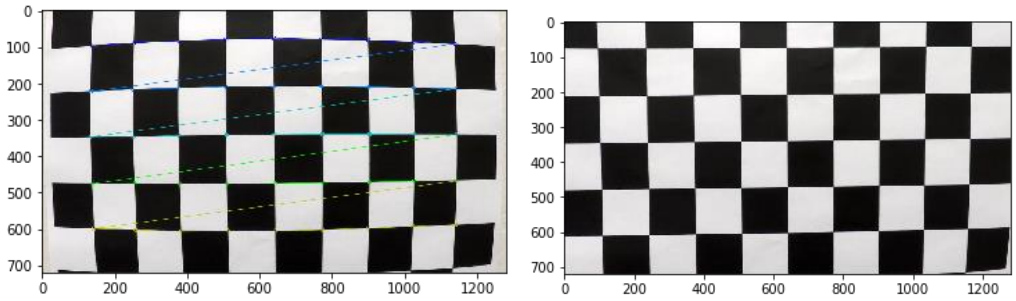
Meredith Meyer

Project 2: Advanced Lane Finding

Camera Calibration

I completed the camera calibration step in the same way as within the lessons. I utilized cv2.findChessboardCorners and cv2.DrawChessboardCorners to produce the first image below. With the object points and image points obtained through that step, I applied cv2.calibrateCamera to get the necessary information to apply the cv2.undistort function within the pipeline. After applying the warping to correct for camera distortion, the checkboard looks like the second image below.



Pipeline (images)

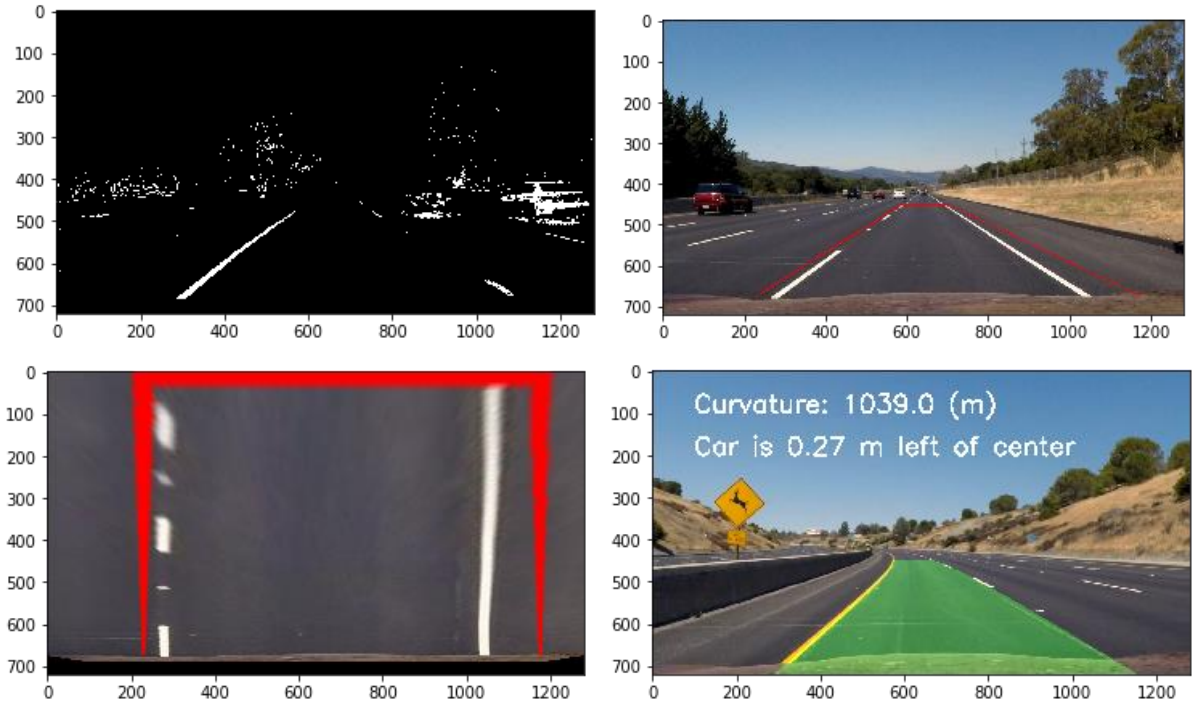
To generate the final result for each image, I followed roughly 6 steps:

1. Calibrate the camera (once, described above)
2. Undistort the image
3. Apply a binary threshold
4. Perspective transform
5. Find lane lines
6. Determine curvature

Then, finally displaying this information on the final image.

I did some experimenting with color channels to see which threshold would be most appropriate and robust. The S channel was consistently finding lanes on test images, so I utilized that one, as well as the Sobelx threshold. For perspective transform, I did some experimenting to find the points in the straight-line test images that produced straight, parallel lines after the transform. I used cv2.getPerspectiveTransform and cv2.warpPerspective to apply the transform. I used the sliding window method to find the pixels that makeup the left and right lane lines, and then I calculated curvature using the formula from the lesson. Finally, I drew the lanes on the image in green, and printed the curvature and offset values. The four images below show: the binary threshold image for a test image, perspective transform points, the warped image after perspective transform, and the final result image.

Meredith Meyer



Pipeline (video)

For the video processing, I used the same basic functions as in Project 1 to process a video frame-by-frame. My processImage function runs the functions I made to complete the 6 steps described earlier.

Discussion

This project required a lot of guess-run-check for me. One of the hardest parts for me was finding the points to use for perspective transform. What seemed to work for one straight-line image didn't produce the same clean result for the other, and the blurriness of the warped images make it difficult to see what might need to be changed. The thresholding step was another in which I had a hard time deciding when to be "done" and accept the current output. Therefore, I believe there to be myriad opportunities for the pipeline to be improved.

Most of my time, however, was spent trying to get a good result on the video. Smoothing/averaging the lines over frames was something I had trouble with. A simple average is probably not the best method, however I don't have sufficient knowledge for a better way to do this. It could definitely be improved.

From my time working on this project, I think that the lane fitting step caused the most issues. In some frames, one of the lane may have had fewer lane pixels, which throws off the curvature value, and sometimes the drawing of the lanes.

I would like to be able to improve the thresholding values to work on the challenge videos, as well as streamlining my code with functions, but for now I am satisfied with my output for the project video.