Project 2 Report

ENPM 673 - Perception for Autonomous Robots

Team 5

Prof. Dr. Mohammed Charifa

Nikhil Lal Kolangara (116830768)
Kartik Venkat (116830751)
Kushagra Agrawal(116700191)

Contents

1 Problem 1

Here, we aim to improve the quality of the video sequence provided above. This is a video recording of a highway during night. Most of the Computer Vision pipelines for lane detection or other self-driving tasks require good lighting conditions and color information for detecting good features. A lot of pre-processing is required in such scenarios where lighting conditions are poor. Now, using the techniques taught in class your aim is to enhance the contrast and improve the visual appearance of the video sequence. You can use any in-built functions for the same.

1.1 Approach

Step 1: The video file is a recording of an highway from a camera on the car. The video sequence is shot under low-light conditions during night. The video file is imported and processed frame by frame using openCV.

Step 2: Every frame of the video is converted into grayscale.

Step 3: Gamma correction is done on these gray scale images. Gamma correction function is used to correct an images luminance.

Step 4: A Contrast Limited Adaptive Histogram Equalization (CLAHE) object is created. The image is divided into small blocks called "tiles". Then each of these blocks are histogram equalized. So, the histogram is confined to a small area. The frames after gamma correction is passed to the CLAHE object.

Step 5: A filter is applied to the frame obtained in the above step.

1.2 Result

Original video with gamma correction(Output 1) = It is the video obtained after the gamma correction. The gamma value is kept at 1.9. In this frame we can see a increase in the brightness and the video has become clear compared to the original video. After gamma correction we can see information which was missed out in the original video because of the lack of brightness.

Gamma corrected video after CLAHE (Output 2) = The frame after gamma correction is passed to a CLAHE object to improve the quality of the video. The brightness of the video increases but in doing that the noise in the frame increases.

Final video output after applying 2 D filter to gamma corrected frame (Output 3) = A filter is applied on the frame after doing gamma correction to remove the noise, and the output which we get is the best of all as the video is bright enough to detect the objects such as bridge and the hoardings on the side of the road. The noise in the video is also very less.

2 Problem 2

In this project we aim to do simple Lane Detection to mimic Lane Departure Warning systems used in Self Driving Cars. You are provided with two video sequences (both are required for this assignment), taken from a self driving car (click here to download). Your task will be to design an algorithm to detect lanes on the road, as well as estimate the road curvature to predict car turns. Also keep in mind, that lane detection is a very popular problem and there are many implementations available on the Internet. If any part of your implementation is not designed by you, you will loose a considerable amount of points.

2.1 Understanding of Homography

- Homography is the transformation of an image from one plane to another. The image is manipulated using the 8 degrees of freedom. The use of homography is to calculate the top view of a specific region of interest where the lane is always detected. We do this so that we can isolate the lane from the image and visualize the image in its true from, bringing back its parallelism instead of a projection variant and can utilize these properties.
- We have calculated the homography matrix using the getPerspective function in OpenCV betwee the region of interest points in the image and a set of image points decided by us to make the image upright. Then we pass this matrix to the warpPerspective function of openCV and then use it to transform the image.

2.2 Understanding of Hough Lines

- The Hough transform is a feature extraction technique. It is used mostly for detecting lines but can be extended to find circles and ellipses.
- In general, the straight line y = mx + c can be represented as a point (m,c) in the parameter space. However, vertical lines pose a problem. They would give rise to unbounded values of the slope parameter m. Hence, in Hough transform we consider 'r' which is the distance from the origin to the closest point on the straight line and theta is the angle between the x-axis and the line connecting the origin with that closest point.

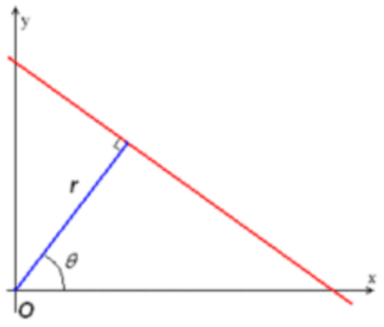

Figure 1: Hesse Normal Form

- Given a single point in the plane, then the set of all straight lines going through that point corresponds to a sinusoidal curve in the (r,θ) plane which is unique to that point.A set of two or more points that form a straight line will produce sinusoid which cross at the (r,θ) for that line. Thus, the problem of detecting collinear points can be converted to the problem of finding concurrent curves. Sinusoid corresponding to the co-linear points intersect at a unique point.

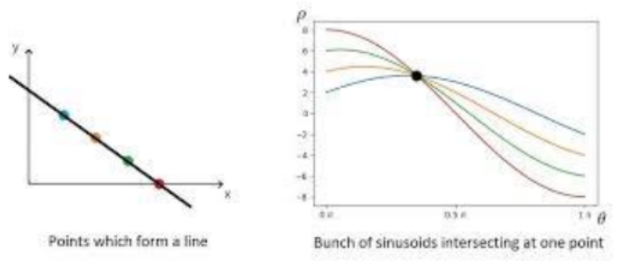

Figure 2: Hough Transform of collinear points

2.3 Pipeline for Lane detection

Step 1: The given sample images are loaded using a function and stored onto a list.

Step 2: In order to detect the lanes the images are warped first using the warPerspective function. The source and the destination points are are defined to make the lanes parallel from vanishing point perspective.

Step 3: The warped image is converted into a grayscale image to reduce the computation as the other color channels does not provide extra information for the purpose of this project.

Step 4: Threshold the grayscaled image and convert it to a binary form.

Step 5: The threshilding emphasizes the white lanes which are traced using sliding window technique. Here a rectangle window is defined by setting an appropriate size. The lanes that are emphasized using thresholding is determined by taking in the coordinates with highest histogram equalized values.

Step 6: The windows help in differentiating the right and left lanes by defining a rectangular box for each separate lane respectively.

Step 7: The polyfit function helps us to plot lines on the lanes. The function outputs a warped image that returns lanes that are colored.

Step 8: The warp perspective function is called again to reverse the image from parallel form reversing to its original format.

Step 9: The output image from previous step is again converted to grayscale and again into binary format. The purpose of this step is to combine the outputs of both the current image and the original image.

Step 10: The colored lanes are displayed on the original image (inverted form) by performing bitwise AND with the binary data of the current image and the original image.

Step 11: The radius of curvature is calculated using the following formula:

Rcurve=|*dy*2*d*2*x*|[1+(*dydx*)2]3/2

The average of radius of curvature of left and right lane is calculated.

Step 12: We used the farthest point on the left lane, to find the slope between it and a point in the middle of the lane.
When the turn is left the gradient is positive and is negative when the turn is right. So, by comparing the gradients we say whether the turn is right or left. If the slope is 0 then it is a straight road.

3 Output

3.1 Problem 1

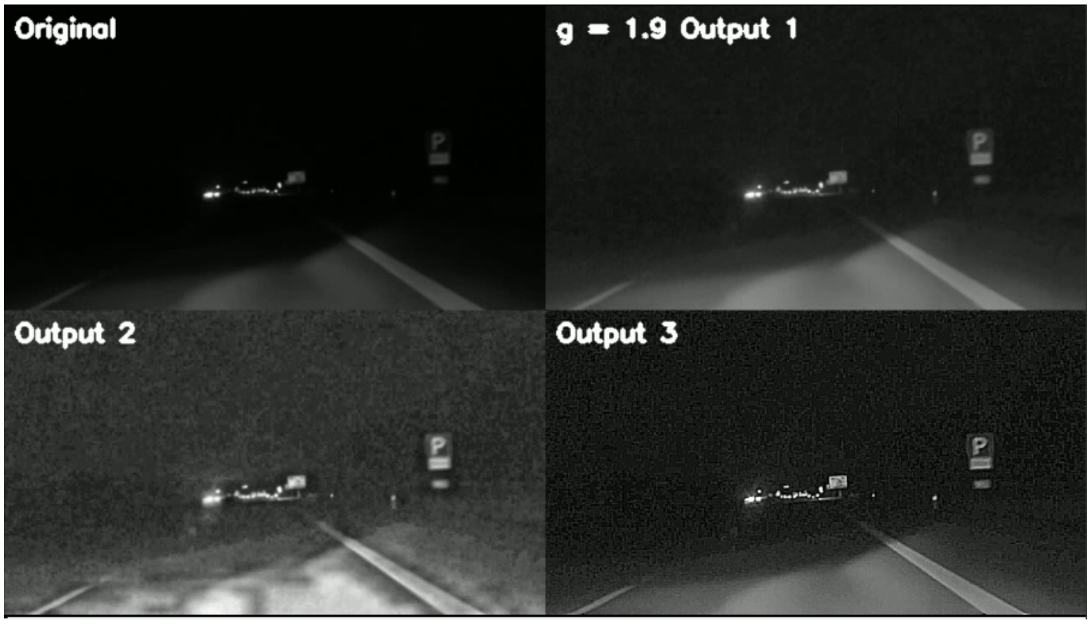

Figure 3: Output Video Snap

3.2 Problem 2

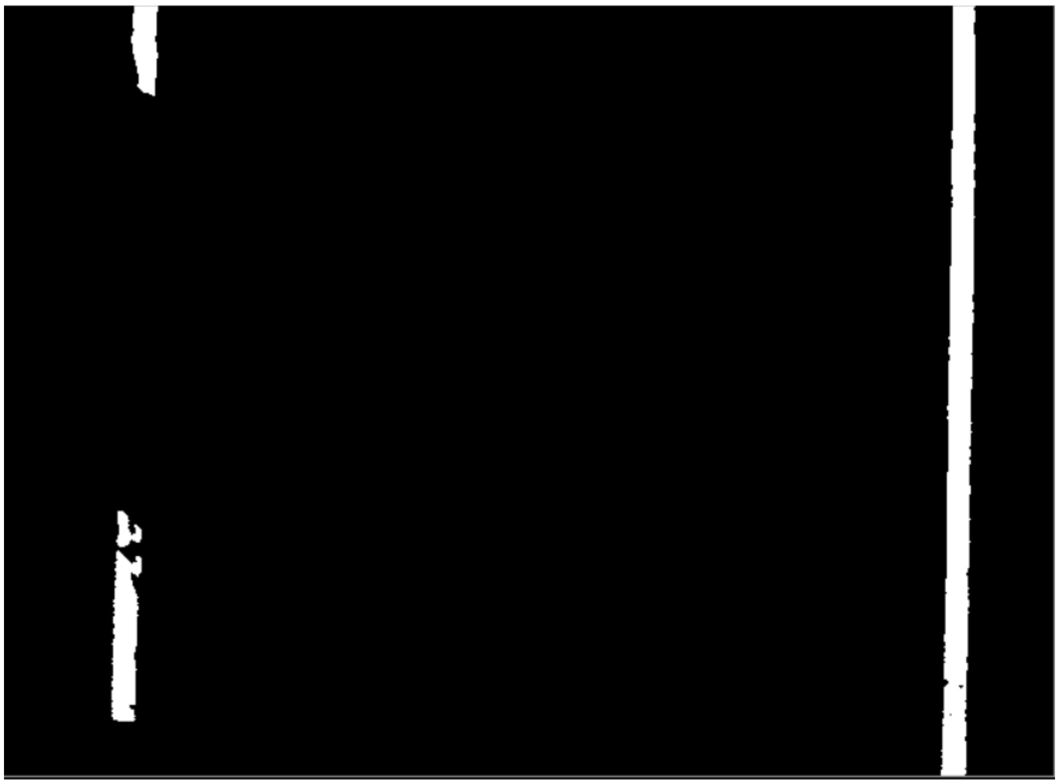

Figure 4: Binary Warped Image

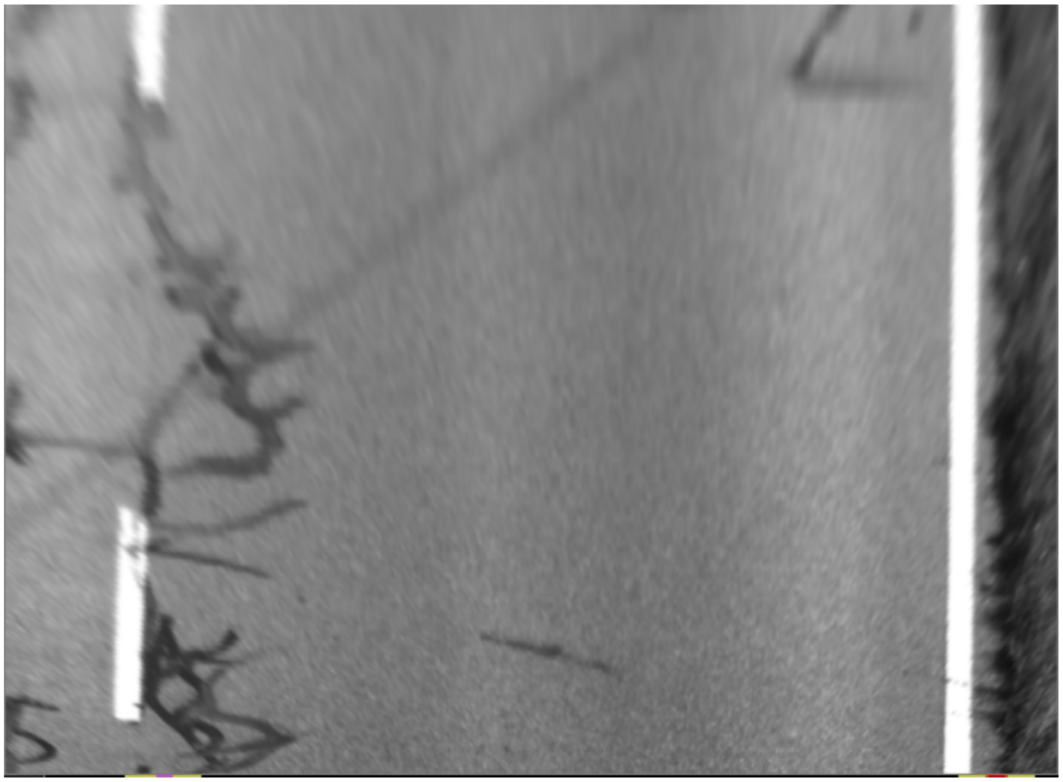

Figure 5: Warped Image

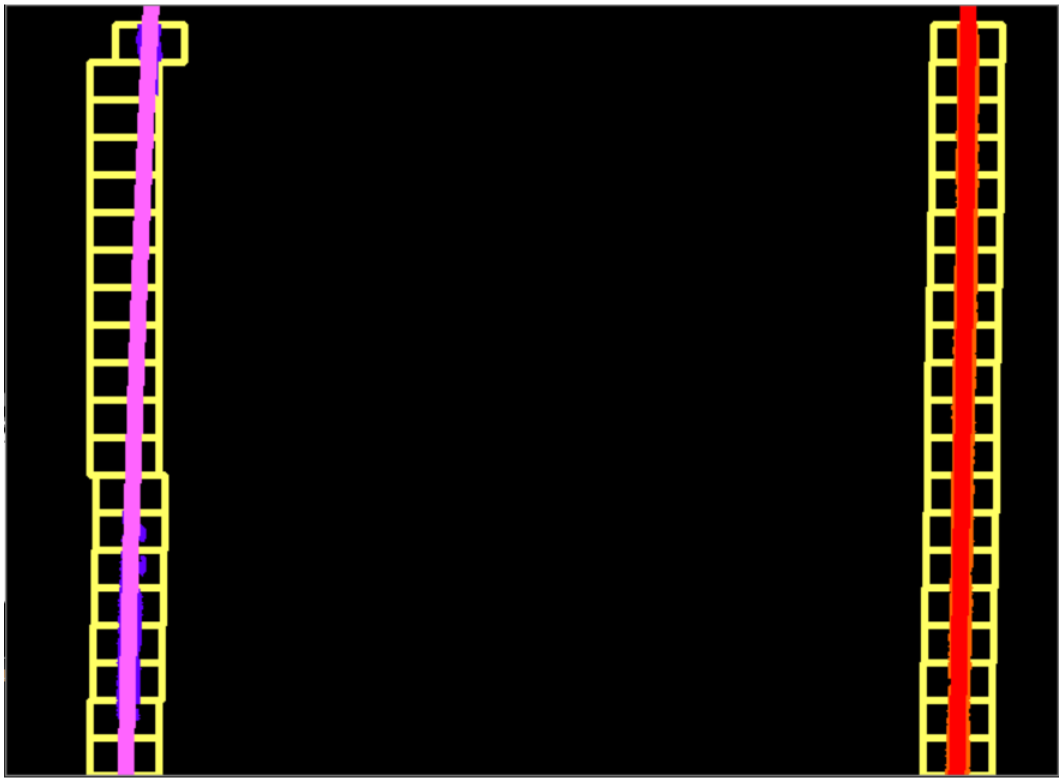

Figure 6: Sliding Window

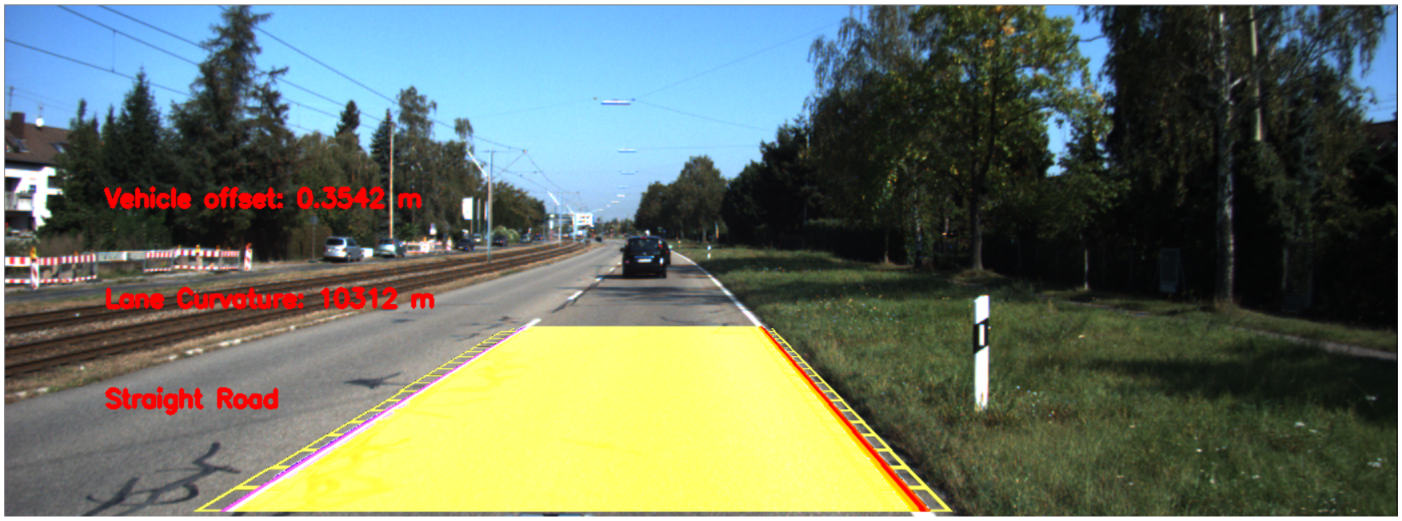

Figure 7: Final Output Snap

3.3 OUTPUT VIDEOS DRIVE LINK:

`http://https://drive.google.com/drive/folders/1R4A5_f2jZA8iUW22EzCjdMACD7pWoooB`