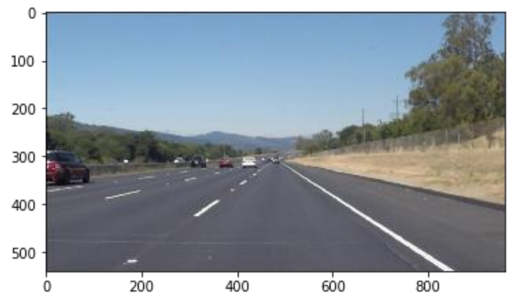
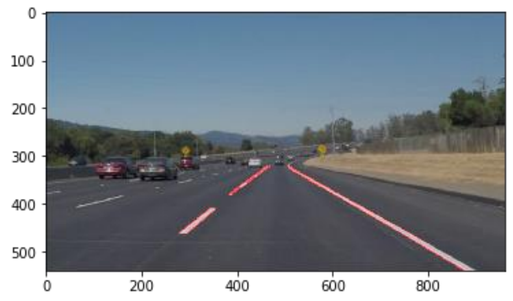
Finding Lane Lines on the Road

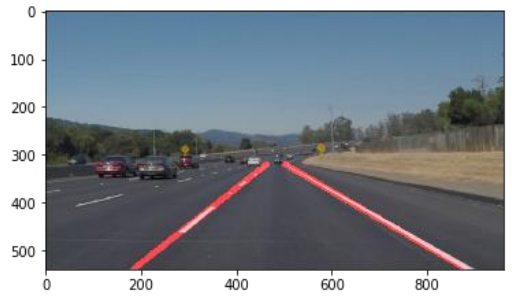
The goal of this project is to build a simple pipeline to detect lanes lines in given road images and short videos taken from car driver view. The result is not only defining the detected lanes segments but also estimates a fully connected lanes on the right and the left of the view. Figure 1 below illustrates the pipeline input and the final expected output.



(a)



(b)



(c)

Figure 1 (a) The raw input image, (b) the detected lane segments, (c) the final output showing only two connected lines for the left and the right lanes.

Lane Detection Pipeline

Step 1: Reading The image

The first step includes reading the raw image and converting it from the RGB color space to a gray scale as a part of the preprocessing for the coming steps of Canny edge detector. The gray scale provides clear contrast between the colors.

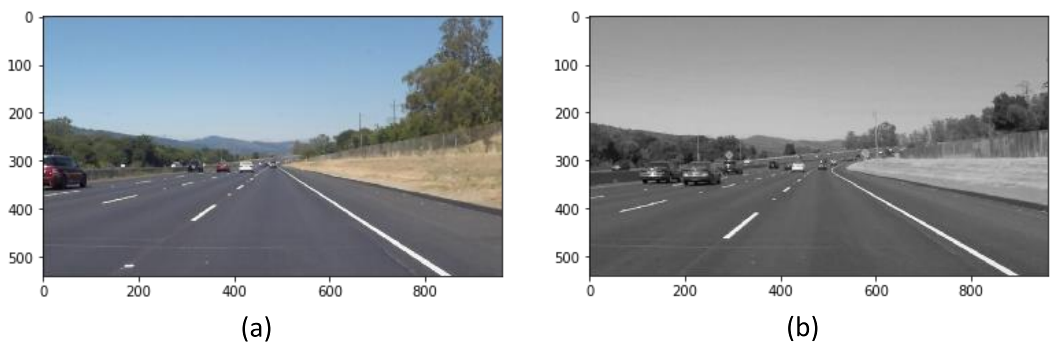


Figure 2 input image (a) vs gray converted one (b)

Step 2: Applying Gaussian smoothing

This step filters out noisy edges by smoothing the image which reduces noisy contrast, so in the next step of edge detection a lower number of edges is detected, these edges are considered more significant.

Step 3: Canny Edge Detection

Applying Canny edge detector filters out every thing in the image except for the edges (lines). This step requires manual tuning for the thresholds required by OpenCV implementation for Canny detector, the chosen values for lower and upper threshold are 50 and 150.

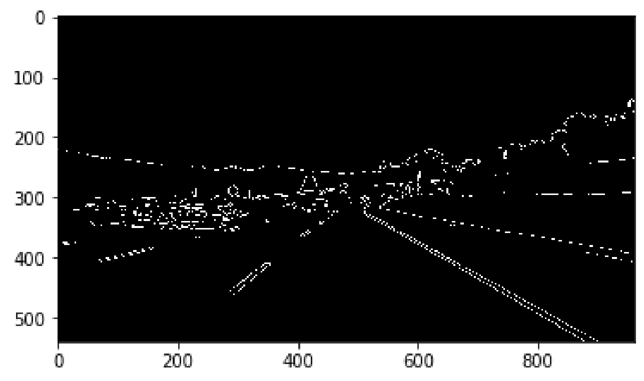


Figure 3 Detected edges after applying Canny edge detector

Step 4: Defining the ROI for the edges in the image

The image contains a lot of edges out of our scope of interest, to filter these edges out we can set a region of interest (ROI) covering the path of the car.

This is done by creating a masking image containing a polygon around the ROI. The pixels inside the ROI are set to Ture while these out of it are set to False. Then by using logical ANDing between the masking image and the previous output from Canny detector we get edges only inside the ROI.

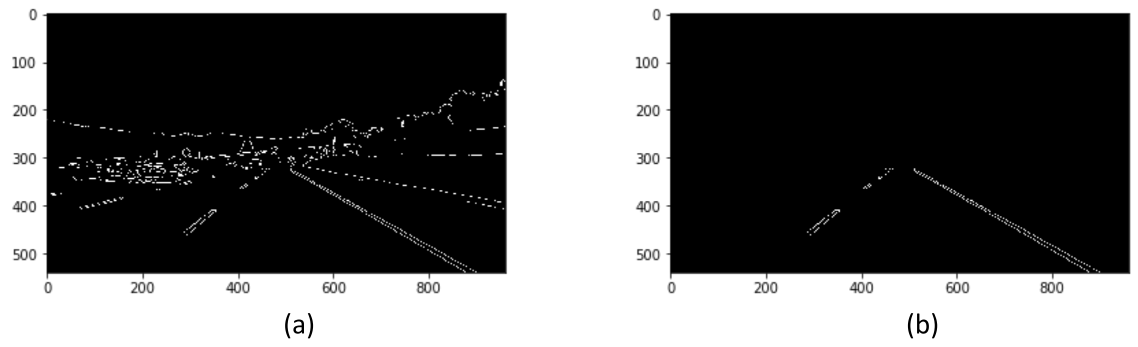


Figure 4 the result of applying ROI masking (b) on the image in (a)

Step 5: Hough Transform

Now we are ready to apply Hough transform on the last masked image. Hough Transform extracts lines by identifying all the points laying on them. This step also requires manual tuning for OpenCV implementation of Hough Transform.

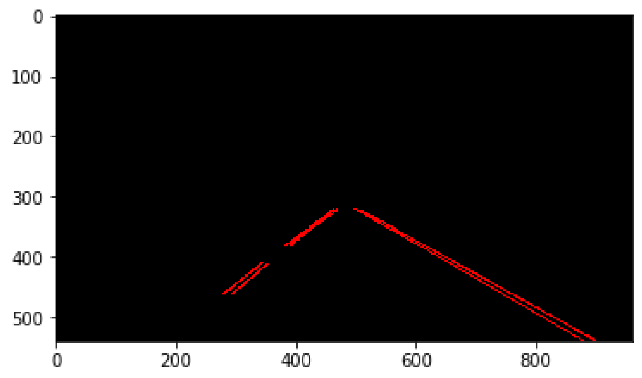


Figure 5 Extracted lines by Hough Transform

Step 6: Final image for Lane Segments

For displaying lane segments the above steps are enough, we can do this by overlaying the last image showing the colored edges on top of the original image, the results is shown below.

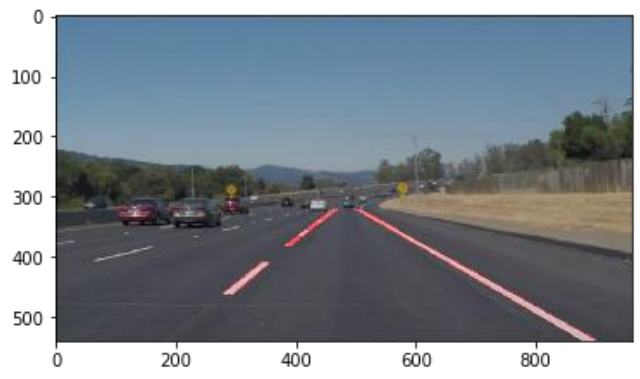


Figure 6 Colored lane segments

Step 7: Connecting the segments

Connecting the segments is done by fitting all the points of the left or right lane into a single line equation, to do that first we have to separate the left and right segments; this is simply done based on the value of the x coordinate.

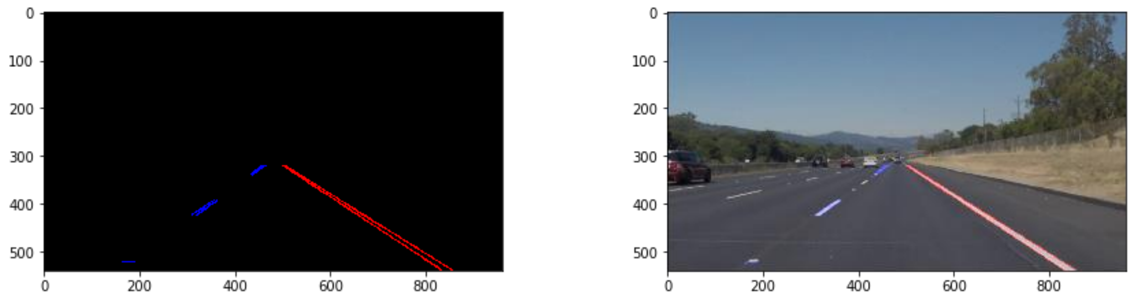


Figure 7 left and right lanes are clearly seperated

After that we can use the OpenCV function fitLine() to fit each set of points into a line, the output of this function for a 2D line is four elements representing the line normalized direction in x and y orientations, and a point coordinates on that line.

Using the extracted lines information we can find the x coordinate for the starting and ending points of left and right lines representing the connected lanes. Keep in mind that we know the top y coordinate which is the starting of the ROI and we also know the bottom y coordinate which the last row in the image.

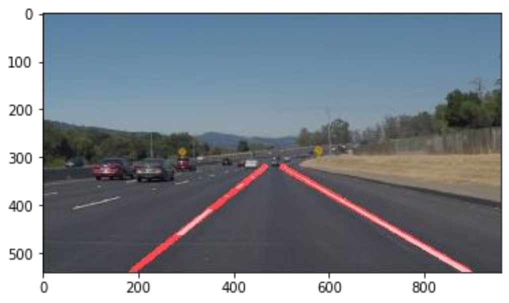


Figure 8 Final output showing connected segments

Reflections

The pipeline is tuned to work in certain lighting condition which make it not robust for all cases like the case of the Challenge video. The manual tuning is a big issue her, if we have parameters that is suitable for more cases it would be great. Some suggests working on HSV color space to eliminate the light intensity issue.