CS6640 – Project 4 Hough Transform
Assigned Wed Nov. 12, 2014
Due Wed Nov. 26 (Just before midnight)
Instructor: Guido Gerig
TAs: Avantika Vardhan, Padmashree Shrinivasa Shetty Teeka,
TA office hours Mo 3-5pm,Wed 11am - 1pm, office WEB 2626

Required Readings: Book DiP Ch10.27, pages 733-738
Course notes and slides GG

Problem 1: Hough Transform for straight lines without edge orientation

Write a program that calculates the Hough Transform for straight lines using the parametric form $\rho = x\cos\theta + y\sin\theta$. With (x,y) being the image coordinates, the origin for the HT is put into the left corner of the image.

Each point in image space generates a cos-curve which is accumulated in parameter space. The increment can be chosen as 1 for each point, but could also be proportional to the edge-strength which is the brightness attribute for each contour point (see contour images described below), so that the accumulation reflects the *importance of edge points*.

You may use the test images provided with this project, the simulated scene containing edges and lines "edges-lines" and the airport scene, but feel free to choose images of your choice that contain straight edges to be detected.

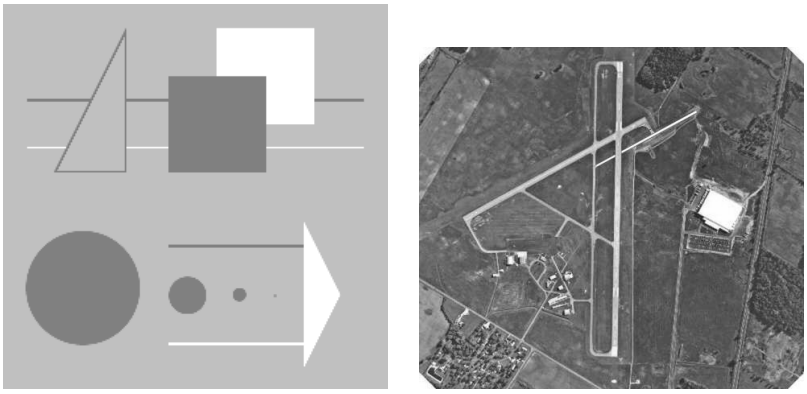

Figure 1: Example of images presenting straight line features.

a) Images need to be preprocessed with edge detection and thresholding to get a small set of edge candidate pixels to be processed. Use your Canny edge-detection as implemented in Project 2.

b) Implement the Hough-Transform and create an accumulator that corresponds to a 2-D image array with axes θ and ρ. Choose the cell size $\Delta\rho$ and $\Delta\theta$ as a parameter in your program for doing your own tests. Defaults could be 1 deg and 1 pixel steps, but you may also choose a coarser accumulator grid of speed is an issue and if you can sacrifice precision.

c) After calculating the HT for your image points and accumulation of curves, detect the N largest maxima and write the pairs $(\rho_j, \theta_j, nvotes_j)$ to a file for checking (nvotes are the accumulated votes per maxima).

d) Calculate the corresponding N straight lines and draw these lines back into the image space for comparison with the original image. Nice would be to overlay the lines in color onto the original gray-level image. Examples of overlays in Matlab can be found in the Matlab instruction files from the first week.

e Hint on peak detection: There may be several peaks nearby due to discretization of the accumulator. You may implement one of the following options:

- Find maximum peaks as local maxima in the accumulator (i.e. cells that have larger votes than all neighboring cells).
- Gaussian smoothing of the accumulator with a small kernel to reduce noise. Then apply the procedure above for finding peaks. Select the Gaussian width parameter based on the notion that the "pixels" are (ρ_j, θ_j) parameter cells.

- Display the original image, the accumulator space (please make sure that you enhance the image so that it is not all black, just use some image editing program), and the resulting lines. You can overlay the selected peaks to the accumulator, and the corresponding lines to the original image.
- Apply the HT to two images you select from the test images or other image libraries.

BONUS question for specialists

Please note that peak detection in the accumulator is the biggest challenge of the Hough transform. The better we select single peaks and suppress secondary nearby peaks the better the result. This is the reason why the decrementation strategy described in the handout notes and ICCV87 paper is recommended, eventually with a preceding Gaussian filtering of the accumulator.
Apply the "decrementation" strategy proposed by Gerig et al. in the handout to simplify the accumulator before finding the major peaks. Please note that decrementation is practically the same transformation but instead of incrementation you decrement all cells except the single maximum per parameter curve.

Problem 2: Hough Transform for straight lines using edge orientation

We discussed in the course that knowing the local edge orientation can significantly reduce computation time. Instead of accumulation of a whole curve for each point, you only need to

accumulate one cell or a cell and a few neigbhors since location and orientation may not be precisely known.

- Choose the images which you also processed above, again apply your Canny edge detection algorithm but this time also use the local gradient orientation angle. This results in a gradient magnitude map and an orientation image (see project 2).
- Threshold the edge-magnitude image as before.
- Apply your Hough Transform as before, but this time limit accumulation to a the angle as given the the edge orientation image, or to a small angle range θ of a few degrees from the edge orientation, e.g. $\pm 5 deg$. You therefore only accumulate a small piece of the whole curve centered at the known edge orientation.
- Do peak detection as before, and plot the resulting lines in the image and/or as an overlay on your original image.
- Compare the method with full accumulation of all angles and this reduced accumulation in regard to the lines found by both versions, discuss the time savings you observe.

Instructions, Requirements, and Restrictions

1. Please use your name **"NAME"** in all reports and submitted materials to uniquely identify your projects.
2. Your report should summarize your approach to solve the problems, show graphs, images of the results and include a critical evaluation/discussion of the results. Please do not copy the project description into your report, just the title is sufficient, but provide a short description what you did for each experiment and how you did it. The report can be written with any text program of choice, and the handin needs to be in pdf format.
3. You should have in your report a short description of each algorithm you used and documentation on how your code is organized. A short summary on most essential core information is sufficient, don't expand too much.
4. We **do not allow to use Matlab toolbox functions** (e.g. Imaging Toolbox) or other existing image processing libraries or solutions found elsewhere. We want you to experience challenges of implementation and give all students the same conditions for code development [1].
5. Your project report will be in the form of a pdf file.
6. Write your project code in a single directory, called `project1-NAME` and the best is to create a compressed tar file of code and images used to run it. Submit a separate pdf report and the compressed code/images file to canvas.

[1] The core MATLAB package comes with several rudimentary functions that can be used to load, save, and perform custom functions on images.

7. **Please remember to look-up the honor code and requirement to provide your own solution as discussed in the syllabus.**

8. **Please look up the late policy as defined in the syllabus**