Project1: Finding Lane Lines on the Road

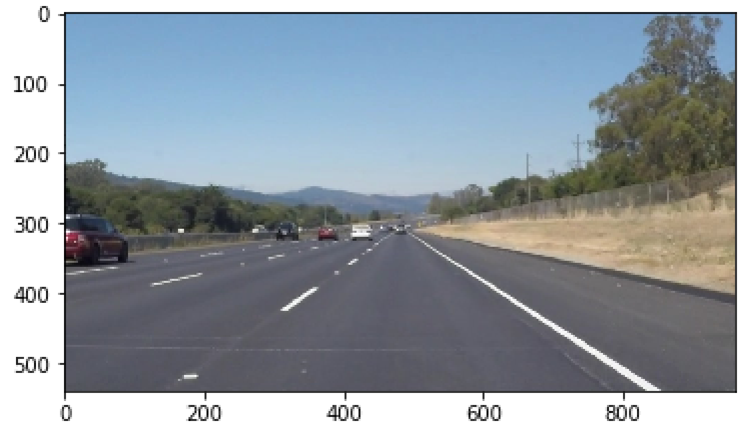
Introduction:

The goal of this project is to build up an image pipeline, which takes a frame from a video and process it then return a modified frame. Through this process, the lane line on the road should be detected and marked up.

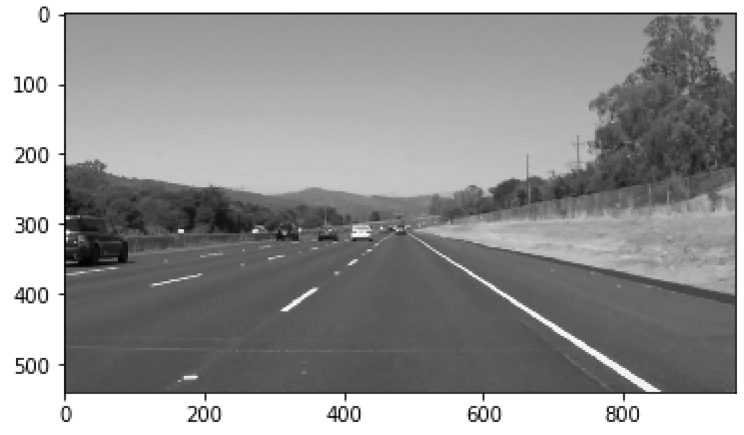
Pipe Line Discription:

The image pipeline consisted of 7 steps.

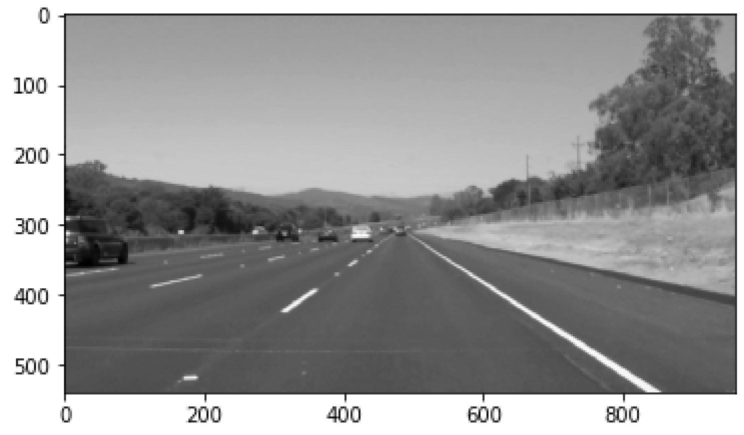
1.Read in image: At first, the pipeline takes a single 3 - channel RGB image



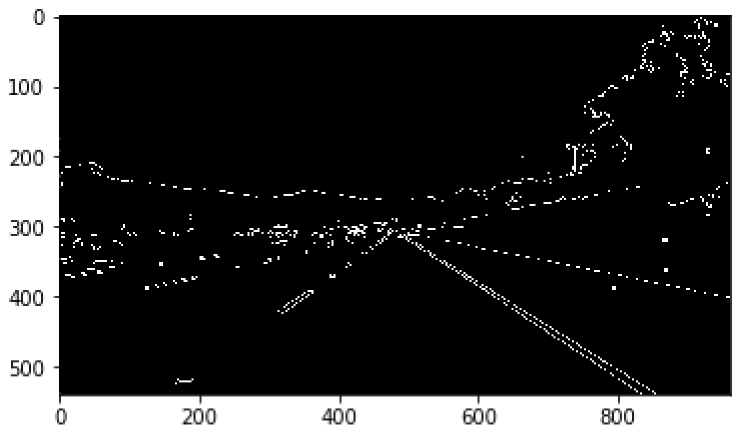
2.Gray scale transform: Then converts the images to grayscale



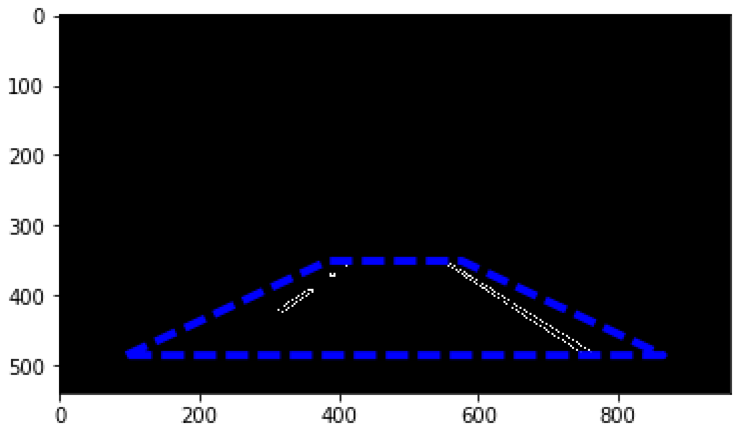
3.Gaussian smoothing: A Gaussian kernel is applied in order to blur the image and eliminate the noise



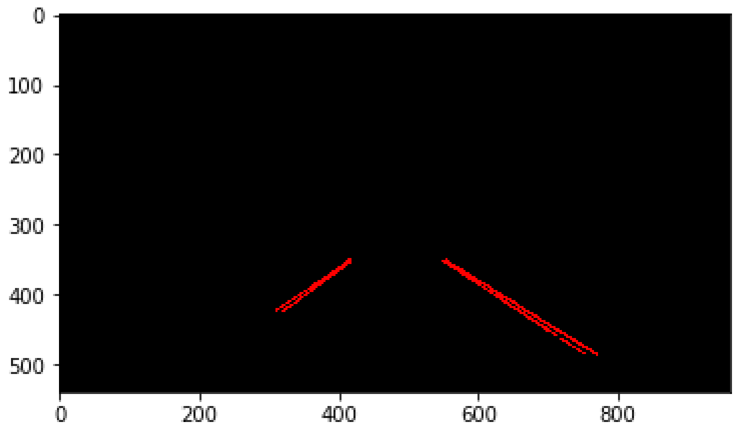
4.Canny trasform: Use canny edge detector to detect the edge of a lane line. The parameter of the gaussian filter and canny detector has been tuned in order to keep the detected lane edge always available and in good condition.



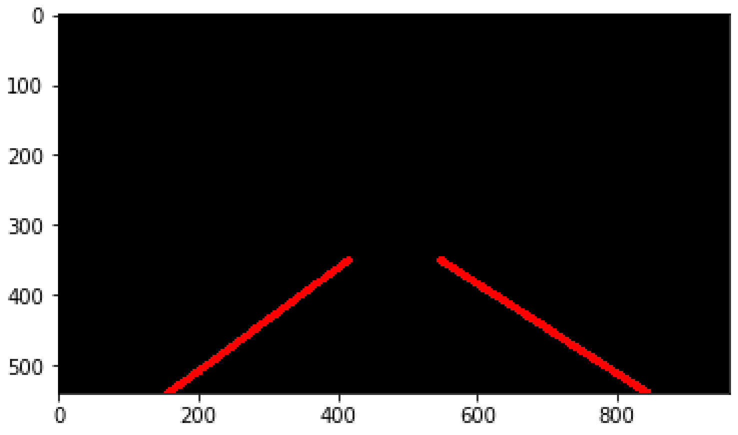
5.Region of interest mask: For next step, a mask of interst region is applied to filt out objects which is not interested in the image.



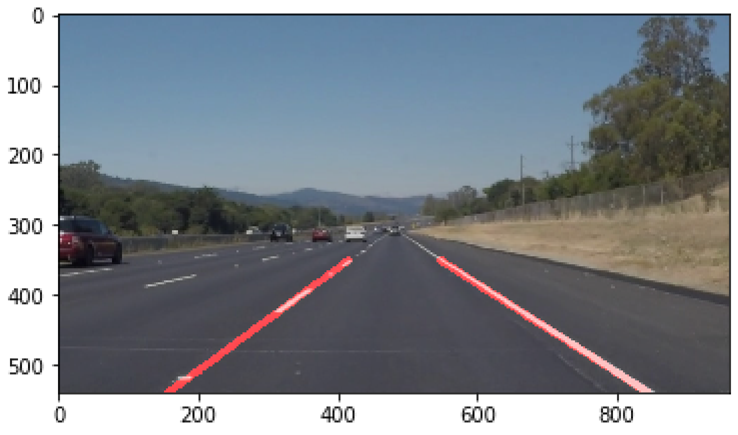
6.Hough Transform and Hough lines drawn: Apply Hough transform with a tuned parameters to find a lines in the image and draw red lines to mark out them on the image.



However, there are many red lines on the image and jitter a lot, so I've also modified the draw_lines() function by devide lines detected by hough transform into 2 groups: Left_Lane_Lines and Right_Lane_Lines. In each group, the mean slope and the mean intercept point are calculated then a single and solid line is drawn with these data.



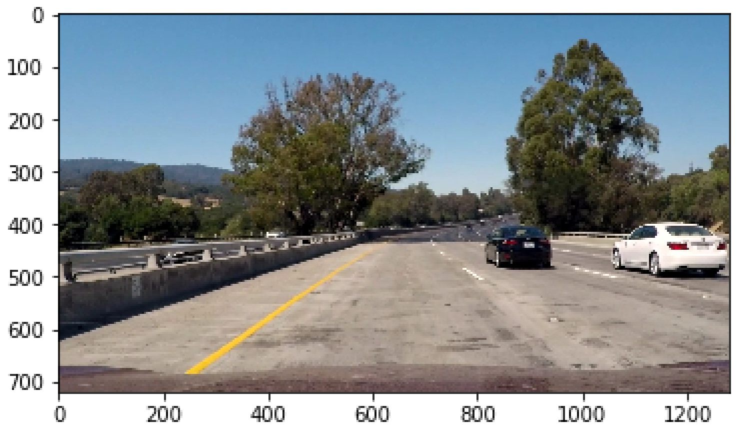
7.Return modified image: Combile the Hough line image with the original image and output the modified image.



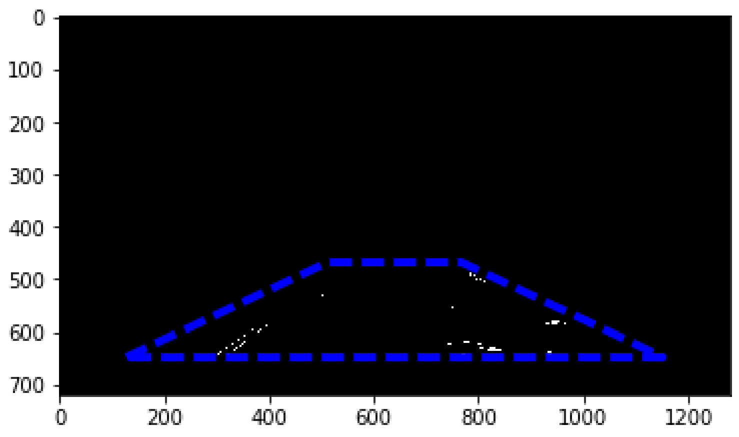
Potential Shortcomings with the Pipeline:

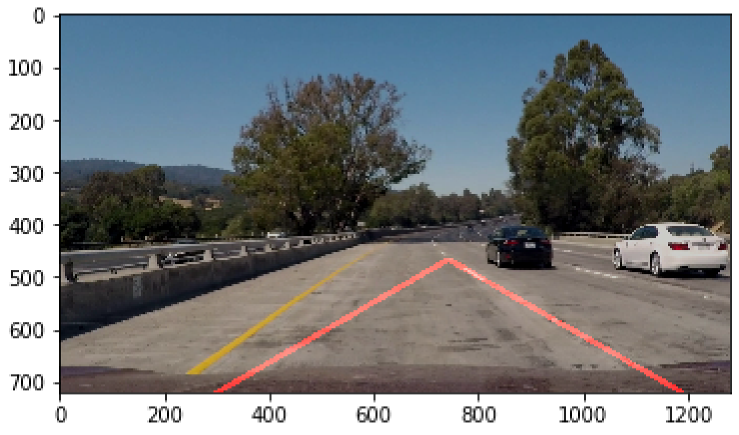
The pipeline seems working fine with a single picture. However, the detected lane lines are not stable enough while dealing with a video input. In order to get more stable result. I've been tuned the parameters in canny and hough transform.

Now,the pipeline works fine with the firest two test video. After moving to the challenge video, I've found one potential shortcoming. Take the following frame from challenge video as an example.

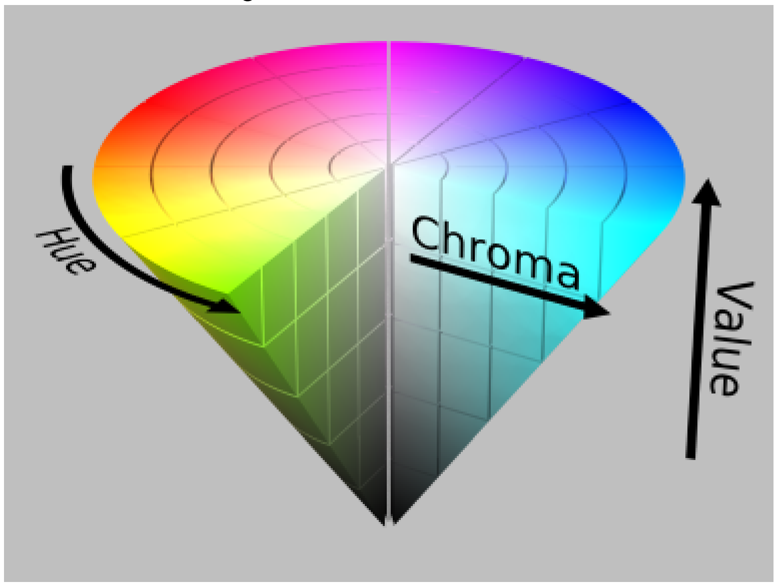


When the car drive through a shadow area, the lane line in yellow and the road in grey actually have the same brightness levels. This problem can be clearly shown under canny transform and clearly it's almost not possible to detect yellow line under this case. The pipeline is failed.



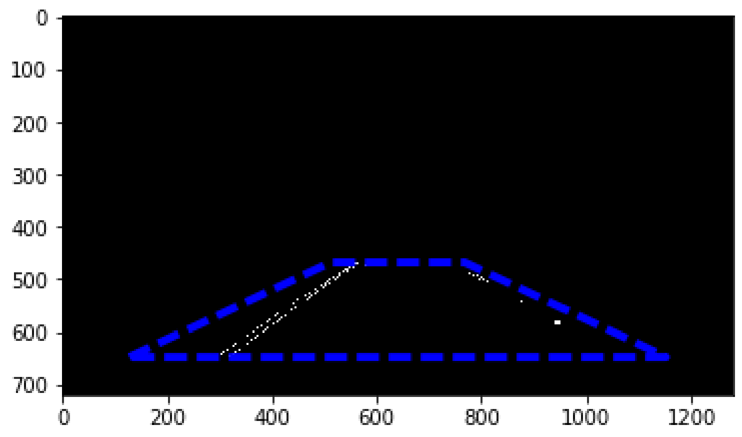


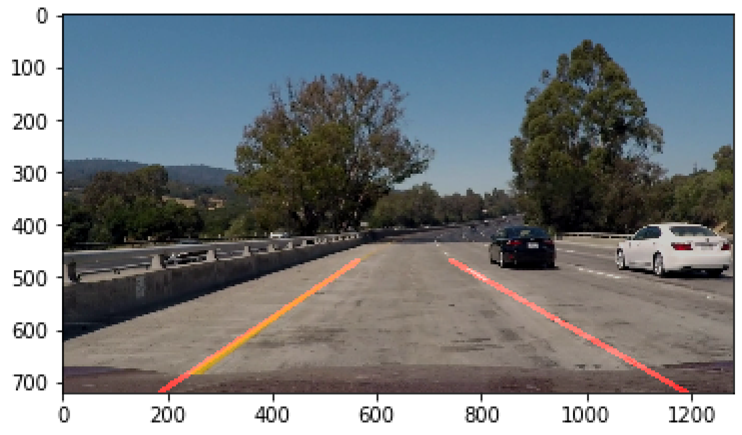
To overcome this problem, instead of using RGB2GRAY transform, a HSV color transform is applied.



Source: wikipedia

HSV is better as RGB when we want to detect a certain color. Especially under low light condition. By apply HSV into the pipeline to replace gray scale transform, the detail of lane lines are preserved even when the car is driving through a shadow area. The results are shown in the following figures.





Another shortcoming is that when the lane line or the vehicle motion is changing rapidly, the detected lane line will also jitter and not stable. To deal with that, I use a buffer, which store stable lines from the previous 30 frames. The new stable line will be the average of these data. This method improve the stability of lines segnificantly.

The last shortcomings is that, this pipeline use Hough line transform, which works not well when the turning radius is too large.

Suggest Possible Improvement to the Pipeline:

Although so far the pipeline working fine with all the cases in test videos, a possible improvement would be apply a kalman filter to further increasse the stability. By doing that, the drawn lane lines will be more stable. We can also use advanced hough transform to detect curvature to further improve to line fitting stability.

In [ ]: