Finding Lane Lines on the Road

Objectives:

- Build a pipeline that finds lanes on the road

Pipeline Description:

- Inorder to ease up the process of tuning the parameters of Canny and Hough Lines, I approached the project by creating gui based tuners

- The first one is draw_roi.py in which upon running with a video file you can select the region of interest using your mouse clicks and creating a polygon.
    - The video is looped infinitely so that you have enough time to select or deselect the vertices.
    - You can deselect the points by right mouse button
    - Finally after selecting the 4 points you can press 's' key to print the vertices on the terminal and close all windows and exit the program execution

- Coming to the main project, it is run by hough_tuner.py and tunerUtils.py, This is where the actual lane finding happens.
    - The hough_tuner.py is a main() function and calls the class Tuner from tunerUtils.py with the filename of video passed to it as arguments
    - The class tuner will create the trackbar for parameters of gaussian blur, canny and hough lines such as – Kernal size, low and high thresholds, No of Votes, Minimum Line Length and Maximum Line Gap.
    - A window will show the edges detected by the canny which is very useful to adjust your parameter values and visualise how good your parameters are.
    - Further down the execution is to classify the left and right lanes from the points returned by hough lines function, this is achieved by calculating the slope (m) from x1, y1, x2, y2 points and if the value of m is positive, it is left lane and if the value of m is negative, then you have the right lane.
    - Once the lanes are classified then we use fitLine() method to get vx, vy, x, y and using this values you can draw a line by calculating x1, y1, x2, y2. This step will reduce your noise and disturbances to an extent.
    - Th next step is to combine the detected lane line with the input image using addweighted() function and on the result window you get the annotated video of lane markings.

Potential Shortcomings:

- One of the short coming with this code is that if the vehicle is on a curve then the detected lane lines will be erroneous.
- Another shortcoming will be if the input frame has a horizontal line with very sharp edges (like that at an intersection) within the region of interest then the lane detection will again fail.

Possible Improvements:

- To make the lane detection working on curves, I may have to avoid using hough transform and use another method that works better on detecting curved lines or using quadratic/cubic regression techniques rather than trying to fit a linear line
- To avoid the algorithm to detect horizontal lines, I would approach to use a tolerance level for the slope of lane lines and if the detected slope is out of bounds then replace with an averaged value of slope in that frame.