Finding Lane lines

The goals / steps of this project are the following:

* Make a pipeline that finds lane lines on the road

* Reflect on your work in a written report

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

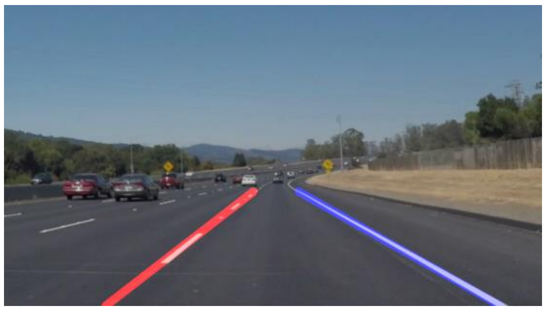
The pipeline consisted of several steps as listed under:

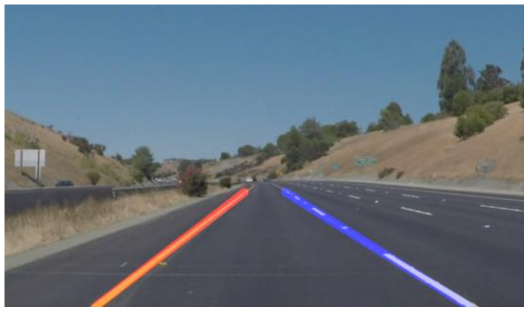
a. The images were converted into grayscale
b. The Gaussian smoothing/blurring function was applied with a kernel size of 7. This value was chosen by playing around with kernel size to see the difference between several values such as 1,3,5,7. Any value of 5 or 7 seems to return similar results.
c. Further, A canny transform was applied on the smoothed image. The low and high thresholds were tuned to be 60 and 100 respectively. The general rule of high threshold being 1.2 -1.5 times of low threshold were considered during the tuning of thresholds.
d. A region of interest mask was applied based on images.
e. Hough transform was applied on the image with the masked region of interest to detect the different line segments
f. Furthermore, the draw_lines function was modified accordingly:
    a. Based on the slope of the lines, the line segments were divided into segments belonging to the left lane or the right lane.
    b. For each line segment belonging to left or right lane, the slope, the intercept (these 2 are critical to define a single line) and the lengths of the line segments was stored
    c. Based on the lengths of the line segments, a weighted average of slope and intercept was calculated for left and right lane separately.
    d. For each lane, the distinct points were calculated using slope. Interface and 2 different Y values.
    e. After finding out the distinct points, cv2.line function was uised with different colors for left and right lane

The plotted lines on the image are representative of the detected lane lines from the pipeline.

Shown below are a couple of example images processed:

[***Note***: Please refer to the 'test_images_output' folder in the working directory for processed images of all the 6 images provided]





Identify potential shortcomings with your current pipeline

There are several potential shortcomings of the current pipeline:

1. Any solid line in the processed video is quite stable. However, the alternate line detected is very jumpy and sometimes is incorrect
2. Since it is a line segment that is detected, any curvature in the lane lines is not well represented.
3. Optional Challenge processed video is horrendous. Pipeline is not able to understand when the image is darker than usual.

Suggest possible improvements to your pipeline

These are some improvements to the pipeline that I could think of:

1. The draw_lines function needs further improvement. Some improvements to reduce jumpy nature are:
    a. Possible to include lane width in calculation to maintain the lane width at all points
    b. Find the mean line of left and right lane. Add half width obtained from previous step to make the lines more stable
2. Use cv2 function to introduce curvature
3. Tuning of the pipeline parameters such as:
    a. Threshold on canny edge detection- Still need to play more with this parameter
    b. Hough transform parameters such as rho, threshold,etc.