Advanced Lane Finding Project*

The goals / steps of this project are the following:

* Compute the camera calibration matrix and distortion coefficients given a set of chessboard images.

* Apply a distortion correction to raw images.

* Use color transforms, gradients, etc., to create a thresholded binary image.

* Apply a perspective transform to rectify binary image ("birds-eye view").

* Detect lane pixels and fit to find the lane boundary.

* Determine the curvature of the lane and vehicle position with respect to center.

* Warp the detected lane boundaries back onto the original image.

* Output visual display of the lane boundaries and numerical estimation of lane curvature and vehicle position.

I start by preparing "object points", which will be the (x, y, z) coordinates of the chessboard corners in the world. Here I am assuming the chessboard is fixed on the (x, y) plane at z=0, such that the object points are the same for each calibration image. Thus, `objp` is just a replicated array of coordinates, and `objpoints` will be appended with a copy of it every time I successfully detect all chessboard corners in a test image. `imgpoints` will be appended with the (x, y) pixel position of each of the corners in the image plane with each successful chessboard detection.

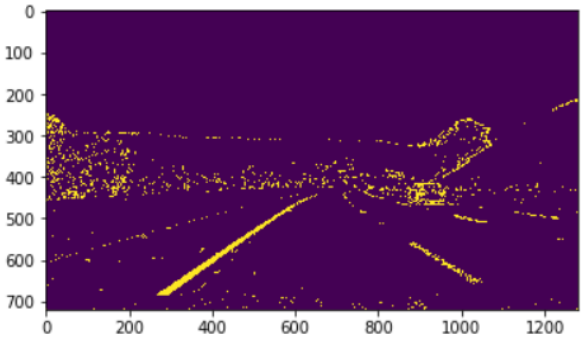
Pipeline (single images)

1. Provide an example of a distortion-corrected image.

To demonstrate this step, I will describe how I apply the distortion correction to one of the test images like this one:

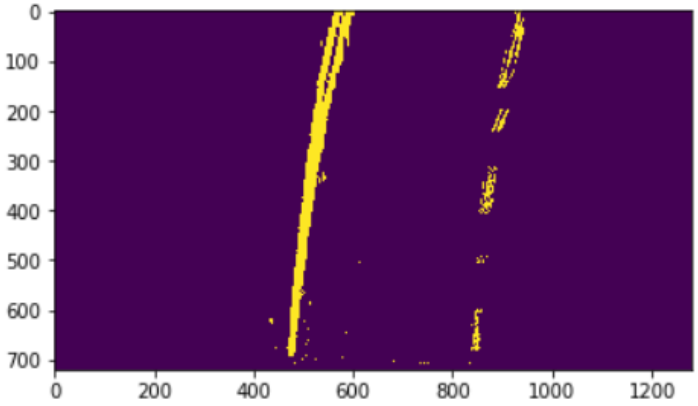
2. Describe how (and identify where in your code) you used color transforms, gradients or other methods to create a thresholded binary image. Provide an example of a binary image result.

I used a combination of color and gradient thresholds to generate a binary image. Here's an example of my output for this step.



3. Describe how (and identify where in your code) you performed a perspective transform and provide an example of a transformed image.

The code for my perspective transform includes a function called perspective_transform() which appears in the jupyter cell containing all the helper functions. This function takes in an input image and performs perspective warping according to manually selected source and destination points on the image. This function also calls the region_of_interest() function to take in only the area considered for work. The result obtained for the above image is as follows.



I verified that my perspective transform was working as expected by drawing the `src` and `dst` points onto a test image and its warped counterpart to verify that the lines appear parallel in the warped image.

4. Describe how (and identify where in your code) you identified lane-line pixels and fit their positions with a polynomial?

The draw_lane_curves() function takes in the warped image and performs the operation of identifying the left and right lane pixels, fitting an estimated polynomial function on the lanes and calculation of the radius of curvature.

5. Provide an example image of your result plotted back down onto the road such that the lane area is identified clearly.

The function plot_back() unwarps the image and draws a lane area diagram on the detected lane space. The radius of curvature values are also printed on the top left corner of the screen.

The final video is titled “project_output_video.mp4” which is the final result of the processed input video stream.

Discussions

1. Briefly discuss any problems / issues you faced in your implementation of this project. Where will your pipeline likely fail? What could you do to make it more robust?

For the project, the problems faced were in the challenge part of the section. During a sunny day, when shadows are present when the car goes under bridges and trees, the lanes get covered by the shadows, making it difficult for detection. The algorithm performs suboptimal in these conditions as lanes aren’t detected properly.

One way it could be improved is by extrapolation. That is, understanding the lane orientation before and after the shadow/obstacle and estimating the region in the shadow.