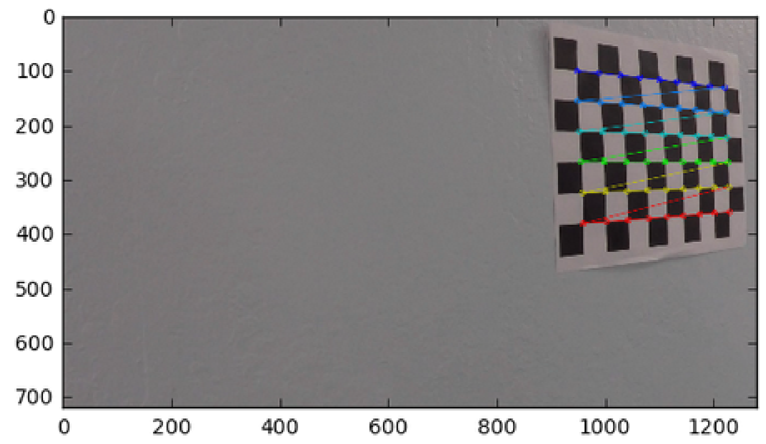
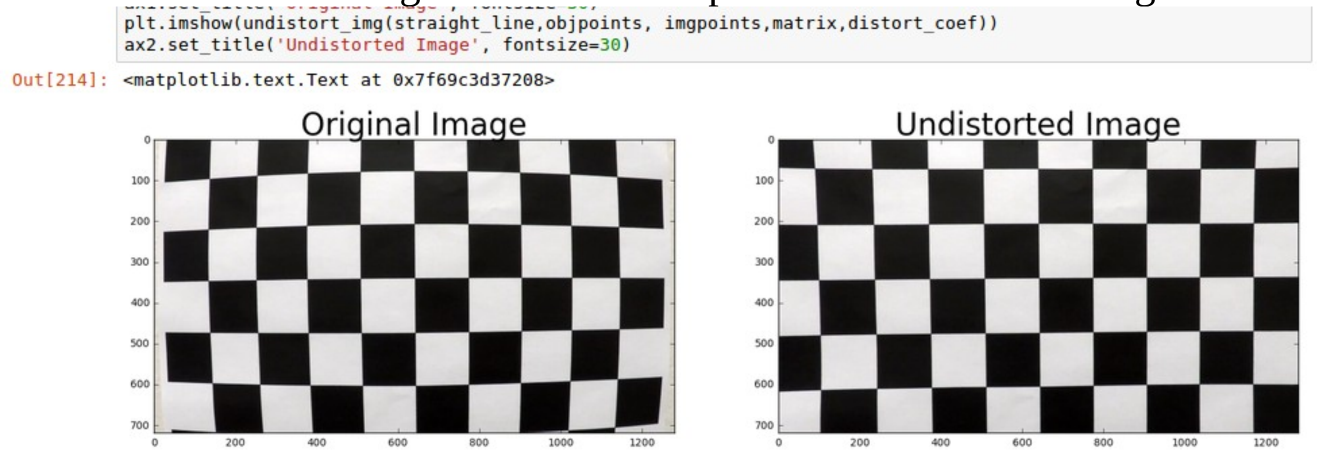
Advance line tracking problem

The first step I went through in order to solve this problem was to calibrate the camera. The code to achieve this part of the work is included in cells 1 to 4



after calibrating the camera, we now have the calibration matrix that can be used to undistorted images. Here is a sample of an undistorted image:



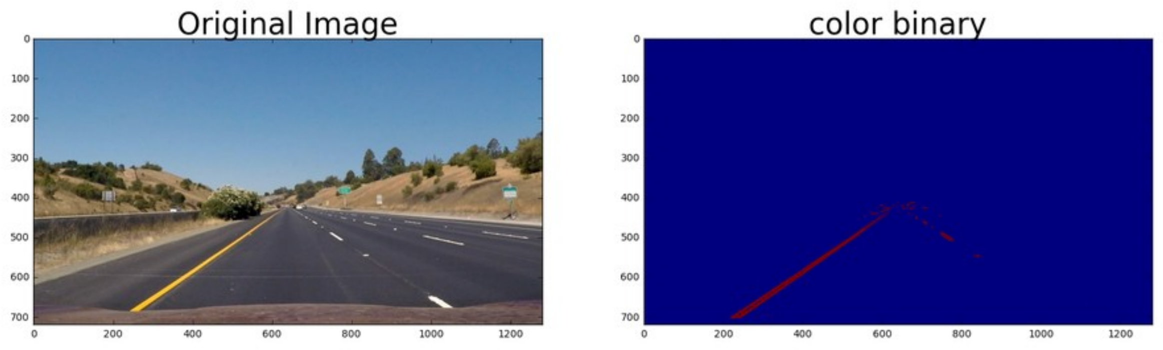
```
plt.imshow(undistort_img(straight_line,objpoints, imgpoints,matrix,distort_coef))
ax2.set_title('Undistorted Image', fontsize=30)
```

```
Out[214]: <matplotlib.text.Text at 0x7f69c3d37208>
```

The undistorted image is the starting point of processing pipeline.

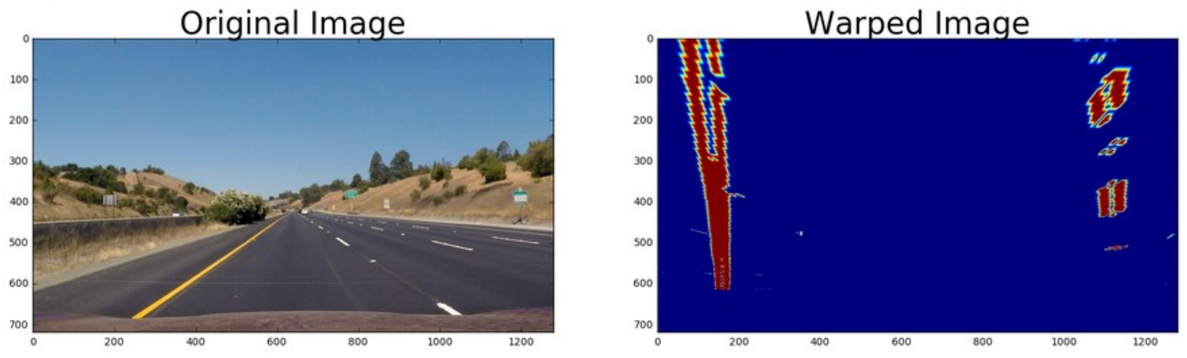
First I applied sobel, magnitude gradient and directional gradient thresholding to the undistorted image. All the thresholding results are then combined into a binary image.

```
Out[18]: <matplotlib.text.Text at 0x7fa2d4202320>
```



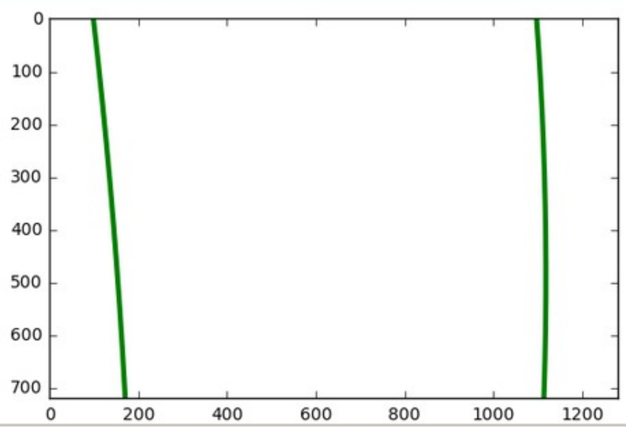
Then the binary image is warped via a perspective change to better isolate the lines

```
Out[19]: <matplotlib.text.Text at 0x7fa2d4154cc0>
```

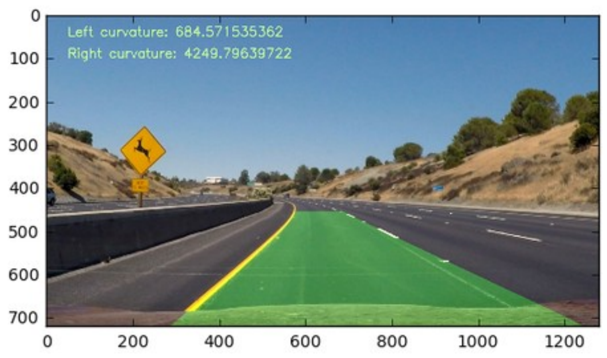


a line is detected in the warped image and extrapolated to provide the following:

```
app.launch_new_instance()
```



after the lines are correctly detected in the perspective space, the region of interest is warped back into the original image.



Reflexions:
it was very hard to manually find thresholding values and play with different thresholding methods, I think there is still room for improvement in terms of finding the right combinations of thresholding values. The pipeline might fail for many reasons including change of brightness, change of colors due to the presence of shadows for instance. Another important reason why the pipeline can fail is that we are doing a second order polynomial fit which will not work with roads with a lots of turns.