HW5 (line Hough) CSx73 22sp CV+CNN due: 03.22.22

This homework is about Hough transforms for finding lines.
It may be implemented in either Matlab or Python. Matlab is recommended and the simplest.

Matlab and Python have functions for Hough transform: obviously these functions are not allowed in this homework. You may use them to test your functions, but please remove them from your submission. You may use the built-in Matlab functions for edge detection and grayscale conversion (no need to implement those yourself).

The Matlab template `hw5_22sp73.m` guides your code development.
It also shows you what input images to use and output images to generate for each question.

The homework has 3 parts, each computing lines from an image using Hough transform. The images become progressively more difficult to find the lines, so your code will require some tuning as you proceed. Please read the entire homework before starting, and browse the code. Part 1 works with a clean 2x2 square image; Part 2 with a noisy 2x2 square image; and Part 3 with real images. Parts 2 and 3 are for 673/773 only. Each part computes edges, builds an accumulator array, computes peaks in this accumulator array, and draws the associated lines. Parts 2-3 add Gaussian smoothing as a preprocessor. Part 3 adds grayscale conversion as a preprocessor.

Part 1: Hough line transform on a noise-free toy image (2x2 square)

edge detection

Compute an edge image (binary image, white iff edge pixel) using your favourite edge detection algorithm.
Use Matlab fn: *edge*

Hough accumulator array

Populate an accumulator array.
The Matlab documentation for hough, which you are simulating, should help you interpret the parameters.
Your function: *hough_lines_acc*.
Simulate (but do not use) Matlab fn: *hough*.

hints

- use the polar representation of a line in an image coordinate system (where x moves right and y moves down) as follows: $(\rho, \theta) = (x\ cos(\theta) + y\ \sin(\theta),\ \theta)$
- but recall that the image is stored in a matrix coordinate system (y, x)
  (first coordinate moves down, second coordinate moves right)

Hough peaks

Write a function `hough_peaks` that finds the peaks of the accumulator array: indices associated with local maxima. You will call this function to find the 10 strongest peaks (10 strongest lines).
Then draw these 10 peaks using your function `draw_highlighted_peaks`.
Your functions: *hough_peaks*, *draw_highlighted_peaks*.
Simulate (but do not use) Matlab fn: *houghpeaks*.

draw Hough lines

Extract the lines from the peaks (translate back from Hough space), and draw the associated lines in green overlaid on top of the image. Infinite lines are fine (no need to clip to finite line segments as in the associated Matlab function houghlines).
Your function: *hough_lines_draw*.
Simulate (but do not use) Matlab fn: *houghlines*

Hint: explore the Matlab function `fimplicit`.
To hold your image while you draw the lines, use the hold option to imshow.
I showed you how to output an image of the present display to a file in the previous question.

Parts 2-3 on the next page.

Part 2: Hough line on noisy toy image (673/773 only)

This question repeats the Hough pipeline on a more challenging toy image. Smooth the noisy image to prepare for edge detection. This was unnecessary in Part 1, since the image was noise-free. Then perform the sequence from Part 1, outputting your intermediate images: edge detection, peaks of accumulator array, lines.
Discussion (in hw5_22sp73_discussion.md): Discuss the refinements you made from Part 1. For example, you may to need change your edge detection algorithm, and tune it differently; adjust your bin sizes, threshold, and neighbourhood size, and so on. I anticipate a paragraph or two of discussion, focussing on concise and precise details on the changes you needed to make.

Part 3: Hough line on a real image (673/773 only)

This question repeats the Hough pipeline on a more challenging real image. Translate to grayscale (this image is now colour) and Gaussian smooth. Then perform the sequence from Part 1, outputting your intermediate images: edge detection, peaks of accumulator array, lines.
Discussion: Discuss the refinements you made from Part 2.
Additional test data: draw the lines you find in the test images in the img directory. You may shrink the large images if you want.

Part 4: reading the literature (773 only)

1-2 page report on Tuytelaars 1998 ICCV (cascaded Hough)

Deliverables

Zip up your code, discussion, and output images, and deliver as zip on Canvas. It should contain the following files:
hw5_22sp73.m
hough_lines_acc.m
hough_lines_draw.m
hough_peaks.m
draw_highlighted_peaks.m
hw5_22sp73_discussion.md
Part 1: square_edges.png, square_acc.png, square_peaks.png, square_lines.png
Part 2: noisy_edges.png, noisy_acc.png, noisy_peaks.png, noisy_lines.png
Part 3: real_edges.png, real_acc.png, real_peaks.png, real_lines.png
Additional line images (only line images foo_lines.png) for the rest of the test data images.

Bonuses

- vote using gradient to cull votes
- store pixel with vote, so you can build the line segment