Project 1: Lane Detection.

ENPM 673, Robotics Perception

Utsav V Patel

UID – 115816756

Following are the steps that I took to complete this project

1) **Denoising the image-**
   Any image taken by image capturing sources has some noises, it is important to perform denoising operation to remove the high frequencies before applying any other mathematical operations. Here, I used median filter to denoise the image as it reduces the noise without reducing the sharpness of the image.

Fig(1) – Input Image (Noisy)

Fig(2) – Denoised Image

2) **Converting the image from RGB to grayscale-**

After denoising the image, I converted the denoised image to gray scale, the converted image is shown in the fig(3).

Fig(3) – Grayscale Image

3) **Binarizing the image-**

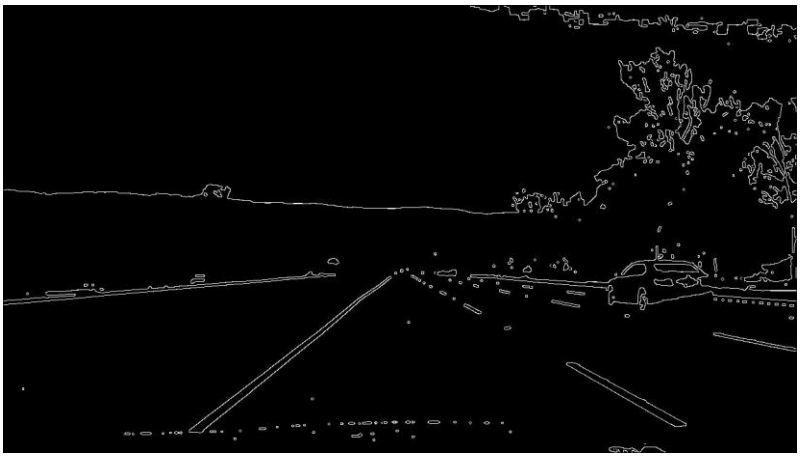
In this step, I converted the grayscale image into binary (BW) image. The threshold value for binarizing the image is selected based on the histogram and trial and error. The binarized image should clearly show the lanes of the road.

Fig(4) – Binary Image

4) **Detecting the edges-**

After binarizing the image, I applied the canny edge detection method to find the edges in the binary image. I read the documentation from the official website of MATLAB. The link is given below.

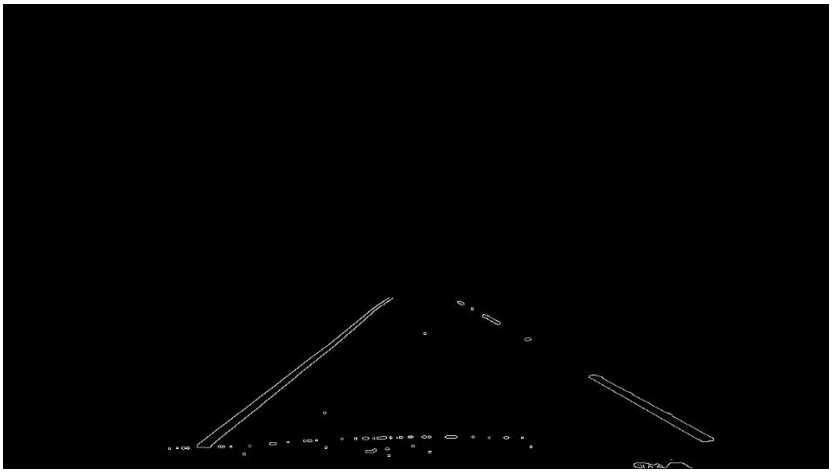
https://www.mathworks.com/help/images/ref/edge.html



Fig(5) – Image with edge detected

5) **Masking the image-**

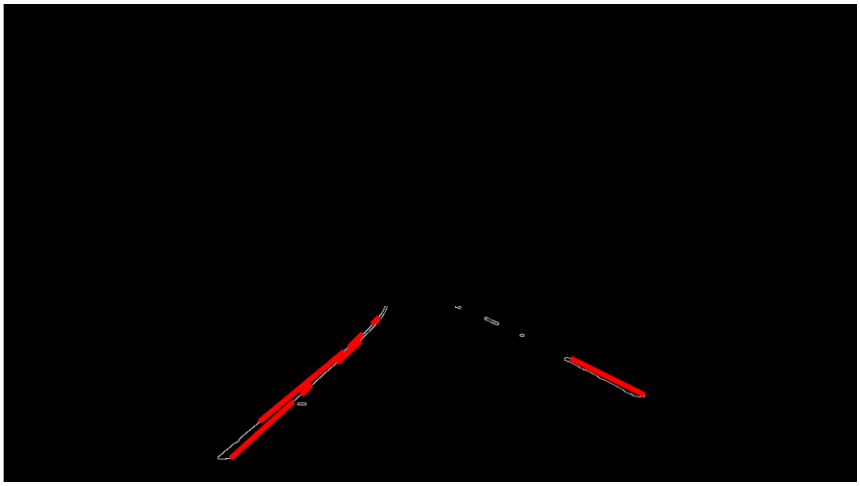
The canny edge detection finds many edges around different objects in the image. All of the found edges are not useful, so we mask out the nonrequired portion of the image. Here, I kept the trapezoidal shaped area, and masked out the remaining area. The masking operation reduces the computational time as the nonrequired area is removed and number of pixels on which operation is required to be done is also reduced.



Fig(6)- Image with unwanted part masked

6) **Applying Hough Transform-**

After applying the masking operation, I the applied Hough Transform to the image. As a result of that, I got houghlines and houghpeak. The value of various parameters were obtained by trial and error method.



Fig(7)- houghlines

7) **Dividing the houghlines in left and right lanes-**

The obtained houghlines are then divided in left and right lanes. This operation is done by comparing the initial and final point of the houghlines with the center line of the image and by comparing the slopes of the houghlines.

8) **Polynomial fitting –**

After dividing the houghlines in left and right lane, a polynomial of first order is passed through the points of the left and right lanes. Now, the slope and constant of the polynomials are obtained. The slope and constant are then used to obtain final points

for the given initial points. The obtained final and initial points are then plotted on the image.

9) **Turn prediction –**

To predict the turn, I used the slope of the left line. The turn is predicted by comparing the slope of the left line with certain threshold values, the threshold values are obtained experimentally. The method is collaboratively developed by me and my classmates. The names of the classmates are following.

Gunjan Khut
Dipam Patel

Fig(8) – Turn prediction

Challenge Video-

For the challenge video, I used the same code, I just changed the value of thresolt to convert image into binary image. The code is showing the turn correctly, but it is not working beyond certain number of frames.

Fig(9)- Challenge Video