Qamar Syed
ENPM 673
4/4/2022

# ENPM 673 Project 2 Report

(output link at the bottom of the report)

## Problem 1

The solution to problem 1 was fairly straightforward. In order to apply histogram equalization to the image, it was initially split into each color channel (B, G, and R). A function was made to make a histogram with bins for each intensity in a given image array, generate a CFD by comparing the bin count of the intensity and those below it to the number of pixels/points and then multiplying the intensities by their correlating CFD value.

Adaptive equalization was done by iterating over the image but in x by x tile sizes, separating each color channel, and then applying the equalization function detailed above to each color for each tile. Color channels are recombined after equalization of all channels for both to get the final image

As can be seen in the images below, global equalization increases the contrast across the picture and allows for higher clarity on edges and such, but detail that is similar in color becomes more lost. Adaptive equalization allows for an increase in object clarity across the image and makes all portions of the image more visible with added contrast

Original Picture

Qamar Syed
ENPM 673
4/4/2022

Global Histogram Equalization

Adaptive Histogram Equalization

## Problem 2

The program for problem 2 initially applies multiple masks to the image. The first of which only leaves a triangular area going from the edges into the centerline at a specified height. This allows the later processes to only focus on the lanes and the road, and ignores other features with edges and similar colors such as the sky or features outside the road (since this question was framed as detecting the lanes to keep the car within the boundaries it was assumed that the other lanes on the road were irrelevant and only the ones the car was between matter to keep it from leaving the lane).

A mask was then applied to find the white of the lanes using HLS color space to filter for lightness. After the lanes were found, Probabilistic Hough Transform was done to find the ‘most confident’ line and this was determined to be the solid lane. The lane associated with the line was colored red and the rest green and then this was applied onto the images using the mask indexes to color in the location of the lanes on the actual image.  This pipeline is fairly well designed to apply to any video with a single solid line and multiple dashed ones as it just finds these lanes with color filtering and makes the most prominent line detected by cv2.houghlines the red one.

Qamar Syed
ENPM 673
4/4/2022


Problem 2 Video Screenshot
(The bottom overlay is just from google drive, it has the same overlay as youtube but this is my video uploaded on drive, not someone else's)

**My understanding of Hough Line Transform as applied above:**
Hough Line Transform is a voting mechanism that is aimed at finding straight lines in an image. This is done by creating pairs of points, finding the line that would connect these points and 'voting' for it. This is done across all point pairs and the lines that are voted for the most can be returned via a minimum votes parameter. A line that connects many points would indicate those points are positioned at a similar linear slope from each other, and these points make a line

## Problem 3

The solution for problem 3 has a high runtime due to difficulties mainly with the numpy polyfit function. The SciKit RANSAC function was considered as this would have made the solution much easier, but was unsure whether more than just OpenCV, Numpy, and Scipy are allowed for these projects.

The mask to isolate the closest lanes was applied similar to problem 2 at the beginning of the program. After this, HSV and HLS color space masks were applied to find the white and yellow lanes on the image. Using edge detection followed by Hough line transform finds the white and yellow lanes. The voting cutoff for Hough Lines was kept very low so that the program would

always find the approximate location for the lane lines. The lanes are analyzed in much more depth later and that involves the actual part of the program that works when a lane is not detected, not applied here just yet.

Using the points from these line segments (HoughLinesP was used), four points were made from the corner of each which outline the segment of the lane in front of the car. Using findHomography and warpPerspective, the lanes were then projected onto a flat rectangle providing a top down view of the lanes.

**My understanding of homography:**
Matrixes are transformations and a homography matrix is no exception to this. A homography matrix is the matrix that transforms a set of points from one perspective view to another and can be found by selecting the position of an initial set of points and then their desired endpoints after the 'perspective transform'. Using linear algebra, the matrix that would transform one set of those points to the other is then found/estimated. This matrix when applied to an image of that specific initial perspective should transform it to the goal perspective.

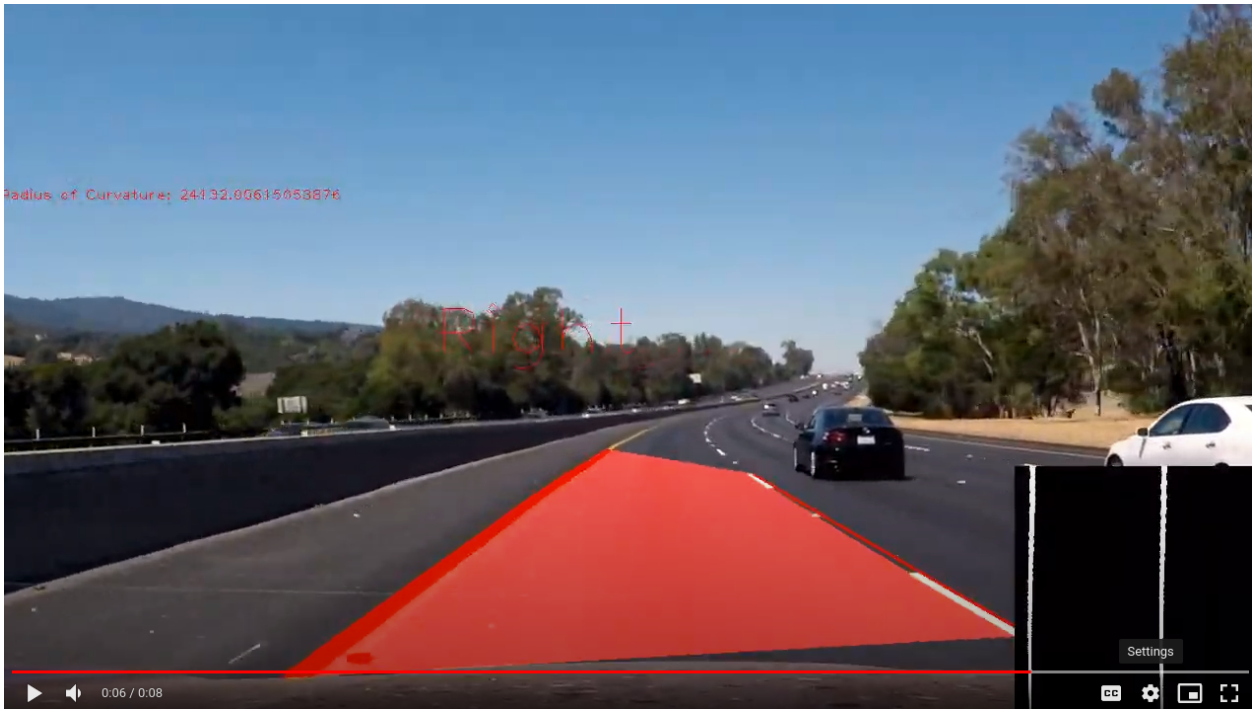
The lanes were then detected via color mask (LAB color space was used to find the yellow lane as it has a value that is low at blue and high at yellow). Then, the lane points were applied to numpy polyfit to find a quadratic equation that best fits all the lane points. This is where issues started to come in. Standard regression is not the most reliable for finding a goal line and a system such as RANSAC would be much more useful here. To take into account an accurate lane or not and continue the program with accurate lanes, the residual per point was calculated for a frames polyfit and if it was not lower than a maximum requirement, the previous polyfit was kept instead. Although this worked for the average case, polyfits that did not match the curve often found themselves with residuals equal to those of accurate fits resulting in a dilemma of filtering out good lines and essentially having very few lanes detected across the video or allowing a few outliers.

Filters such as erosion successfully limited the outliers but a few still remained. Calculations for the turn direction and radius of curvature were based on the values of the polyfit at a point which was found at a specific y range across the lane (loop through lane points starting at a height until it finds a lane point at that height). The turn direction was simply determined by evaluation if the derivative of the polyfit on the lane point was negative or positive indicating the direction the lane is turning. The radius of curvature was calculated with the equation given in the project instructions.

Homography and the lane color mask was then used to apply a colored lane and lane line back onto the original frame, the text for the turn and radius of curvature and top down view were also applied onto the video.

Qamar Syed
ENPM 673
4/4/2022

There is a fair amount of instability with the calculations that most likely would not be there if RANSAC was used as it is very good at keeping the estimated line at an ideal location but polyfit lines were very unstable from frame to frame.




Problem 3 Video Screenshot
Signals Right Turn in the middle (somewhat difficult to read)

## Link to Google Drive output (contains output for all 3 problems):

https://drive.google.com/drive/folders/1FbsfJsH9RkDNWCzaDUpwu9QnItVC9cV8?usp=sharing

Videos stored in .avi file with XVID formatting which worked for Ubuntu 20.04
Pictures stored in .png file