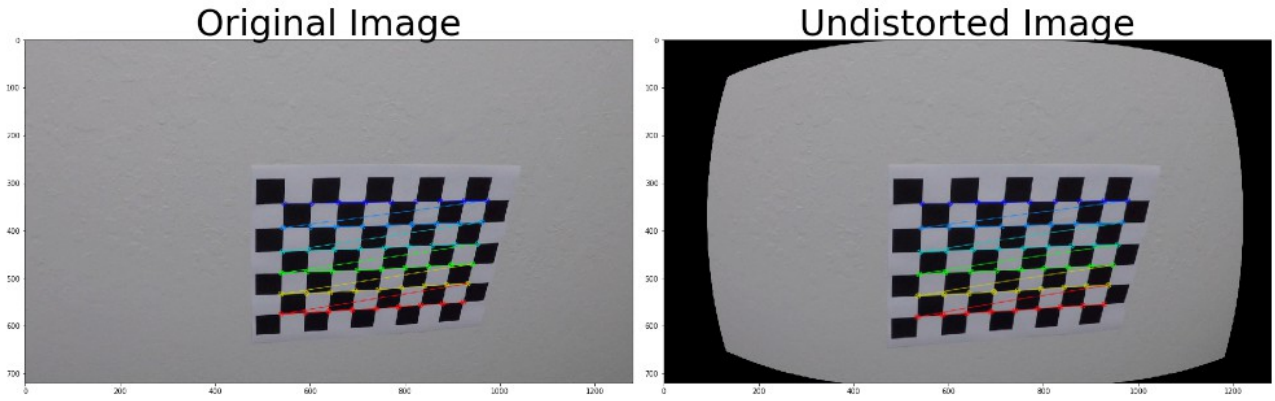
Advaned Lane Finding

1. Camera calibration and undistortion

First I define two functions for camera claibration and for undistorting images. I took the provided images, find the chessboard corners and undistort the images. for success control I printing out original and undistorted image.

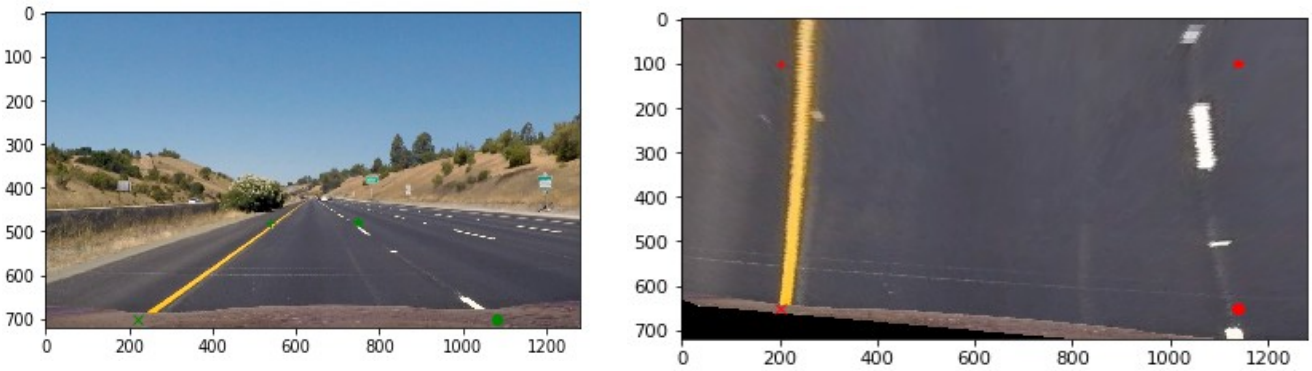


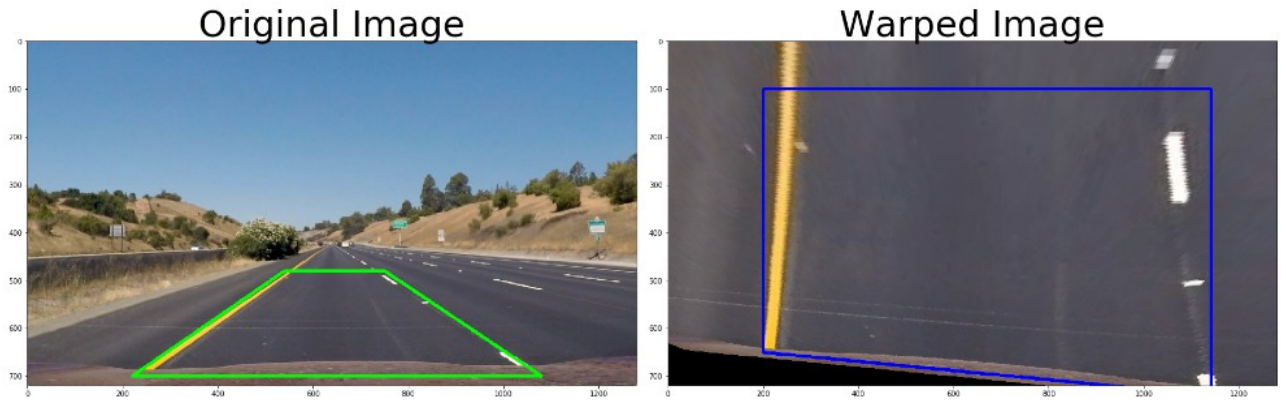
2. Preprocessing

After Cameracalibration I applied the ideas from the lessons to prepare the images. For that I define some functions. The Sobel operator to detect the horizontal and vertikal lines from canny edge their magnitude and the direction of them. For better results in difficult situations like shadows and different colours of lane lines I convert the image into hls colorspace. The last function in this block is the warp function. By that function I realized the perspective transform from original to birds eye view.

3. Source and destination points

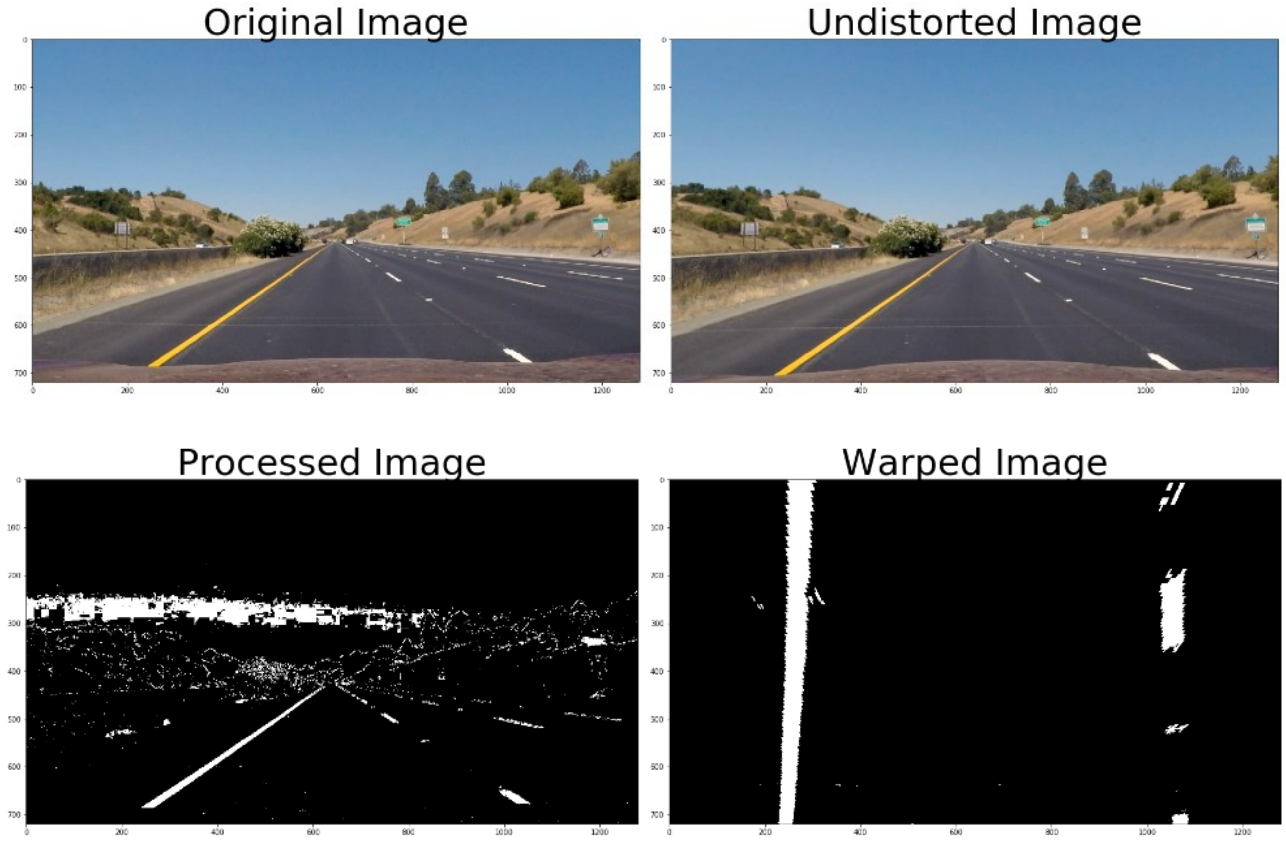
In this block I append source and destination points on original and warped images and fit lines through them. The result looks a little bit strange. My expectation was that there were two straight parallel lines in the top down image but it is a little bit distorted.



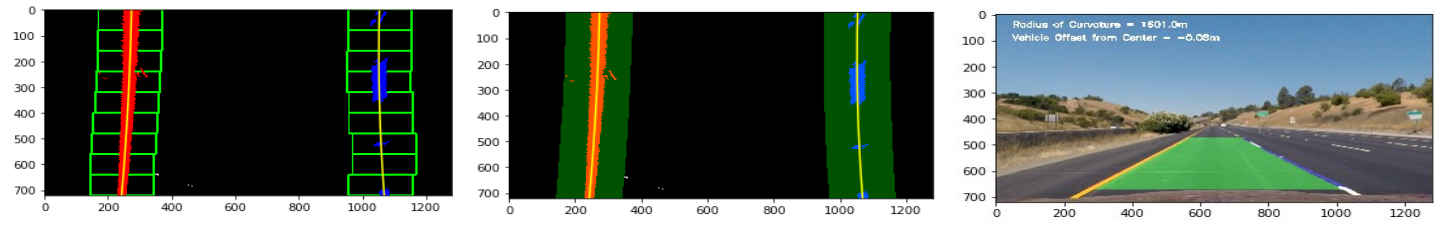


4. Processing sample images

In this block I go through the sample images, preprocess and undistort them, calculate the Transformation Matrix and the inverse Transformation Matrix and finding the lanes by applying the sliding window method from the lesson. Applying a polynomial and curvature measuring are also from the lesson.Visualizing the results are the last steps in the block.




```
Left Curvature 1902.54 m
Right Curvature 1298.54 m
Vehicle Offset from the Middle -0.08 m
Average Curvature 1600.54 m
```



5. Video clip pipeline

A leaner version of the above image process is using for the video. To aviod jittering I´m implemented a buffer and a monitor to restrikt curvature change. Rest is carryover from image pipeline.

6. Processing the video

This part nearly drove me crazy. It took a long time to understand.. no better to handle this fault:

OSError: [WinError 6] The handle is invalid

Restarting the Kernel was the solution.

7. Discussion

The present code work well on the project video because the challange is not very sophisticated. Running on the second video shows some weak points. The behavior when lighting changes happen like sunny and shadow areas or sunbeams is really terrible. I think implementing a more powerful outlier function to eliminate strong changes in curvature will help to reduce jittering. Another helpful improvement is to optimize image preprocessing.