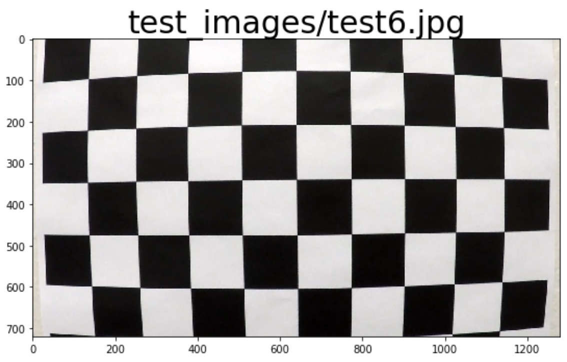
Advanced Lane Finding Project

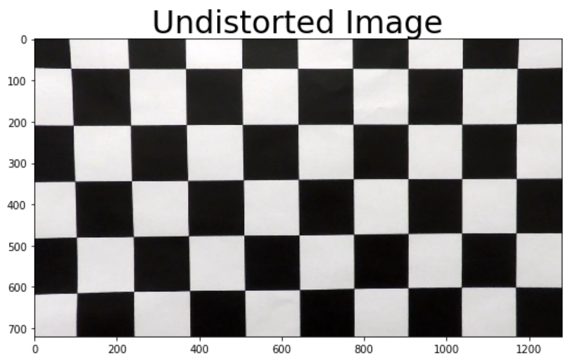
Camera Calibration

Have the camera matrix and distortion coefficients been computed correctly and checked on the calibration test image?

Here I have calibrated the camera using the functions as discussed in the tutorials. There are 9 corners in rows and 6 corners in columns. The images in camera_cal folder have been used to calibrate this camera.

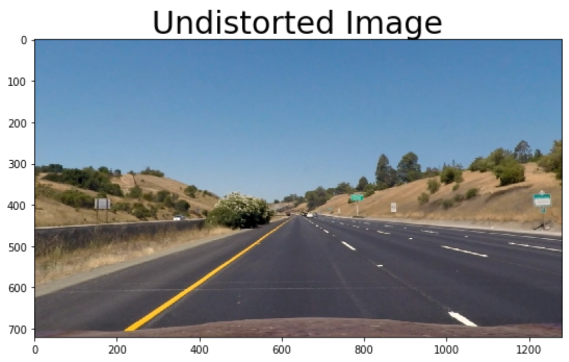
The object points refers to the 3d points on the chessboard in real world space. The image points are calculated using cv2.findChessboardCorners(). These object and image points are further used to calibrate the camera using cv2.calibrateCamera() function.





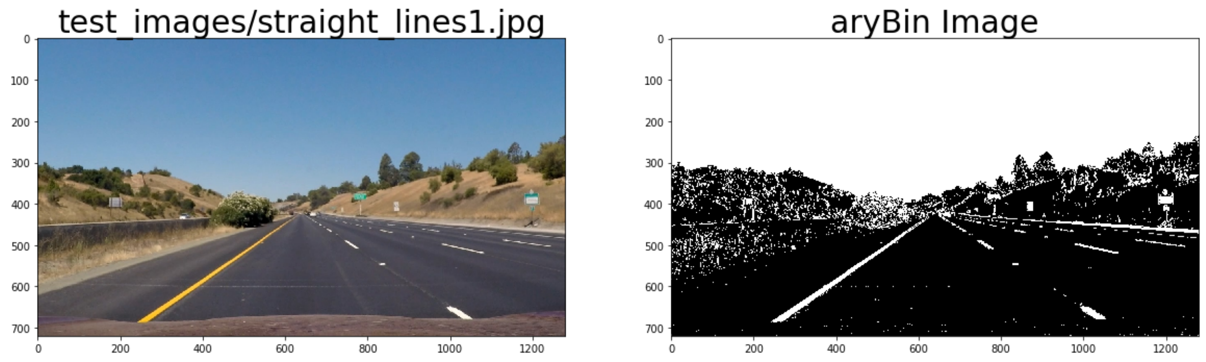
Pipeline (single images)

- **Has the distortion correction been correctly applied to each image?**
  Yes! You can find them at test_images/undistort. It looks something like this:



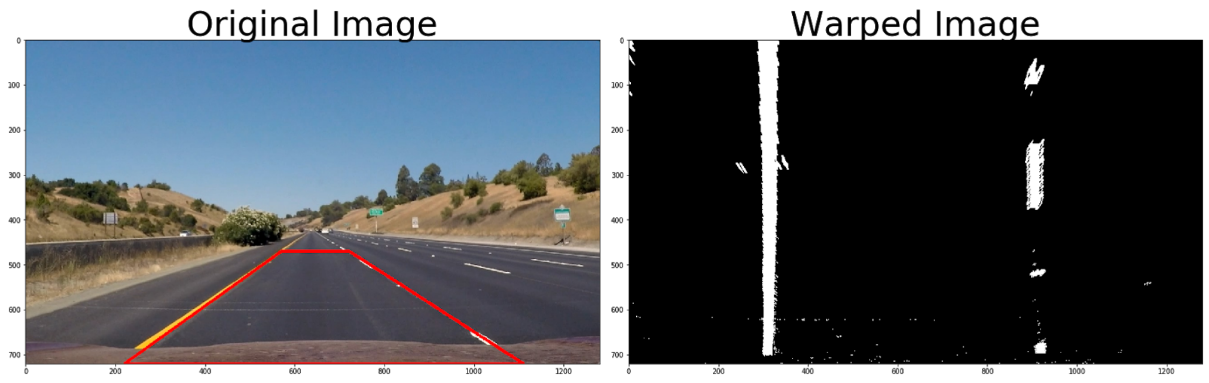
- **Has a binary image been created using color transforms, gradients or other methods?**

The binary image has been created by taking the saturation channel, red channel and the x-gradient threshold. These thresholds were chosen after experimenting in different channels and thresholds. Although it works in the video, it doesn't do well in intense brightness or broad daylight, for which, the lightness channel also needs to be incorporated.



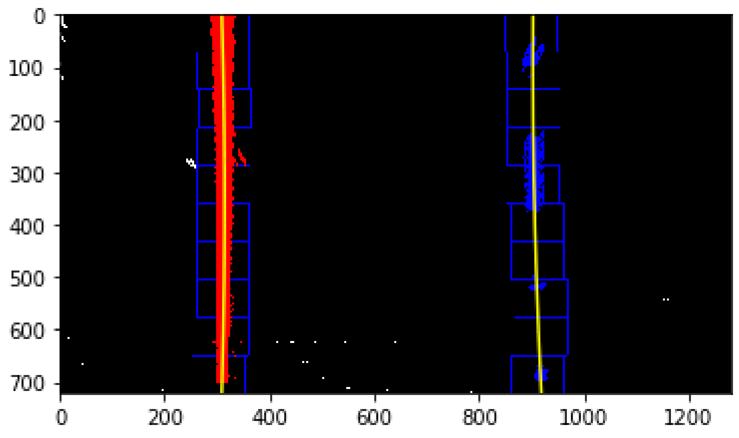
- **Has a perspective transform been applied to rectify the image?**

Yes! Here, the source and destination points have been entered manually. These points are used to get the transformation matrix *M* and the inverse transform matrix *M_inv* using the function cv2.getPerspectiveTransform. You can see the bird-eye-view of the image below.



- **Have lane line pixels been identified in the rectified image and fit with a polynomial?**

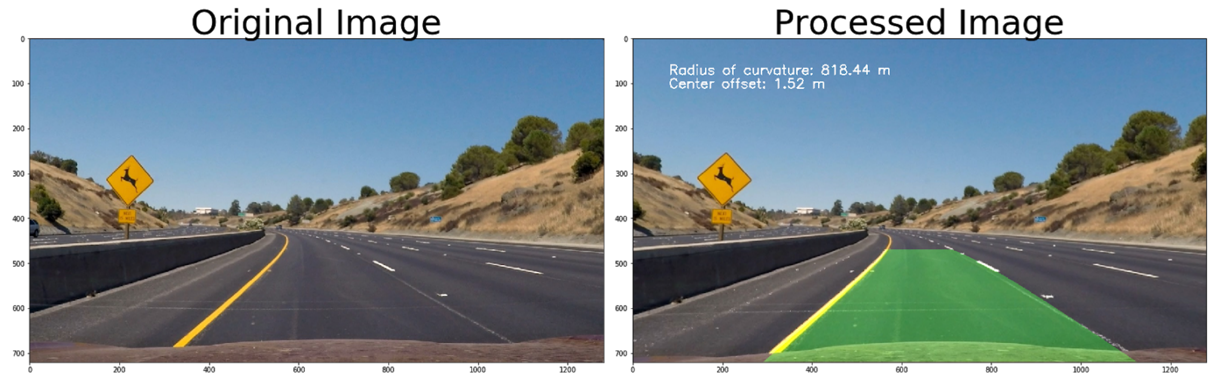
The sliding window technique has been used to detect and find lane lines. This is an initial brute force technique to find lane lines. Later in the pipeline, this step has been avoided(wherever possible) and an average of previous 10 frames has been taken to reduce computation and improve accuracy.



- **Having identified the lane lines, has the radius of curvature of the road been estimated? And the position of the vehicle with respect to center in the lane?**

  Yes! The formula mentioned in the tutorials has been used here. Also, the final value has been converted from pixels to meters to give real world data.

- **Has the result from lane line detection been warped back to the original image space and displayed?**



Pipeline (video)

In the pipeline, a few global variables have been used to keep track of the previous 10 frames which are used to take average of lane lines. In case we have appropriate values from previous frames, the search is done apriori as mentioned in the tutorials.

To detect outliers, apart from the threshold values used to create the binary image, imperfect lane findings have also been removed. It compares the current mean difference between the left and right lanes and the previous averaged mean difference. If it goes beyond a threshold, then the lane detected is considered as an outlier.

In the end, the warped image is unwarped using the *M_inv* matrix and is filled with green color, followed by writing the corresponding offset and ROC on the video.

Pitfalls and scope of improvement

The binary conversion doesn't do well in broad daylight and shadowy conditions leading to unwanted results. Further, the perspective transform should be taken on a smaller section to work well on steep curves. Also, taking an average of 10 frames is not good enough because it slows down the pipeline. Taking average of fewer frames mi