Reflections

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

    My pipeline functions as follows:

    1. The image is read, a copy is made, and the image is converted to gray scale
    2. The gray image is then processed using Gaussian smoothing
    3. The smoothed image is then processed through a canny edge detector to find edges in the picture
    4. A masking algorithm is then used to hide away parts of the picture that are not the lane lines
    5. A Hough transform is used to define lines on the image where edges are found
    6. Within the Hough transform the draw_line() function is used to draw the lines, this function works in the following way:
        a. First the lines found by the Hough transform slopes are calculated. Lines with a negative slope and below a certain distance threshold are left lane lines and lines with a positive slope and above a certain distance threshold are right lane lines
        b. Once a line is determined to be a left or right lane line they are placed into arrays. These arrays are then passed into a numpy's poly fit function and poly1d to interpolate the lines of best fit.
        c. With these lines of best fit for the left and right lanes, the region_of_interest points are used to define where the lines go and they are draw using open cvs line function
    7. Once the lines are drawn, they are placed onto a copy of the image using the weighted_img function.

2. Identify potential shortcomings with your current pipeline

    Some of the potential shortcomings of my pipeline include changing lighting conditions (shadows, night time, changing road color) and curved lines (due to rudimentary region_of_interest approach using fixed polygon rather than auto updating the region_of_interest depending on the picture). Both of these will currently break my pipeline as can be seen in the optional challenge

3. Suggest possible improvements to your pipeline

    A possible improvement that could be made to my pipeline is keeping some of the data from the previous frames during the video to try and reduce some of the gitter during the video.