Finding Lane Lines on the Road

Writeup Template

You can use this file as a template for your writeup if you want to submit it as a markdown file. But feel free to use some other method and submit a pdf if you prefer.

Finding Lane Lines on the Road

Reflection

1. Describe your pipeline.

My pipeline consisted of 7 steps. The steps are inspired from the study by Li et al. (https://www.hindawi.com/journals/am/2018/8320207/).

1. Filtering out the white and yellow colors from the image Since the lanes are in white and yellow colors, we can concentrate only on the white and yellow parts of the image. Hence, the first step in the pipeline is to extract the white and yellow parts of the image. This removes the unnecessary processing of the unrelated parts of the image.

To do this task, first I tried to extract the required regions defining ranges in the RGB channels. Later, I found that HLS and HSV provide better representations. Hence, the final implementation extracts yellow and white color regions from the image in the HLS domain.

2. Grayscaling the image The next step to do is extract all the lines in the image which are potential lanes. To detect edges in an image, we do not need the color information. We just need the luma values. Hence, grayscaling the image to remove the unnecessary color information is the next step.

3. Blurring the image Noise exists in high frequencies. Here also, to decrease the false detections and combine nearby lines, we blur the image using Gaussian Blur of kernel size 15x15.

4. Detecting Canny Edges Now that our image is ready to detect the potential lanes, we detect edges using Canny edge detection.

5. Selecting the region of interest As the lanes exist in the bottom half of the image, we can safely design a region of interest where the road exists and ignore the edges outside this region of interest. The notebook shows the region of interest I used in my implementation.

6. Detecting Hough lines We provide the edges detected to the Hough Transform to extract the lines from them. This results in a list of lines, in which the required left and right lanes exist.

7. Extracting and Drawing the lanes To extract the lanes from the lines detected by Hough Transform, we follow the following steps:

1. Aggregate the lines with negative and positive slopes separately.
2. Take weighted average of the two lists separately. The weight of each line is given by the length of it. We choose this because the length of the line is analogous to the confidence of lane being that line.

Now, we draw the lanes on the image as a weighted sum of the actual image and the lanes.

This final image with lanes drawn on it is returned by the function.

For reference, we also return images from intermediate steps.

2. Potential shortcomings

1. I have observed that the detected lanes are not very stable. They are fluctuating a little with time. To decrease the fluctuations, we can increase the threshold, but it will increase false misses. This is a tradeoff we need to explore more.
2. All the static values which define the region of interest, thresholds for edge detection, colors extraction, hough transforms are potential shortcomings. The human factor of selecting the ranges is the shortcoming in this case.

3. Suggest possible improvements to your pipeline

1. To make the lane detection more stable, we can use a control systems methodology to make the lanes not fluctuate with each image. We can restrict the detections to not drastically fluctuate temporally.
2. We can develop a system that can optimize dynamically, for example, deep learning models. They are adaptable to the application of the model we are going to deploy on.