I400/B490 Intro to Computer Vision - Programming assignment #4

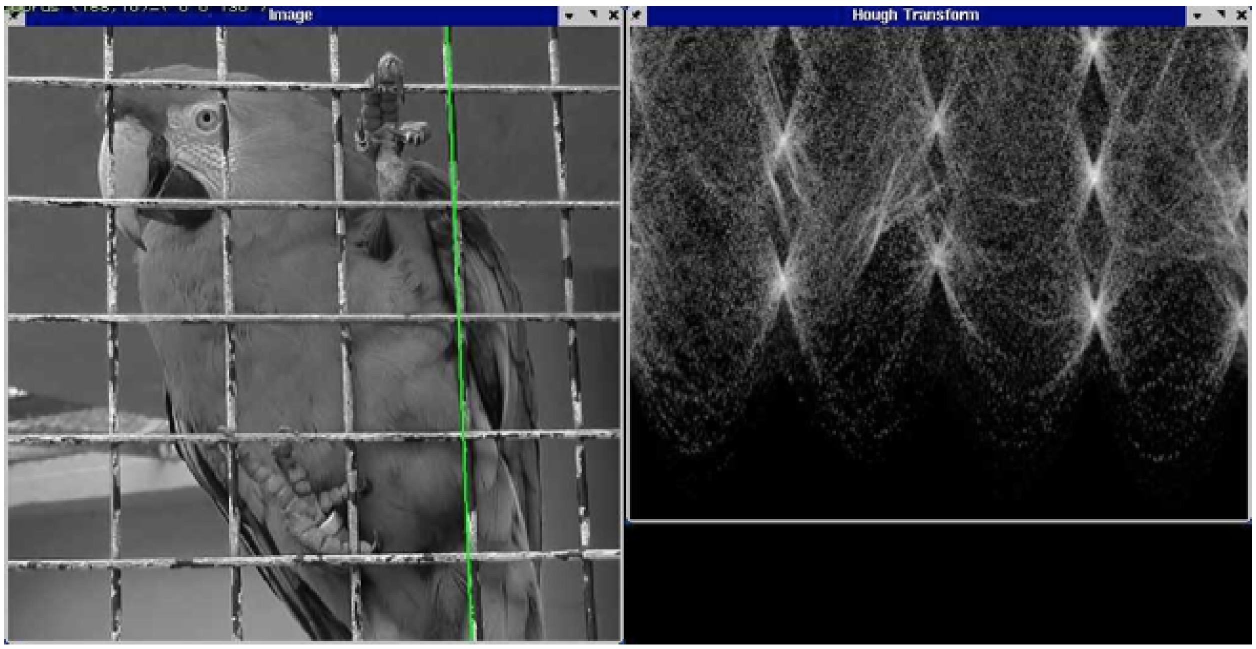
This assignment heavily relies on lecture6_1_hough.ppt . Do *NOT* use Hough transform functions from the `skimage` package directly. Use `skimage` only for the Canny edge detection and peak detection:

Problem 1: Hough transform - line fitting [100%]

0. Download the image from Canvas: “line_original.jpg”

1. Do Canny edge detection by using the function `canny` from `skimage` package: http://scikit-image.org/docs/dev/auto_examples/edges/plot_canny.html . Save the result as “line_canny.jpg”.

2. Implement the Hough transform (slides 9-16) yourself by considering a line with the following equation (slide 14): $x\cos\theta + y\sin\theta = d$. Create a 2-D Hough space (i.e., parameter space) where the two dimensions correspond to θ and d. Each edge pixel (from the step 1) should cast multiple votes in the 2-D parameter space. In this step 2, before doing the actual voting, try visualizing the results as an intensity image similar to the right-hand figure below:



This can be done by setting the histogram bin size to 1. For each edge pixel (x,y), iterate through each θ value from 0 to 360. For each θ value, compute the corresponding d, and increase the histogram bin value by 1: H[θ,d]++.

Save the image and “line_hough.jpg”. Remember to rescale the max value to 255, and change the array type to uint8 before saving the image.

3. Do voting by discretizing the parameter space into multiple (larger) bins. Try multiple bin sizes for the voting. Find the best one. Note that this Step 3 is actually identical to Step 2, just having a different bin size. Save the best voting result as “line_voting.jpg”.

4. Find peaks (i.e., multiple local maxima) from the voting results by using `peak_local_max` function in the `skimage` package: http://scikit-image.org/docs/dev/auto_examples/segmentation/plot_peak_local_max.html .
`coordinates = peak_local_max(im, min_distance=13)`

Visualize all detected lines on the image, similar to the left figure above. Save the result as "line_results.jpg".

5. Use the gradient orientation for more efficient voting (slide 24). Save the result as "line_results2.jpg".

Problem 2 (extra): Hough transform - circle fitting [20%]

0. Download the image from Canvas: "circle_original.jpg"

1. Do Canny edge detection. Save the result as "circle_canny.jpg".

2. Implement the Hough transform for the circle fitting (slides 26-33). This time, the parameter space is 3-D. Do voting, similar to the step 3 of problem 1. [10%]

3. Detect peaks (notice that this will be tricky since it will be 3-D peak detection), visualize circles on top of the original image, and save the result as "circle_results.jpg". [10%]