identified in our frame.

Fig. 6 – Cars identified by YOLO framework.

III. WORKING

The pipeline implements an advanced driving assistance system that effectively combines lane detection and object recognition techniques to enhance driver awareness and safety on the road. The methodology employed in the code consists of several processing steps that integrate the advantages of well-known computer vision algorithms and deep learning models.

The lane detection pipeline begins with preprocessing the input image by converting it to grayscale and applying a Gaussian blur to smoothen the image. This step helps in reducing noise and highlights the essential features required for further processing. Next, the Canny edge detection algorithm **[1]** is employed to identify the edges in the image, which represent potential lane boundaries. The algorithm works by detecting areas of rapid intensity change in the image and suppressing the non-maximum gradients, leading to thin, continuous edge lines.

After obtaining the edge image, the Hough Transform **[2]** is used to identify lines in the image. This technique works by converting the points in the Cartesian coordinate system to a parameter space, where the intersections of curves represent potential line segments. By setting an appropriate threshold, the algorithm filters out spurious line detections and retains only the prominent ones.

To further refine the results and segregate the lines into left and right lanes, the slopes of the detected lines are calculated. Based on the slope values, the lines are categorized as left or right lanes, assuming that left lanes have a positive slope and right lanes have a negative slope. A single representative line for each lane is then obtained by averaging the slopes and intercepts of the lines in each category. These averaged lines are drawn on the original image, providing clear visual cues for the driver.

For object recognition, the state-of-the-art YOLOv8n deep learning model **[3]** is employed to detect and track vehicles in the scene. This model is known for its high accuracy and real-time performance, which makes it well-suited for driving assistance applications. The position, size, and proximity of the detected vehicles are analyzed to provide the driver with information on the number of cars on the left, right, and ahead, as well as any nearby vehicles that may pose potential hazards.

This information is crucial for maintaining a safe driving distance and assisting the driver in making informed decisions while navigating the road. By overlaying the object recognition and lane detection results on the original image, the system provides a comprehensive solution that enhances safety and situational awareness on the road.

In conclusion, the driving assistance system presented in the code combines the strengths of Canny edge detection, Hough Transform, and YOLOv8n object recognition to offer a powerful and reliable solution for improving driver awareness and road safety.

IV. EXPERIMENTS AND RESULTS

A. Experiments -

While the data collection was done on the dataset image provided in the Udacity database. They have provided with 3 .mp3 files for testing algorithms. Here, we have evaluated our performance in lane detection and object recognition separately. This is done by calculating accuracy over frames.

The three datasets differ as given –

1. solidWhiteRight.mp4 – Here the lane solid lane is to the right while the left lanes are dot-dash.
2. solidYellowLeft.mp4 – Here the lane is yellow in color and the dot-dash line is to the right.
3. challenge.mp4 – This is a difficult setting car-dashpot video with curvilinear lanes and irregular lighting.

Since our Lane finding algorithm relies on finding slopes using Hough transform, if Hough transform fails to find single line through curvilinear road then our algorithm fails, which happens a lot in challenge.mp4.

B. Results -

We will find results individually for each video and then combine it to find overall accuracy of our algorithm.

Lane Detection

The type of errors in lane detection with its example are –

a. Incorrect slope –

In the figure shown below, a snap from challenge.mp4. The left lane is incorrectly recognized, and this may lead to hazardous situations while autonomous driving.

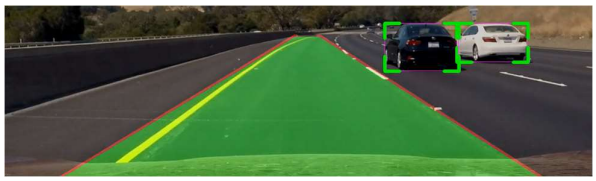


Fig. 6 – Incorrect slope

b. Cross-poly –

While figuring out the two lines with opposite slopes, some points from either sides get mixed to form a cross-polygon because of intersecting lines. This occurrence is frequent with highly curvilinear lines. It can be shown as below –

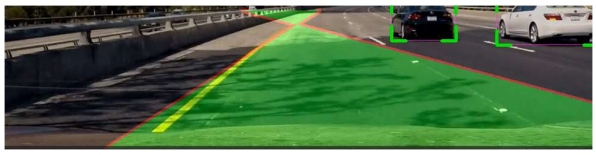


Fig. 7 – Cross-poly

c. Slope not detected –

Sometimes because of highly irregular distribution of points, slope remains empty which causes failure in lane recognition.

DIFFERENT TYPES OF LANE DETECTION ERRORS IN VIDEOS

Clip	Type of error			Number of Frames
	Slope incorrect (SI)	*Cross-poly (CP)*	*Slope not detected (SND)*	
solidWhiteRight.mp4	14	0	0	221
solidYellowLeft.mp4	58	18	0	681
Challenge.mp4	160	72	1	251

Fig. 8 – Lane Detection error table

From the table we can see that our pipeline's performance is worst in the challenge video which is expected. The total accuracy for each video is calculated based on weights assigned to the type of error. For example, incorrect slope is given a weight of 0.15 because the trouble in driving assistance is negligible with this type of error, Cross-poly error has a weight of 0.35 and if slope is not detected then our lane detection is blind that's why the weight given is 0.5.

Thus, total lane detection error (LDE) is calculated as –

LDE = 0.15 x SI + 0.35 x CP + 0.5xSND

Using the above formula, the error for solidWhiteRight.mp4 is ~ **%0.95**. Error on solidYellowLeft.mp4 is ~ **%2.32**. Error on challenge.mp4 is ~ **%43.98.**

Object Recognition

We will find customized performance metric for vehicle detection as we did in lane detection. For vehicle detection, these are the following errors we see –

a. Incorrect classification –

This error happens when a vehicle is incorrectly classified, i.e. a background object which is non-vehicle is classified as a vehicle. For example, look at the image below –

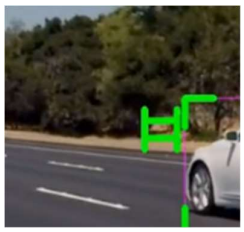


Fig. 9 – Incorrect classification

A tree shrub is classified as vehicle in the above image.

b. Incorrect information –

This happens when a far vehicle is classified as a near one or vice versa. For example,

Fig. 10 – Incorrect information

Even if the black car is not in front, an alert is raised. This happens when on a curved road the right or left vehicle center comes into the central view of our camera.

DIFFERENT TYPES OF OBJECT RECOGNITION ERRORS IN VIDEOS

Clip	*Incorrect Classification (IC)*	*Incorrect Information (II)*	*Number of Frames*
solidWhiteRight.mp4	38	0	221
solidYellowLeft.mp4	36	0	681
Challenge.mp4	42	0	251

Fig.11 –Object recognition error table

Fortunately, we don't see any incorrect information displayed on any of our videos. The incorrect classification consists of - incorrect classification, did not classify as vehicle, double classification for a single vehicle, etc. The error of vehicle recognition is %17.19, %5.28, and %16.73 for solidWhiteRight.mp4, solidYellowLeft.mp4, and challenge.mp4 respectively. With total error ~ %13.

V. CONCLUSION

In conclusion, the advanced driving assistance system presented in the code effectively combines lane detection and object recognition techniques to enhance driver awareness and safety on the road. However, there are still challenges to be addressed in the current implementation, particularly in the lane detection component.

As observed in the results, classical computer vision-based lane detection is prone to errors, such as incorrect slope identification, cross-poly issues, and failure to detect slopes. These issues primarily arise due to the limitations of

traditional edge detection and Hough Transform-based approaches, especially in complex road scenarios with highly curved lanes or varying lighting conditions.

To further enhance lane detection performance, it would be worthwhile to explore the potential of deep learning-based methods, such as semantic segmentation or instance segmentation. These approaches could provide more robust and accurate lane boundary identification by leveraging the power of neural networks to learn complex road features and structures.

Regarding object recognition, the current implementation using YOLOv8n yields relatively good performance. However, it may be beneficial to explore the use of more advanced and larger YOLO models, such as YOLOv8l or YOLOv8x, to further improve the accuracy of vehicle detection and reduce the number of incorrect classifications. These larger models have been designed to capture more intricate features and patterns in images, which may lead to better performance in recognizing and classifying objects, even in challenging scenarios.

Overall, the advanced driving assistance system demonstrates the potential to improve driver safety and situational awareness by combining lane detection and object recognition techniques. However, further research and experimentation with advanced deep learning models could help mitigate existing limitations and provide an even more robust and reliable solution for enhancing road safety.

Acknowledgment

We would like to express my deepest gratitude to my mentors and colleagues who have provided their valuable guidance and support throughout the development of this project. Their expertise and insights have been instrumental in the successful completion of this work.

We would also like to thank my friends and family for their constant encouragement and belief in my abilities, which helped me stay motivated and focused during challenging moments.

Special thanks go to the open-source community and the creators of the libraries and tools used in this project, including the developers of YOLOv8 and OpenCV. Their contributions have made it possible for researchers like me to build upon existing knowledge and explore new frontiers in computer vision and autonomous driving.

Lastly, I am grateful for the opportunity to contribute to the advancement of the field and hope that the findings presented in this work will inspire further research and innovation.

REFERENCES

[1] J. Canny, "A Computational Approach to Edge Detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-8, no. 6, pp. 679-698, 1986.

[2] P. V. C. Hough, "Method and means for recognizing complex patterns," U.S. Patent 3 069 654, Dec. 18, 1962.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," arXiv preprint arXiv:1506.02640, 2016