Finding Lane Lines on the Road

Finding Lane Lines on the Road

The goals / steps of this project are the following:

* Make a pipeline that finds lane lines on the road

* Reflect on your work in a written report

1. Describe your pipeline. As part of the description, explain how you modified the draw_lines() function.

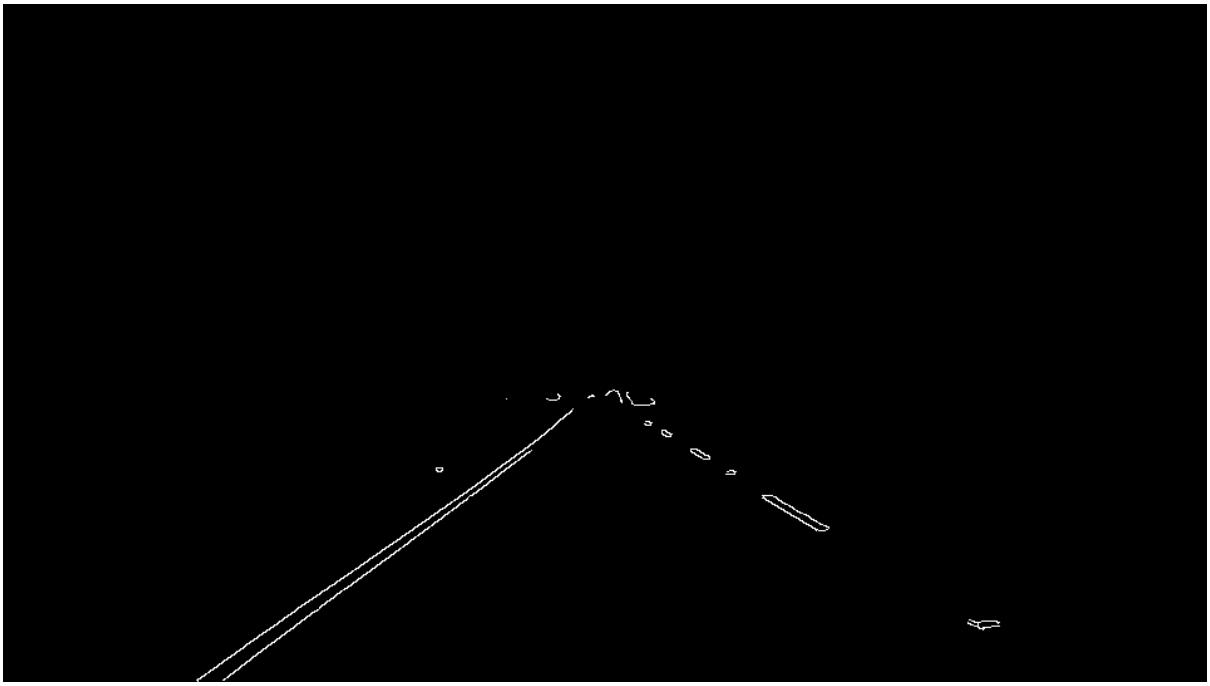
For making a pipeline that can find lane lines on the road, my pipeline is consisting of 5 major steps which are explained as follows –

1. First we need to convert our colour image into a grayscale image

2. Now we will add Gaussian Blur into this grayscale image. I have used kernel of 5 for this Gaussian Blur

3. Now in this Gaussian Blurred image, we apply the Canny Edge detector with a low threshold of 150 and high threshold of 250 to detect the edges

4. After applying the Canny Edge, we now extract the Region of Interest which is only the lane lines in the road



5. As soon as we extract the region of interest, we apply Hough Transform to draw the extrapolated lines in our original image as shown

The Hough Transform parameters which I have chosen to get this result are as follows –

- Rho = 1
- Theta = pi / 180
- Threshold = 40
- Min_line_length = 1
- Max_line_gap = 10

The extrapolated lines as seen in the last step are drawn using an iterative process, which means that repeatedly slopes of the left lane and right lane are detected if they are positive or negative and then they are accumulated in separate lists and then drawn according to our simple equation of a line y = m*x + b.

2. Identify potential shortcomings with your current pipeline

Since this lane detection pipeline is one of the simplest, it assumes that we only have straight lane lines on a road and because of which we can see that in our images the straight lane lines are detected quite well. But, in real life, all the lanes are curvilinear in nature. The lanes and roads both are always turning from left to right and right to left which means it is not straight, so this lane detection is not generic and will not work well in real life situation.

Another problem that might occur is when we have lots of cars and obstacles in our roads and when they are interfering with our lane region of interest. In this case, the lanes will not be detected properly and the results will be highly unreliable for real life driving situations.

The last shortcoming which I can think of is when the lanes merge where this pipeline is definitely going to fail as lane merging are complex in nature and this algorithm doesn't account for those complex changes.

3. Suggest possible improvements to your pipeline

One of the most intuitive improvements to this pipeline can be using s curvilinear equation to account for the curvilinear changes in the lane lines. This will be able to grasp the dynamic changes occurring in the lane lines.

Another improvement which can be made is that to include information of various other factors such as obstacles, cars, pedestrians etc as to account for the interference caused by them.

Much of this algorithm can be improved if drivable surface detection methodology using Semantic Segmentation is used where the road segments or drivable surface can have their own specific segment separated from other objects as categorical data.