Project 2 Report

PRATIK SUNIL BHUJBAL

UID:117555295

Problem 1:

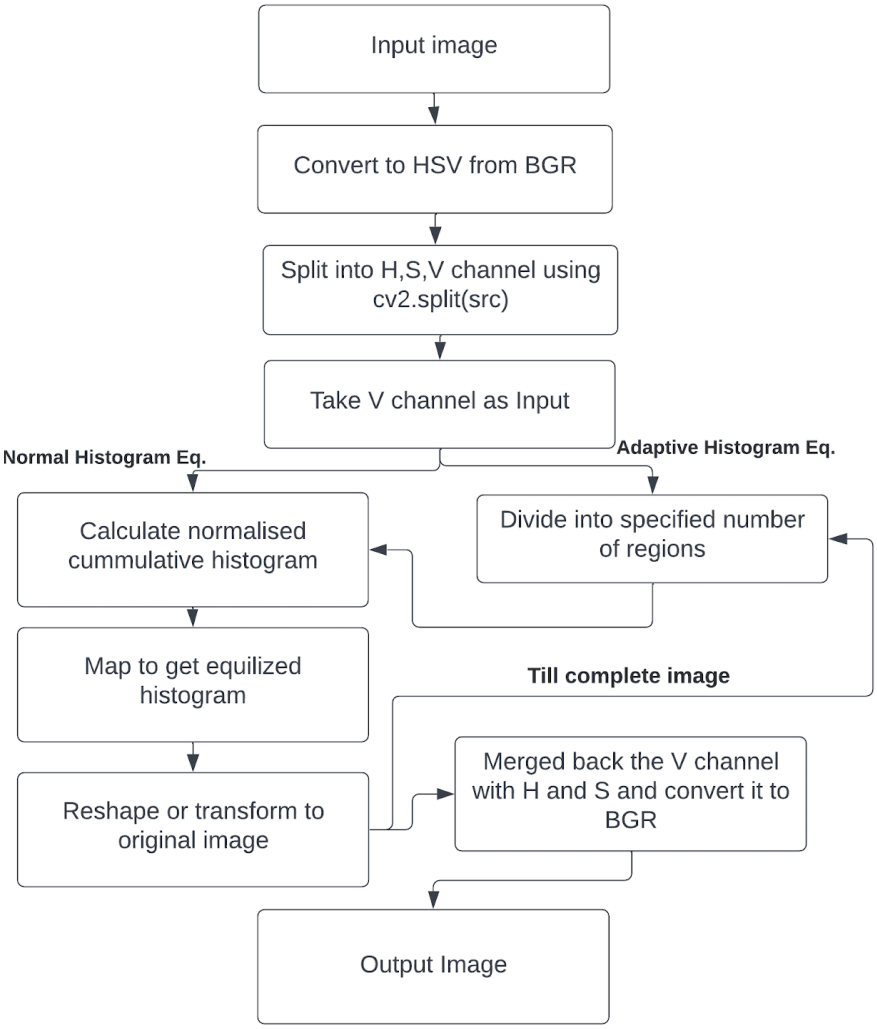


Fig1. Problem 1 flowchart

Take all images in an array sorted according to the frame number. Apply the above methods to each image also as follows: For normal Histogram Equalization (HE) first, convert the image into HSV and use the V channel. Then convert the array to a 1D array and calculate the cumulative histogram by dividing the frequency of each bin by the total pixels in the image. Map the intensities to get equalized histogram and then transform it back to the original image. For Adaptive Histogram Equalization divide the v channel array into distinct blocks and computes normal HE for each section as mentioned above. Then merged back all the sections.

After performing the equalizations merge back the V channel and convert it to a BGR image which will give an output of Normal and Adaptive HE.

Results:

Fig2. Original Image

Fig3. Normal Histogram Equalization

Fig4. Adaptive Histogram Equalization

The above figures show results for normal and adaptive HE. Clearly, results show that Adaptive HE has some advantage over the normal HE that it enhances the contrast locally i.e by

dividing into more sections we can get better and better results. However, this process is slow and in some regions, we can see the over enhancement of noise.

Problem 2

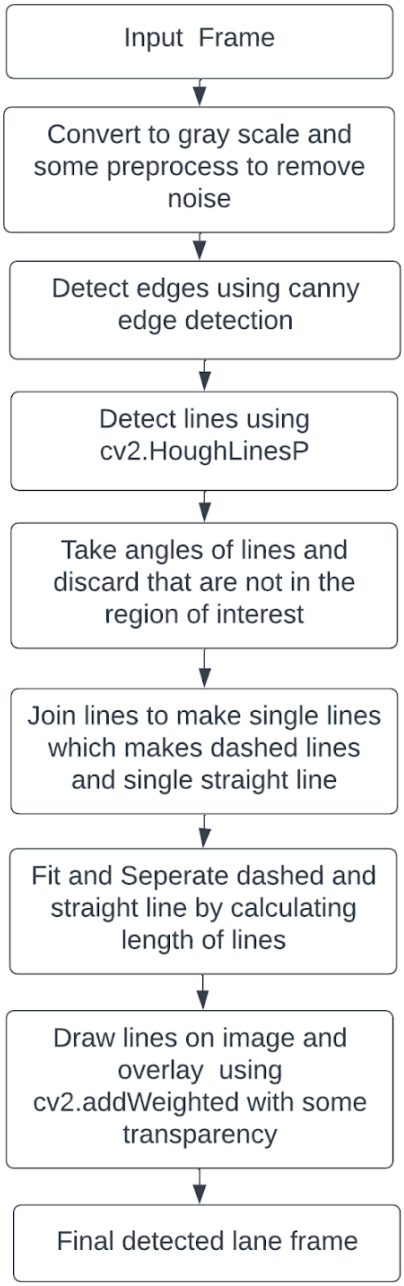


Fig5. Problem 2 flowchart

After preprocessing and edge detection using canny, transformed these edges into lines using Probabilistic Hough Line Transform.

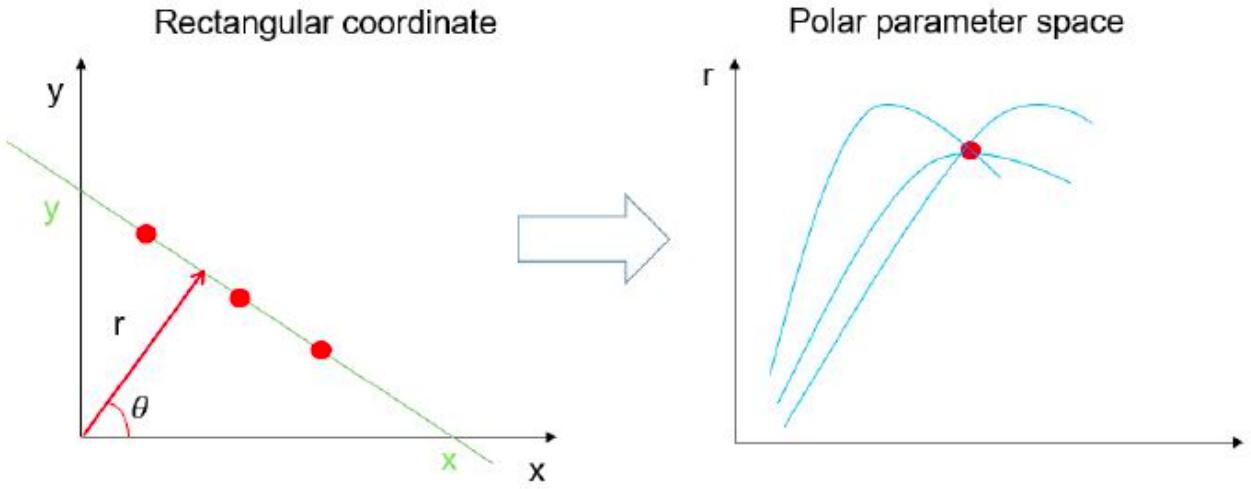


Fig6. Hough transformation (Source)

The basic concept behind the Hough transformation is to create a coordinate system to represent any straight line or curve on an image, and then convert that straight line or curve to parameter space. Here each edge pixel in "Image Space" will have become a line or curve in "Hough Space".

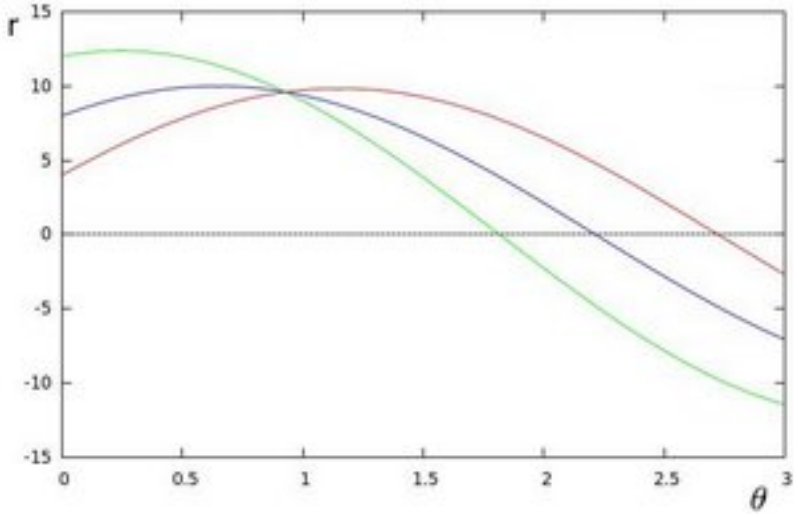


Fig6. Sinusoids curves (Source)

For Hough Transforms, lines are represented in Polar Coordinate System because in a vertical line slope is infinity and infinity cant be represented in Hough Space. In the polar coordinate system, lines are represented with the equation **r=xsinθ + ysinθ**. Where "r" is the radius between the origin points with closest point on the straight line, "θ" is the intersection angle between x-axis and the line which make connection from the closest point to origin point. Considering for all point in the image such that **r > 0; 0< θ < 2*pi*,** gives multiple sinusoids intersecting at one point as example shown in figure 6. And finding the bin which has the most

intersections and can use those m and b values to determine the line of best fit which gives us lines. Parameters can be defined for thresholding the number of intersections, distance resolution, angle resolution, etc. Line detection using HoughLinesP is shown in figure 8.

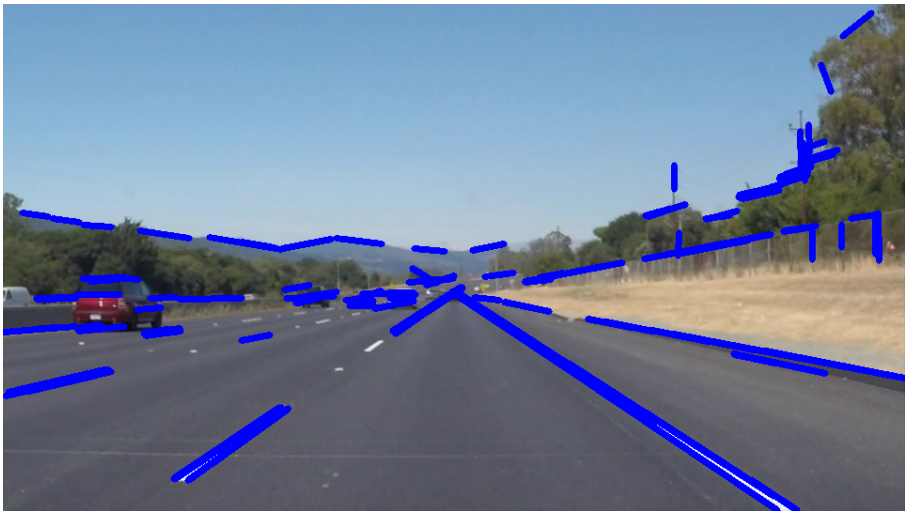


Fig8. Detected Lines

Region of Interest (ROI) using line segment angle:

By determining the angle for each line segment detected by the above method, line segments that are not in the region of interest can be eliminated as shown in figure 8. An angular range within (**$28^o < \theta_L < 40^o$ or $-40^o < \theta_R < -28^o$)** was not discarded.

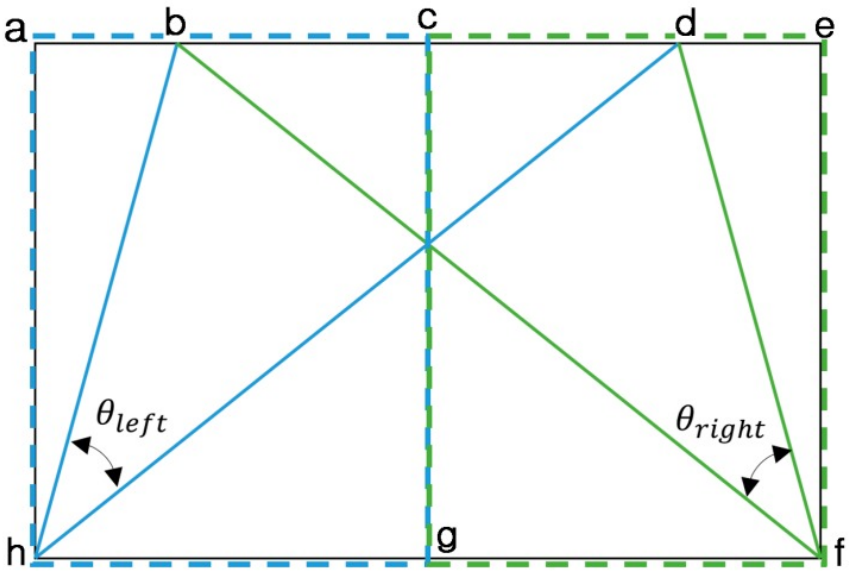


Fig8. ROI (Source)

(a)

(b)

Fig9. a) Line segments before, b) After elimination using line segments angles

After line fitting and overlaying on the lane with the solid line as the green color and dashed as the red color, the final output is shown in figure 10.

Results:

Fig10. Lane Detection

Fig11. Lane Detection after the input flipped

Problem 3

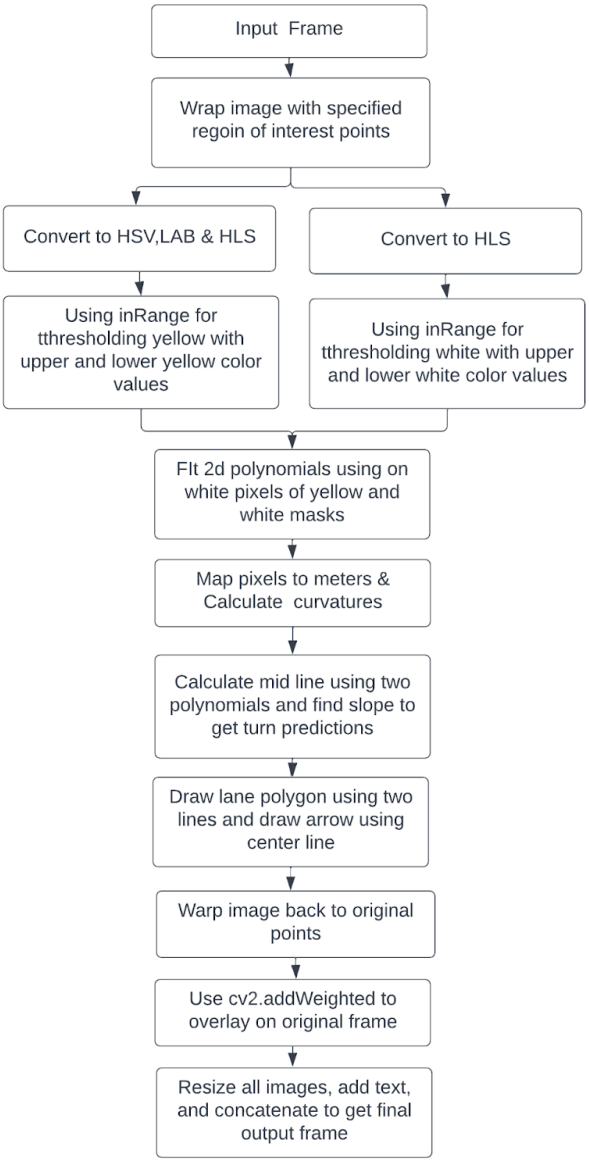


Fig 12. Problem 3 flowchart

By reading the video frame by frame, first transform the perspective of the image to a bird's eye view with specified ROI (Figure 13 b). Convert the BRG image to HSV, LAB, and HLS for the yellow and white masks (Figure 13 c). After getting the white pixels of yellow and white masks, fit the curve using those points and draw the line on those curves (Figure 13 d). Draw polygon and arrows and warp back image to original points. Resize all images, add text for curvature, image info, etc and concatenate to get the final output(Figure 14).

Curvature and Turn Prediction:

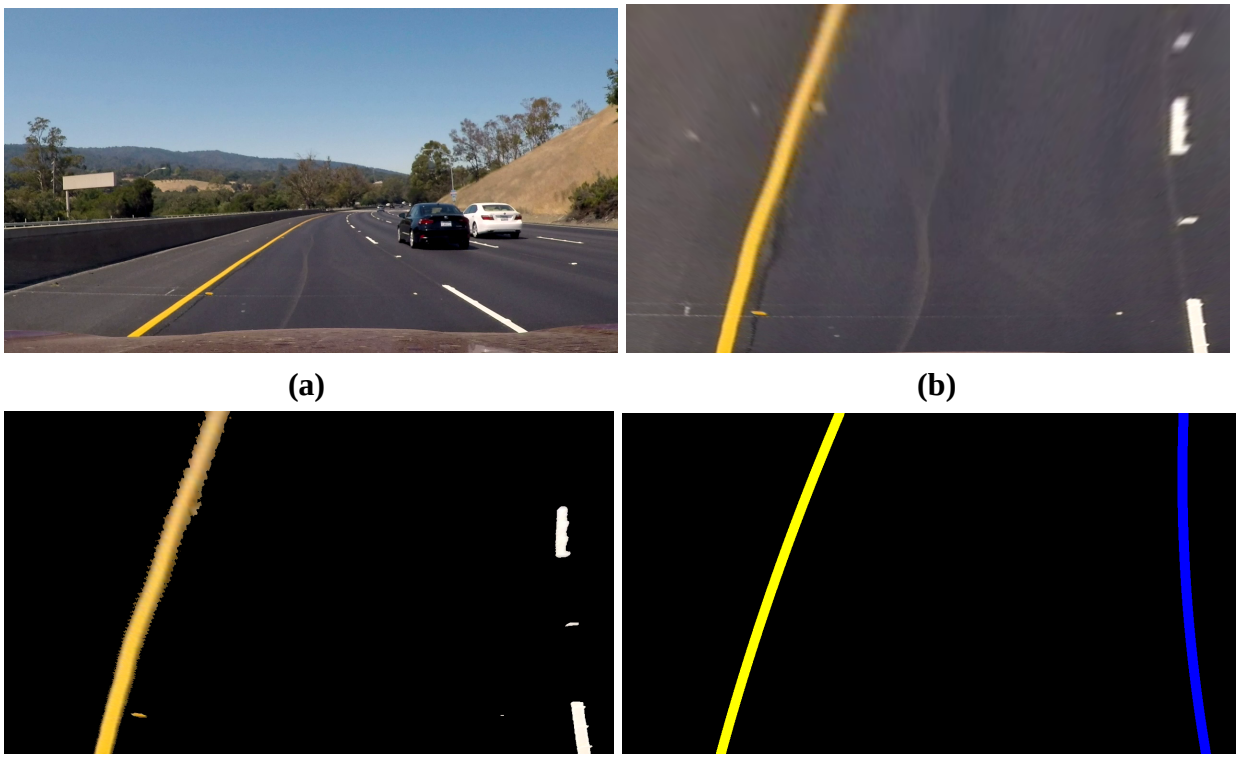
Calculate curvature for left and right lines using the below formula. Lane curvature is computed at the maximum value of y i.e closest point to the vehicle.

$$R_{curve} = \frac{(1+(2Ay+B)^2)^{3/2}}{|2A|}$$

- Source

The slope of the centerline is used for turn prediction. Based on the slope, negative or positive turn prediction and be defined.

Results:



(a) **(b)**

(c) **(d)**

Fig 13. a) Input Image; b) Warped Image; c) Masked image; d) Polynomial Fit

Fig 14. Final Output

Adaptability for similar videos:

This pipeline for both the problems should work well with similar videos after adjusting the ROI and since most of the lane colors are yellow and white, this should work widely. However poor lighting conditions may affect lane detection and polynomial fitting. Threshold conditions need to be adjusted for the different or sudden changes in lighting conditions.

Videos-link:

https://drive.google.com/drive/folders/1IDlf01PBbaaSjvry2oHF3dA8poVJm2LS?usp=sharing

References:

[1] Hoang, T.M.; Hong, H.G.; Vokhidov, H.; Park, K.R. Road Lane Detection by Discriminating Dashed and Solid Road Lanes Using a Visible Light Camera Sensor. Sensors 2016, 16, 1313. https://doi.org/10.3390/s16081313

[2] Rahmdel, P.S., Comley, R., Shi, D., & McElduff, S. (2015). A Review of Hough Transform and Line Segment Detection Approaches. VISAPP.

[3] https://www.intmath.com/applications-differentiation/8-radius-curvature.php.

[4] https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html.