Project 2

Perception for Autonomous Robots

April 5, 2022

Instructors:
Samer Charifa

Student:
Prateek Verma

Semester:
Spring 2022

Course code:
ENPM673

Contents

1 Problem 1

The aim is to improve the quality of the image sequence provided. Only 25 frames are provided in the dataset which needs to be considered as a video sequence to enhance the contrast and improve the visual appearance using Histogram Equalization and Adaptive Histogram Equalization. The algorithm is designed by using the histogram equalization function developed from scratch.

- Y-channel of OpenCV YCRCB filter is used for computation.
- For adaptive process, 8x8 tiles are used which yields better results.

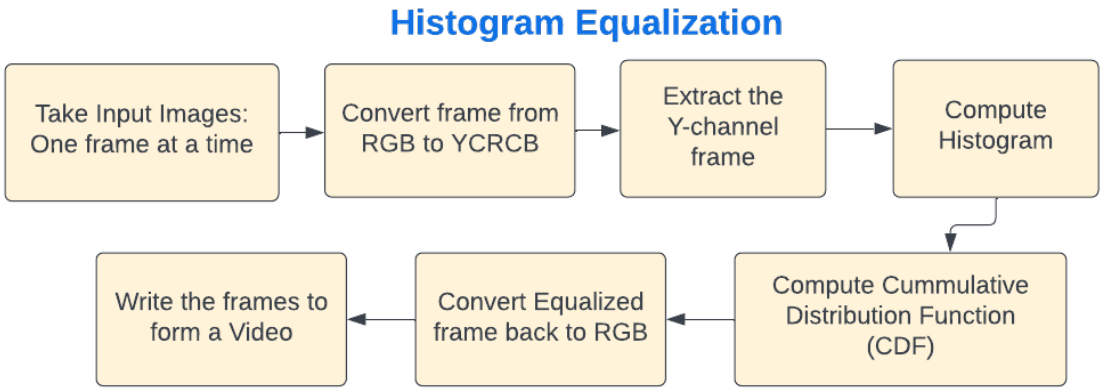

Figure 1: Pipeline for Histogram Equalization

Figure 2: Output for Histogram Equalization

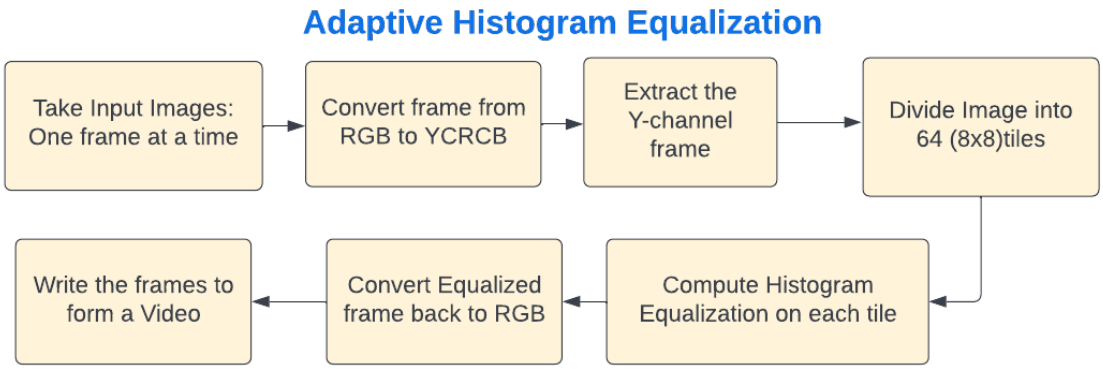

Figure 3: Pipeline for Adaptive Histogram Equalization

Figure 4: Output for Adaptive Histogram Equalization

The program file is: **histogramCode.py**

2 Problem 2

Aim to do simple Lane Detection to mimic Lane Departure Warning systems used in Self Driving Cars. The algorithm is designed using HoughLine transformation, which is applied after pre-processing the frame and applying the triangular mask to isolate the region of interest.

- the solid and dashed lines are highlighted using green and red line respectively.
- the lanes are assumed to be straight.

The pipeline/block-diagram for this process is presented below:-

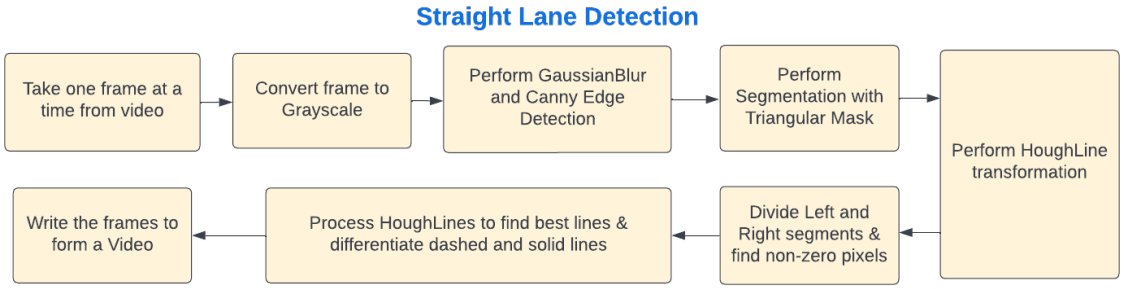

Figure 5: Lane Detection

Figure 6: Output for Lane Detect

The program file is: **detectLane.py**

3 Problem 3

Aim to detect the curved lanes and predict the turn depending on the curvature: either left or right turn. The algorithm is designed by defining a regions of interest to isolate the lane and then perform homography and warping methods followed by histogram to find required points. Next functions like polyfit is used to get the curvature.

- Based on change in slope of the lanes turn left or right is indicated.
- the lanes are assumed to be straight.

The pipeline/block-diagram for this process is presented below:-

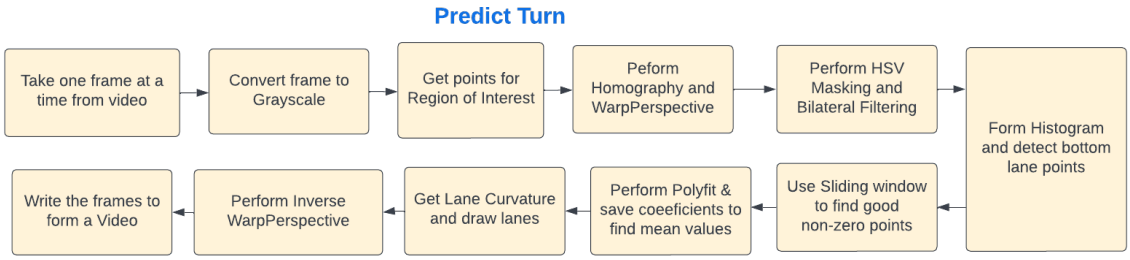

Figure 7: Predict Turn

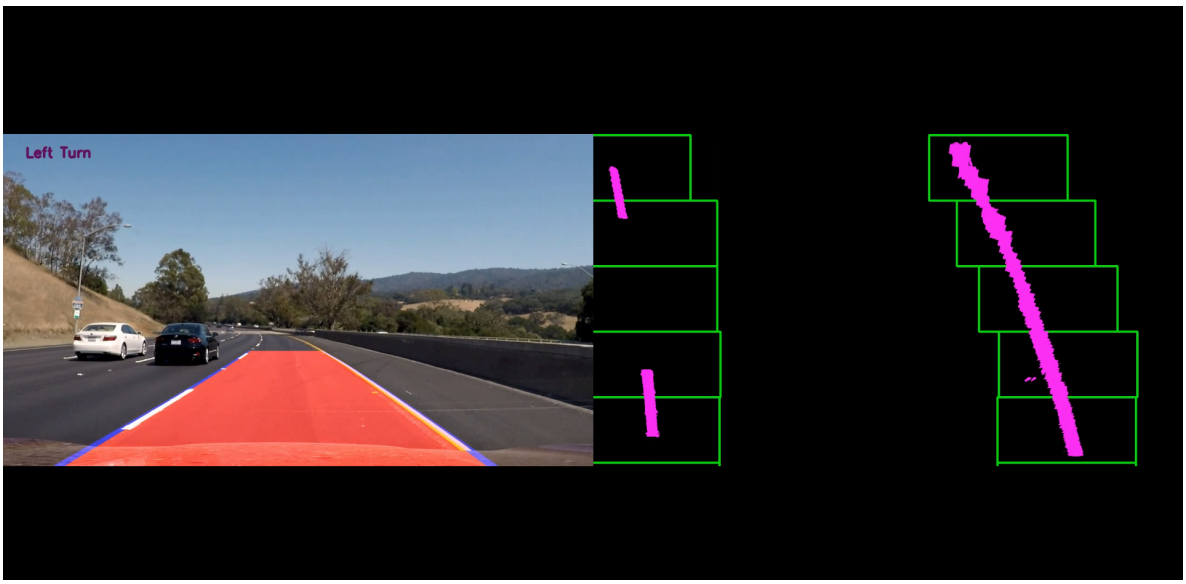

Figure 8: Output for Turn Predict

The program file is: **turnPredict.py**

4 Extras

1. **Homography**
    - transformation that is occurring between two planes.
    - In image processing, it is a mapping between two planar projections of an image.
    - represented by a 3x3 transformation matrix in a homogeneous coordinate space.
2. **HoughLine Transform**
    - convert lines in Polar co-ordinates from Cartesian co-ordinates.
    - helps in calculating points on the line and point of intersection between two lines.
3. **Pipeline**
    - can be generalized for similar inputs.
    - turn prediction can be performed on flipped video input.

Google Drive Link for Output:
ENPM673-Project2-verma