Advanced Lane Finding

The aim of this project is to find the markings on the road. This pipeline is more robust than the previous one, which employed Hough transforms and Canny edge detection, in a way that it even detects lanes around a corner. Curvature and offset estimates are also found out in this pipeline. This report describes the pipeline in detail, shortcomings and the probable improvements in order to make it more robust.

The following two techniques were used in order to complete this project:

1) Camera Calibration
2) Un-distortion
3) Perspective Transformation
4) Lane searching algorithm – Histogram peaks and sliding window techniques

Fig 1: Test Image

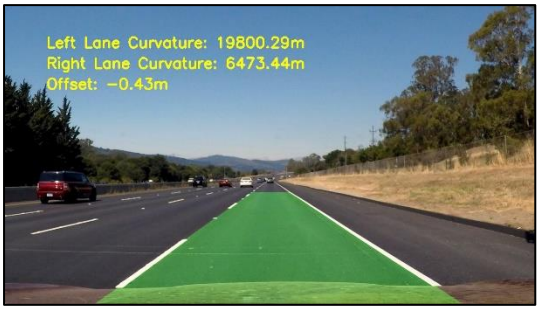
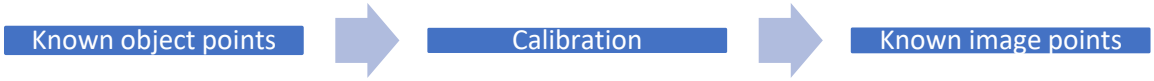


Fig 2: Detected Lane Lines on the test image

Pipeline:

The following describes the pipeline in detail. It describes first for a given image, and then extends the same pipeline to work on a video, because a video is nothing but a collection of images or frames.

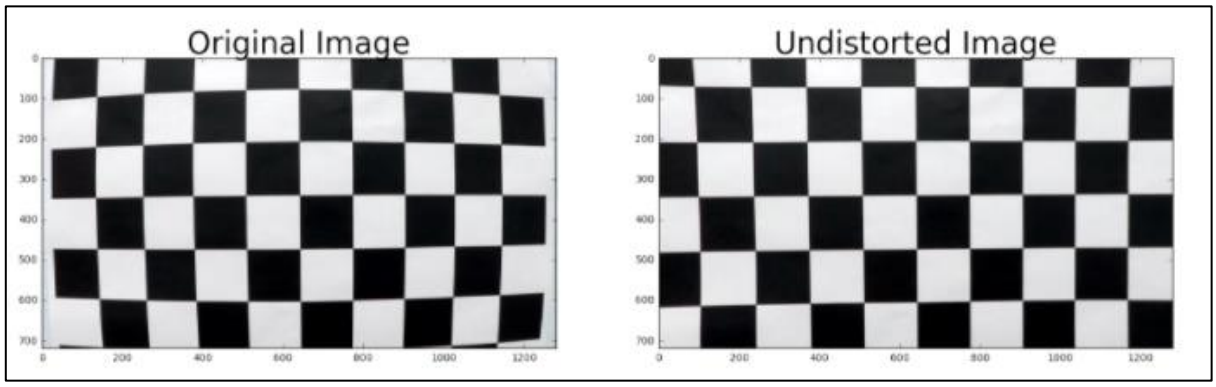
1) **Camera Calibration:** The first step involves calibrating the camera. Camera, being the sensor here, is far from giving out perfect images that describe the real-world scenes in their truest sense. There are certain errors that are inherent to the captured frames/images that may give rise to erroneous measurements and hence errors in the judgements made by the autonomous vehicle. Hence, the first step involves calibrating the camera in order to get rid of these incorrections. Following is the technique used to calibrate cameras.



Here, we make use of different images of a chessboard (the size of the chessboard is known). We then compare the real-world co-ordinates (3D with z=0) of the corners in a chessboard with the image points of the same co-ordinates. We make use in-built OpenCV functions (`findChessboardCorners & calibrateCamera`) to calibrate the camera. On doing the step, following are the parameters and coefficients that we obtain as a result:

1) Camera matrix – a matrix containing the intrinsic and extrinsic parameters of the camera that is being used
2) Rotational and translation vectors that describe the orientation of the camera in the real world
3) Distortion coefficients – radial and tangential coefficients that undistort a given image

Anywhere between 10-20 images would be required to calibrate the camera.



2) **Thresholding:** The aim of this step is to filter the given undistorted image/frame in such a way that we're able to extract enough information about the lanes we want to detect. The filtration that I carried out was based on three conditions:
   - **a.** The derivates were calculated using the Sobel operator and then thresholding was carried out on the images' gradient data thus obtained
   - **b.** The saturation channel of each image was extracted and then thresholding was carried out on this data
   - **c.** Lastly, the thresholding on pixel intensity of a grayscale image was carried out. This filtering did not help much in most of the test-images, but helped me extract some data that was otherwise lost in the first two steps

   The outputs of the above filtering were converted to binary-images (0=black & 1=white) and then combined together to render a single image. Masking was then carried out on this image in order to pass on only the important data to the next step.

Fig 3: Undistorted Image

Fig 4 – Filtered & Masked Image

3) **Perspective Transformation:** Now that we have our region of interest filtered out well, we can go ahead and find parameters that describe/define the lane lines – e.g. position of the car and the lane curvature. This will be part of the data that will be used to calculate the vehicle control parameters that keep the vehicle well under its and its surrounding's limits. Following are the steps followed to find the 'equation' of the lane lines even when they are curved:

   a. The filtered image obtained in step 2 is perspectively transformed in such a way that a 'bird's eye' view of the lane lines is obtained. Such a view is beneficial since it becomes easier to estimate the lane's shape when viewed orthogonally.

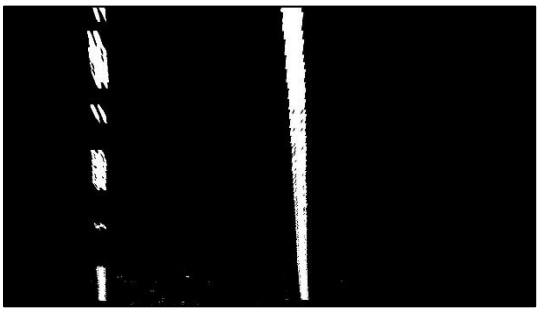


Fig 5: Bird's eye view of the lane lines in the region of interest

   b. Next, we use the transformed image to fit a curve over the activated pixels (lane-lines). This is done by first finding the start point(**i**) of each lane, and then using a search window(**ii**) to follow the lane all the way to the top of the image. All the pixels that lie within this window are accepted, and the rest are rejected. The window dimensions are user defined.

      i. The pixel values along columns are added, and then the peaks are found out within these sum values. The peaks are nothing but the location in X where the lane lines begin; so, this is where the searching begins.

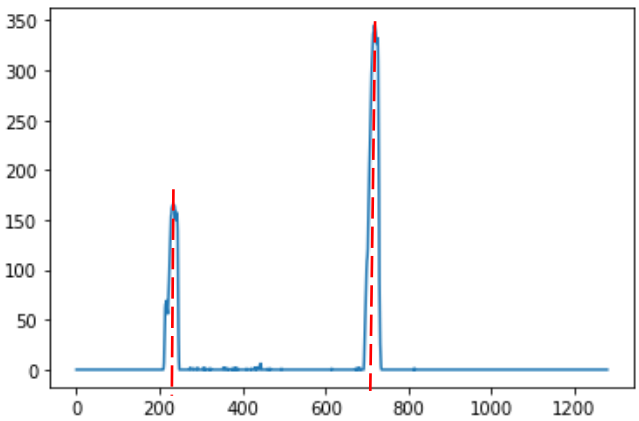
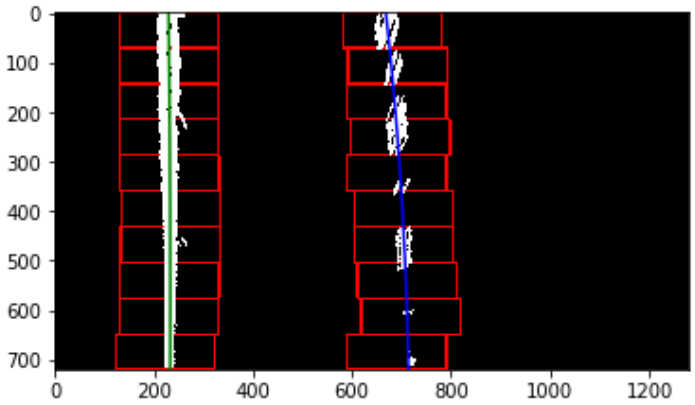


Fig 6: Peaks for search start point estimation

      ii. Using the start points and the search window, we look for activated pixels that lie within the search boundaries. Using these pixels, we then fit a second-degree polynomial to obtain an equation that represent the lane lines. All the required curvature and offset calculations are carried out using this equation



This concludes all the major steps in lane estimation. The following steps include projecting the lane lines obtained, onto the original image**.**

4) **Inverse transformation:** Here we transform the image containing the bird's eye view of the region of interest (defined by the lane lines obtained in step 3)**,** back to the real-world perspective for better visualisation. We use **inverse** of the transformation matrix obtained in step (3a) to transform the image. We then overlap this image onto our original image to get the final output.

a. Here, the radius of curvature is measured using the following expression:

$$\rho(t) = \frac{\left|1 + f'^{2}(t)\right|^{\frac{3}{2}}}{|f''(t)|}.$$

Where f(t) is the polynomials obtained in step 3. Considering that the polynomial was found out in terms of pixels, in order to have curvature and offset values in terms of meters, we scale the x and y using factors that define the no. of meters per pixel in x and no. of meters per pixel in y.

b. The offset values are obtained by assuming that the camera is mounted at the centre of the vehicle, and hence the image. The value is obtained by estimating how far the centre of the image is from the centre of the lane lines obtained, again scaled using the 'pixels per meter factor'

Video pipeline:

1) For processing the video, everything remains the same except that the starting point for the lane line, instead of being estimated at each and every frame using the histograph technique, is obtained via the previous frame's measurement. This reduces the time to process each and every frame
2) A smoothing is also carried out in order to get a smoother output. In this step we smooth the polynomials by taking a mean of the polynomial obtained in the previous n frames. This particularly helped in the region where there were sudden changes in the scene in terms of shadow & the colour of tarmac.

Shortcomings and Improvements:

1) This pipeline works well where the lane curvatures are not very low, and hence this would fail in the harder challenge video when it will not be able to find the lanes at all. In that case, the region of interest would have to be increased and thresholding the image based on gradients and colour spaces would have to be more robust.
2) As mentioned above, the thresholding can be better in the current pipeline as well. The pipeline seems to be pretty shaky whenever there are shadows in the image or there are lines in the region of interest that are other than the lane lines. Hence, I intend to better this filtering by exploring the different colour spaces available.
3) Currently, the smoothing has been carried out by taking the previous 10 frames. This can be bettered by taking a non-uniformly weighted average instead of a uniform weighted average; by giving most weightage to the last frame.
4) Currently the lane start points at each frame are taken from the previous frame. This can be further improved by simultaneously checking the goodness of fit value in the previous frame. For a higher fit, the margin within which the next start point is estimated can be lower than the instance in which the previous frame had a bad fit. If the quality is way below the accepted levels, then the histogram search method can be employed instead.