ENPM673 – Perception for Autonomous Robots – Spring 2022

Project 2

Gokul Hari, Sakshi Kakde, Jayesh Jayashankar, Samer Charifa

Due date: 4 April 2022, 11:59PM

Submission guidelines:

- This homework is to be done and submitted individually.
- Your submission on ELMS/Canvas must be a zip file, following the naming convention **YourDirectoryID_hw1.zip**. If your email ID is abc@umd.edu or abc@terpmail.umd.edu, then your Directory ID is **abc**. Remember, this is your directory ID and NOT your UID.
- Please submit only the python script(s) you used to compute the results, the PDF report you generate for the project and a detailed README.md file.
- For each section of the project, explain briefly what you did, and describe any interesting problems you encountered and/or solutions you implemented.
- Include sample outputs in your report.
- The video outputs are to be submitted as a separate link in the report itself. The link can be a YouTube video or a google drive link (or any other cloud storage). Make sure that you provide proper sharing permission to access the files. **If we are not able to access the output files you will be awarded 0 for that part of the project.** On elms you can only submit one .zip file which can be of maximum 100 MB.

Problem 1: Histogram equalization [30]

Dataset for Problem 1

Here, we aim to improve the quality of the image sequence provided above. This is a video recording of a highway during night. Most of the Computer Vision pipelines for lane detection or other self-driving tasks require good lighting conditions and color information for detecting good features. A lot of pre-processing is required in such scenarios where lighting conditions are poor.

Now, using the techniques taught in class your aim is to enhance the contrast and improve the visual appearance of the video sequence. You **cannot** use any in-built functions for the same. You have to use **histogram equalization** based methods, both **histogram equalization** and **adaptive histogram equalization** and compare the results.

Note:

1. We understand that adaptive histogram equalization will be slow, that is why only 25 frames are provided in the dataset.

Problem 2: Straight Lane Detection [35]

Dataset for Problem 2

In this problem we aim to do simple Lane Detection to mimic Lane Departure Warning systems used in Self Driving Cars. You are provided with a video sequence, taken from a car. Your task is to design an algorithm to detect lanes on the road, and classify them as dashed and solid lines. For classification of the line type, you have to use different colors. Use **green for solid** and **red for dashed.**

Note:

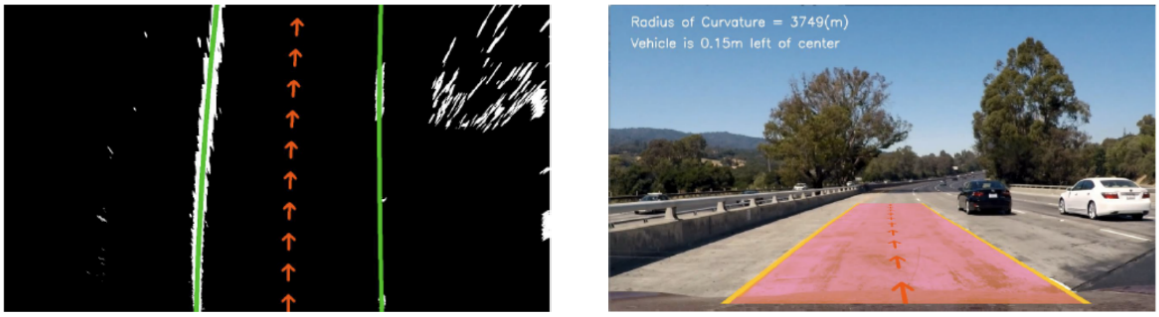
1. For this problem, you can assume lanes to be straight.
2. Keep in mind that lane detection is a very popular problem and there are many implementations available on the internet. If any part of your implementation is not designed by you, you will lose a considerable amount of points.
3. Do not classify the left lane as dashed and right lane solid. Your code must work if the video is horizontally flipped.

Problem 3: Predict Turn [35]

Dataset for Problem 3 :

In this problem, we aim to detect the curved lanes and predict the turn depending on the curvature: either left or right turn. The dataset provided has a yellow line and a while line. Your task is to design an algorithm to detect these lanes on the road, and predict the turn. Please note that for the output of the lane detection, you have to superimpose the detected lane as shown in fig.4. Also compute the radius is curvature for the lane using the obtained polynomial equation. Refer to this.

When your lane detection system loses track(cannot find lane lines), you must manage to use the past history of the lanes from the previous image frames to extrapolate the lines for the current frame.



(a) Polynomial fitting (b) Backprojection onto the original image

Figure 4: Polynomial fitting and an example of the final output

General Suggestions:

1. (**ONLY for problem 2**) : You are allowed to use inbuilt functions for hough line, histogram equalization , homography computations, warp perspective, and all numpy function including polyfit
2. We expect an output like this.

Report:

For each section of the project, explain briefly what you did, and describe any interesting problems you encountered and/or solutions you implemented. Include the following details in your writeup:

1. Your understanding of homography and how it is used (if you used it).
2. Your understanding of how HoughLines work (whether you used them or not!)
3. How likely you think your pipeline will generalize to other similar videos.

Please include all the results and images in the report, along with a link to your videos. The report must be a typed PDF only.

Acknowledgement:

We are using the Advanced Lane Detection dataset from Udacity - Self Driving Nanodegree program and KITTI dataset.