# Advanced Lane Finding

# Aims

The main aim of this project is to accurately find the lanes on the road using the camera image mounted in the car. Based on the camera images the road lanes are found using traditional computer vision techniques. The project is performed using the following steps:

1. Computation of the camera calibration matrix and distortion coefficients using given a set of chessboard images.
2. Application a distortion correction to raw images.
3. Using of color transform, gradients, etc., to create a threshold binary image.
4. Application of perspective transform to obtain bird’s eye view of the binary image
5. Detection of lane pixels and fitting of the lane boundary.
6. Vehicle position and lane curvature determination.
7. Warp the detected lane boundaries back onto the original image.
8. Displaying the output with determined lane markings, estimated lane curvature, and the vehicle’s position

# Camera Calibration

The problem with camera images is that it can have distortions. In order to remove the distortions from the camera should be calibrated. The distortions can be removed using distortion coefficients and the camera matrix. The distortion coefficients and the camera calibration can be calculated if a picture taken in the camera contains an object whose position is known. In order to do this chessboard is used. Several images of a standard chess board are taken from different angles and distances. Since points of the corners in the chessboard are known, which are called as object points, they are compared with the corners found in the image, which are called as image points. The opencv library in python provides in built functions to find the corners of the chessboard and to calibrate the camera. The first few cells in the jupyter notebook titled “AdvancedLaneFinding.ipynb” shows camera calibration. Several chessboard images are read, “*cv2.findchessboardcorners()*” function is used to find the image points then the object points and the image point are feed to “*cv2.calibrateCameera()*” function to find the distortion coefficients and camera matrix. The distortion coefficients and the camera matrix, which are used later to remove distortions from the road images, are then stored in a pickle file “CameraCalibration.p”. The below picture shows an example of distorted chessboard image and the same image of the chessboard with its distortions removed. It can clearly be seen that the radial distortions, which makes the edges of the chessboard curved are removed in the undistorted image.

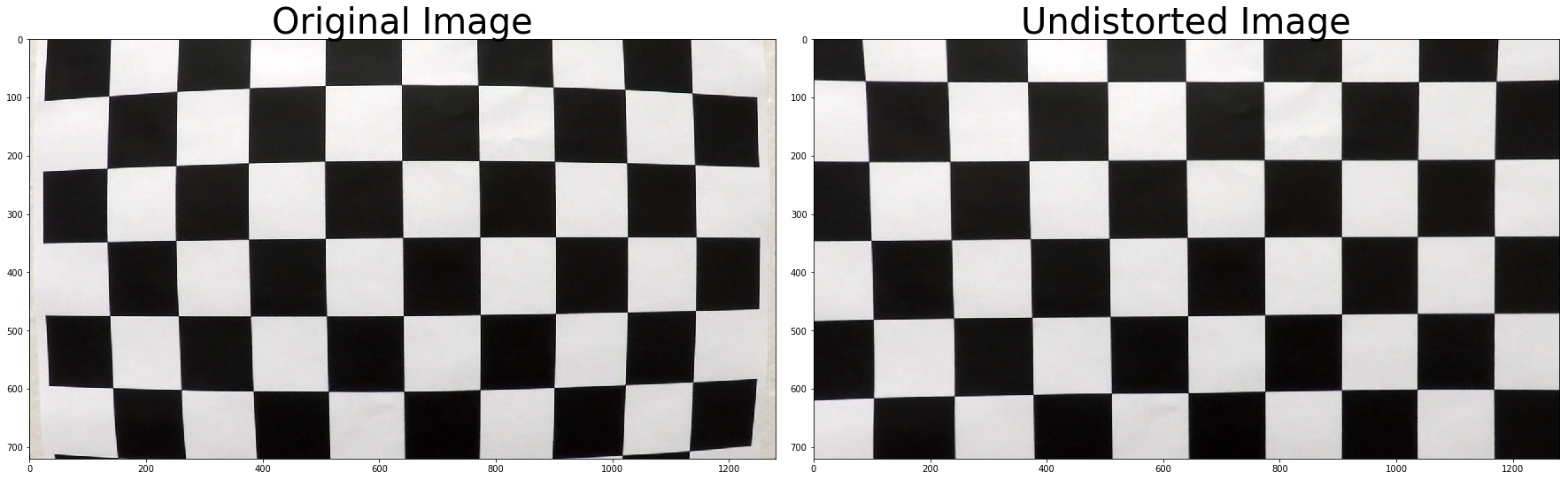


Fig : Example of distortion removal from chessboard image

The picture below shows the distortion removal in one of the test images used to find the lanes. There is not so much of the difference between the distorted and undistorted images but there are small differences such as the position of the white car and the shape of car’s engine compartment.



Fig : Example of distortion removal in one of the test images

# Binary Image

In order to detect lanes on the road from the images the features in the image such lines should be extracted. In the first project the lanes were detected using canny edge detections algorithm which was working well for some images but it was not able accurately detect edges in cases such as bad lighting, conditions, images with shadows etc. In this project several methods are used to extract lane markings which are gradient detections, color transformations. In the gradient detections the sobel transform is used to calculate the gradient images along the x and y directions. Based on the gradient of images a predefined values are chosen in the gradient values of the pixels within the range are set to 1 and the rest of the pixels are set to zero giving a binary image. The HLS and HSV color transform are applied and the binary images are obtained for different color channels. Finally a binary image is obtained using a combination of different methods. The table below shows different methods used for getting the binary image.

|  |  |  |
| --- | --- | --- |
| Method | Threshold Value | Reason |
| X gradient | 20 - 100 | Detects edges in the X direction. |
| Y gradient | 20 -100 | Detects edges in the Y direction |
| Magnitude of X and Y | 70 – 100 | Detects edges in both X and Y directions |
| V channel (HSV) | 150 – 255 | Detects shadows and dark regions in the image |
| L channel (HLS) | 50 – 255 | Detects shadows and dark regions in the image |
| S channel (HLS) | 100 – 255 | Detects the regions based on the level of color saturation. The road lanes have saturated color so it is useful to detect the lanes, especially continuous lane. |

The picture shows the example of the binary image compared to a test image.



Fig : Binary image using gradient and color transform

# Perspective Transform

When perspective transformation can be applied to an image a new image from a different perspective i.e. different view point can be obtained. Once the binary image showing the road lanes are obtained perspective transform can be applied to get a bird’s eye view of the road lanes. This helps to accurately detect the position of the lane and also more information such as curvature of the road, direction of curvature etc. In order to apply perspective transform the transformation matrix should be calculated by choosing four points in the original image and where these points would be located in the transformed image. Then the transformation matrix can be used to warp the image to a new perspective. The opencv library has functions to calculate the transformation matrix “*cv2.getPerspectiveTransform()*”and to warp an image to a new perspective “*cv2.warpPerspective()*”. In order to get the points for applying perspective transform the lane detection algorithm in the project “Finding lane lines” is used. When the binary image is feed to this algorithm it detects lines on the image using Hough transforms in a masked region and detects the lane lines, which are a set of four points two for each lane. These four points are the source points for the perspective transform. The destination points are calculated based on the simple assumption that the lane lines are straight and parallel when looked from a bird’s eye view. The pictures below shows the result of perspective transform on a test image and the binary threshold image.



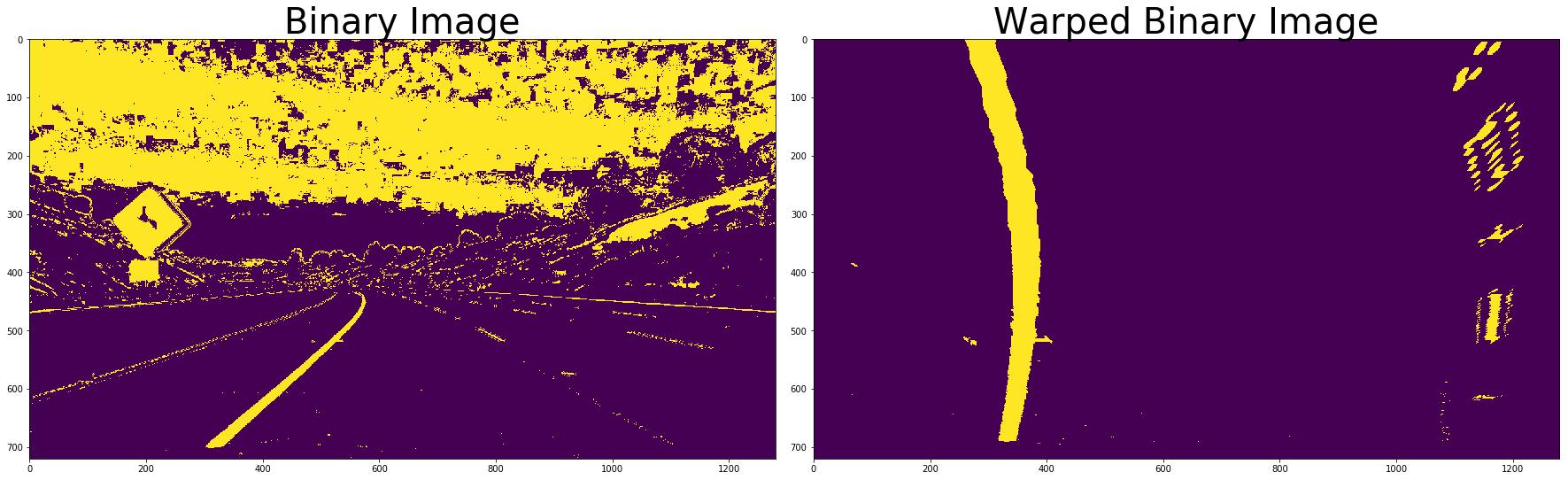


Fig : Example of Perspective Transform applied on a test image and binary Threshold image

When the source and destination points are switched in the calculation of the transformation matrix an inverse transformation matrix can be obtained. The inverse transformation matrix is helpful to project the lanes to the original image after its detection in the top view.