Finding lane lines

Vashist Valsaraj

January 2021

Introduction

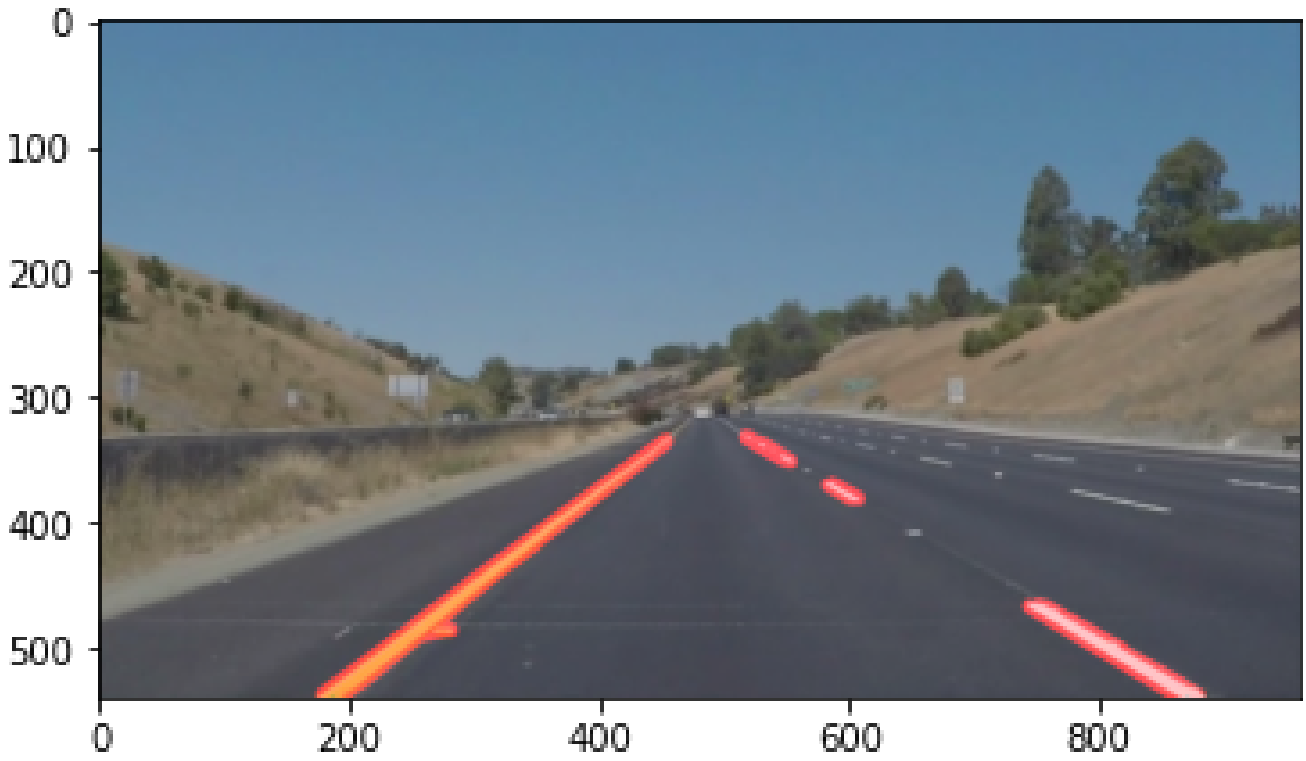
Finding Lane Lines on the Road

- Make a pipeline that finds lane lines on the road
- Reflect on your work in a written report

Reflection

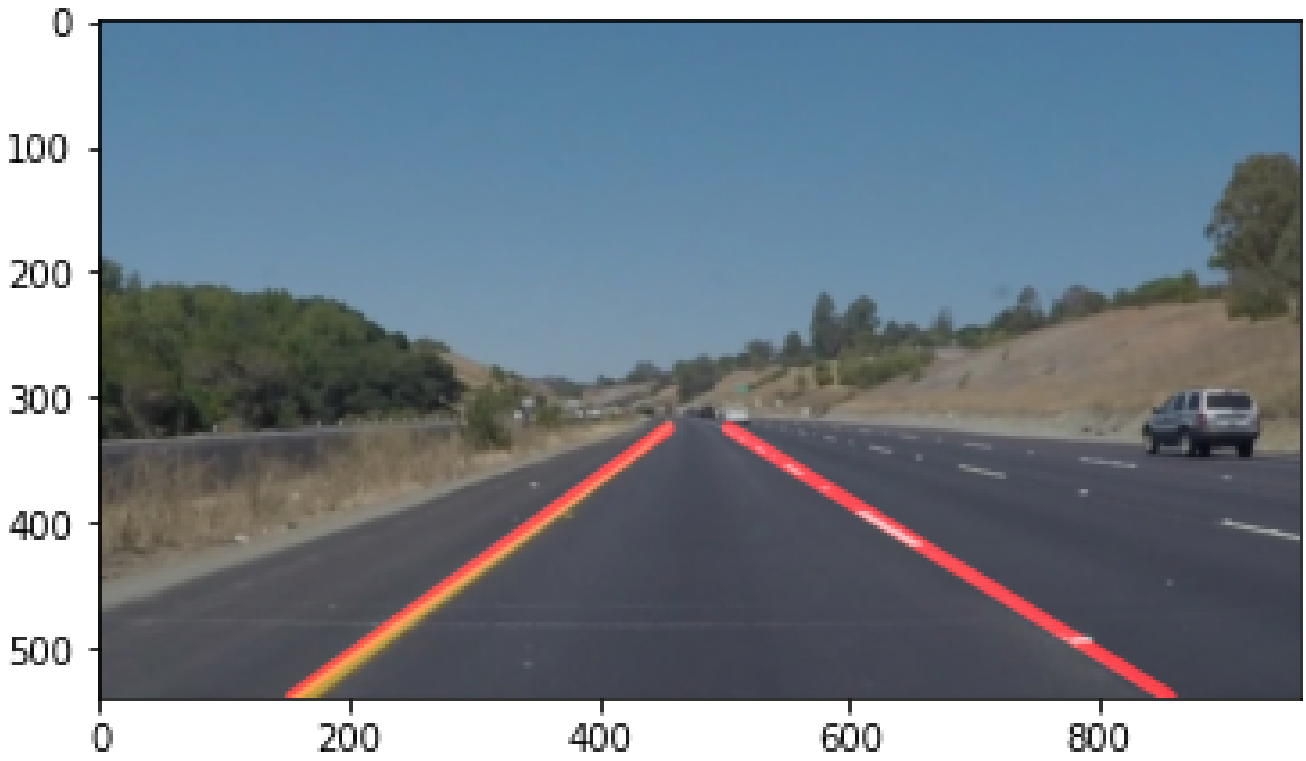
Pipeline

- Initially, I tried to find the white and yellow pixels in the image.
- Then I went on to find the region of interest and blacked out every pixel that was outside my region of interest.
- I converted the RGB image to a gray-scale image as the canny algorithm would find the edges where the pixel values in each x and y direction varies a lot.
- I applied a Gaussian Blur over the image to eliminate certain noisy pixels and areas. I chose a kernel size of 5.
- Then I applied the canny edge detection algorithm to find the edges in the image. I used 50 for the low threshold and 150 for the high.
- Finally I performed a Hough transform with Rho = 1, Theta = $\pi/180$, threshold = 10, minimum line length = 20 and maximum line gap = 15.
- I then overlayed the lines over the original image using the weighted average function.



Averaging the lines

Initially, I accumulated the slopes of the lines into 2 separate lists based on whether the slope was positive or negative to differentiate left and right lane markings. I kept track of the bottom extreme points for each case. I then found the mean slope of each lane and used the point to find the end points in my region of interest. Then I drew these lines over the image.



Potential shortcomings with current pipeline

- These lane markings wouldn't work in lanes that are curving.
- If the lanes are parallel to the image plane, the we might get NaN values while calculating the slope while averaging all the lines.
- Distortion at the ends of the image makes the lane markings go inward into the road.

Possible improvements to pipeline

- We could implement a sort of tracking mechanism, like averaging the last three frames of the image with the previous images being weighted down, so as to avoid jerks in the line being predicted.
- We could tune the hyper-parameters better to get a much better lane marking.
- Instead of averaging the lines in the m and b domain, we could average the lines in the rho and θ domain just as it is done in hough transform.