Project: Advanced Lane Finding

Akhil Waghmare

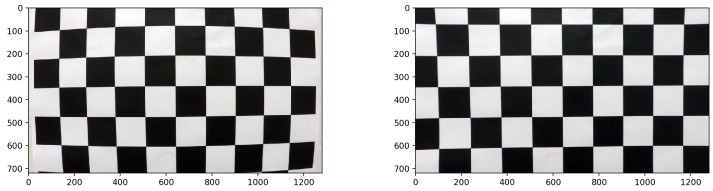
September 2017

Camera Calibration

1. *Breifly state how you computed the camera matrix and distortion coefficients.*

   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.

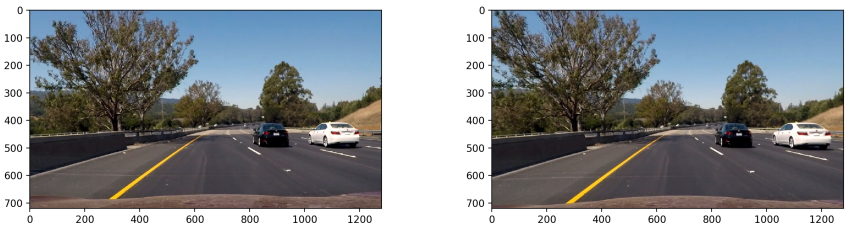
   I then used the output objpoints and imgpoints to compute the camera calibration and distortion coefficients using the cv2.calibrateCamera() function. I applied this distortion correction to the test image using the cv2.undistort() function. Here is an example of an original (left) and undistored (right) pair of images:



Pipeline (single images)

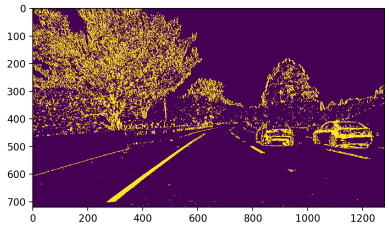
1. *Provide an example of a distortion-corected image.*

   Here is an example of an original (left) and undistorted (right) pair of images:

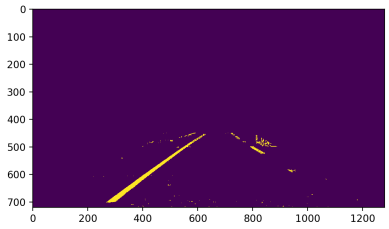


2. *Describe how you used color transofrms, gradients, or other methods to create a threhsolded binary image.*

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

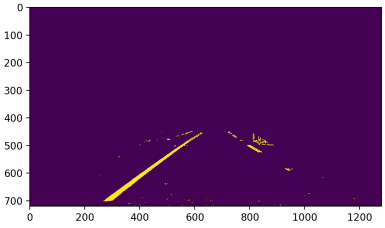


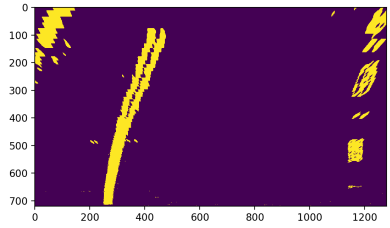
   To focus in on the lane, I applied a quadrilateral mask to the image, resulting in something like:



3. *Describe how you performed a perspective transform.*

   The code for my perspective transform includes a function called transform(), which appears in the file helpers.py. The transform() function takes as inputs an image (img) and the transformation matrix (M). This matrix is generated using source and destination points in the perspective_transform_matrices() function. I chose the hardcode the source and destination points. Here is an example of the perspective transform effect:



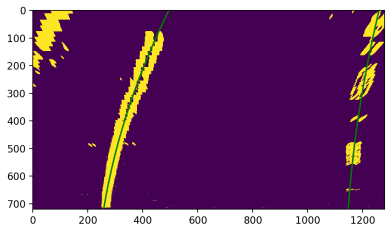


4. *Describe how you identified lane-line pixels and fit their positions with a polynomial.*

   I first examined a histogram on the bottom half of the image to determine the starting points for each lane. I then applied the sliding-window approach to trace the line going upwards. After identifying

the pixels for each lane, I used the polyfit() function to determine the quadratic curve which best fit the pixels.

Here is an example:

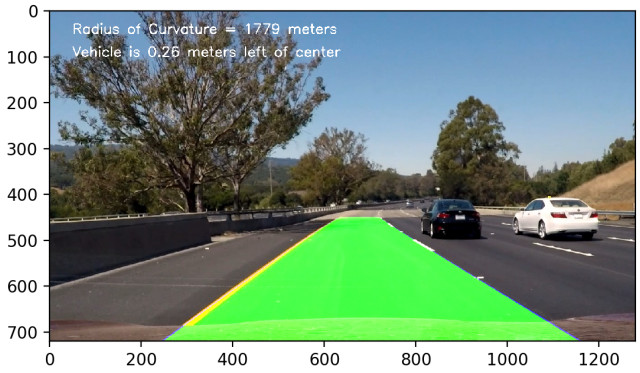


5. *Describe how you calculated the radius of curvature and the position of the vehicle with respect to center*

   After the polynomial fits for each lane were found, the radius of curvature was found using the formula provided. This left me with a measurement for each lane. In the video, I decided to report the average of the two curvatures.

   To determine the vehicle position with respect to center, I first calculated the center of the lane by finding the midpoint of the lane polynomials at $y = 720$ (the bottom of the image). I then compared this to the true center of the image (assumed to be the vehicle's center). The sign of this difference determines whether the vehicle is to the left or right of lane center.

6. *Example of pipeline final result.*



Pipeline (video)

Here's the link: `https://youtu.be/u-8R7YZh-YA`

Discussion

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

    - **Efficiency:** The pipeline recalculates the lanes at each frame from scratch. To make the process more efficient, it would be better to store the previous frame's calculation and search just in a small area around this region.
    - **Curvature accuracy:** At certain parts in the video, the radius of curvature measurement seems to grossly be miscalculated (on the order of 10,000 meters) for just a split second. There is generally also a lack of stability in the radius of curvature measurements, so this could definitely be improved.