Advanced Lane Finding Project Report

The goals / steps of this project are the following:

- Compute the camera calibration matrix and distortion coefficients given a set of chessboard images.
- Apply a distortion correction to raw images.
- Use color transforms, gradients, etc., to create a thresholded binary image.
- Apply a perspective transform to rectify binary image ("bird's-eye view").
- Detect lane pixels and fit to find the lane boundary.
- Determine the curvature of the lane and vehicle position with respect to center.
- Warp the detected lane boundaries back onto the original image.
- Output visual display of the lane boundaries and numerical estimation of lane curvature and vehicle position.

Camera Calibration

- The Camera calibration portion of code is the second code cell portion of the code just after the imports. I read all the images using glob function and initialize *obj points* and *img points.*
- Obj points - chessboard corners (x,y,z) where z = 0 since chessboard is on flat surface(z = 0) always.Same for each calibration image.
- In the process of calibration, all the obj points and image points are appended whenever the corners are found in a ***calibration/test*** image.
- Img points is appended with each of the corners (x,y) pixel position found in each successful detection of corners on a test image.
- Obj points and img points are then used to calculate the calibration matrix for the camera and also the distortion coefficients using the cv2.calibrateCamera() function.
- Distortion correction is done using cv2.undistort() function.

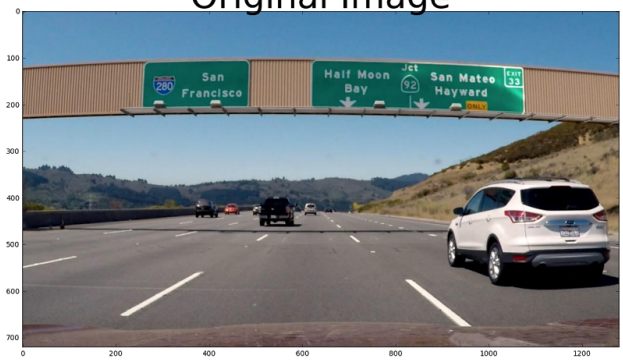
Describe how (and identify where in your code) you used color transforms, gradients or other methods to create a thresholded binary image. Provide an example of a binary image result.

Pipeline(Images)

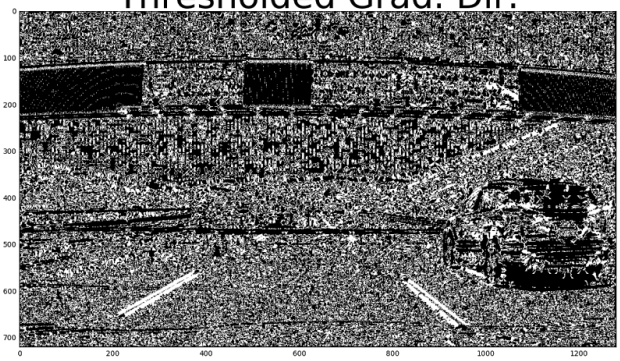
Combining Threshold

- I used various combinations of thresholds and plotted them to check the best one and the outputs are following.
- Among them, the best output appears for Threshold Magnitude.

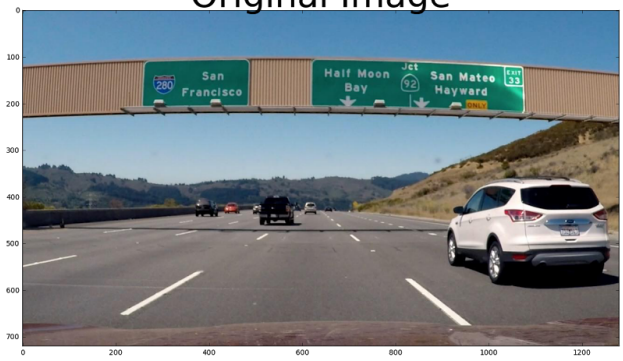
Original Image



Thresholded Grad. Dir.

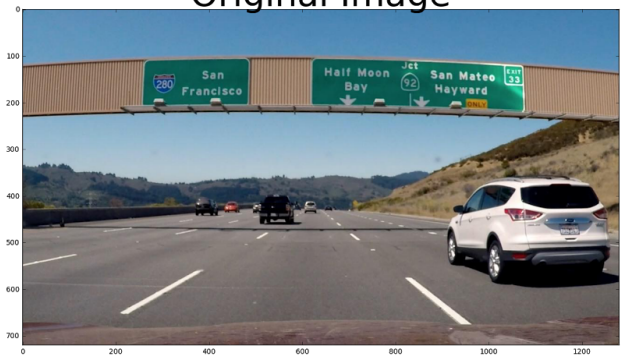


Original Image

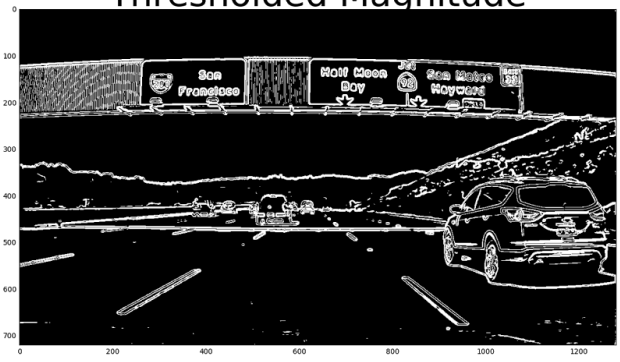


Thresholded Gradient

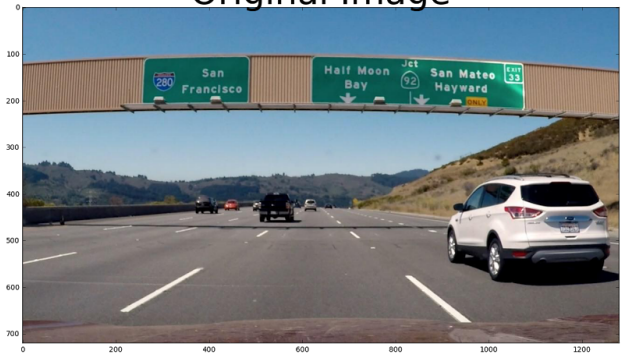
Original Image



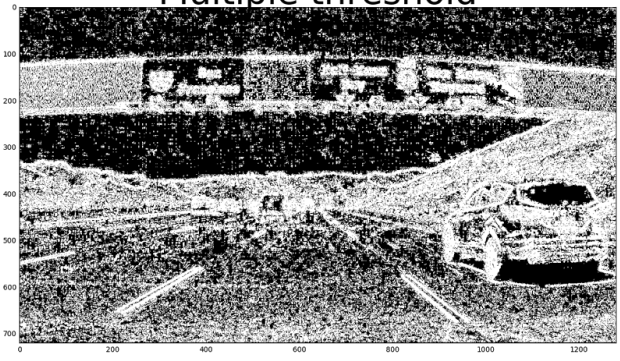
Thresholded Magnitude



Original Image

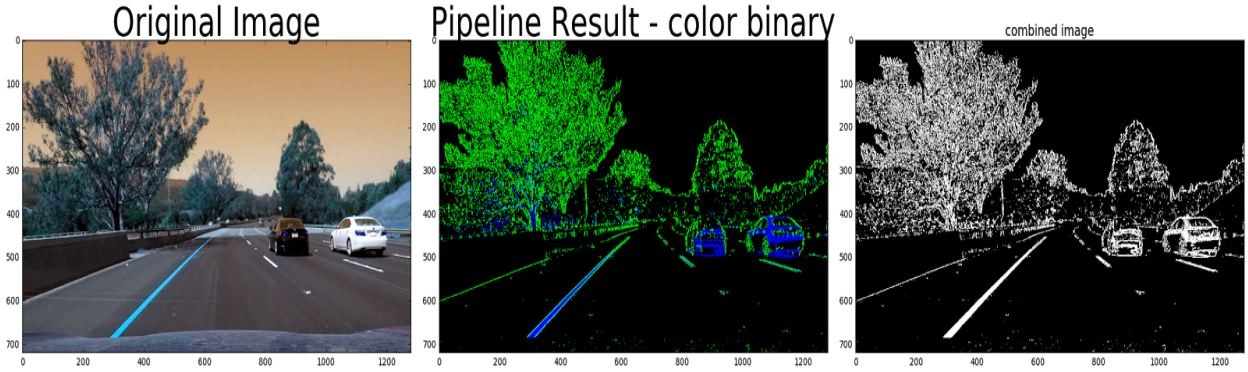


Multiple threshold



HLS and Color Thresholds

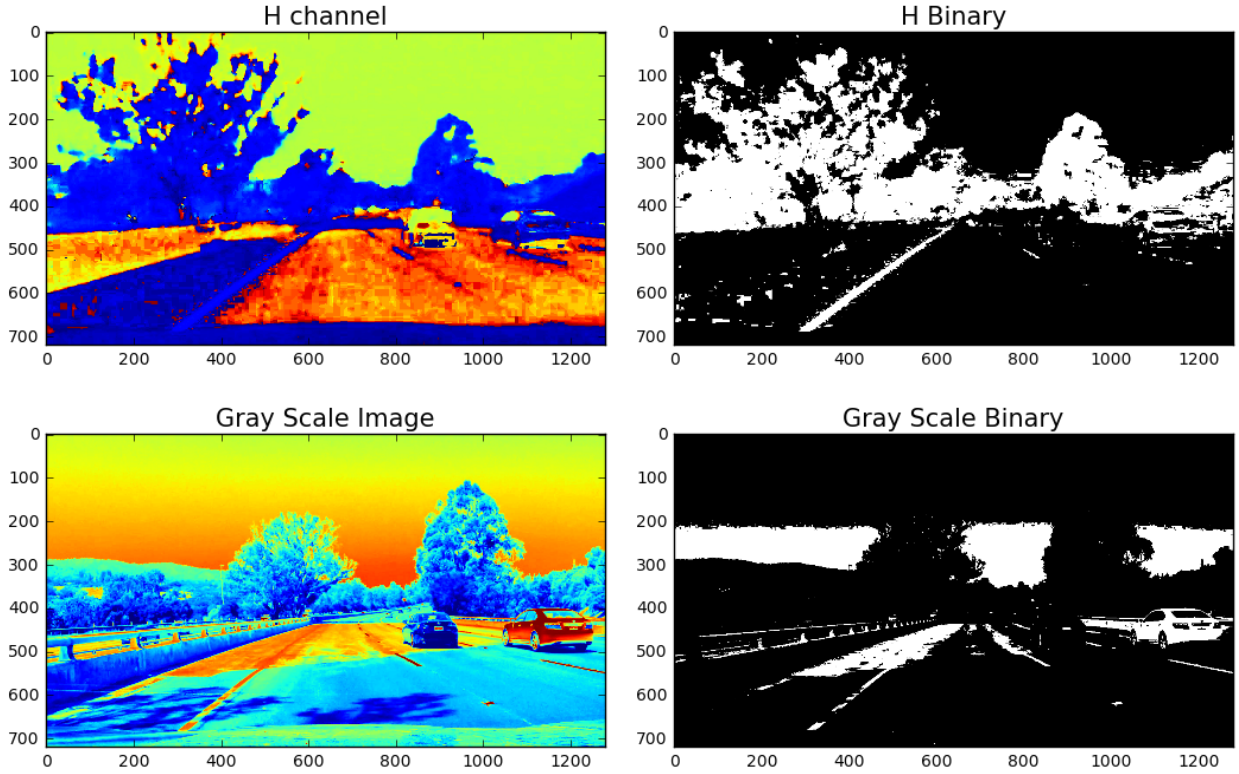
The below output is when a combination of sobel-x and sobel-y operators. This helps in clearly identifying the lanes with different colors.

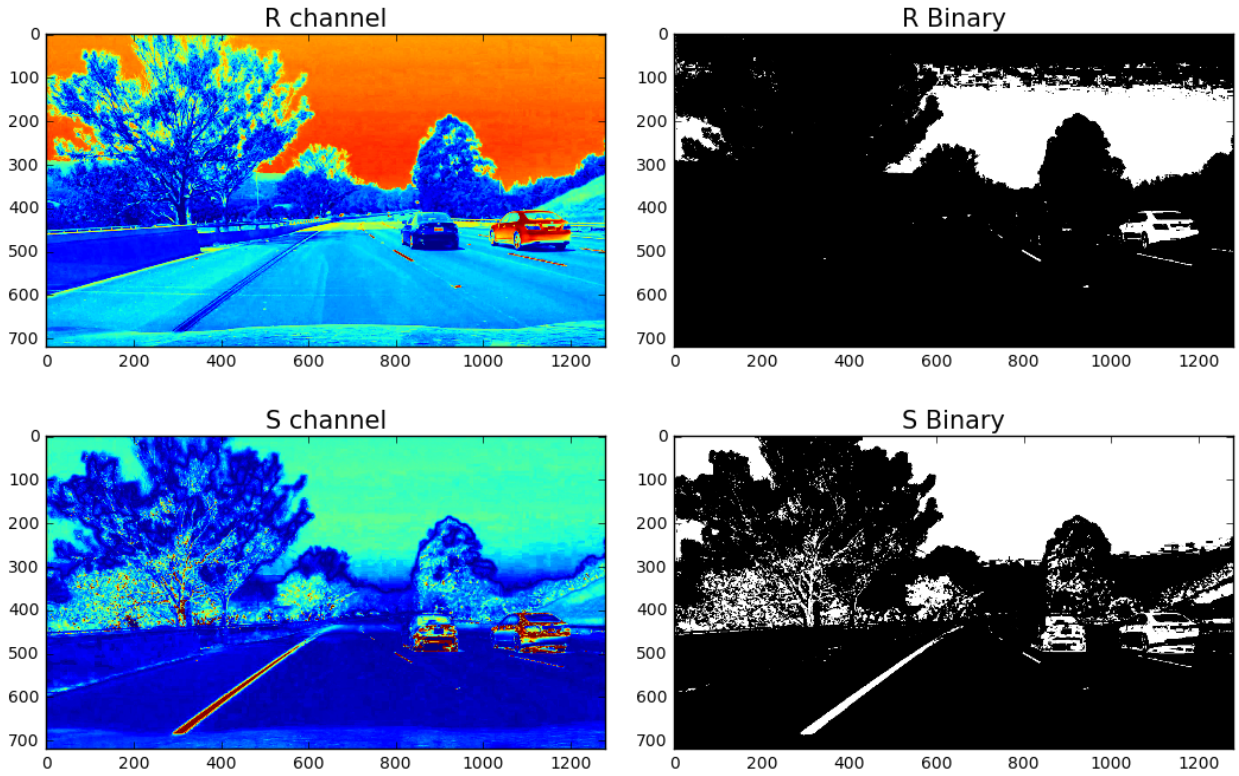


Color and Gradient

GrayScale, S, H, R channel binary

- Among the below images, S channel binary is better compared to others

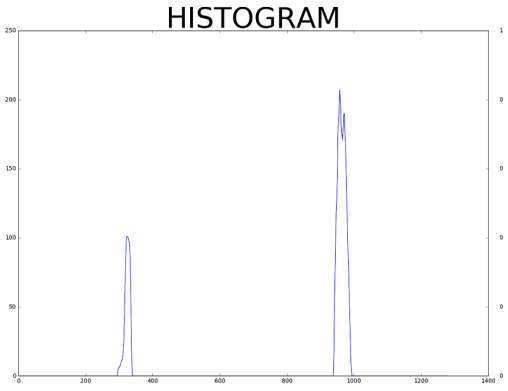




Describe how (and identify where in your code) you performed a perspective transform and provide an example of a transformed image.

Perspective transform

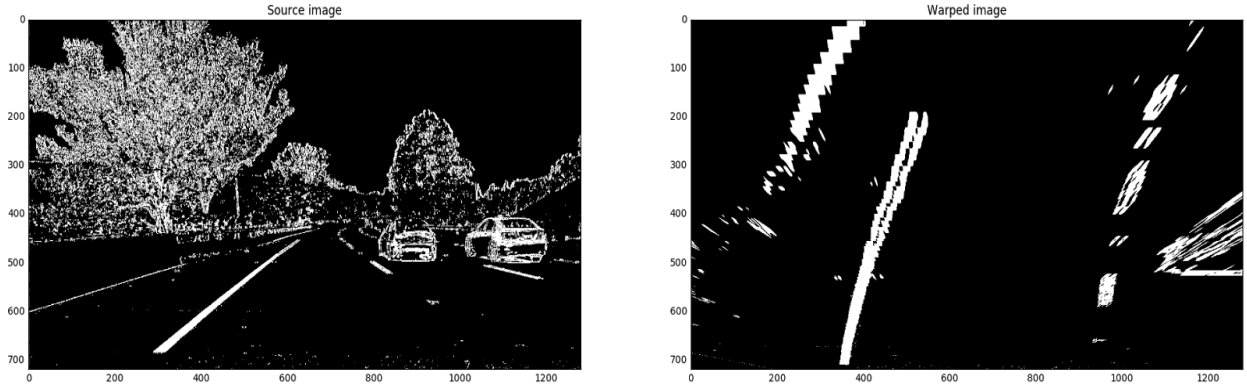
An example of finding histogram in the process of perspective transform.



The hardcoded src and destination points are as follows.

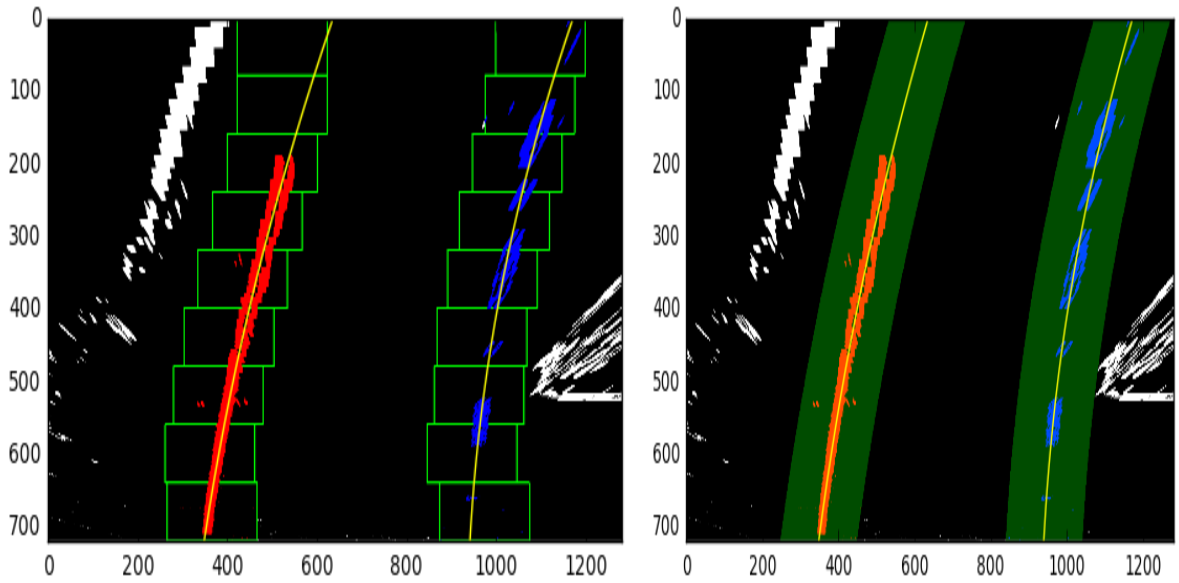
src = np.float32([[585, 460], [203, 720], [695, 460],[1127, 720]])
dst = np.float32([[320, 0], [320, 720], [960, 0], [960, 720]])

My example output of a fully perspective transformed image is below.



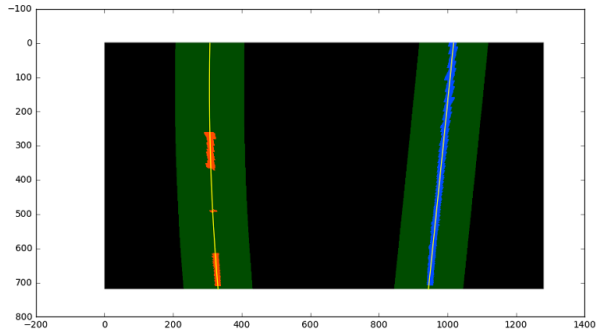
Finding Lane Lines

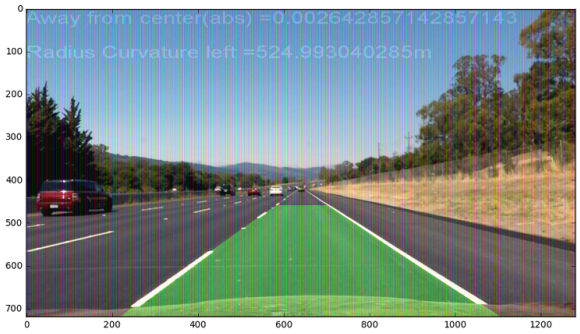
Here is an example of plotting the lanes and the fitting polynomial on the warped image.



Measuring Curvature

Provide an example image of your result plotted back down onto the road such that the lane area is identified clearly.





Video(Pipeline)

Provide a link to your final video output. Your pipeline should perform reasonably well on the entire project video (wobbly lines are ok but no catastrophic failures that would cause the car to drive off the road!).

Here is a link to my Project Video Output

Discussion

Briefly discuss any problems / issues you faced in your implementation of this project. Where will your pipeline likely fail? What could you do to make it more robust?

Due to the time constraint , i couldn't work on developing the solution for challenge videos. The main challenge in implementing this solution was to write everything in a single function and apply it to video. I initially started working individually on various portions of the code.

The main portion of code has been tested on all the test images and then proceeded to the videos.

I also included a folder called trials where i implemented various sections of solutions individually - **Finding Lane Lines, Combining Thresholds, sliding window etc.**

The output_images folder contains the outputs of project at different stages with a proper naming convention used.