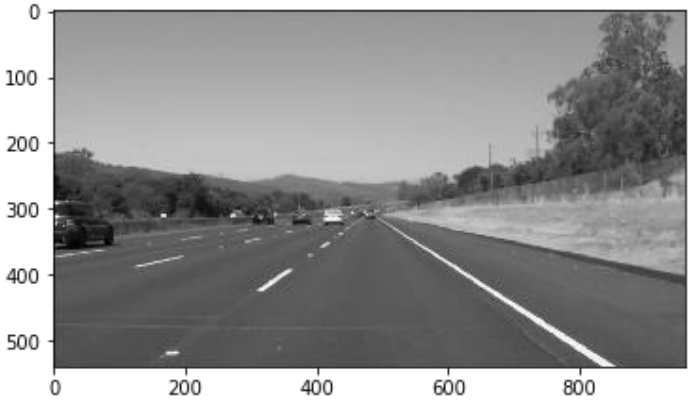
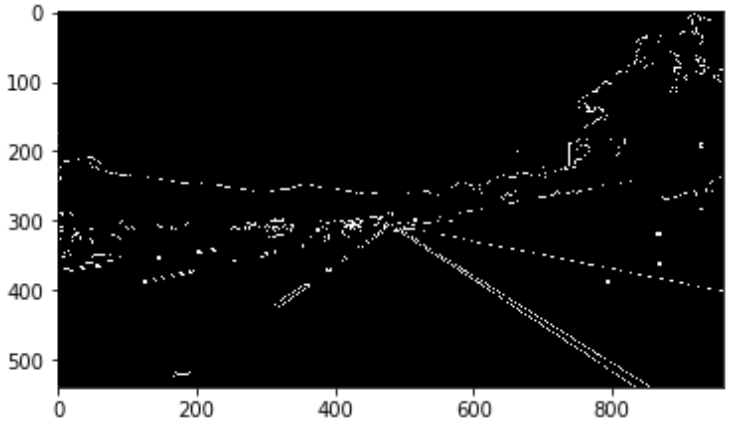
1. Describe your pipeline. As part of the description, explain how you modified the draw_lines () function.

My pipeline consisted of the following steps:

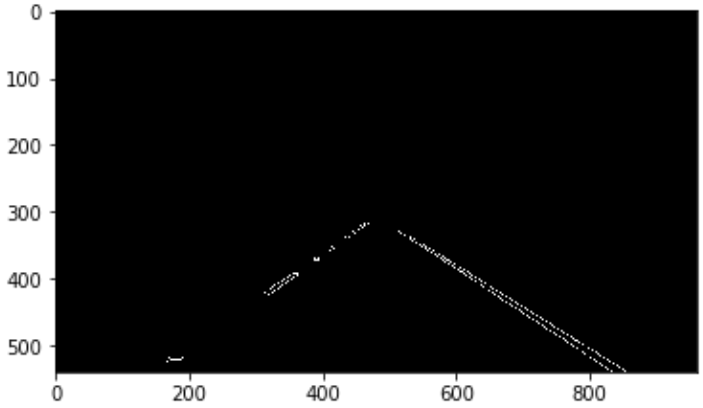
- Convert the image to grayscale



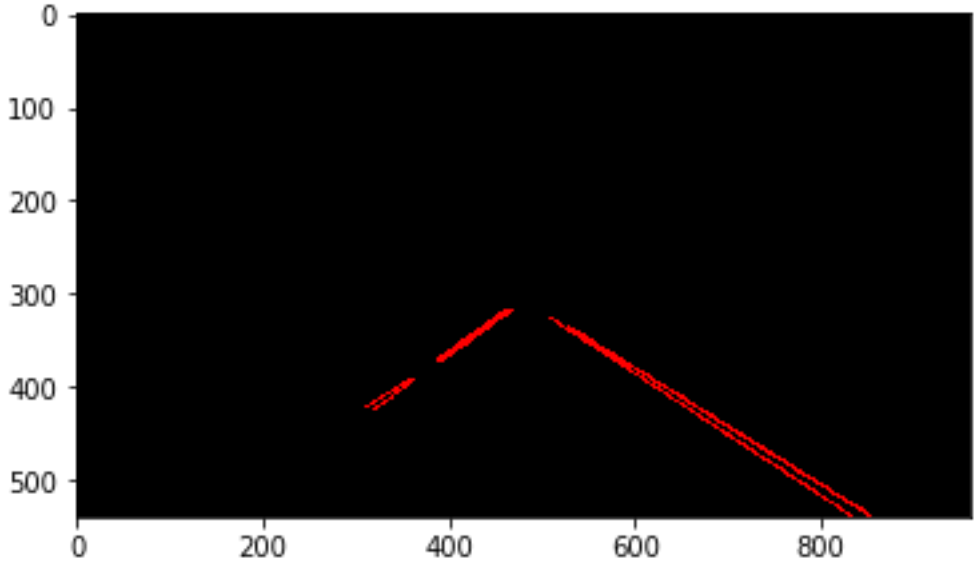
- Blur the image and apply canny edge detection



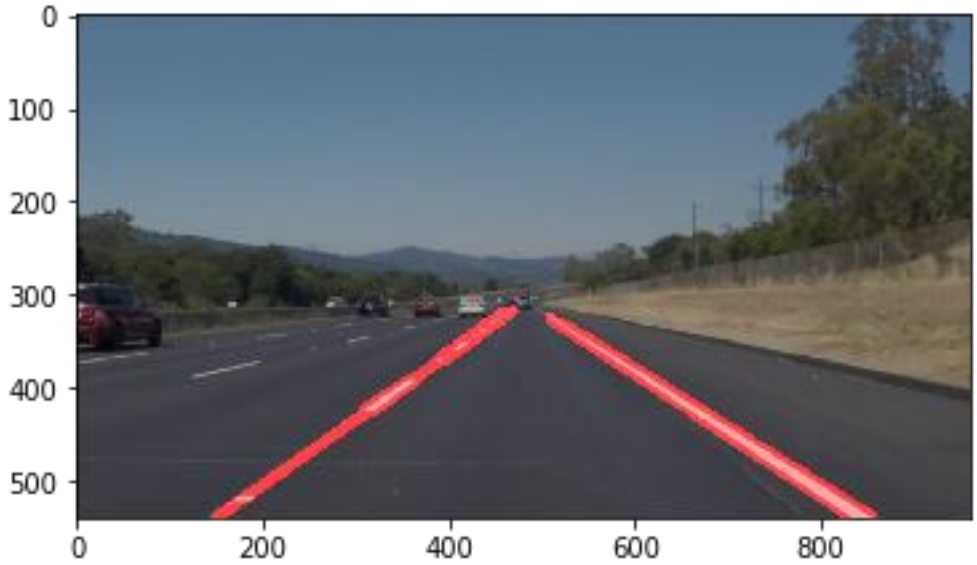
- After the canny edge detection, I am masking the area. In other word, I am choosing the area (polygon using functions in OpenCV) of interest.



- Next, I am using Hough Line transform.



- Since the left lane marking is not continuous, there is a requirement to make it continuous. There is a need to improve draw_line() function. I created a new draw_line_interpolate() function to create a continuous lines using the foundations of interpolation.



- Lastly, I am displaying the annotated image.
- I am repeating the above steps and on a pipeline of images.
- Lastly, considering videos as a pipeline of images, the same steps are applied on videos.

2. Identify potential shortcomings with your current pipeline

One potential shortcoming would be what would happen when there is a curve and the road lines doesn't look like a line but a curve. The challenge problem is based to draw attention to this limitation.

Another shortcoming could be that same algorithm might not be robust for different weather. For example, see the figure below

Lastly, the algorithm is not robust if the line markings are inconsistent and absurd.

3. Suggest possible improvements to your pipeline

A possible improvement would be to interpolate the discontinuous lane marking. Draw_Line_Interpolation function is a rudimentary yet effective way to tackle to fix it.

Another potential improvement could be to navigate at curves. This script is not comprehensive as it cannot perform in the curves like in the mountainous terrain.