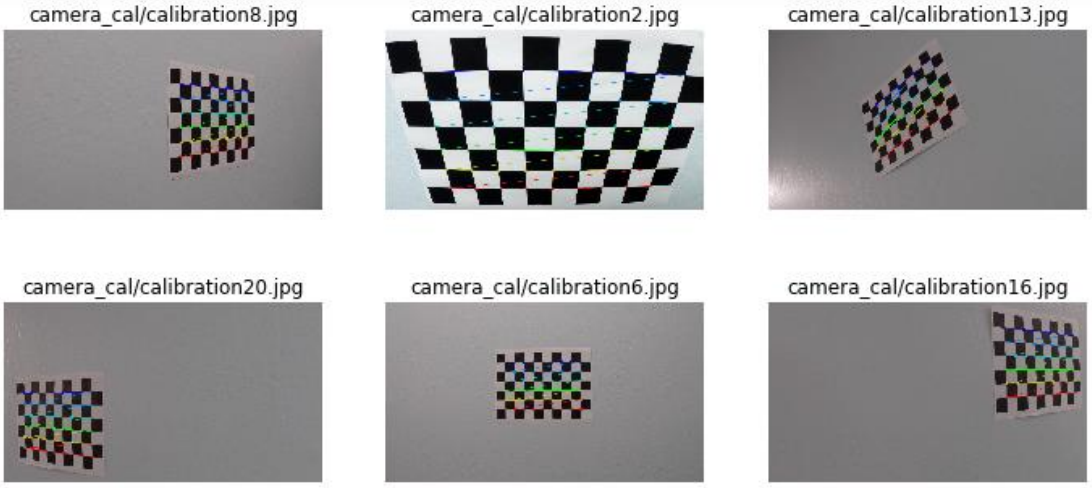
Advanced Lane Lines Detection

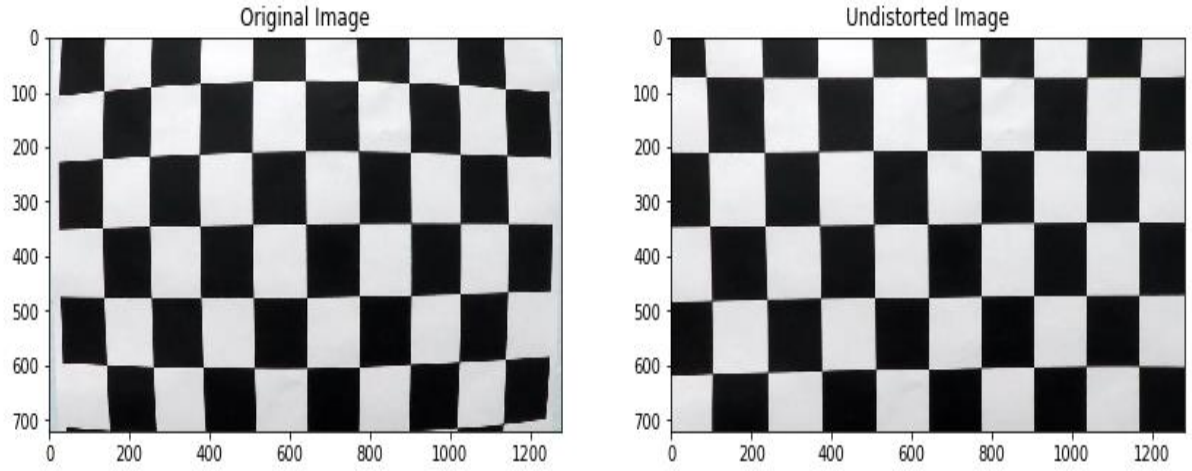
Camera Calibration

This process done to remove the distortion in the images, I used the chess board images to calculate the distortion coefficients



Remove Image Distortion:

Using the distortion coefficients from the last step, we can easily remove the distortion from the images as shown below



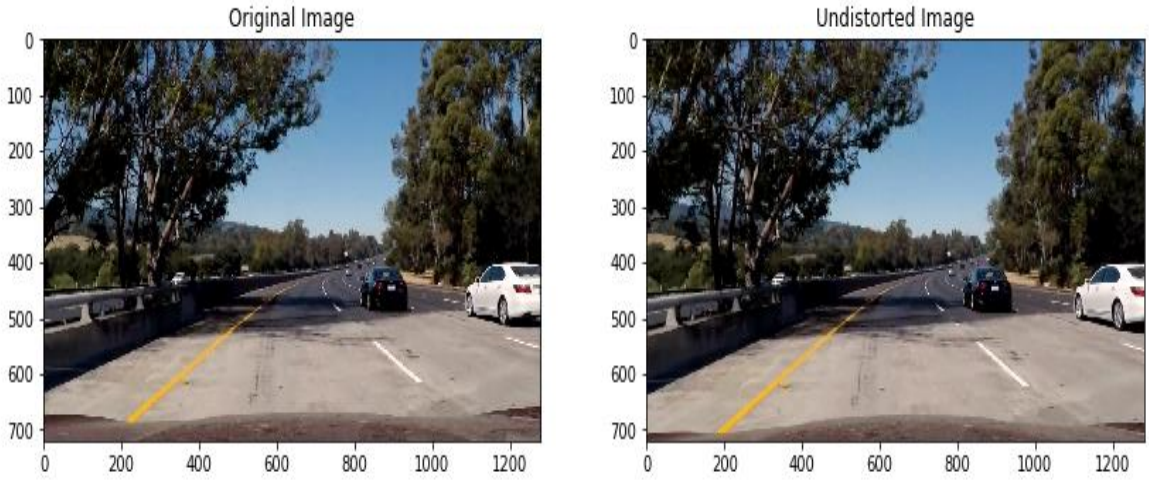


Image Gradient:

In order to focus only on the lane and find its curvature we need to convert the image to a binary image, and to remove the effect of the lightness we convert the image from RGB to HLS and apply the Sobel filter to the S channel.

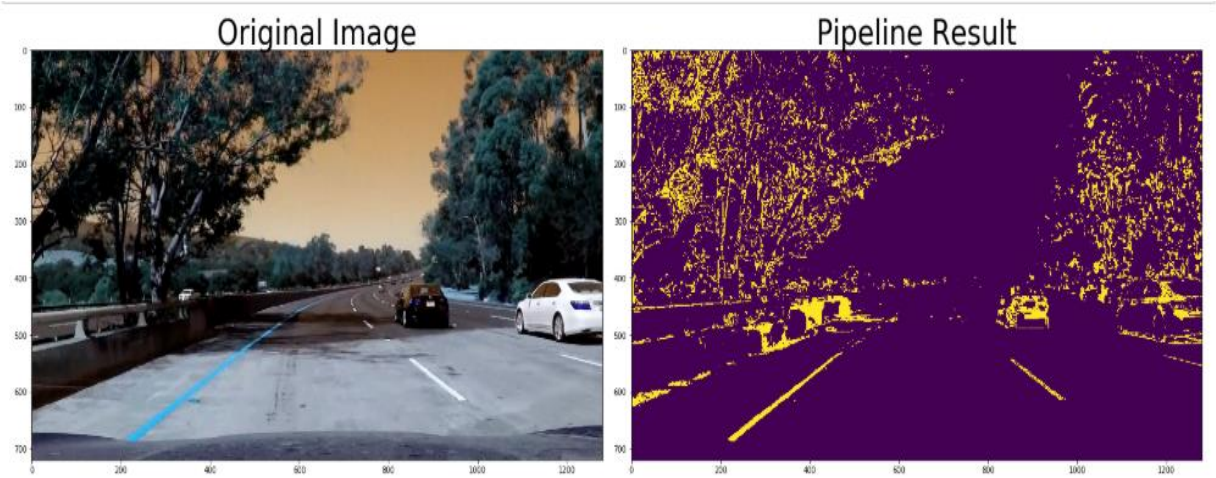


Image Transformation:

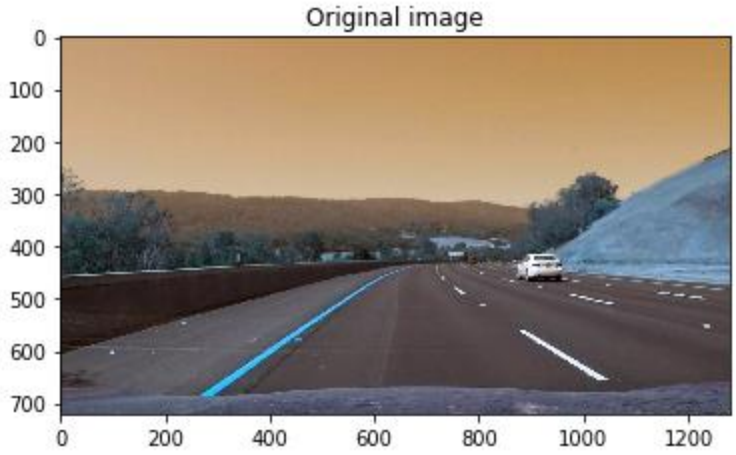
In order to keep the lane lines, parallel a bird's eye view transformation was applied as shown below:

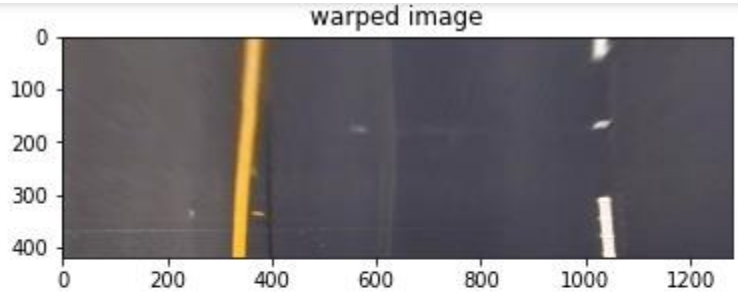
Color Masking:

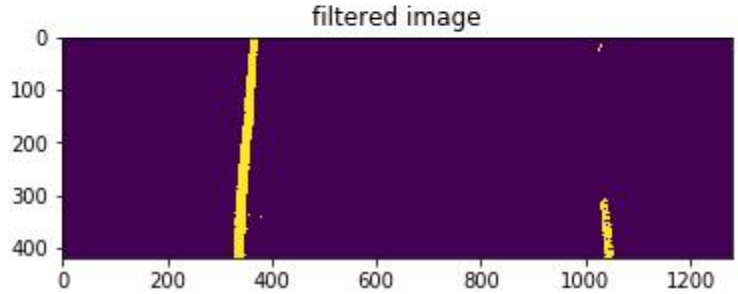
I developed a color mask to cancel any noise by removing all colors except for white and yellow

The lane curvature:

in this section I split the image in two halves to find the lane curve in each half and then draw it in the image, the last step, I performed the inverse transform to draw the curve in the original image as shown below


Original image
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


warped image
0
100
200
300
400
0
200
400
600
800
1000
1200


filtered image
0
100
200
300
400
0
200
400
600
800
1000
1200

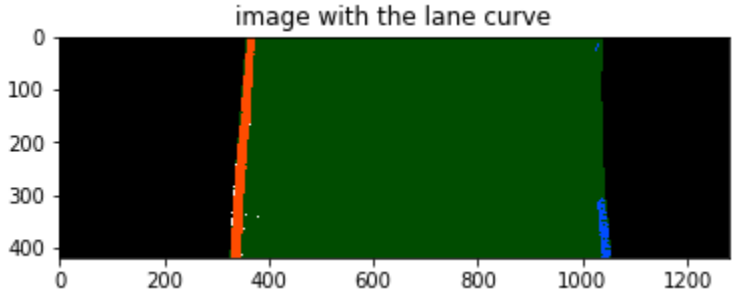
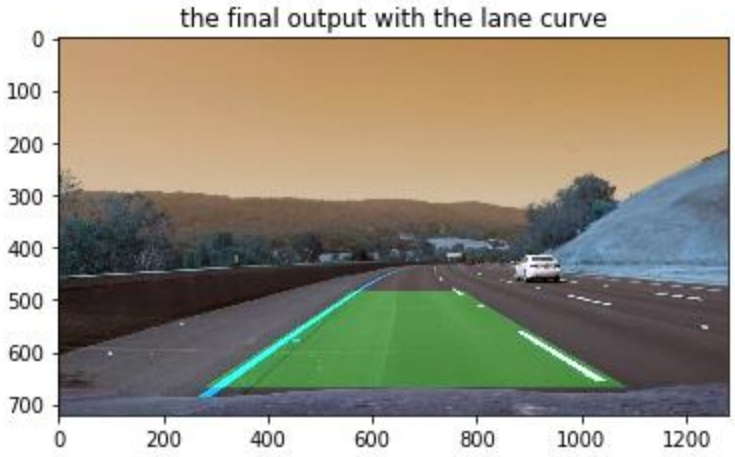


image with the lane curve
0
100
200
300
400
0
200
400
600
800
1000
1200



I applied the same technique to each frame in the video as shown here

problems/issues:

failing in shadows and brighter lane: I overcame this problem by enhancing my color mask to be able to detect the colors in all cases

Conclusion:

- **Step 1**: Apply distortion correction using a calculated camera calibration matrix and distortion coefficients.
- **Step 2**: Apply a perspective transformation to warp the image to a birds eye view perspective of the lane lines.
- **Step 3**: Apply color thresholds to create a binary image which isolates the pixels representing lane lines.
- **Step 4**: Identify the lane line pixels and fit polynomials to the lane boundaries.
- **Step 5**: Determine curvature of the lane and vehicle position with respect to center.
- **Step 6**: Warp the detected lane boundaries back onto the original image.
- **Step 7**: Output visual display of the lane boundaries and numerical estimation of lane curvature and vehicle position.