Finding Lane Lines on the Road

Project Writeup

Submitted by Anirudh Tibrewal

Code file: TibrewalAnirudh_project_lanes.ipynb

1 Pipeline

My pipeline consisted of 6 steps:

1. Converting the image to grayscale so the lane lines can be easily distinguished.
2. Applying a Gaussian Noise Kernel by specifying the kernel size and using the cv2.GaussianBlur function.
3. Distinguishing the edges in the image, the Canny Transform was applied with a low threshold of 100 and a high threshold of 200.
4. Applying the region of interest function to only show the lane lines and mask rest of the image.
5. Parameterizing the Hough lines to obtain a Hough Transform that only identifies lane lines.
6. Using the cv2.addWeighted function, superimposed the lines on the original image.

The result obtained were multiple lines in the left and right lane boundaries. I had to combine them into two lane lines. For that I made a copy of the draw_lines function called draw_lines_lr. First, I defined two lists, left_fit and right_fit to separate left and right lane lines. I used numpy.polyfit function to find the slope and y-intercept of each line and appended to left fit or right fit depending on the slope.

For both the lists I found the average values of slope and intercept and I made a function find_points, that uses the image dimensions and slope and intercept values to find the endpoints. The function is used with the average of each points and the results are plugged into cv2.line to draw the lines in the image.

For the video files, the averages of a maximum of 30 frames were taken to determine the lane lines as shown in the function draw_lines_video, which can be used in place of the draw_lines function. This helps minimize errors due to issues in video output and result in more stable and determinate lane lines.

I tested the pipeline with all the test images and videos. The output of the videos is attached with the repository and the image outputs are attached below.

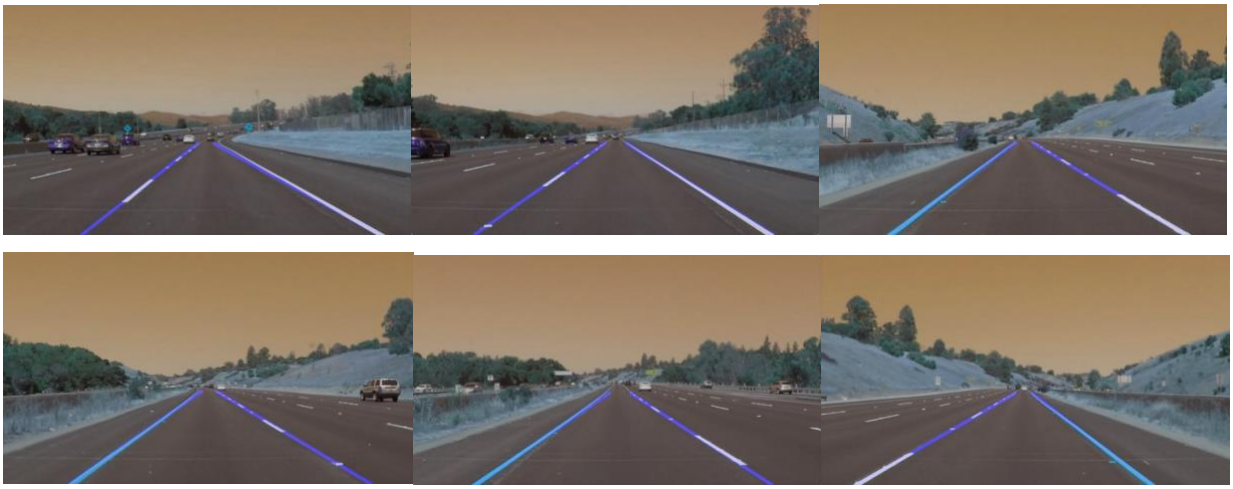

Figure: Lane lines imposed in the provided test images

2 Potential shortcomings

The pipeline could identify solid lines easily, however the middle lanes were sometimes difficult to identify, especially in the video, where the lanes are sometimes not clearly visible in a few frames.

Another shortcoming is in curves or turns; the program sometimes identifies lines from different lanes as part of the same lane.

3 Possible improvements

While taking average of left and right lines, the outliers can be made disqualified to account for imperfections in lane markings or small objects in the road identified by the program as a lane line.