**Project 1 Write-Up Finding Lane Lines on the Road**

**Background Info**

A **starter kit** was provided for this project, which included the following

* A (.yaml) file which contains the required python packages and dependencies
* 6 test images
* 2 test videos
* 1 challenge video (this is not used in my original submission)
* An example Jupyter notebook that contained some helper functions and could be used to confirm the environment is set up properly

**Goals**

* Create an image-processing pipeline that identifies and annotates lane lines on the road.
* Successfully apply the image processing pipeline to simple test videos
* Provide reflection and comments on areas of improvement

**Description of pipeline**

The pipeline follows the following steps:

* Color filtering (identifying yellow and white lane lines)
* Select a Region of Interest (ROI) to mask lanes other than the two the car is driving in.
* Convert to gray scale and reduce noise with Gaussian smoothing
* Apply Canny edge detection algorithm
* Apply Hough Transform to the output of the Canny edge detection to identify lines
* Use lines detected in previous step to determine the left and right lane lines and average/extrapolate the results
  + This step took the longest time to fine tune. Part of the problem was that during my original solutions – the short line segments that were detected were skewing the estimated slope of the lane line and during the video processing the projected line would flicker off the actual lane line.
* Display projected lane lines on the image

*An example of a test image before and after image-processing pipeline has been applied is provided at the end of this document.*

### Potential shortcomings with current pipeline

### The method for isolating a region of interest is not ideal. It does not dynamically deal with curved lines. This is one of the key reasons the challenge video did not work with my pipeline.

### The short line segments of lane lines introduced some problems and occasional flickering of the line projection. This should be smoothed out in future versions.

### The algorithm hard codes color filtering but does not account for changes of lane color during night driving.

### Situations where another car merges into the lane in front of the autonomous car would disrupt the lane detection because it would mask the actual lane lines.

### Suggest possible improvements to your pipeline

* I’d like to improve the smoothness of the line projections
  + Possibly using a moving average of the projections to deal with flickering
* Make the color-filtering step more robust to deal with changes in daylight.
* Make a more robust region selection algorithm to allow for curved lines.
* Include a step for dealing with strange exceptions that may occur. For example, if no line is detected you'll be dividing by zero while computing slope and it will throw an error. This occurs at some point during the processing of the challenge video.
* After making some improvements I will apply them to the Challenge video and resubmit.

**Results on videos**

* The processing pipeline was successfully applied to both test videos. The annotated videos can be found in the ‘test\_videos/myResults’ folder.
* There is slight flickering of the line projections in (solidWhiteRight\_outputVideo.mp4) but the overall processing is pretty good.
* The (solidYellowLeft\_outputVideo.mp4) is not as good but is still doing what was intended. There are one or two flickers that project the lines very poorly. I think this could be resolved with more time to fine-tune the functions parameters.

|  |  |
| --- | --- |
| Before | Space Station:Users:Tgallow5:Desktop:Project 1 Submission - Lane Line Detection:test_images:solidYellowLeft.jpg |
| After | Space Station:Users:Tgallow5:Desktop:Project 1 Submission - Lane Line Detection:test_images_annotated:processed_solidYellowLeft.jpg |