Finding Lane Lines on the Road

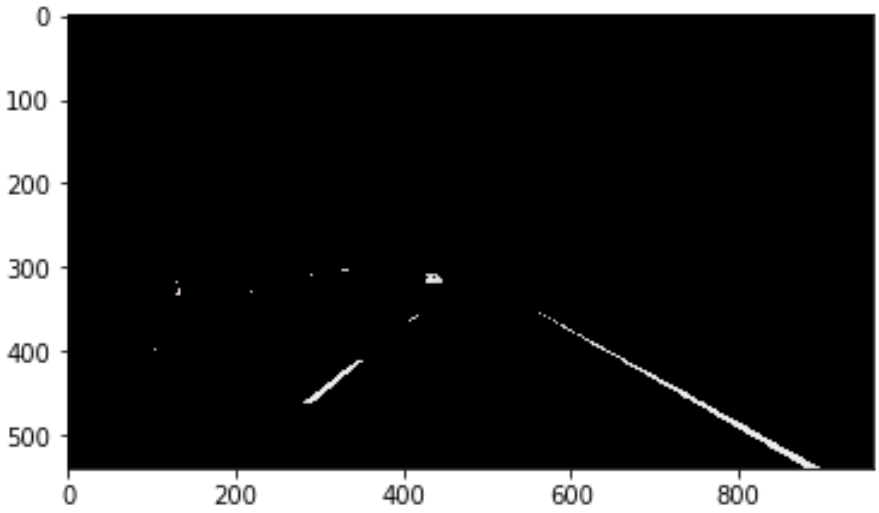
The motive of the project is to find lanes on image and then use the image to find lanes on the video. I have used Python and OpenCV to find lanes on images and videos.

1. Describe your pipeline. As part of the description, explain how you modified the draw_lines() function

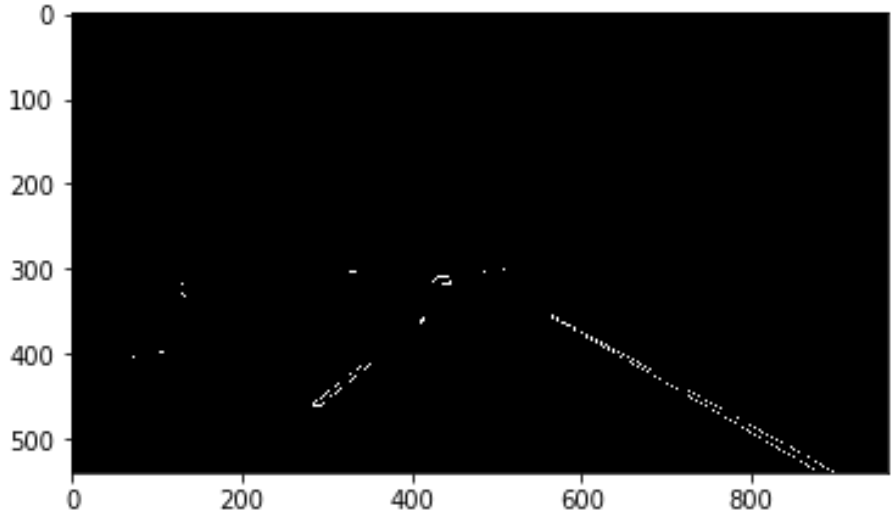
   a. Import the images from test_images folder and read the images using imread function of matplotlib.image library

   b. Apply the gaussian blur on the image to blur the image based on the kernel size.

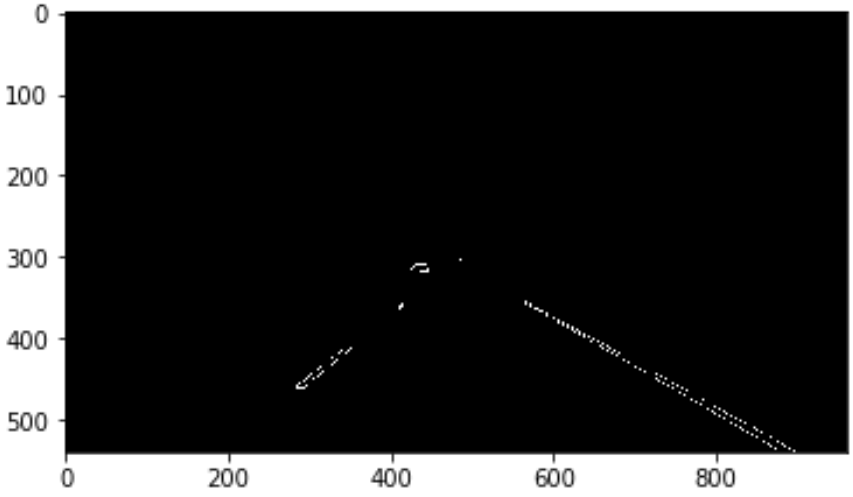
   c. Apply the color threshold to the image. The final image has pixels which are greater than a threshold and the pixels below the threshold are blacked out. In the below image, only the lane is seen, everything else is blacked out.



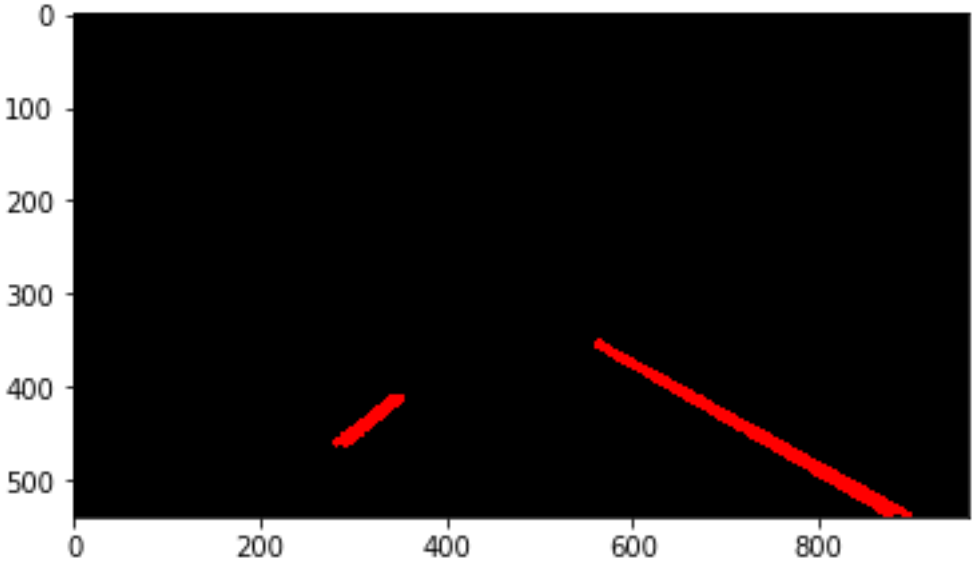
d. Apply Canny edge detection technique to find the edges of the lane lines in the image with low_threshold = 50 and high_threshold = 150



e. Find the region of interest by providing the vertices of the area to be taken into consideration. For this case, the vertices were (0,image.shape[0]),(430, 300), (500, 300), (900, image.shape[0]) which creates a quadrilateral around the lanes of interest. Thus, it just gives the image within this region, and blacks out everything else.
image.shape[0]) = 540

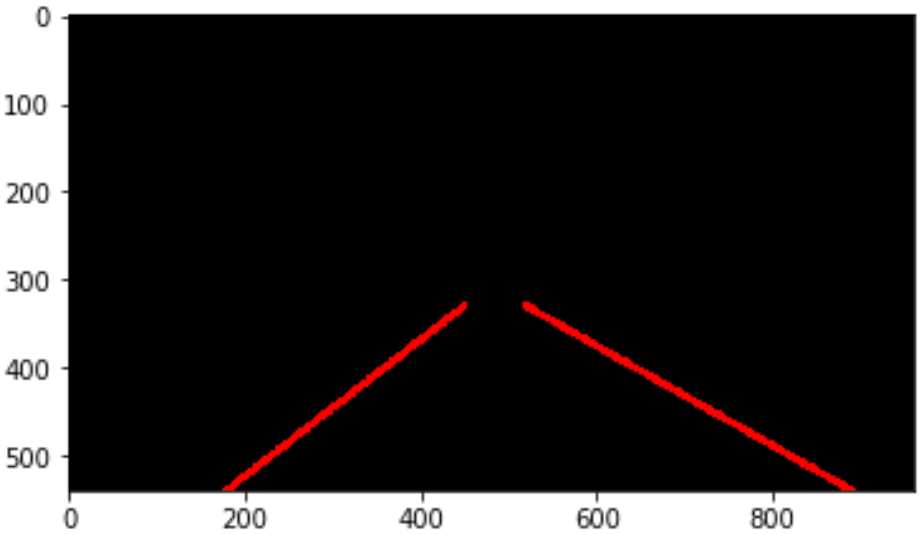


f. Apply the Hough transform to the above picture with rho = 1, theta = np.pi/180, threshold = 30, min_line_len = 30, max_line_gap = 20



g. Change the draw_lines function to average and extrapolate the line for left lane.
For extrapolating the dotted lane, following steps are taken:

i. Find the slope(m) for each line(2 points) and find intercept with that slope
ii. Check if slope is negative or positive. If the slope is negative, check if the slope is greater than -0.8 or less than -0.2, continue to the next line since the slope is too less or too high and the points are threshold.
iii. Similarly, for right line, the slope should be position and if the slope is less than 0.2 or the slope is greater than 0.8, the line is skewed and can be treated as the threshold.
iv. If the line is not a threshold, append the slope and intercept to either left or right values depending on slope.
v. Set y_max as image.shape[0] (540) which is the max value of y axis on the image and y_min to 330 which can be treated as the upper threshold limit of y value on the image.
vi. If the left or right slopes exits, find mean of slopes and intercepts for each line and find the left and right x values using the y = mx+c formula and correspomding values calculated in the above steps.
vii. Now the new averaged and extrapolated coordinates have been created.
viii. Create the corresponding line for the extrapolated coordinates.



h. Call the weighted image function which will apply apply the extrapolated lane image on the original images generating the following final image.

i. Save this image in the test_images_output folder and apply the same procedure for all the other images in the test_images directory.

2. Identify potential shortcomings with your current pipeline.

A possible shortcoming to the current pipeline is it wont properly work on sharp turns. It is designed for straight roads and wont be able to detect curvy/sharp turns easily.

Another shortcoming can be that the current approach has al the hard coded parameters, so it might not work where the region of interest is out of these hard coded values. It might also not work on up-hill or down-hill roads where the region or interest might change.

3. Suggest possible improvements to your pipeline

The possible improvement may be to include the algorithm to work for curves/sharp turns. To do so, the x and y values can be calculated from the equation of curves instead of y = mx + c

Another improvement can be to use flexible region of interest based on the type of road.