Finding Lane Lines

Overview

This is a basic approach to detect lane lines of a road. Since this is basic, the algorithms are written to detect most of the straight roads and not the curvy ones, with very little noises.

Reflection

My pipeline is consisting of 5 steps.

1. Convert the original image to a grayscale
2. Apply the Canny Edge detector with appropriate threshold values
3. Do a Gaussian Smoothing
4. Select the Region of Interest
5. Draw Hough Lines on the selected region
6. Keep the best fit lines (One for left and one for right) and eliminate the other lines
7. Weighted the lines

I took some specific frames of the given videos to test how my model is goanna perform in different road backgrounds.

And final applying these filters to a video stream as a pre-defined function (processe_image())

First 4 steps are general image processing functions. The real implementation is on draw 2 best fitted Hough Lines.

- To do so I first used the general Hough transformation in selected region.
- Then using polyfit() function I fit the 2 best lines for left and right lanes by taking there slopes. Positive slope for left region and the negative slope for right region/lane line.
- Then I defined a make_points() function to output the x and y coordinates of those 2 lines.(x1,y1,x2,y2)
- Final I called the draw_lines function so because of I build the necessities in hough_lines() and make_points() functions, the draw_lines() function could be used without any modification

2. Identify potential shortcomings with current pipeline

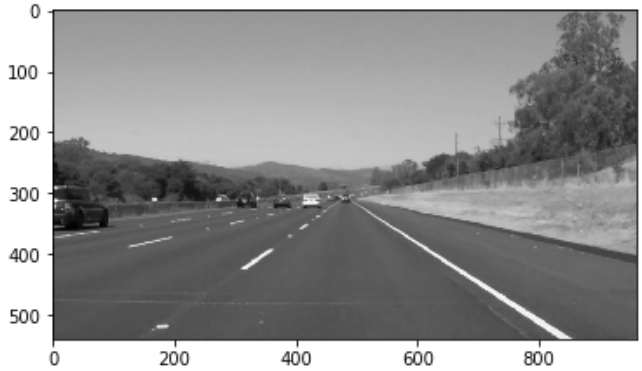
When the process_image() function is applied to the video stream, for some frames, the lines are jittering. Some glitches could be seen in the lines of the video.

This pipeline is not doing any good in ‘challenge.mp4’ video and was not able to function well in shadowing regions or less intensity road lines, even with completely deferent parameter values. (thresholds, max_line_length etc)

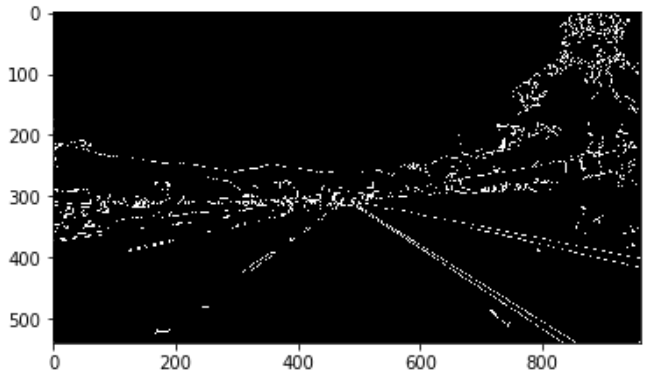
3. Suggested possible improvements to pipeline

This pipeline could be using some help of Convolution Neural Networks, in order to draw the best fitted lines specially in like cases of shadowing images and less intensity roads.

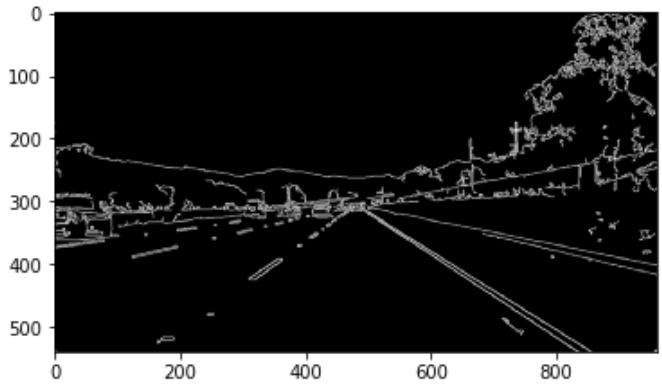
Here are some processed images of my software pipeline.



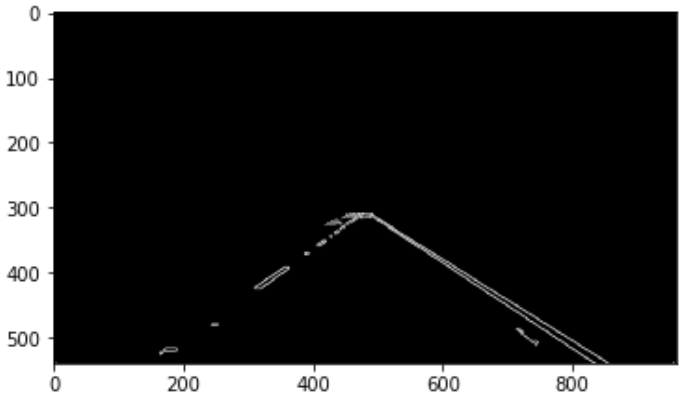
Gray-scaled Image



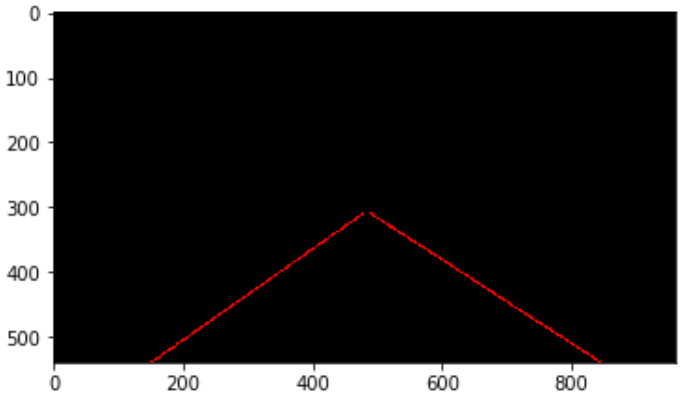
Applying Canny Edge Detector



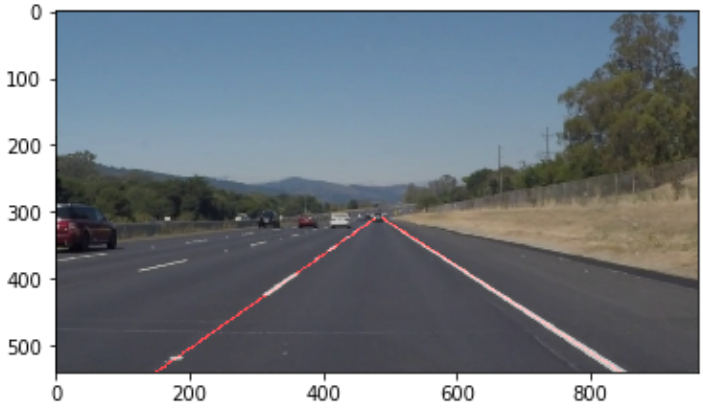
After Gaussian Smoothing



Cropping the Region of Interest



Weighted Lines



Final Outcome/Combo Image