ENPM 673: Perception for Autonomous Robots
Project-2

Shreya Gummadi
Ghanem Jamal Eddine
Sneha Ganesh Nayak

13 March 2019

Contents

1 Lane Detection and Turn Prediction

The following project detects the lane the car is driving on. In order to do so, we first used the camera calibration and distortion matrix to undistort the frames generated from the video. We then proceeded to perform homography, edge detection and getting the lane pixels in order to detect the lane detection.

1.1 Homography

Homography links two images on the same planar surface in space. It has many applications, one of which is perspective transformation. We use this to view the road from a bird eye view perspective. We took a frame with straight lanes and found two points on each lane and mapped them to desired points to get a top view. We mapped the points at the bottom of the lane to have y co-ordinate 720 and the ones on the top to have y co-ordinate 0. As stated in the pipeline for the project we assumed that this homography would work for all frames.

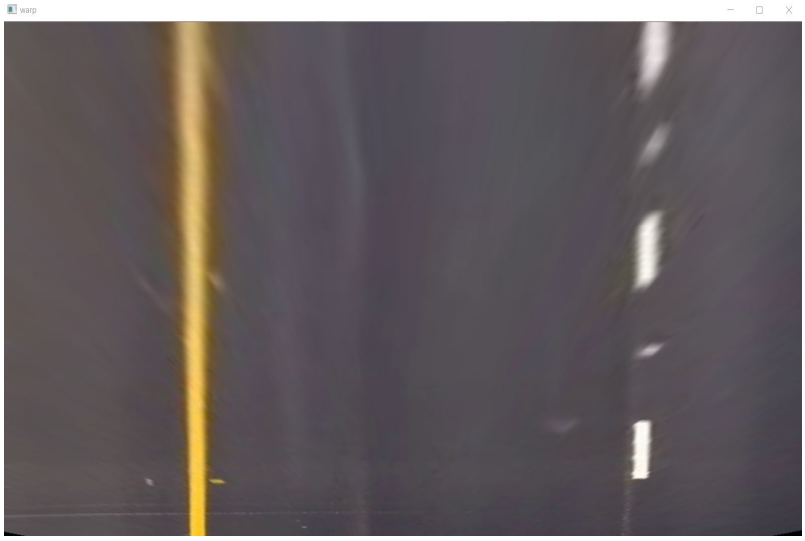

Figure 1: Top view projection.

1.2 Edge Detection

Once we have the top perspective we use it to detect the lanes. We first use filtering to reduce the noise and then pick out pixels that are white and yellow which correspond to the lane. We then extract that part of the image and run Canny edge detection on it.

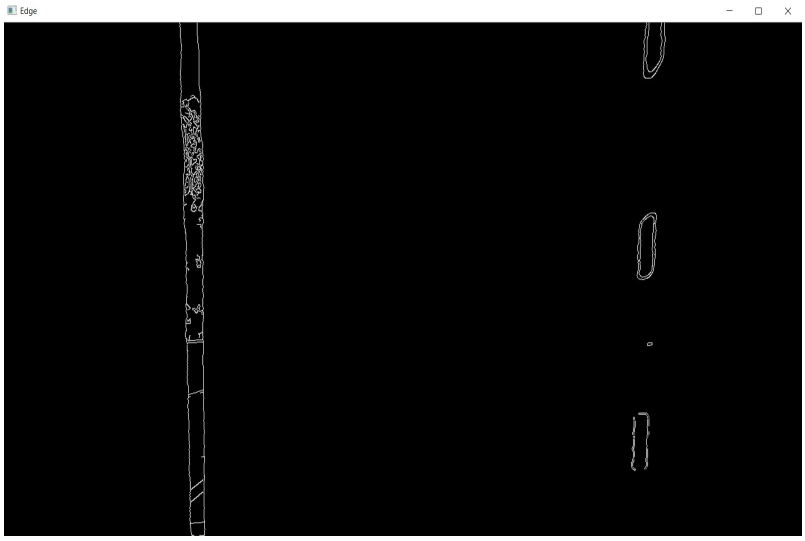

Figure 2: Lane detected by Canny edge.

1.3 Hough Lines

Hough Line is a transform used to detect straight lines. Any line can be expressed in the image space in the polar co-ordinate by (d,θ). Here d is the perpendicular distance from the line to the origin and θ is the angle this perpendicular makes with the x axis. The line can then be expressed as $d = xcos\theta + ysin\theta$.We can then get this for each edge point based on image gradient. We can generalize it to detect curves if they can be parametrized. But it doesn't work that well. So instead for our project we made use the pipeline for histogram of lane pixels.

1.4 Histogram of Lane Pixels

As stated in the pipeline, the region of interest is the bottom half of the image. We take the bottom half and sum over the columns and get the maximum argument's position which corresponds to the lanes. The peak in the first half gives us the position of the left lane while the peak in the second half gives us the right lane.
We then built windows on starting from these base points and going up. In the window we got locations for pixels that were non-zero. This gave us the lane pixels.
We then used these lane pixels to find the polynomial coefficients and fit a polynomial to them. This completed the lane detection part of our project.

1.5 Turn Prediction

In order to predict the turn, we found the centre line points and fit a polynomial to it. We then found the gradient of the centre line. If the gradient is negative it corresponds to left curve, a positive gradient corresponds to right curve and an approximately zero gradient corresponds to a straight road.
We also use the polynomials fit to the lanes to get the left and right radius of curvature and then average it to get the radius of curvature of the lane.

2 Results

We were able to project the lanes back on to the original undistorted image and detect lanes. For the project video, we predicted the left,right curves and the straight road.As for the challenge video we are able to predict the right curve.

Figure 3: Lane Detection and Turn Prediction-1.

Figure 4: Lane Detection and Turn Prediction-2.

Figure 5: Lane Detection and Turn Prediction-3.

Figure 6: Lane Detection and Turn Prediction-4.

While working on the project, we played around with the window dimensions and the number of windows in order to get better results. The turn prediction fails for a few frames while predicting right curve. This could be due to the fact that less number of points are detected on the right lane than on the left lane. We tried to make it better and played around with the range for the prediction. Right turn prediction still fails for a few frames.
We applied the same pipeline to the challenge video. It did not work as well as it did with the project video. The polynomial fit to the lane sometimes goes out of the lane which could be corrected to get better detection.

Bibliography

1. https://www.rapidtables.com/web/color/RGBColor.html
2. https://opencv-python-tutroals.readthedocs.io/en/latest/pytutorials/pyimgproc/py colorspaces/pycolorspaces.html
3. https://docs.opencv.org/3.0-beta/doc/pytutorials/pycalib3d/pycalibration/pycalibration.html