Project: Advanced line Finding

1. Pipeline

   The following is the pipeline to detect lane lines. Each steps output is an input to the next.

   a. Get a frame from the video to work with
   b. Get a combined binary image
      i. Binary from gradient thresholding
      ii. Binary from color thresholding
   c. Apply mask with a region of interest (to flush out unwanted pixels)
   d. Perspective transform (Birds-eye view)
   e. If first frame or when lost lane confidence
      i. Apply histogram on the lower half of the image
      ii. Dynamic window search using histogram peak positions as a starting point
      iii. Fit a 2^{nd} order polynomial
   f. For non-first frames or with a prior confidence
      i. Search for lane pixels around previously fitted 2^{nd} order polynomial (with a margin)
      ii. Fit a new 2^{nd} order polynomial
   g. Measure radius of curvature and center error
   h. Unwarp the and post-process the image with lane markings and text.

2. Shortcomings

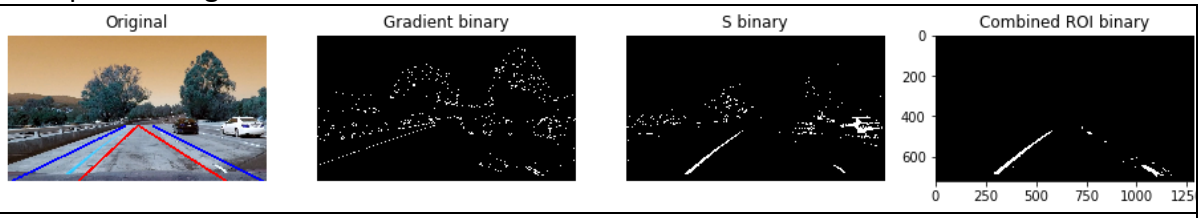
   This is a great lane finding approach. However, this approach didn't address the following challenges:

   a. What happens when changing lanes
   b. We had some assumptions like ROI, points for perspective transform. Will these assumptions hold in real implementation?

3. Possible improvements

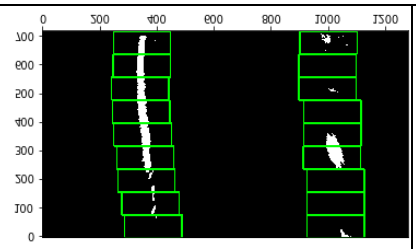
   a. My code doesn't work great on the challenge video, I will extract some sample frames to work with.
   b. I am sure, Deep learning has an answer to some of the limitation in this project. I will get back to this project once I finish all the lessons. Excited for the next project!

4. Pipeline images

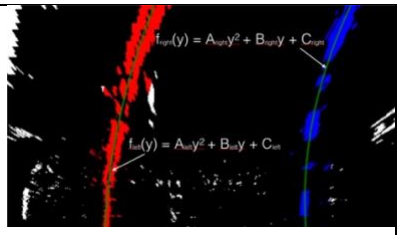


From left to Right: Undistorted original image with ROI marked, Binary from gradient, Gradient from color (S-channel), Combined binary image with ROI mask applied.

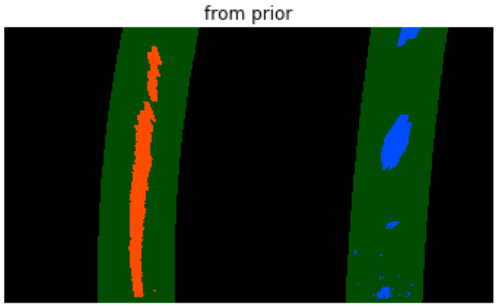
Bird's eye view of the combined binary after applying dilation filter.



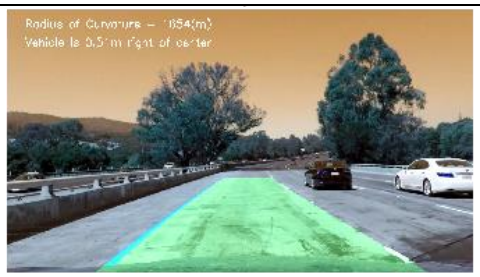
Sliding window method of finding pixels



(This image from lecture). I fit second order polynomial on the lane pixels.



Searching for lane pixels using prior knowledge.



Final processed image with radius of curvature and center error measurement.

The code is self-explanatory, and the functions were named meaningfully. Please go through the code for a detailed understanding of the project.

Happy Learning!!!
Santosh.