**Project Video Code steps for lane detection**

1. We Read our video using MATLAB function “Video Reader” and run a for loop for each frame , so that we can apply our image processing on each loop.

2. Open writer object to store our outputs into one video , so that we can stitch them together for one video output.

3. Convert RGB Image to gray image and apply appropriate threshold to it , so that we ignore any sort of disturbances/marks on road.

4. Use canny edge detector and apply sobel filter for clean edges.

5. Define your roi region by using roipoly function, after getting coordinates ,apply those coordinates and use them to make polygon of interest.

6. All ones in roi represent polygon, so multiply element wise to our edge detected output.

7. Now we have edges in our image, we can apply concept of HOUGH lines into it and detect lines

**Points to notice in Hough lines :**   
a) Need to use less threshold for finding more number of lines, some frames may give higher number fo shorter lines or might not give any line, decreasing threshold will give more number of lines.

b) Use appropriate number in Fillgap (30) and minlength (26) for better detection of lines. These values can be lot lesser but some of the frames have higher number of lines as they can be closer to each other in hough plane.

8. Calculate slope for each line, if you run the code for each frame separately, the key observation made would be, all lines on right side of image have slope greater than 0.2 and left slope is less than -0.2. separate each point as leftpoints and rightpoinst arrays.

9.For each side also save slopes in array to calculate mean slope of each side. This would be used to predict turns. Define appropriate thresholds for turns depending on frame observations.

10. As we found our right and left points separately in each array. Now we can use polyfit to curve them into one line/curve (curve of higher degree gives weird output in some frames because of noise and badly detected points). Then use polyval(for the polynomial found from polyfit ) to calculate the y coordinates for evenly distributed x values.

11. Further, use max and min points of each to calculate the slope of final line which is getting plotted.

12. If lines are in roi and not crossing road, we can extrapolate the lines using Matlab function interp1 and using linear way of extrapolation.

13. If we have found all the lines and they are present in area of interest, we show patch on road using “patch” function .

14. After lanes are detected, we need to convert the output figure into frame and save it to the video using writeVideo function.

15. Running code faster, we use command “draw now” command before each plot/drawing/patch. If used before all plots/patch, Whole video takes around 10~20 mins to run.

**Challenge Video Steps:**

Here we are using same method for lane detection as shown above but using HSV color space for detecting lanes before detecting edges.

1. Convert your rgb image to HSV space.

2. Create a mask of appropriate mask with appropriate threshold to detect yellow and white lines separately.

3. Convert the threshold values into uint8 so as to be able to combine it with rgb image.

4. Using bwareaopen to clear out small holes in image.

5. combine mask and rgb images to get an image, and then proceed to convert it to black and white image to perform further operations. Then add white & yellow detected black/white image (56 line).

6. Quick observation: It was better to not apply canny edge detection and give image directly to houghlines and get better results.

Now repeat steps 7 to 15 as earlier to get the Video Output.

**References:**

Used references, guidelines and links attached in Project 1 question. For challenge video:

1. https://www.mathworks.com/matlabcentral/fileexchange/28512-simplecolordetectionbyhue--

For detecting colors in lane.