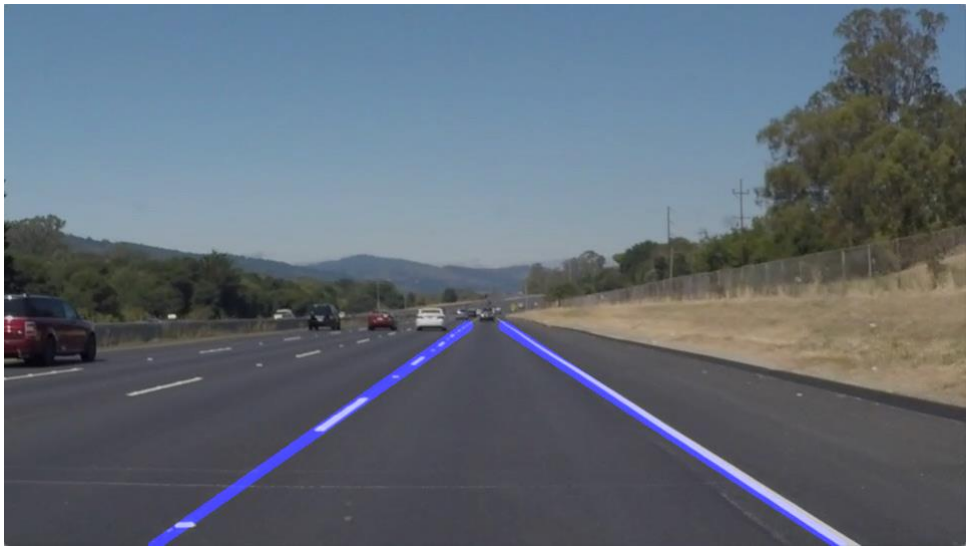
Reflection

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

My pipeline consisted of 6 steps.

1. Imported the image and converted it into grayscale image
2. Applied Gaussian smoothing to suppress noise and spurious gradients by averaging
3. Applied Canny function to detect the edges
4. Masked the region of interest in the image
5. Applied Hough transform to connect the dots (output from canny function)
6. Last step was to create a solid line (and not dashed lines) which detects both – left & right lanes. To do this I modified the draw_lines() function as explained below.
    a. I applied a polyfit function to find the linear fit of the lines detected by Hough transform. So I was able to get the slope & intercept for each of the lines.
    b. In order to segregate the dimensions of left lane and right lane, I simply observed the sign of the slope. If the slope was +ve, we know that it should be part of left lane whereas if the slope was –ve, it should be part of right lane.
    c. Subsequently, I took median of slope and median of intercepts and extrapolated them to draw it on the image



Above is a sample output

2. Identify potential shortcomings with your current pipeline

One potential shortcoming would be with the curved roads. Algorithm won't be able to detect the lanes properly if the road is curved.

3. Suggest possible improvements to your pipeline

A possible improvement would be to make the algorithm work on curved roads. Instead of averaging the slope & intercept and extrapolating it to draw it on the image, we could split the image into multiple smaller sections so that we can compute the slope & intercept separately for each small region and then connect them to find the lane.