PROJECT 1: FINDING THE LANES ON THE ROAD USING COMPUTER VISION ALGORITHMS

GOALS OF THE PROJECT:

- To Write a pipeline to detect the lanes on the roads on images.
- To Stream a video and apply this pipeline frame by frame.
- Extrapolate the lane lines on the video and reproduce the annotated video with the lane lines within which the car should be within.

STEP BY STEP EXPLANATION OF THE PIPELINE:

1. Retrieve the image and make the copy of the image. This will be helpful for us to append the line marking after detecting the lines.
2. Convert the image to greyscale. This reduces the image to a function of brightness and in-turn helps us to simplify our processes and work with only one value, in terms of edge detection.
3. We apply a gaussian blur to filter out higher frequency noises. We do this because, the canny edge detector is method based on gradient, the high frequency noise will be erroneously detected as edges, which further increases processing complexity.
4. We need to decide our area of interest on the image. This will again help us reduce processing complexity by blacking out the rest of the image.
5. We carry out hough transform to detect straight lines in the image. This is theoretically done by checking out the intersection point in the hough space in the hough grid.

PARAMETERES TO TUNE:

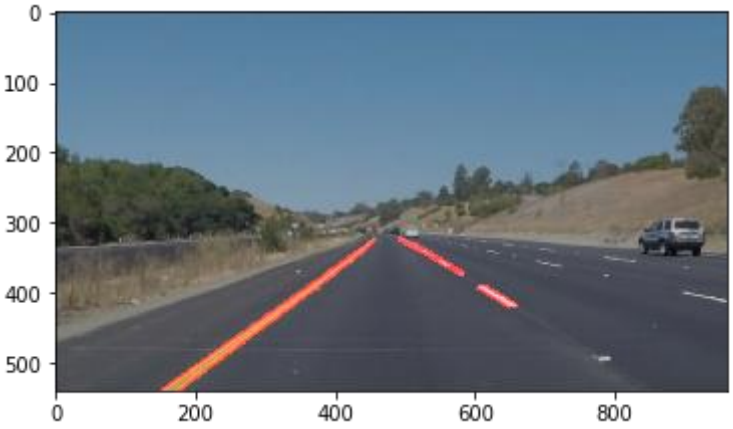
1. **CANNY:** upper threshold and lower threshold. The values above the upper threshold is retained and the values below the lower threshold is eliminated. The values between are retained if the pixel is connected with values above upper threshold.

   The final lower and upper threshold we set ended up to be: 50 and 175 respectively.
2. **The hough parameters:**
   rho = 2 # distance resolution in pixels of the Hough grid
   theta = (np.pi/180) # angular resolution in radians of the Hough grid
   threshold = 15   # minimum number of votes (intersections in Hough grid cell)
   min_line_length = 40 #minimum number of pixels making up a line
   max_line_gap = 20# maximum gap in pixels between connectable line segments

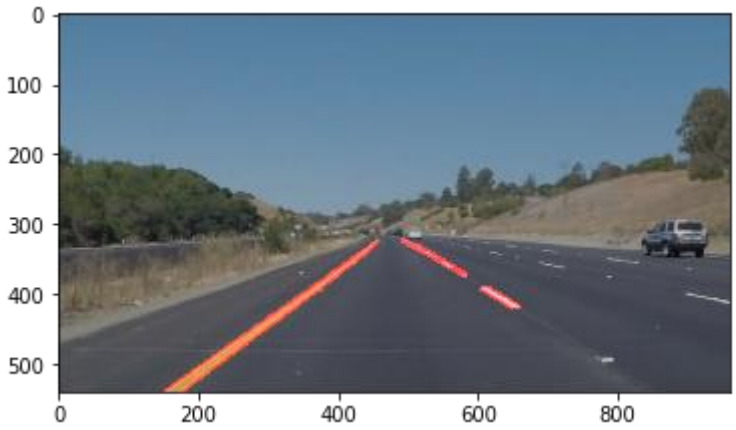
Images after implementing the pipeline above.

SOLIDWHITERIGHT

SOLIDWHITECURVE



SOLIDYELLOWCURVE



SOLIDYELLOWCURVE2

We need to improve on this problem now. We have to improve on the draw lines() functions to extrapolate the line till the end and give two fixed boundaries, the car should travel in.

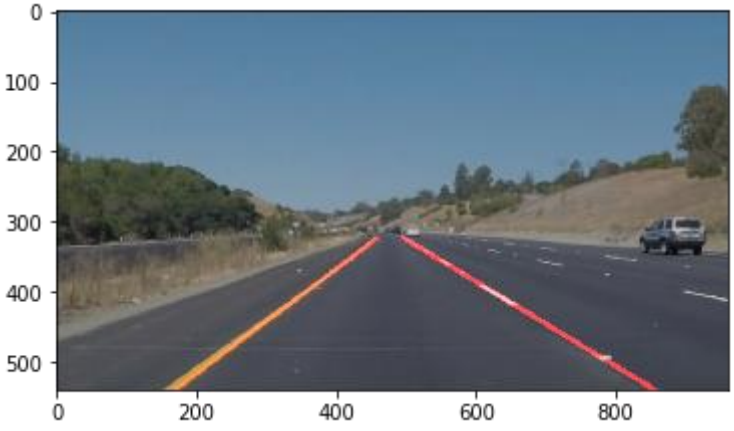
IMPROVING THE DRAW LINES FUNCTION:

1. We have the starting point and the ending point of all the lines now. We need to extrapolate the line based on the points.
2. First, sort the points(left or right) based on the slope between the starting and ending points. The slope formulae- (y2-y1)/(x2-x1)
3. Store all the left slopes and right slopes too.
4. We have the basic line equation y=m*x+c where m is the slope and c is the y intercept.
5. Remove all the slopes which is of high variance from the mean left and mean right slope.
6. Now we need to compute the mean y intercept(c mean intercept) for both left and right lanes.
7. We need to compute the average x-intercept using the average slope and average y intercept. (average x-intercept, 540) is the starting point of the extrapolated line.
8. Again using the average slope, average y intercept and the minimum y value, we can compute the corresponding x_value. (corresponding x_value, minimum y_value) gives the other end of the line.
9. Now we can draw a line between these two points.

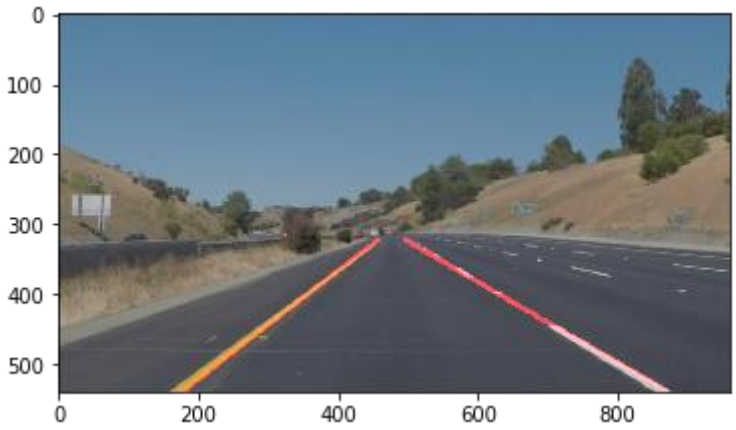
Images after implementing the above.

SOLIDWHITERIGHT

SOLIDWHITECURVE



SOLIDYELLOWCURVE



SOLIDYELLOWCURVE2

PROBLEMS:

There were compilation errors when there were no hough lines detected in the frame. This was solved by adding an if condition. But this also created a problem. There was a problem of lines not drawing in specific frames as no houghlines were created. A feature was added to the code such that a line is drawn between a fixed average position.

Noisy houghlines results in change in line angles during a video stream. This can be avoided by filtering out houghlines in horizontal direction.

Change in brightness or reflection on the road may result in detecting those specific areas as edges. This can be avoided by using different color filter and then applying canny edge detector.