Edit 1. Edited after review, reviewed part is at the bottom of this writeup.
Write-up
Advanced Lane Finding Project.
The goals / steps of this project are the following:

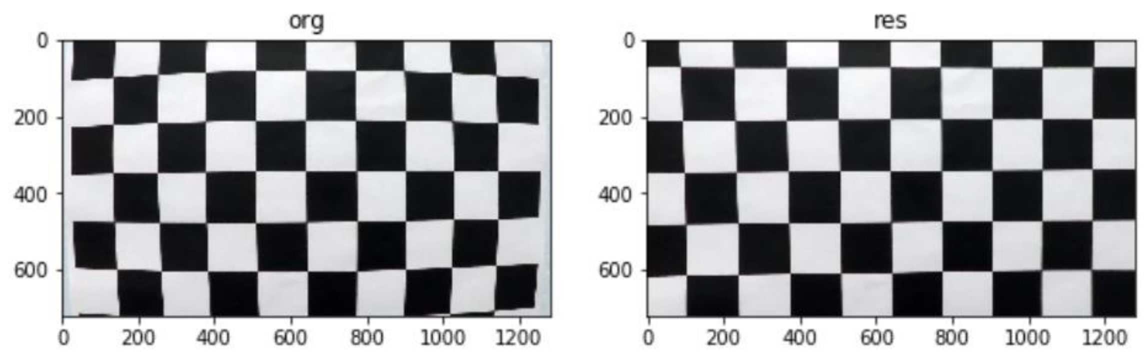
- Compute the camera calibration matrix and distortion coefficients given a set of chessboard images.
- Apply a distortion correction to raw images.
- Use color transforms, gradients, etc., to create a thresholded binary image.
- Apply a perspective transform to rectify binary image ("birds-eye view").
- 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.

Note: The cell numbers are kinda jumbled in sequence , I am new to jupyter notebook, Sorry for the inconvenience.

Camera Calibration

1. Briefly state how you computed the camera matrix and distortion coefficients. Provide an example of a distortion corrected calibration image.
The code for this step is contained in the first code cell of the IPython notebook located in the home directory. The file name is p2.ipynb
So, cv2.calibrateCamera needs object points and image points so for calibration I needed object points and image points.
Image points if obtained by cv2.findChessboardCorners and object points are fixed , In my code you will find different values for nx and ny that is because some of the chessboard images were not being detected and I had to pick smaller corners.
mtx, dist is returned by cv2.calibrateCamera which will be useful for distortion correction.

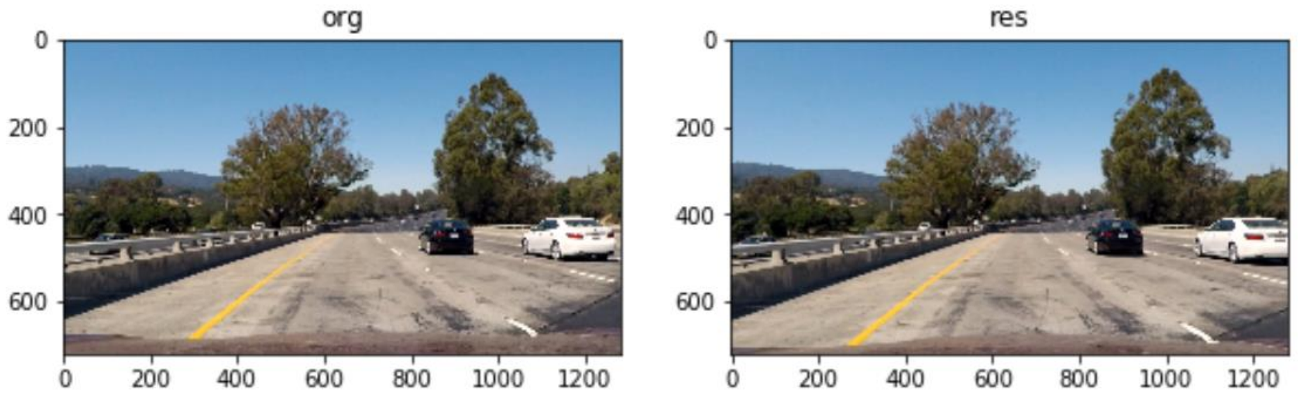
Example:



Pipeline (test images)

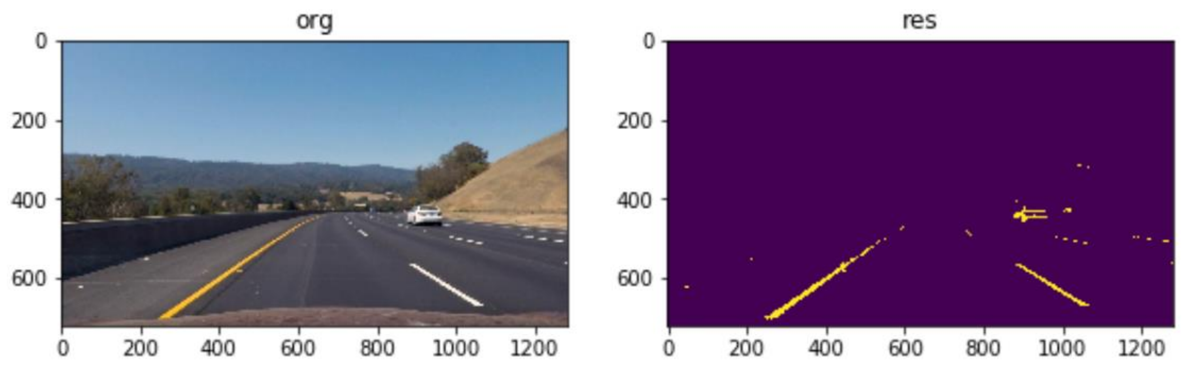
1. **Provide an example of a distortion-corrected image.**

The mtx and dist values returned in previous step, when passed to cv2.undistort gives us a corrected or undistorted image.

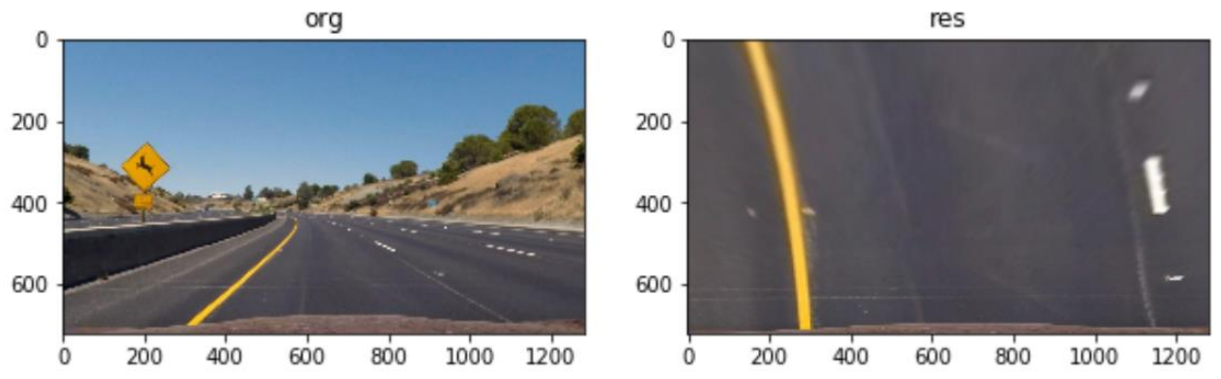


2. Describe how (and identify where in your code) you used color transforms, gradients or other methods to create a thresholded binary image. Provide an example of a binary image result.

In notebook cell number 5 , with title- 2.Thresholded Binary Image lies the code for the following,
Used S channel in HLS, white and yellow in HSV, HSL and RGB for thresholding.
Following is the resultant image.



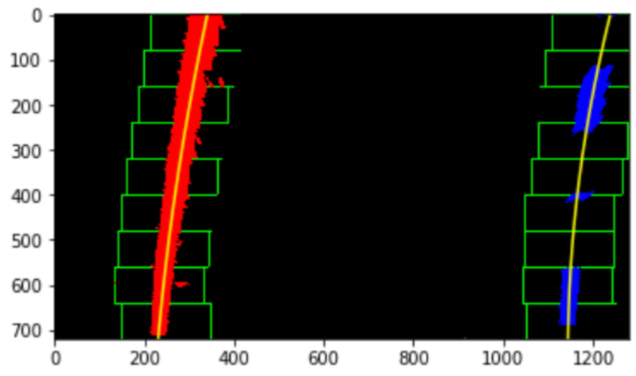
3. Describe how (and identify where in your code) you performed a perspective transform and provide an example of a transformed image.
Cell no 7 in the notebook.
Manually selected some values for src and dst,
cv2.getPerspectiveTransform transforms the image from src coordinates to dst.
We store Minv (M inverse so as to undo this transformation later)



4. **Describe how (and identify where in your code) you identified lane-line pixels and fit their positions with a polynomial?**
Cell no 14 is where the code resides.
Note that most of the code in this block is not a part of the pipeline and lesson quiz was used as a reference point.
With the warped binary image from last stage, To both left and right lanes I fit a 2nd order polynomial .
I calculated the histogram of the bottom half of the image, then portioned the image into 9 horizontal windows at the histogram peaks. Then starting from the bottom windows, put in a 200 px margin of windows at left and right peak.

I moved my way up on the windows and found pixels in the region and recenter my center if needed.
You'll have left and right lane candidate pixels, then fit a 2nd order polynomial to each set, and you'll get estimated left and right lanes.

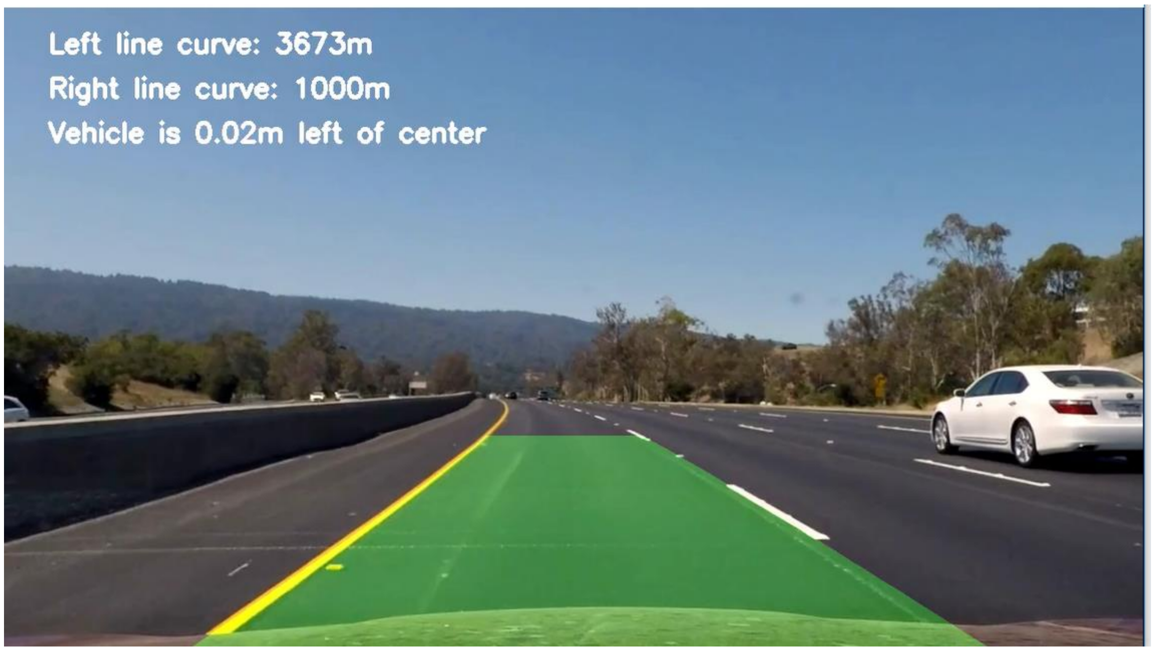
```
Out[13]: <matplotlib.image.AxesImage at 0x7f9ccd6d3710>
```



5. Describe how (and identify where in your code) you calculated the radius of curvature of the lane and the position of the vehicle with respect to center.
Cell 9 has function get_lane_lines with the code.
https://www.intmath.com/applications-differentiation/8-radius-curvature.php
I used the above page as a reference for calculating the radius of curvature.
Converted the distance from pixels to meters on a 30,3.7 meters of axis which was given in the lectures.

Continued..

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



Pipeline (video)

1.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!)

Hi!, my video is in the main directory itself, with the name :
result_project_video.mp4
Thank you!

Discussion:

1.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?

Apparently, The udacity jupyter notebook workspace dosen't run on a proxy at my workplace spent hours just trying to find a solution to that.

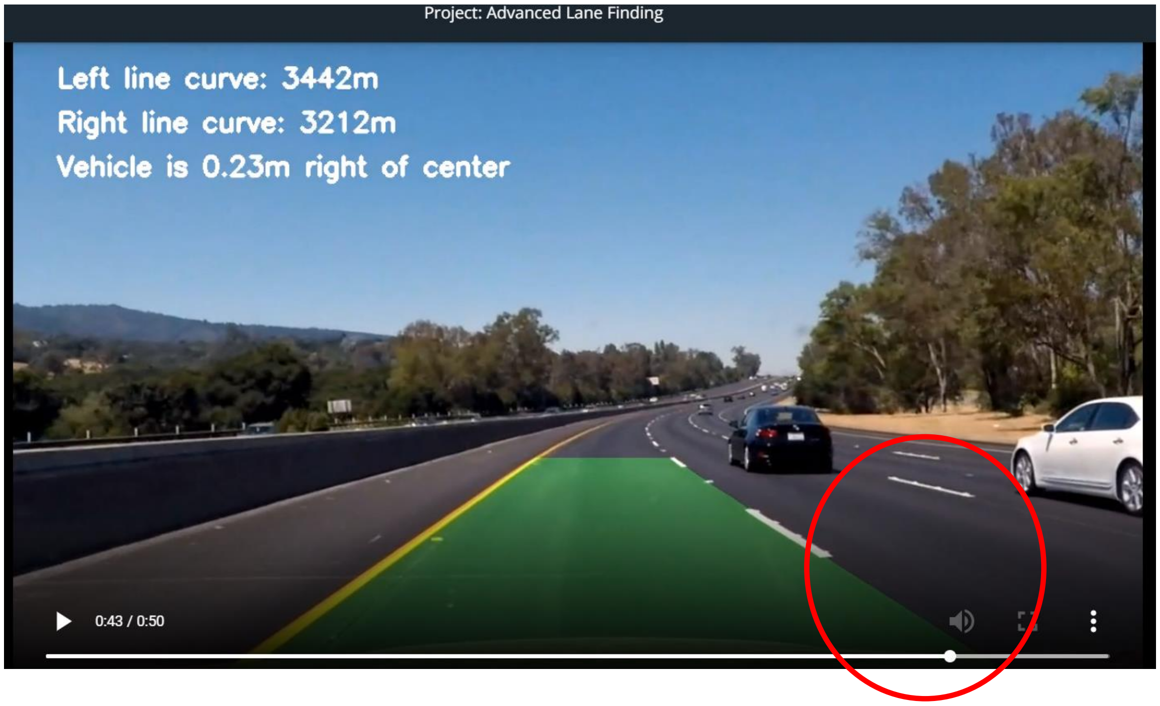
As for the pipeline, Testing it on more dataset and different weather conditions and low light conditions will give me a better idea on what can go wrong.

Changing the camera position will have an impact too.

Edit:

Thanks for the review, I have made some Schanges in the code , here is the snapshot of the time frame you mentioned from the new output.

The new output video is saved as "result_project_video_2.mp4" in the main directory.


Project: Advanced Lane Finding
Left line curve: 3442m
Right line curve: 3212m
Vehicle is 0.23m right of center
0:43 / 0:50