Finding Lane Lines on the Road

Goal

The goal is to identify lane lines on an image and create a pipeline. Since, videos are simple image streams we can apply videos to same pipeline.

- process an image and find edges on this image with canny edge detection
- detect lane lines in the edges with hough transformation
- unify the dashed lane lines as a single line
- combine all of these as a pipeline to detect and mark lane lines on an image
- since videos are stream of images, apply video to this pipeline and detect and mark the lane lines on a video
- Try the edge cases such as curved lane lines.
- Analyze the solution for the real world problems and identify the shortcoming of the solution.

Reflection

The pipeline consisted of 7 steps. All steps are listed below

1. Turn image into gray scale, since we don't need color.

   Since we use canny edge detection, we need to calculate gradient's strength, we don't need color of the image. So, we can turn the image into gray scale. Actually, it is the easiest part.

2. Blur the image

   Prior to canny edge detection, we need a way to eliminate the noise and factitious gradients. We are going to average them applying a gaussian smooting filter. Don't forget the gaussian filter dimension has to be an odd number.

3. Apply canny edge detection

Canny edge detection simply help us to catch the changes on the brightness of the pixels. If the brightness of a pixel is higher than a threshold value, we can consider it as an edge and we can consider adjacent pixels are part of the edge until a pixel is below another threshold value. Basically edges are between higher threshold and lower threshold. And a lane line could perfectly be detected with this approach. What are the best threshold values? As usual there is no easy answer to these question, you should make some experiment and find out best values for your case.

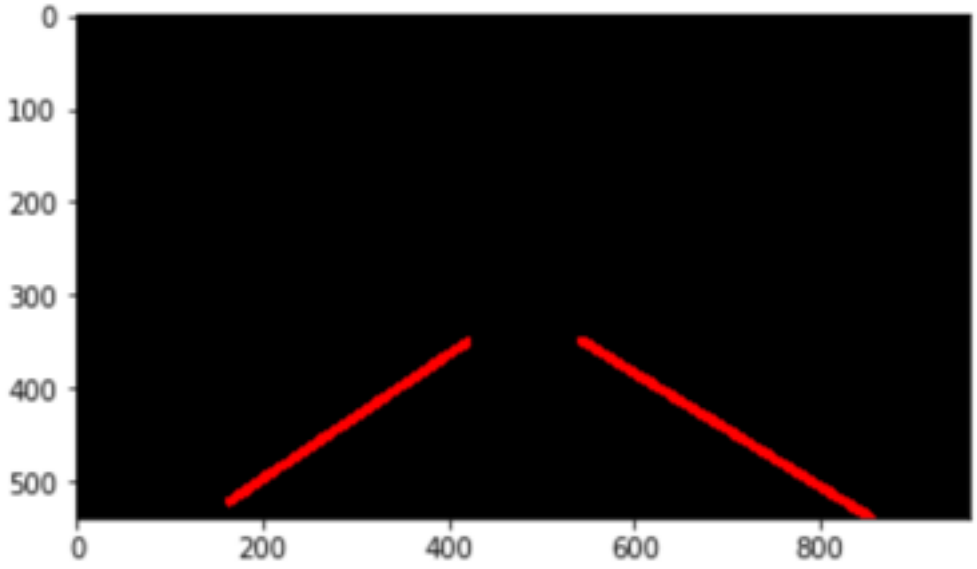
4. Chose the region to mark right lane lines

More or less we are able to detect the edges, so we are able to detect the lane lines. But which lane lines should we direct out attention. Since we are not considering changing lane any time soon, we can just pay attention to the lane lines in front of us, and its shape is kind a quadrangle. So we need a mask which matches this quadrangle shape, and we can safely omit anything outside of this mask.

5. Apply Hough transformation to mark the lines in the masked region

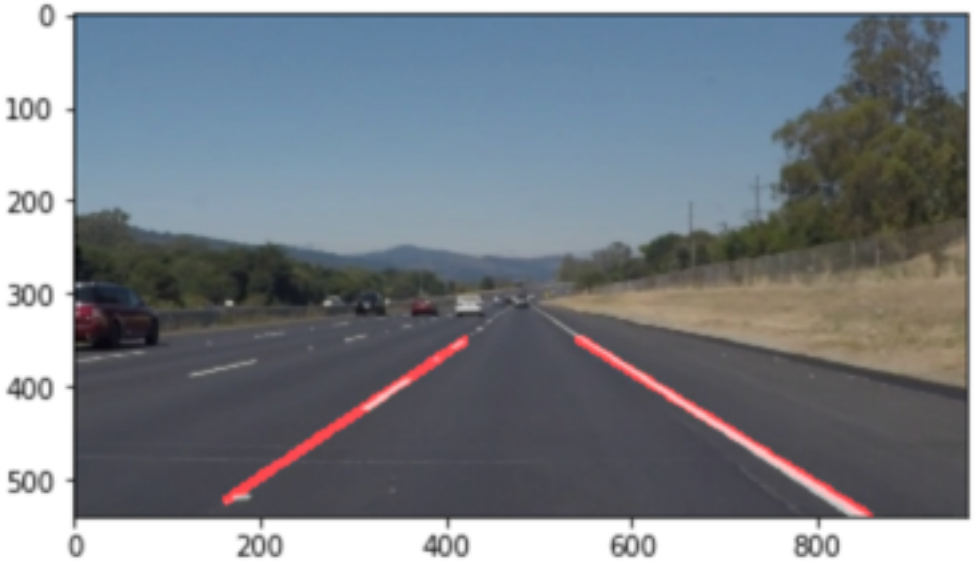
Now, we have a region and we know the edges in this region. But, as you may realized the edges are not seem like a straight line. We need to connect the dots and marked the lane lines. And, right tool for this purpose is the hough transformation. Lets skip the theory behind it(just for now, we will return back to that.) and focus on the implementation.

6.



Apply marked lines on the original images with weight

And we need one final touch, we are going to apply the marked image on the original image.



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

In first version, draw_lines method only draw the same line on the road to the image. We need to unify dashed lines as a single line, so we have modified the draw_lines method. You can find the details about modification below.

go all line in lines
calculate the slope
check slope more than zero than its right
check slope less than zero than it is left
if it is right slope and more then threshold, store min and max value and slope value for right
if it is left slope and more then threshold, store min and max value and slope value for left

calculate mean for right slope
calculate mean for left y_min, y_max
calculate mean for right y_min, y_max
calculate b; y = mx + b —> b = y -mx
calculate x; x_max = (ymax-b)m , x_min = (y_min - b)m

When we have found min and maximum points x,y values, draw the lines.

B. Identify potential shortcomings with your current pipeline

We are able to identify lines, But we could not identify what is a lane lines. If there is any line which formed naturally or artificially, out program could not distinguish them. If it looks line a straight line with a similar slope, our program would consider it as a lane line.

Also, the road is not always straight, there are curves on the road. We are able to identify straight lane lines but we need to identify curved lines as well.

Finally, snow, rain, dark or other outer conditions might effect our algorithm and it might not detect the lane lines. Even some road might

not have lane lines. It shows that we can not only trust lane lines for a safe self driving experience, we need other method as well.

C. Suggest possible improvements to your pipeline

Draw_lines method might be improved slightly. I have made some experiment and decided on a threshold value. I believe we can test with more diverse image and video and We can find a better threshold.

Also, quadriteral space might be improved. The shape of the quadriteral might reflect a car vision better.

Alsoi while we are unify the dots as a line, minimum length and maximım line gap might be optimised.

Also, our program is only works for straight lines, it could be improved to detect curved lines as well.