Finding Lines on the Road: Reflection

1. Pipeline Description

My pipeline consisted of following steps:

1. Converting image to the gray scale
2. Appling Gaussian Blur filter
3. Finding edges using Canny edge detector
4. Defining mask vertices and masking the image
5. Finding lines using Hough Transformation
6. Drawing Lines on the image

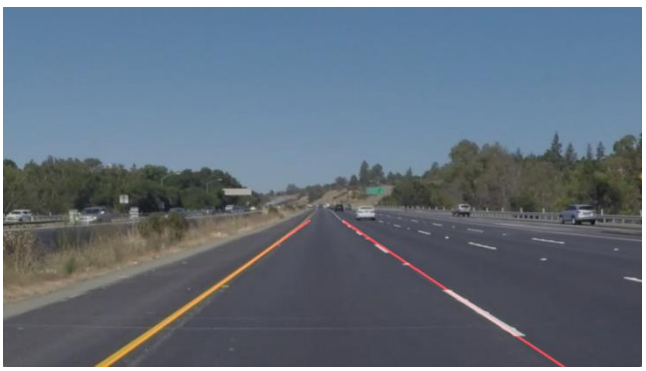
In order to draw a single line on the left and right lanes, I modified the draw_lines() function by:

1. Classifying lines into right and left lines using image slope: (y2-y1)/(x2-x1) i.e. right lines have a slope lower than zero and left lines have slope higher than zero
2. Averaging all right and all left lines
3. Finding coefficients of linear polynomial for averaged right and left line using polyfit funtion
4. Drawing (extrapolated) lines using coefficients estimated in the step 3 for the points having y1 = image_height and y2 = image height/2 + 50

If you'd like to include images to show how the pipeline works:

solidYellowCurve2:

solidYellowLeft:



whiteCarLaneSwitch:

solidWhiteCurve:

solidWhiteRight:

solidYellowCurve:

2. Identify potential shortcomings with your current pipeline

My algorithm is constrained to search for straight lines. If the road is much curved, than straight lines wouldn't fit, as can be seen in the "challenge" video.

Furthermore, the image mask is designated to fit a little bit below the horizon. This assumes that horizon begins approximately at the middle of the image. If the horizon is relatively near the bottom or the top of the image, this might cause problems.

3. Suggest possible improvements to your pipeline

One possible improvement would be to try to fit not only to straight line but also curves by using for example the general Hough transform.

Another potential improvement could be to detect the horizon and then used the detected horizon position to define the mask.