Finding lanes on the road

Objective :

When we drive, we use our eyes to decide where to go. The lines on the road that show us where the lanes are act as our constant reference for where to steer the vehicle. Naturally, one of the first things we would like to do in developing a self-driving car is to automatically detect lane lines using an algorithm.

In this project, a pipe line to detect lane lines in images using Python and OpenCV libraries are described.

Project Pipeline :

As the first step a pipeline was developed to process individual images to convert them into images with lane marking. After successful testing of the pipeline on individual images, the same is applied to multiple video streams.

The overview of the algorithm is shown in **Figure 1** .

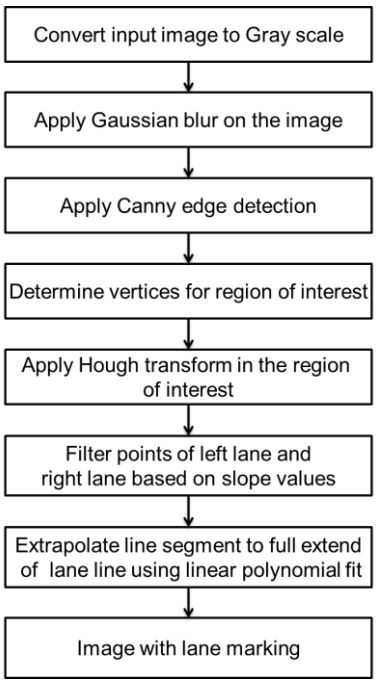

Figure 1 : Overview of the project pipeline

At first the individual images are converted it to grayscale. In the next step, a Gaussian Blur is applied to smooth the image to filter out noises. After this, Canny edge detection is applied to retain the “edges” in the image. In order to filter out the edges pertaining to the lane lines, a region of interest is defined based on the coordinated of a quadrilateral. The vertices of this region are defined, and rest of the pixels are masked. In a next step, Hough transform is applied to find the pixels where line is formed. Then the output of Hough transform is filtered to determine the points (x, y) corresponding to left lane and right lane respectively, based on

the slope values. From the filtered left and right line points, a liner polynomial is fit and extrapolated to form the lane line segment. This lane lines are then overlaid on the original image resulting in the image with lane marking.

Result :

The results of the lane detection pipeline for an example image is shown in **Figure 2**.

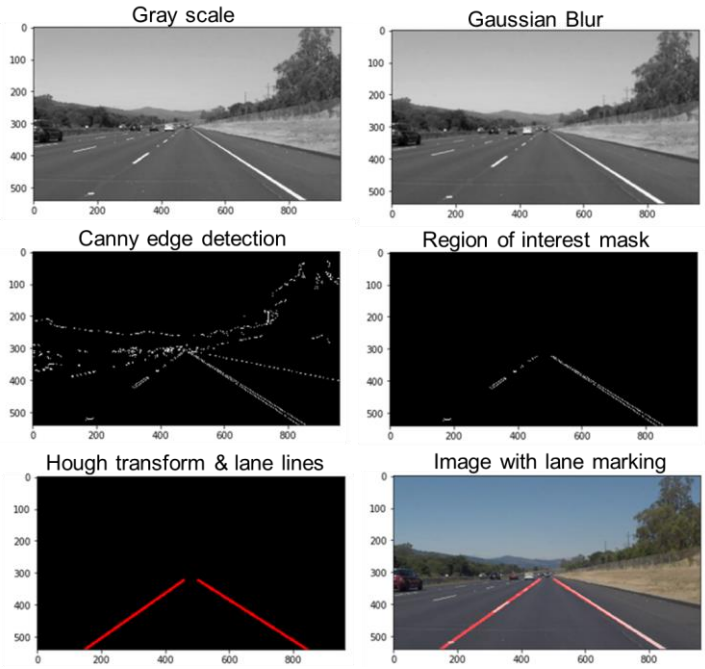

Figure 2 : Results of the lane detection pipeline on individual image

The same pipeline can be applied to other images and even video streams as shown in **Figure 3**

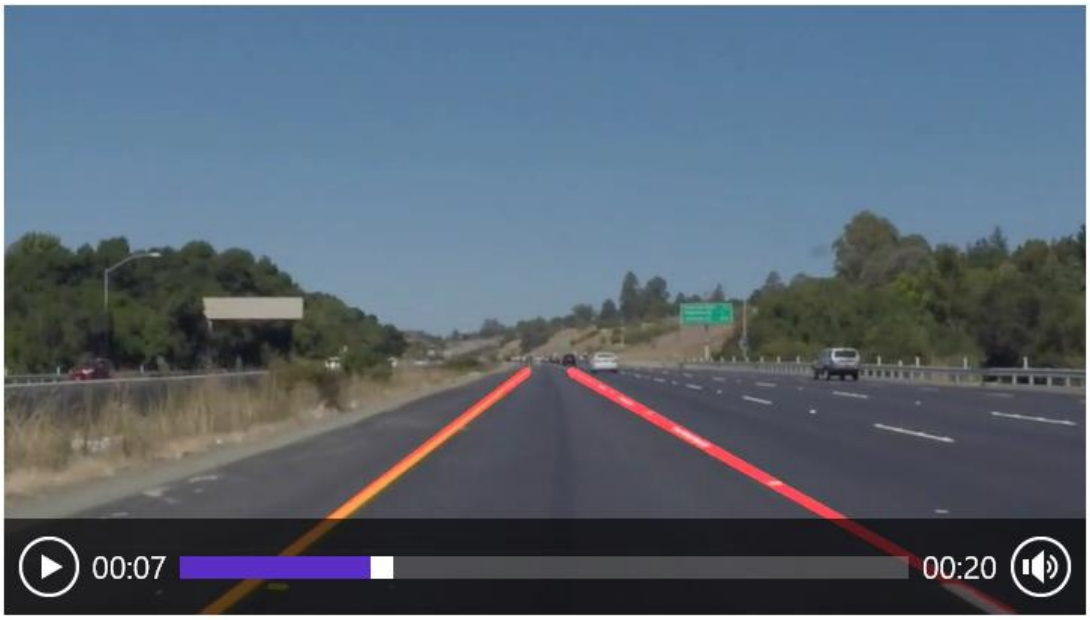

Figure 3 : Results of lane detection pipeline on video stream

Shortcomings :

Following are the short comings of this pipeline.

1. Region of interest is hardcoded coordinates. Hence the camera mount vibration effect is neglected.
2. Linear fit is used to determine the lane lines. Hence, this method fails during curvature.

Possible improvements :

1. Including camera vibration compensation by dynamically including the region of interest
2. Instead of linear fit, may be cubic fit or logistic regression or splines may be used to draw lines for curved lanes.