ADVANCED LANE FINDING

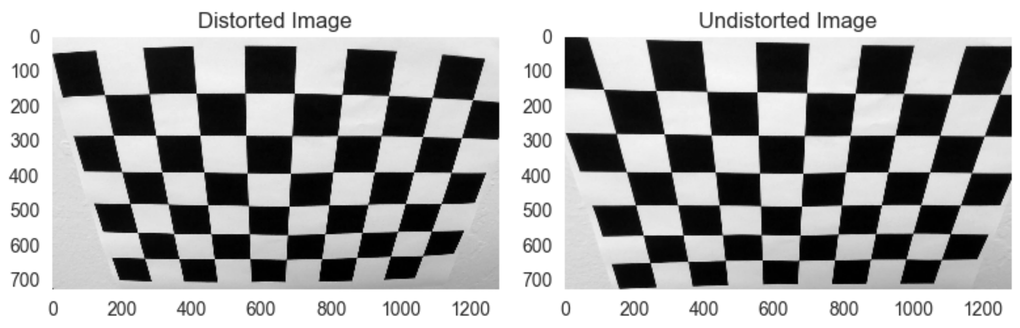
The following is a description of the steps for identifying lanes from a front-facing camera. The steps include:

- **Distortion Correction:** Correct image distortion produced by camera.
- **Image Thresholding:** Create a binary image that only includes lane pixels by processing image color and gradient information within a region of interest.
- **Perspective Transformation:** Rectify the binary image to a "bird's eye-view".
- **Lane Segmentation and Curve Fitting:** Detect right/left lanes and fit boundary curves using information from previous frames when appropriate.
- **Curvature and Position Estimation:** Determining lane curvatures and vehicle position.
- **Visual Display:** Warping detected lane boundaries onto the original image.
- **Smoothing and Filtering:** Averaging lane curves across frames and drop those with poor lane detections.

DISTORTION CORRECTION

Correcting camera image distortion involves using calibration images to derive the camera matrix and distortion coefficients, and then using those parameters to undistort newly acquired images.

The images below illustrate a calibration image before and after correction. In the original image (left) one can see the curvature introduced by radial distortion at the top of the image. In the corrected image (right) radial distortion is drastically reduced.



The images below illustrate correcting distortion in a frame from the project video. In the original image (left), the highway exit sign (upper right corner) appears slanted due to radial distortion. In the corrected image (right) the exit sign appears to be squarely facing the camera

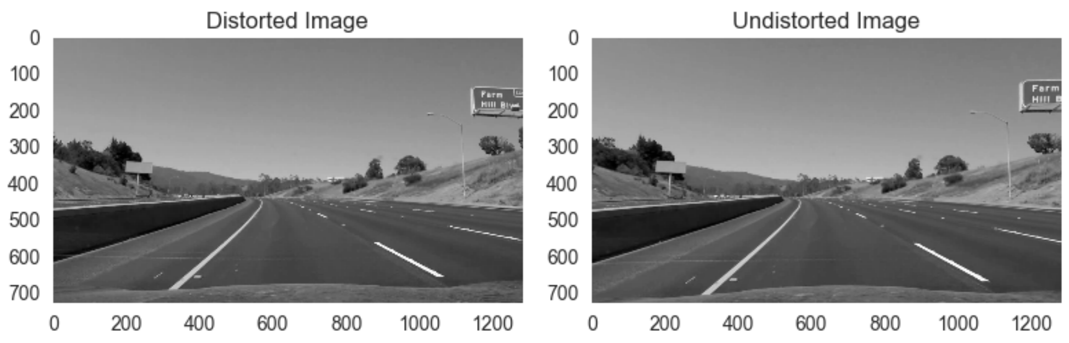
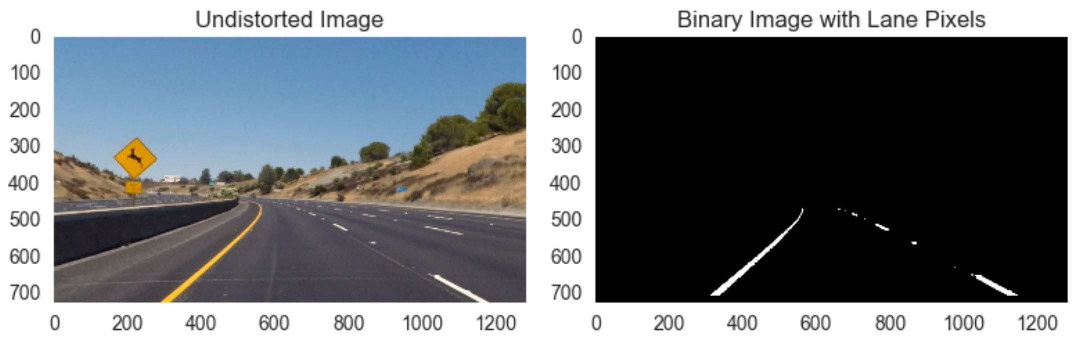
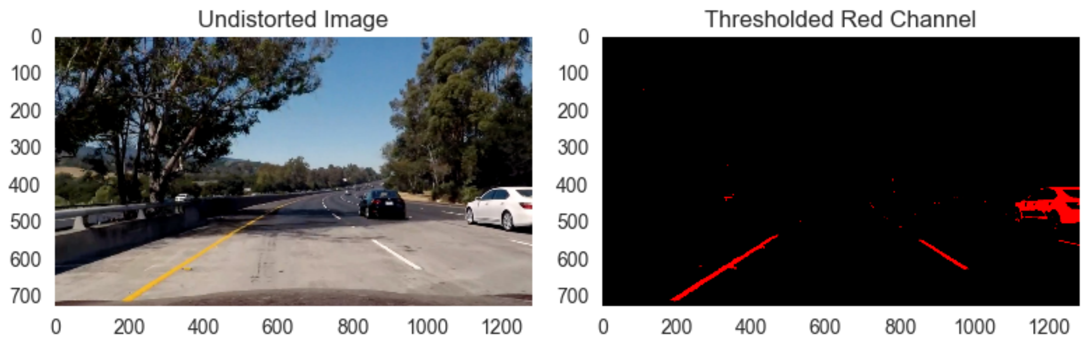


IMAGE THRESHOLDING

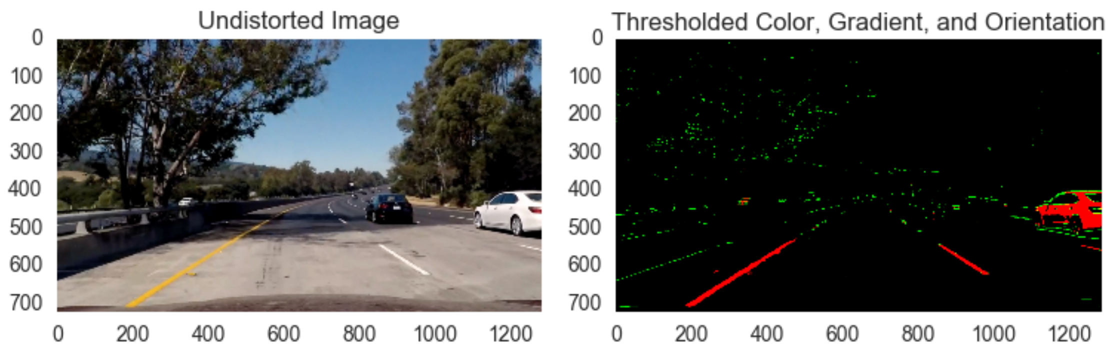
Once distortion is corrected, the image is transformed into a binary image where all pixels other than those belonging to lanes are zero. For example, the images below show a frame after correcting for distortion (left) and the resulting binary image (right). The binary image is obtained by thresholding the ***color*** and ***gradient magnitude*** and ***orientation*** within a ***region of interest***. Let's see how this works with an example.



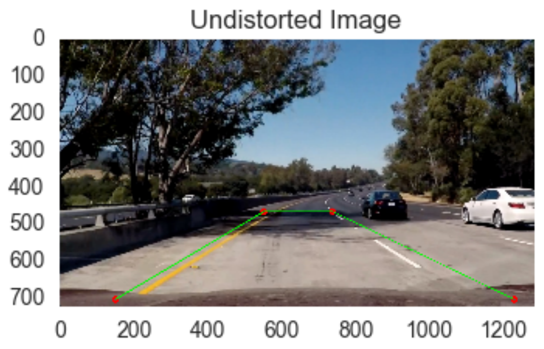
The red channel contributes to the white and yellow lane pixels. So, retaining pixels with high red values should highlight the lanes. As an example, below is the input frame (left) and the result of only retaining pixels with red values greater than 210 (right). As can be seen, a significant fraction of both lanes is identified.

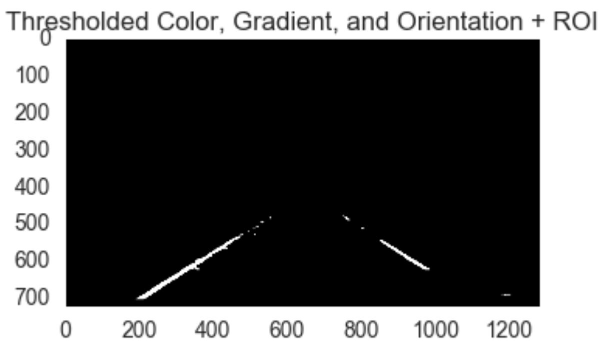


Thresholding the red channel fails to capture lane segments darkened by shadows. These segments can still be detected since they form distinct edges. Below one can see that retaining pixels with gradient magnitude greater than 85 and orientation greater than $\pi/4$ (green pixels) in addition to pixels with red greater than 210 (red pixels) allows us to recover a more of each lane (specifically parts previously obscured by the shadow now in green).



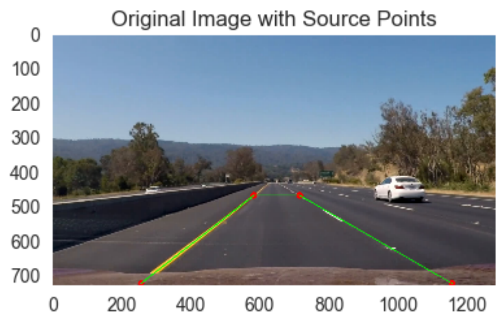
The final step is to retain only the pixels within a specified region of interest (ROI). The images below illustrate the input frame (left) with a defined ROI (green trapezoid) and the result of only keeping pixels that fall within the ROI and satisfy the previously discussed color and gradient thresholds (right). Note how all that remains are the pixels belonging to the lanes.

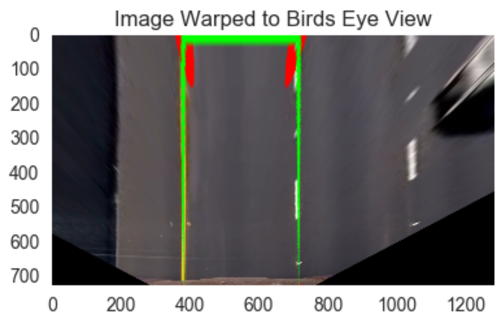




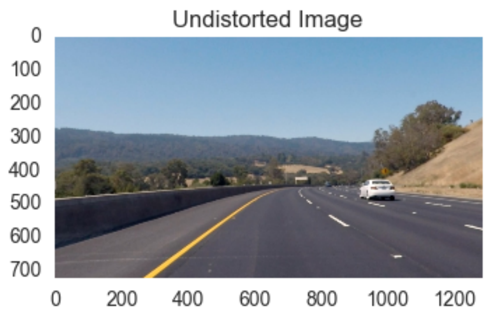
PERSPECTIVE TRANSFORMATION

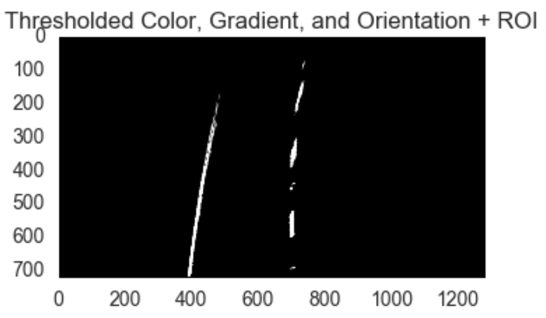
The next step is to transform the binary image into a bird's eye-view. To accomplish this, one must first derive a transformation matrix using four source and destination points. The source points are from the camera view (after correcting distortion) while the destination points specify where in the bird's eye-view to project the source points.





If we select four points defining a trapezoid in an image with parallel, we know that trapezoid will be transformed into a rectangle in a bird's eye-view. Consequently, the four vertices of the trapezoid are the source points and the four corners of the rectangle are the destination points. The images above show a trapezoid in a camera view with parallel lanes (left) and the result of transforming that frame into a bird's eye-view (right). The images below show a camera view with significant lane curvature (left) and the result of transforming the binary image with lane pixels to a bird's eye-view. Note the prominence of the lane curvature.



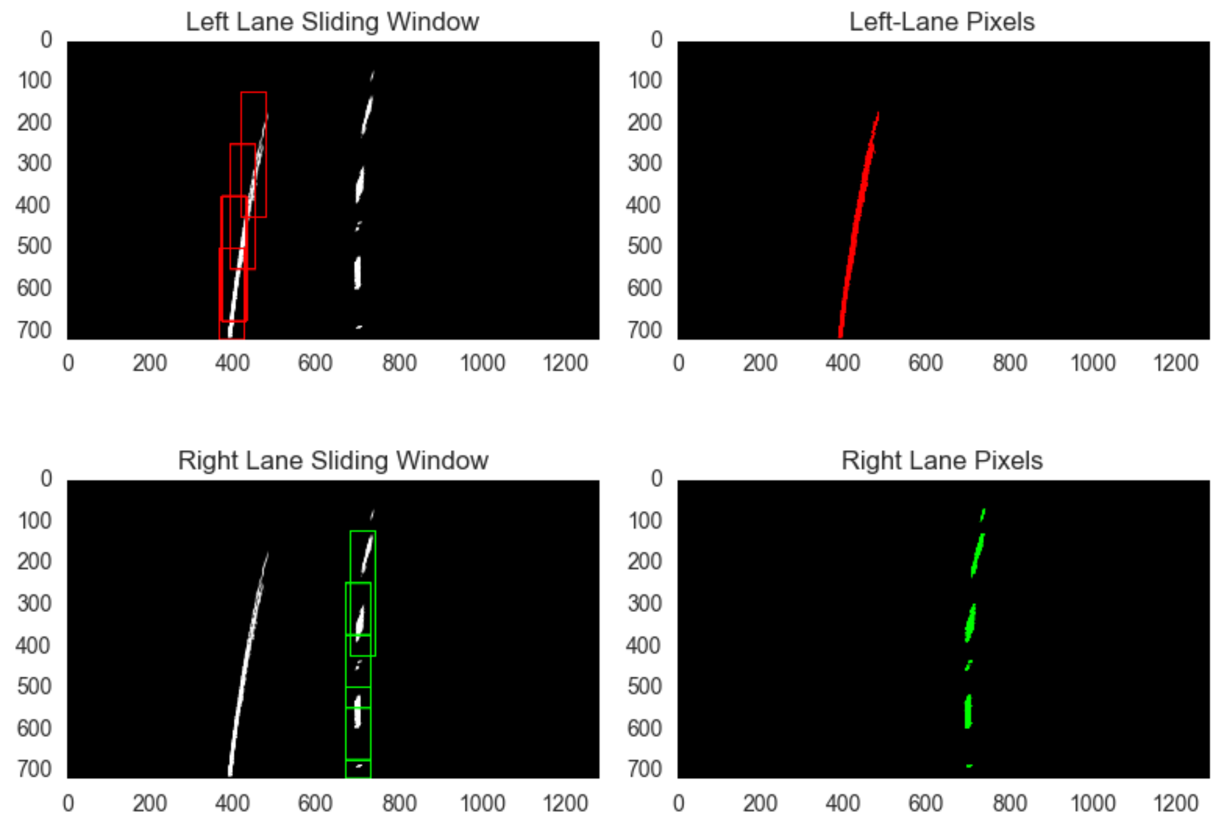


LANE SEGMENTATION AND CURVE FITTING

The next step is to determine which pixels in the warped binary image belong to the left and right lanes, and then to fit a second-degree polynomial to each set of points. Identifying which pixels belong to which lane, or lane segmentation, proceeds differently depending on whether the process utilizes information from preceding frames.

LANE SEGMENTATION <u>WITHOUT</u> INPUT FROM PREVIOUS FRAME

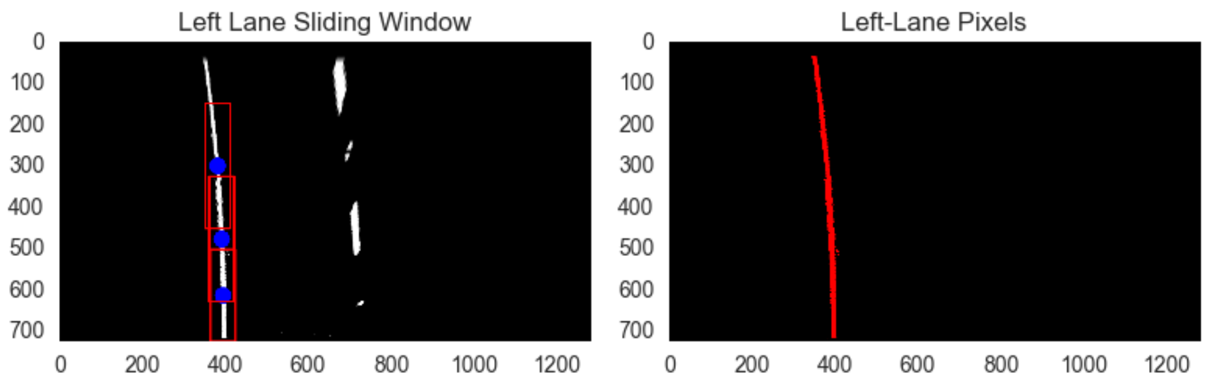
In the absence of information from a preceding frame, a search for the left and right lanes is conducted. The base of a lane (or region closest to the vehicle) is taken to be the column at which the histogram of column-wise pixel sums between rows [500:700] is maximum; for the left lane, the maximum is searched for between columns 0-600, and for the right lane columns 600-1280. Next, a sliding window (height=200, width=30), initially centered at the identified base of each lane, slides upward collecting the pixels associated with each lane.



The images above illustrate left and right lane segmentation. The top panel illustrates how successive sliding windows (in red) collect the pixels associated with the left lane while the bottom window shows how successive sliding windows (in green) collect the right lane pixels.

LANE SEGMENTATION <u>WITH</u> INPUT FROM PREVIOUS FRAME

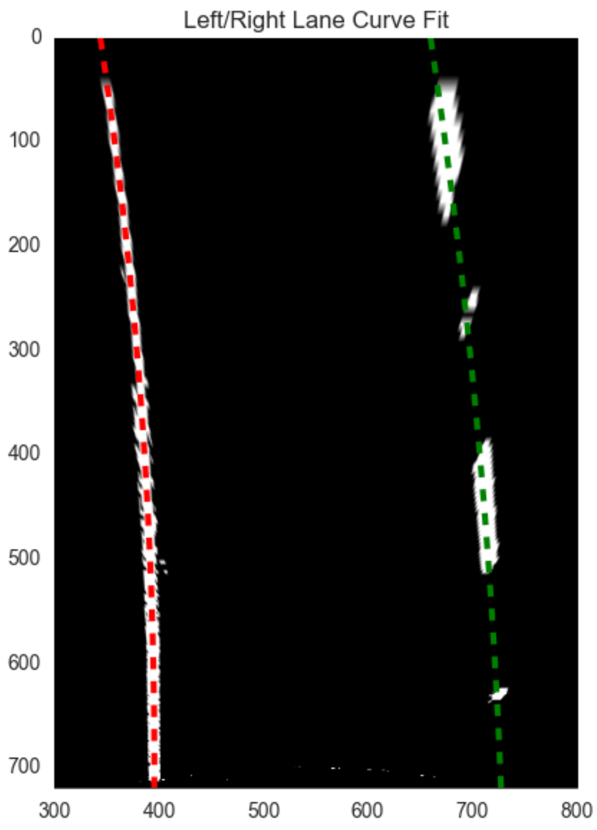
If a lane was detected and fit in earlier frames, we use that lane to establish where to search for lane pixels in the current frame. As an example, the image below shows a binary image corresponding to lanes in the current frame. The blue dots correspond to a few positions of the left lane in the frame immediately preceding. Around each blue dot we collect all the lane pixels falling within the red search windows (height=200, width=30) and ultimately recover all the pixels belonging to the left lane in the current frame. The same procedure is repeated to recover the right lane but using the position of the right lane in previous frames.



LANE CURVE FITTING

Each recovered lane pixel has an associated x (column) and y (row) coordinates. These x,y pairs can be used to fit a second-order polynomial curve (one for each lane) of the form $x = Ay^2 + By + C$.

As an example, the image to the right shows the curve fit using the isolated left lane pixels (red-dashed curve) as well as the curve fit using the isolated right lane pixels (green-dashed curve).



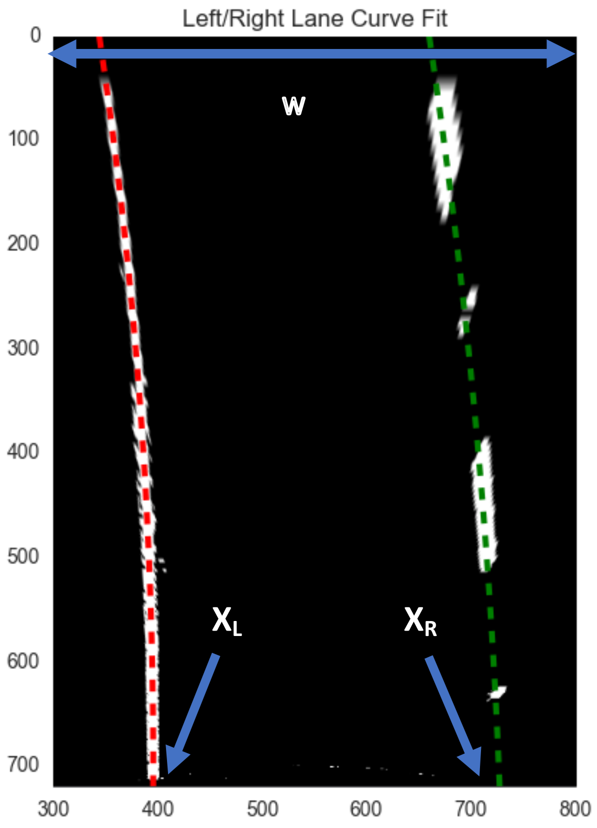
Note that the polynomial in the previous example is fit and displayed in the pixel or image space. To fit the polynomial in physical space, we simply scale the x,y pairs using constants for the number of pixels per meter in the x and y dimensions respectively.

With a parametrized curve for each lane, we can now estimate lane curvature as well as paint a continuous lane on each video frame as illustrated in the upcoming sections.

CURVATURE AND POSITION ESTIMATION

Once the polynomial coefficients A, B have been determined in physical space, the radius of curvature is estimated using the formulas provided in lecture. Estimating the vehicle's position relative to the lane center proceeds as described next.

Let x_L denote the x-coordinate of the left lane fitted curve at the point closest to the vehicle. Similarly, let x_R denote the x-coordinate of the right lane at the point closest to the vehicle. Finally, let W denote the width of the bird's eye-view of the lane as shown to the right.



If a vehicle is centered in the lane, one expects the difference between image center ($W/2$) and x_L to equal the difference between x_R and the image center:

$$\frac{W}{2} - x_L = x_R - \frac{W}{2}$$

If the vehicle is off-center, then the above equality becomes an inequality. For example, if the vehicle is closer to the left lane, then we expect

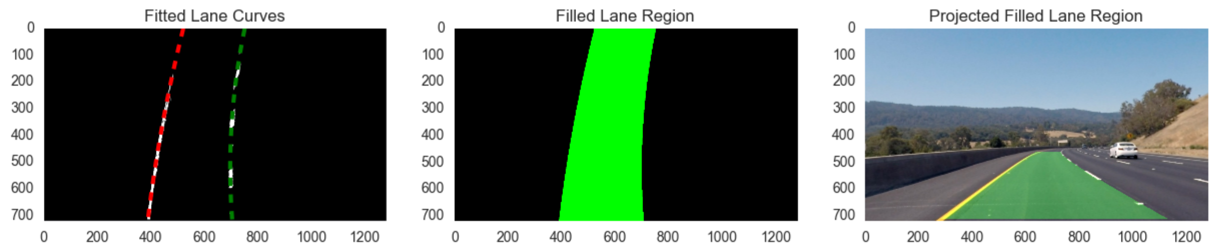
$$\frac{W}{2} - x_L < x_R - \frac{W}{2}$$

More generally, the amount by which the vehicle is off-center ΔC can be quantified shown below. Note that ΔC needs to multiplied by a scale factor to be in meters.

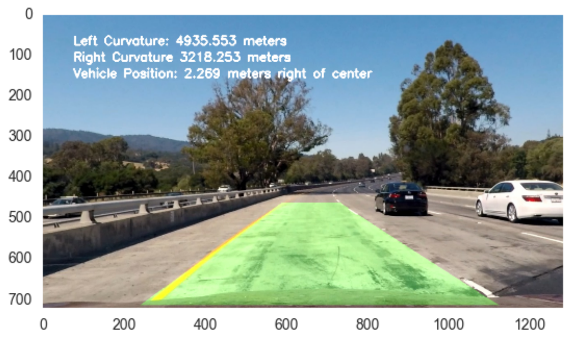
$$\Delta C = \left(\frac{W}{2} - x_L\right) - \left(x_R - \frac{W}{2}\right)$$

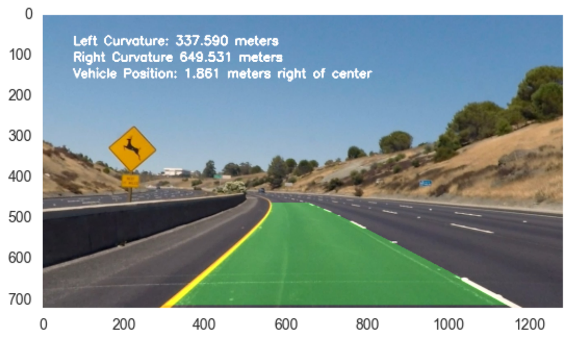
VISUAL DISPLAY

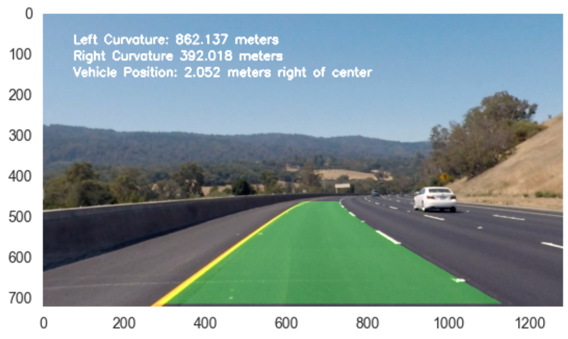
Displaying the detected lane region in the camera view involves the steps illustrated below. First, the fitted right and left lane curves (leftmost image) are used to define and fill the lane region in the bird's eye-view (center image). Next, that filled region is passed through the inverse of the perspective transformation that was used to generate the bird's eye-view. Finally, the projected lane region is overlaid onto the original camera view so that it may be viewed (rightmost image).

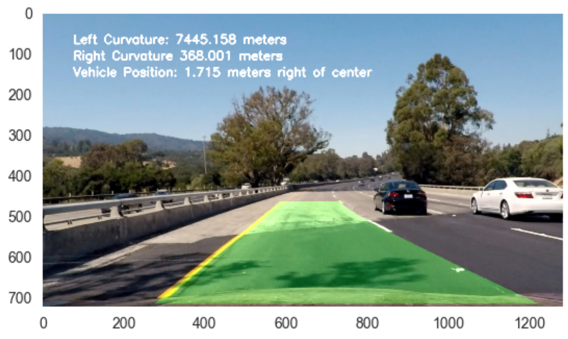


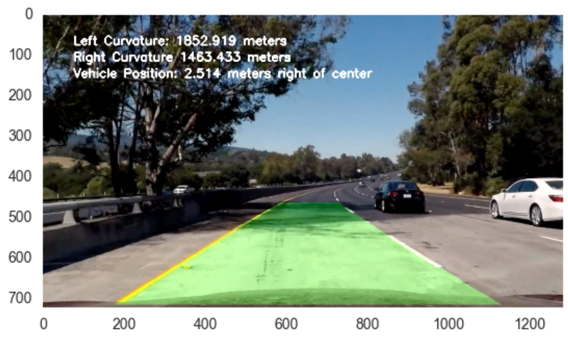
When this process is applied to all the test images provided in this project, the results are displayed on the following page. Although the test images contain a range of brightness and shadows, the lane region is detected and highlighted in each instance. Moreover, on each test image an overlay of the left and right lane curvature is shown along with an estimate of how far off-center and to the right the vehicle is positioned.

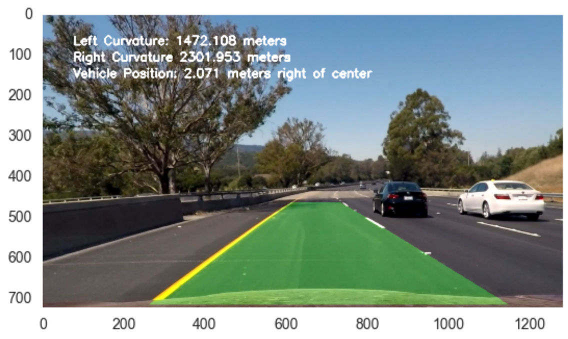

Left Curvature: 4935.553 meters
Right Curvature 3218.253 meters
Vehicle Position: 2.269 meters right of center


Left Curvature: 337.590 meters
Right Curvature 649.531 meters
Vehicle Position: 1.861 meters right of center


Left Curvature: 862.137 meters
Right Curvature 392.018 meters
Vehicle Position: 2.052 meters right of center


Left Curvature: 7445.158 meters
Right Curvature 368.001 meters
Vehicle Position: 1.715 meters right of center


Left Curvature: 1852.919 meters
Right Curvature 1463.433 meters
Vehicle Position: 2.514 meters right of center

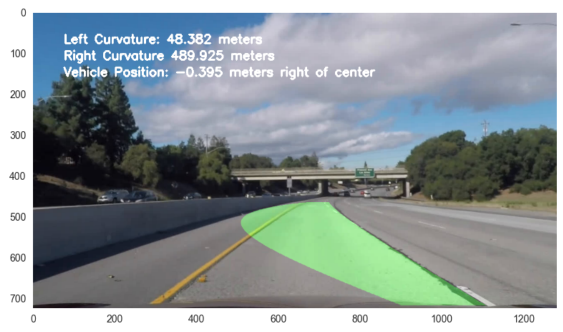

Left Curvature: 1472.108 meters
Right Curvature 2301.953 meters
Vehicle Position: 2.071 meters right of center

SMOOTHING AND FILTERING

When processing the project video, the fitted curves used to define the lane region (as illustrated in the previous section) are only computed independently of previous frames at the start of the video or when no lane has been detected for more than 10 frames. For most frames, the average of fitted curves (with the left and right sides separated) over the previous 10 frames is used to define the lane region that is projected onto the camera view.

It is critical to exclude outliers from from the averaging process. A frame is an outlier whenever the estimated lane width differs by more than 0.75 meters from the expected width of 3.7 meters. This condition is expressed below using the variabiles previously defined but where x_L and x_R have been scaled to be in meters.

$$|3.7 - (x_R - x_L)| < 0.75$$


Left Curvature: 48.382 meters
Right Curvature 489.925 meters
Vehicle Position: −0.395 meters right of center

The image to the right is a frame where lane detection failed resulting in the estimated width to differ significantly from the expected width of 3.7 meters. Consequently, this frame is excluded from the averaging process.