Program 3: Basic Hough Transform for Line Detection
Due Date: Thu 17th October – 11:59 p.m.
Submit electronically at J:\Cs436\Program3

Implement basic line detection algorithm using Hough Transform, as discussed in class. The basic task of this programming assignment is limited to simple implementation of HT using the equation: \( \rho = x \cos \theta + y \sin \theta \). The basic task does not require you to use gradient direction for efficient implementation. Hence, binary edge map can be used as input. Following are the steps that the program should perform:

- Take binary edge map as input. Several example files are provided.
- Decide on quantization level of the accumulator array, and fill it up using the basic HT algorithm for line detection.
- Scale the accumulator array between 0-255 and output it as an image for visualization.
- Use simple thresholding to determine cells in the accumulator array which got more than a certain number of votes.
- Super-impose corresponding lines from all such cells in the accumulator array on the original image for visualization. This involves writing code to plot a line in an image.

Consider as an example, the following input image and associated edge map, obtained through Canny’s edge detector.

Selecting all points in the accumulator array above a certain threshold and plotting them as lines in the original image generates the following output:
Grading scheme for this program is as follows:

The program will be graded on the correctness of results and the quality of report of the basic task. This will include display of intermediate output (accumulator array) and the final output (super-imposed lines). The program must also be run on additional files provided in the website and comments given on the results. An associated short report should explain interesting implementation details, along with comments on performance. The choice of the thresholding parameter should also be reported.

An additional 30% of the maximum bonus grade is available for enhancements to the basic task. Some of the possible enhancements include the use of gradient direction for reducing the time complexity of implementation and comparing the results, using the method described in text (or otherwise) to find actual line segments, rather than infinite lines and coding a better methods to find peaks. Comments and implementation of these tasks will be evaluated for bonus grade.

Note that it is essential to comment on your output/efforts to achieve any grade. Code alone will not be graded, if output is not submitted and commented upon.