Vision-based vehicle assist system for autonomous vehicles

Saksham Dhull
2018186
Saksham18186@iiitd.ac.in

Siddhant Yadav
2018196
Siddhant18196@iiitd.ac.in

Yash Vats
2018204
Yash18204@iiitd.ac.in

Abstract

It's tough at first to wrap our mind around the fact that a car operated by computers could somehow be safer than those driven by humans. In fact, a study by National Highway Traffic Safety Administration revealed that 94% of road accidents are caused by human errors. Self-driving cars still seem like an unachievable desire for Indian roads but some additional safety features like automatic braking, lane correction, rear object distance alert etc. can help us have a safer ride. So, here we are trying to implement these features using image processing and deep learning. First we have implemented Lane detection using computer vision. Then object detection to avoid accidents and clashes has been done using deep learning.

1. Introduction

The lane on the road act as a constant reference for humans to drive. Naturally, detecting those lanes would be the first thing to do in developing a self-driving car. Once we detect the lanes, we can instruct the vehicle to stay inside it to ensure a safe drive.

Secondly, we have detected other vehicles moving on the road which is already being used in the automatic braking system in cars these days. This vehicle detection will also be useful in automated vehicles as it is necessary to instruct the car to maintain distance from other vehicles.

2. Methodology:

Basic features of automated vehicles require lane detection and object detection:

2.1 Lane detection:

Our initial approach to detect lanes was using color selection.

2.1.1 Lane detection using color selection:

We tried to use the white color of the lanes made on roads to our advantage. As the white color has an RGB value of (255,255,255) we kept a threshold of (200, 200, 200) and the pixels having any of the three R,G or B value less than 200 were mapped to (0,0,0) (black). This way we selected the color white out of our image frame. Same has been depicted in fig 1 and 2.

Figure 1: A frame from the input video

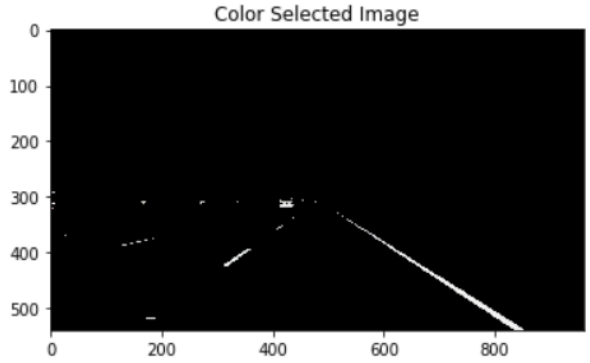

Figure 2: Image after selecting the white color

We can observe that we have got considerable results with color selection.

Now, we considered a hypothetical triangle outlining our white lane and called it our region of interest (fig 3). Then the overlap of the region of interest and white edges (fig 2) gave us our detected edges without any white noise.

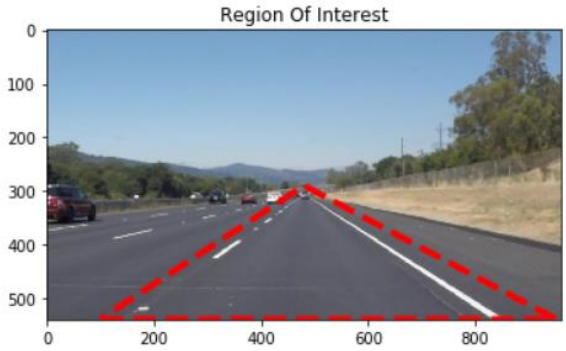

Figure 3: Region outlining the detected lane

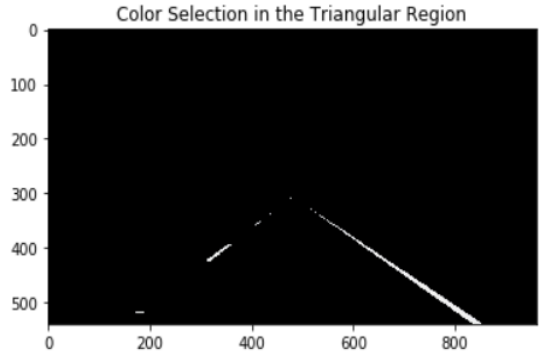

Figure 4: Removed the white noise

Marking the intersection of fig3 and fig4 in green in the original image to get fig 5.

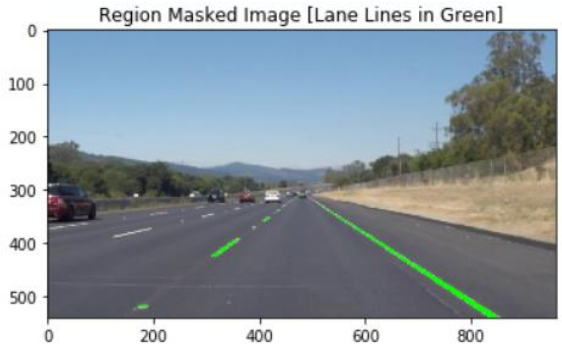

Figure 5: Lane detected image

Hence, we successfully detected lanes using basic technique of color selection. But this detection assumes that lanes are always white. So, it fails in the following case:

2.1.2 Non-white Lane detection :

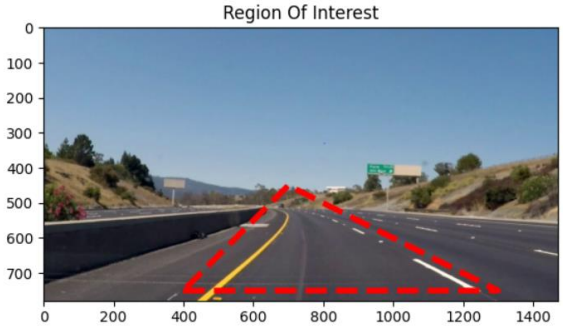

Figure 6: Marking the region of interest

In this case the Lane on the left is not white, hence this algorithm fails to detect it as we can observe in fig 7. So, we tried detecting lanes using Canny edge detection.

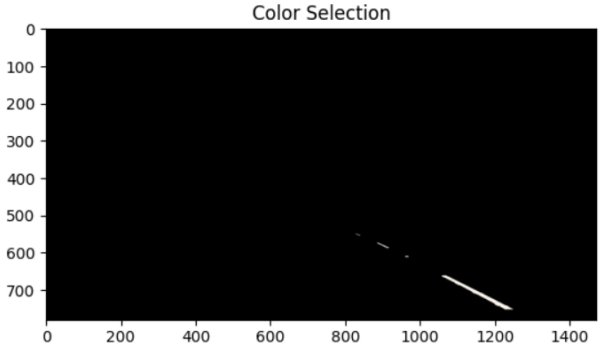

Figure 7: Algorithm fails to detect the yellow lane

2.1.2 Canny Edge Detection:

We'll convert the image to gray scale before performing further operations. Now, a high and a low threshold values will be defined. Canny edge detection algorithm detects strong edges (pixels above threshold) and rejects pixels below low threshold. The leftover pixels between high and low threshold are detected as long as they are connected to strong edges.
Before applying Canny, we used Gaussian blurring to remove the noise and redundant gradients in the image.

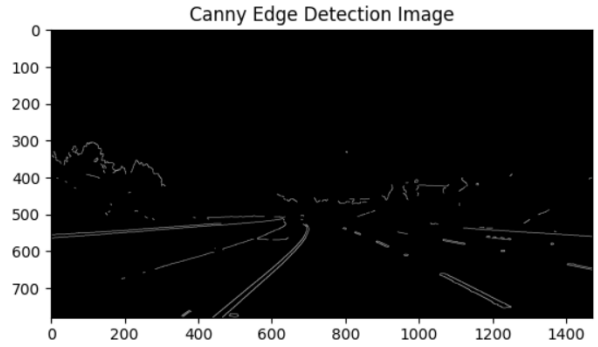

Figure 8: Image after canny edge detection

2.1.3 Hough transform:

We have used Hough transform to find the lines in our region of interest

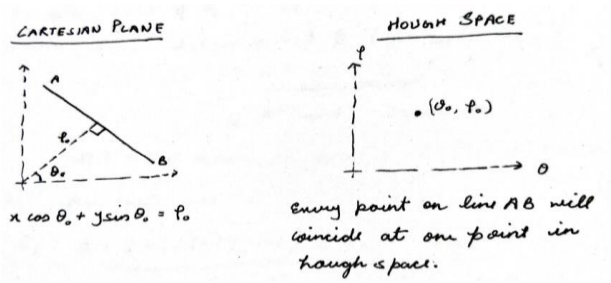

Figure 9: Cartesian and Hough space

As illustrated in fig 9, each point on a line in the image corresponds to one single point (θ,ρ) in Hough space. Accumulation at (**θ**,ρ) in Hough space means that there is a line in the image at a perpendicular distance of ρ and making an angle θ with the origin.

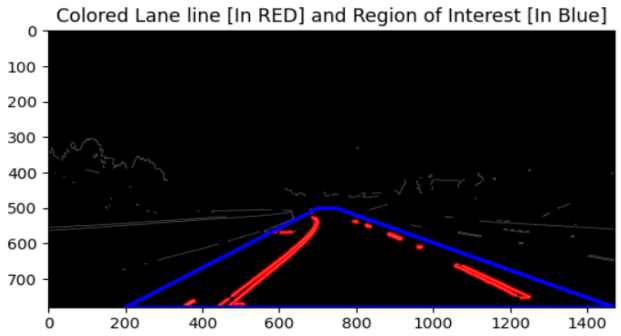

Figure 10: Marking our region of interest for Hough Transform

Applying Hough transform in our masked area of interest to detect the straight lane lines. Filling the detected lanes and adding them to original image to get fig 11.

Figure 11: Final Lane detected image

2.2 Object detection:

For object detection we use the "detect_common_objects" method from the "cvlib" library. This method uses YOLOv3 model pre-trained on the COCO dataset for identifying the common objects.

3. Conclusion:

The paper presents a lane detection algorithm using computer vision. First we tried to detect simply the white lanes on the road but the model failed for any other sort of road lines which were certainly not white in color. Then we used canny edge detection to detect the lane edges and then Hough transform to filter out the straight lanes and ignore all other edges.

Thus system is applicable on any road with well-marked lanes and implemented to the embedded system for the assistance of Advanced Driver Assistance Systems.

3.1. Future work:

We look forward to a better implementation of object detection using CNN as the one that we are using currently is for 80 general objects. A model specifically trained for cars, other vehicles, obstacles and road pits is required for this task. And that would also increase the frame rate of our processed video.

Currently this model is not able to detect curved edges, deep learning can be applied to make it learn about the curves in road lanes.

4. References:

[1] YILMAZ, A., GÜZEL, M., ASKERBEYLİ, İ, & BOSTANCI, E. (2019). A Vehicle Detection Approach using Deep Learning Methodologies. *Computer Engineering Department, Ankara University*.

[Online]: here

[2] Yan, X., & Li, Y. (2017). A method of lane edge detection based on Canny algorithm. 2017 Chinese Automatic Congress (CAC)
[Online]: here