**Finding Lane Lines on the Road**

**Seán McMahon, Cambridge, UK**

**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

**Reflection**

1. **Describe your pipeline. As part of the description, explain how you modified the draw\_lines() function.**

The code for my pipeline is located in P1\_new.ipynb

My pipeline consisted of 5 steps as follows.

1. Converted the images to grayscale
2. Ran the canny edge detector on the grayscale image
3. Did a Gaussian blur of the canny edge detected image
4. Defined the vertices of the region of interest and created a new image based on the vertices.
5. Defined the Hough transform settings and generated a new image with the processed hough transform lines

In order to draw a single line on the left and right lanes, I modified the draw\_lines() function as follows:

1. Separate the Hough Lines into two arrays representing the positive and negative slopes.
2. Sort the arrays by order to the line length
3. Look for the next biggest hough line in both the positive and negative y directions:
   1. the gap between the lines is greater than 6
   2. the new line is longer than 10 pixels
   3. Slope is greater than 0.1 (positive slope) and less than 0.1 (negative slope)
   4. The first x element greater than 300 pixels (positive slope) and less than 660 pixels (negative slope).
4. Generate a new equation for the line using standard line equation
5. Extrude the line to the bottom of the image and increase to (image height/1.7)

If you'd like to include images to show how the pipeline works, here is how to include an image:

**Issue Resolution**

Horizontal Streaks

In the image below, a streak on the road caused the algorithm to pick up the wrong line set. The right side lane reverted to the left side.



This failed as it picked up the horizontal streak at the bottom.

RMaxline [477, 318, 585, 376, 0.53703703703703709, 122.58874336577564]

RMaxline2 [477, 318, 261, 481, 0.53703703703703709, 122.58874336577564]

e.g. (173, 478, 261, 481, 0.034090909090909088, 88.05112151472007)

I introduced requirement slope > 0.1 to fix this issue.

Infinite Slopes

Infinite slopes, where the two y points are identical.

On the last video (solid yellow lane), there were infinite slopes, so I introduced a criteria to reject NaN’s or Inf’s.

*isfinite(slop) != False*

Jumping Lanes

There is intermittent line jitter in the program. A right lane line error is illustrated below.



I decided to introduce an x criteria in order to keep on the correct side of the road.

**2. Identify potential shortcomings with your current pipeline**

One potential shortcoming would be what would happen when a white van appeared just in front of the car. The algorithm would detect a line along the van.

Another shortcoming is that the hough lines are not filtered sufficiently by slope during the transform.

**3. Suggest possible improvements to your pipeline**

A possible improvement would be to introduce a filter to stop the algorithm picking up false positives. The algorithm is currently jumpy, if a large jump in the lane line was detected it could be discarded on first occurrence.

Another potential improvement could be to introduce limits for selection of additional lines based on the equation of the first line.