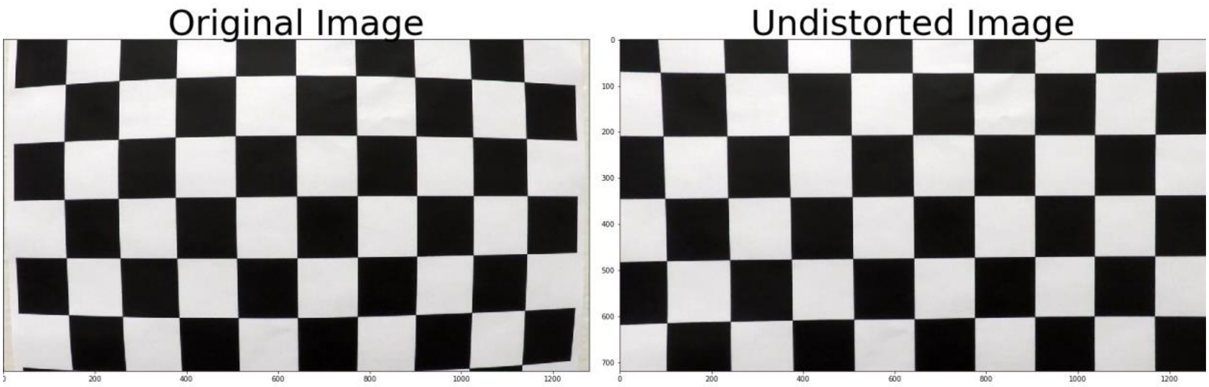
UDACITY Self Driving Car Engineering Nanodegree

Project #4: Advanced Lane Finding

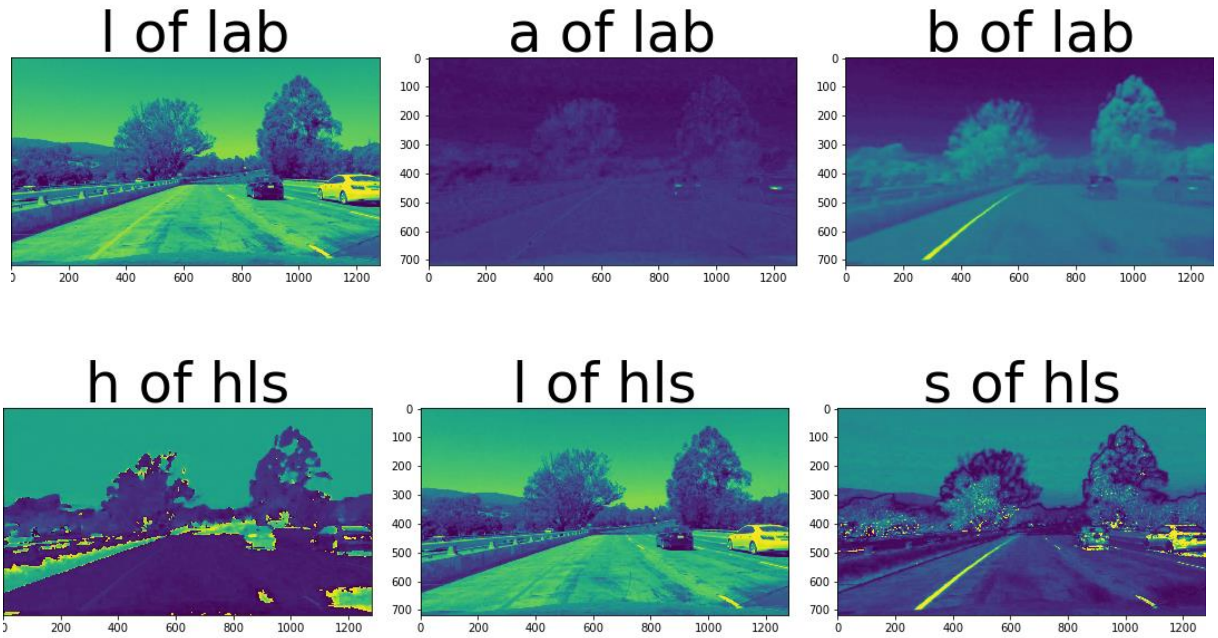
Talha Korkmaz

The Pipeline:

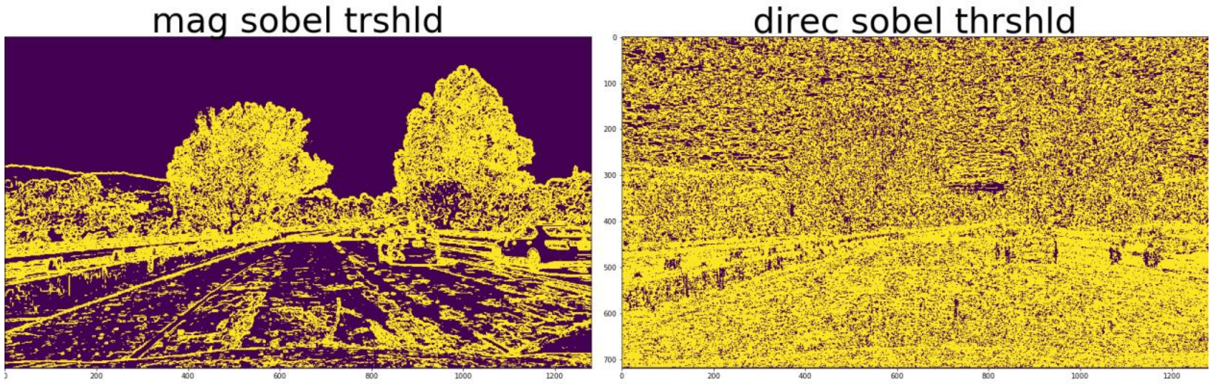
- **Distorsion**
  First, the images are undistorted. To do that, openCV’s corresponding libraries and chess board images are used.



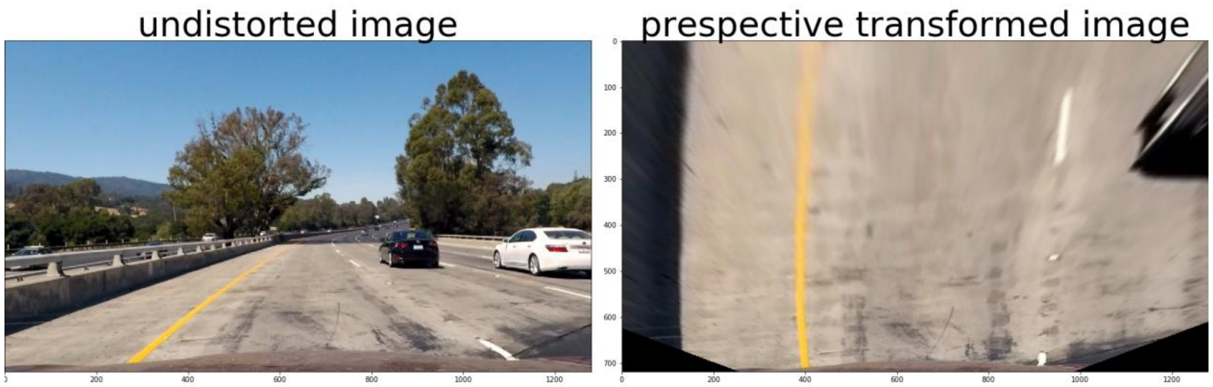
- **Color masking**
  Mainly B in LAB and S in HLS color spaces are used since they are robust to shading & brightness changes. R in RGB is also used in final pipeline to more confidentially detect the yellow lanes.



- **Gradient Masking**
  The image is masked with absolute and directional sobel gradients to detect the edges.

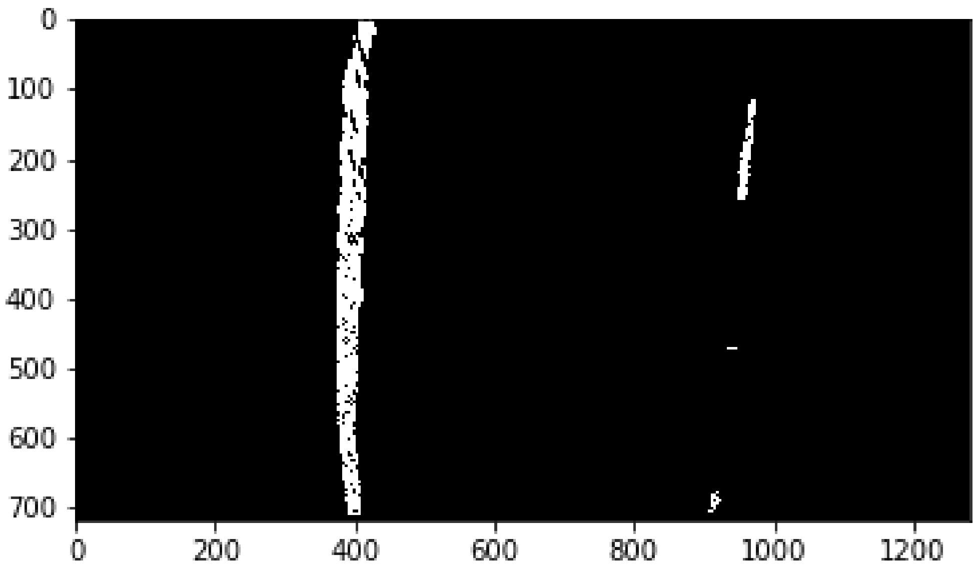


- **Perspective transform**
  It is necessary to look the lanes upside town to correctly calculate radius of curvature. OpenCV’s warp function is used to transform the image upside down perspective.

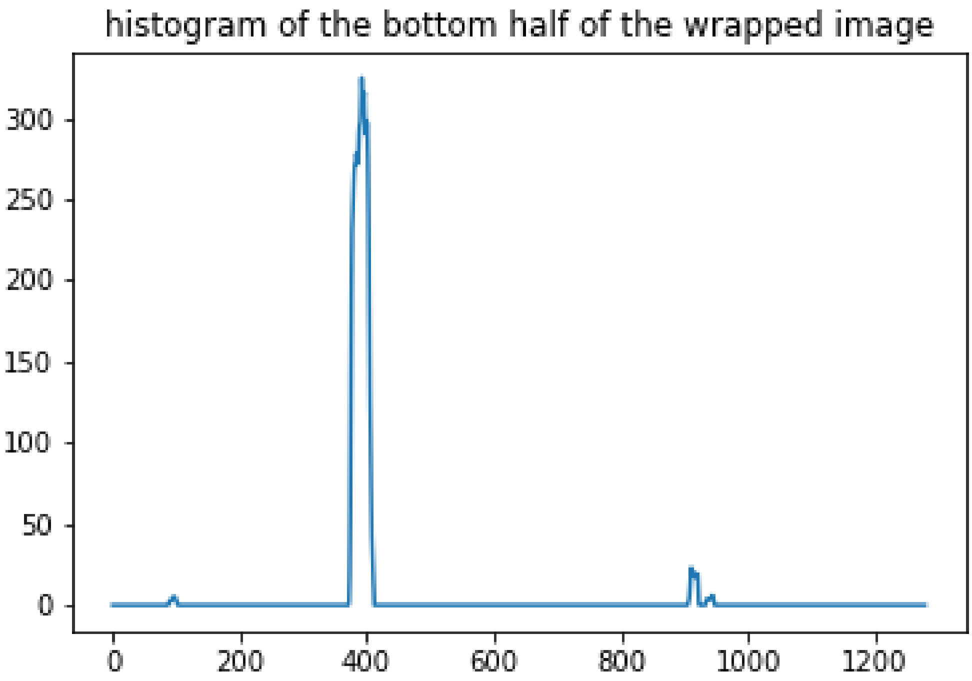


- **Masking**:
  The image is masked so that only potential whereabouts of lanes are left, ie extreme right and left sides are masked out.

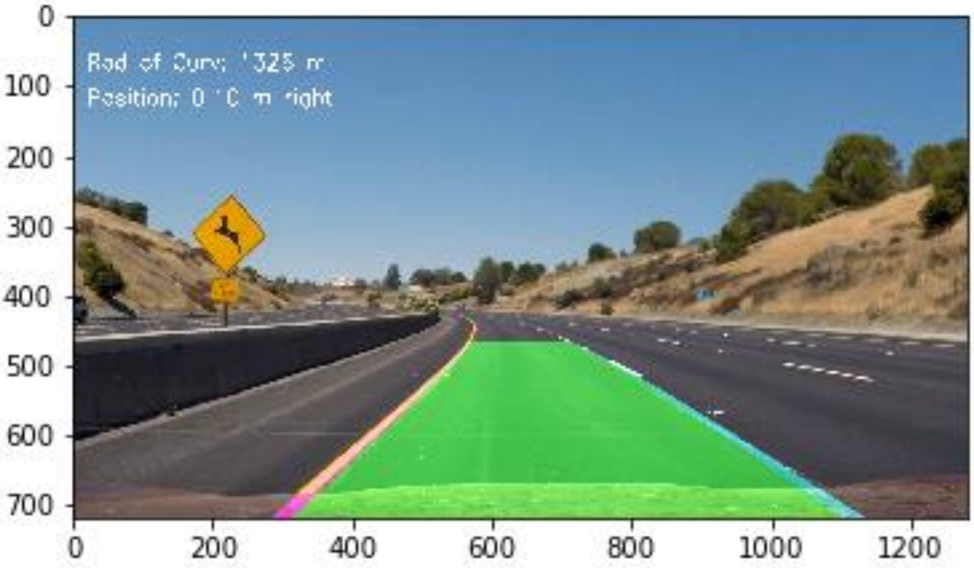
- **Combination of masks**
  The color masks and sobel masks are combined with OR gate within their groups and then color and sobel masks are combined with AND gate to create final mask.

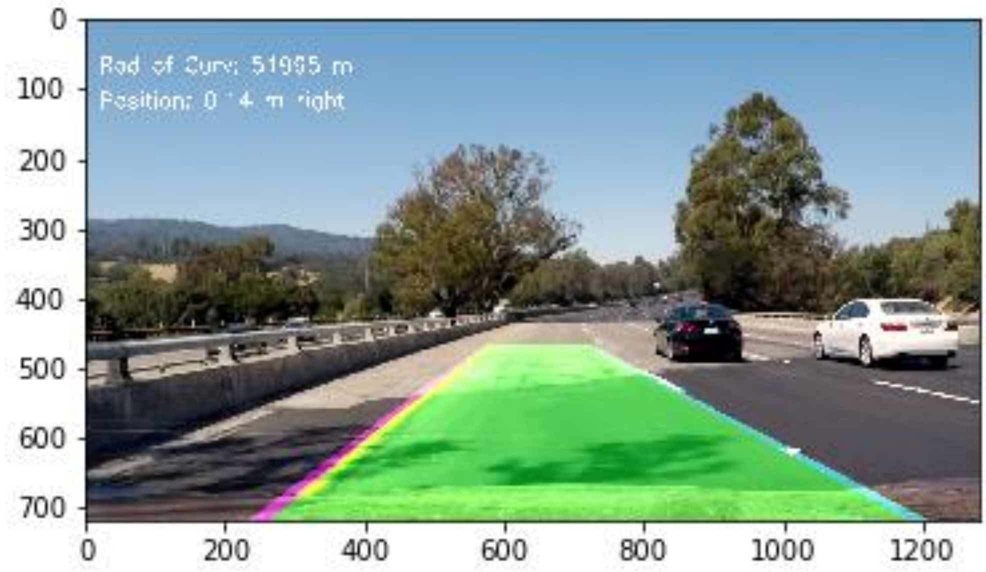


- **Histogram**
  To begin finding lanes, histogram of bottom part of binary image is calculated.



- **Finding lanes**
  Sliding window method described in the class is used to detect the lanes. Then, second order polynomial is fitted to obtain the curve.





- **Radius of curvature and position of the vehicle**

  Once the polynomials are obtained, calculating the radius of curvature is pretty straightforward and the formula is given in the class as well.

  As for the position, the average of left and right lane beginnings is subtracted from the half of the width of the image (ie horizontal center) and vehicles relative drift is found.

  For calculating both the radius of curvature and position of the vehicle coordinates are transformed from pixel space to meter space.

- **Increasing the efficiency**

  If the lanes are found in the previous frame, then it is unnecessary to search the whole space to look for the lanes again. Instead, windows are put in previously found areas and look for the lanes within the window. If not found, then search the whole space. The method is described in the class.

- **Smoothing**

  2 Major algorithm is implemented to smooth the algorithm in videos:

  1- Restrict the change in polynomial constants. There are 3 terms in second order polynomials and they should only change in small amount at each frame since lanes are continuous and smooth. Thus, there are 3 different limits for 3 terms and they can only change that much at a time.

  2- Moving average filter. Last 13 found lanes are taken into account and currently found lanes are smoothed with these previously found ones. Thereafter jitters etc are avoided.