Project1 Lane Line Detection

The goal of this project is to detect the lane lines in an image or video. The key steps *Canny edge detection* and *Hough transform* to find lines from the edge image

1. **Main Pipeline:**

step1: grayscale the image then apply Gaussian blur

step2: canny edge detection

step3: apply a ployfilled mask

step4: Hough transform w/ ***line extrapolation method\****

step5: combine the extrapolated lines with the original image using *cv2.addWeighted* function

***\*Line extrapolation method:***

First slope is calculated on the lines from Hough transform

line\_slope = (y2-y1)/(x2-x1)

Next, left and right slope is recognized by checking the sign of each line’s slope: *positive slope* is right line, *negative slope* is left.

Next, both left and right line slope are averaged using their *Median value* (slightly better than mean due to outliers)

Next average the x and y vertices for both left and right lines to get the *middle point* for both left and right line

Last the *top* and *bottom* vertices for both the left and right line can be calculated using their average slope and middle point

1. **Existing issues:**

*Color sensitivity:* the detected line is sensitive to the image color, white line is easier to recognize, yellow is less easy and a special difficult case is yellow line on concrete surface (the challenge video). **Solution is to keep fine tune the hyper parameters, but I suspect it still not enough due to the varieties of lane line and road color.**

*Line sensitivity:* some time there are very small segment of lane line exist in an image, in such case, the program skip adding extrapolated line on that image. **A potential solution is to use the line from previous frame, that should improve it quite a bit, I haven’t done it yet**

*False line sensitivity:* some time, the Hough transform produce non-lane lines and that mess up the average slope calculation. **I added the slope\_min and slope\_max range so that if see the calculated average slope exceed the range, it will be overwrite by a nominal value (0.6)**

Camera position and image size: in the challenge video, the image size is significantly different from the other two video. **It can be fixed by adjusting the mask vertices.**

1. **Hyper Parameters:**

'kernel\_size': 5,

'low\_threshold': 50,

'high\_threshold': 150,

'ignore\_mask\_color': 255,

# 'vertices': np.array([[(0,540),(450, 320), (490, 320), (960,540)]], dtype=np.int32),

'vertices': np.array([[(100,660),(550, 450), (700, 450), (1200,660)]], dtype=np.int32),

'rho': 2, # distance resolution in pixels of the Hough grid

'theta': np.pi/180, # angular resolution in radians of the Hough grid

'threshold': 15, # minimum number of votes (intersections in Hough grid cell)

'min\_line\_len': 30, # minimum number of pixels making up a line

'max\_line\_gap': 15, # maximum gap in pixels between connectable line segments

'α': 0.8,

'β': 1,

'λ': 0,

1. **Exhibit of detected lane line images**

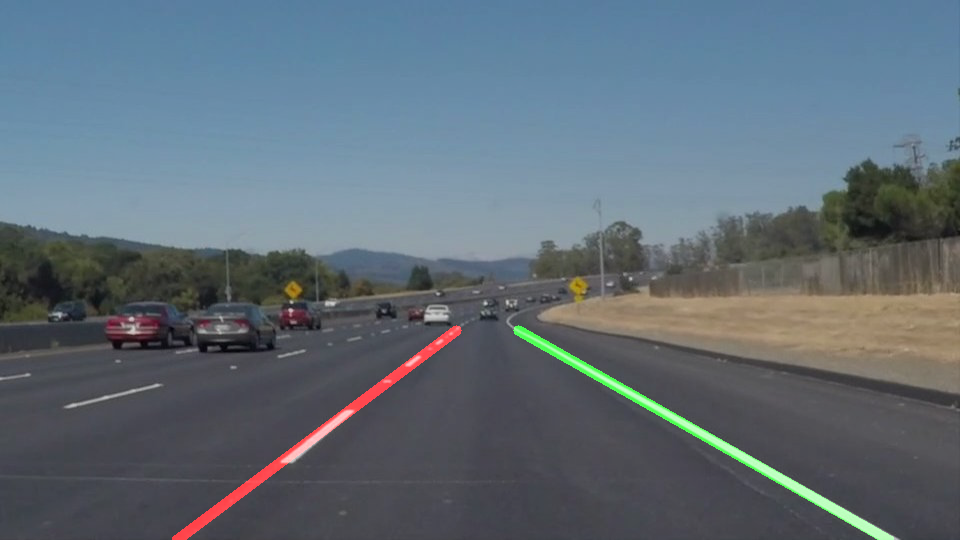


Figure . SolidWhiteCurve

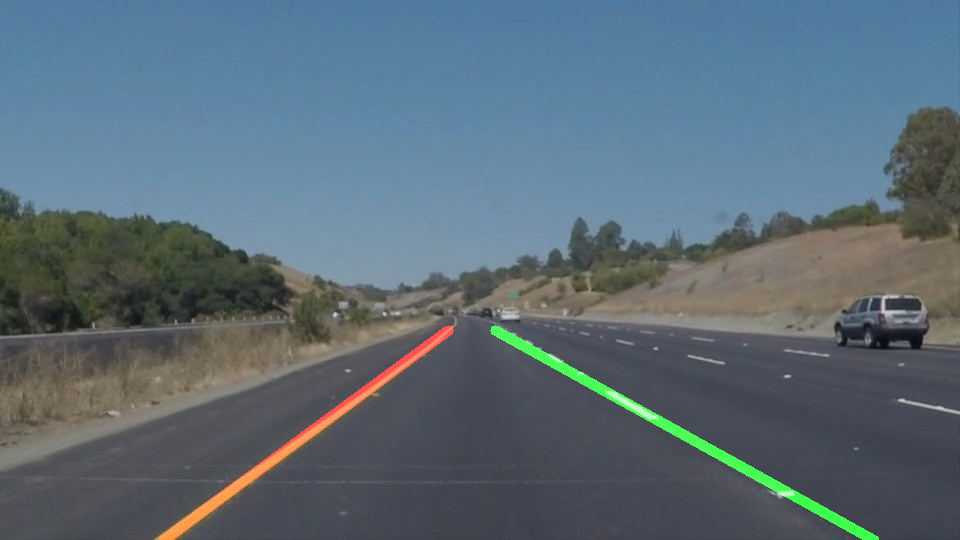


Figure SolidWhiteRight



Figure solidYellowCurve

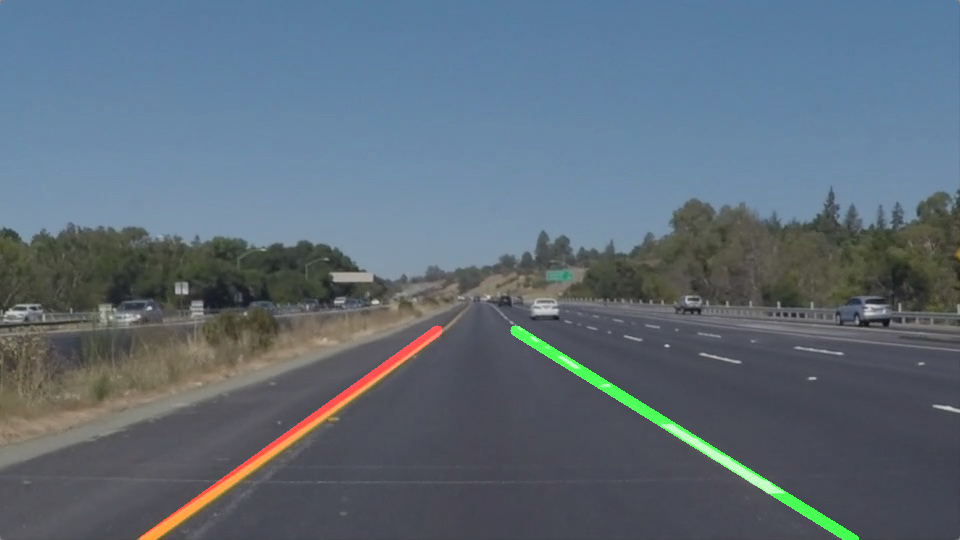


Figure SolidYellowLeft