Homework #2
Due Th. 3/14

You must turn in your code as well as output files. Please generate a report that contains the code and ouput in a single readable format.

Getting Started

- You may want to download Irfanview image viewing software. It handles pretty much any image type, lets you convert, and provides batch processing.

  `http://www.irfanview.com/`

- Download the sample images from the class website.

  `http://www.ee.unlv.edu/~b1morris/ecg795/images/hw2`

Problems

1. Correlation Detection

   This problem requires the UIUC Car Detection Database found at

   `http://cogcomp.cs.illinois.edu/Data/Car/`.

   Notice the images are in .pgm format and cannot be natively displayed on Windows.

   (a) Write a function `corr_detect.m` that will take image template to compare with a test image and a $0 \leq \tau \leq$ threshold for object detection. The function should return a list of the image locations that are above the match threshold τ.

   (b) Test the performance of your detector on the images in the `TestImages` directory and create an ROC curve using the `trueLocations.txt` file. You should compare the performance using image `pos-1.pgm` and `pos-125.pgm` as the templates. Which template performs better? Be sure to evaluate both directions for the templates to get orientation from left-right and right-left.

   Use the overlap ratio to determine TP or FP as described for the Pascal VOC challenge

   `http://pascallin.ecs.soton.ac.uk/challenges/VOC/pubs/everingham10.pdf`

   See "Framework for Performance Evaluation of Face, Text, and Vehicle Detection and Tracking in Video: Data, Metrics, and Protocol " by Kasturi et. al in 2009 for more information about methodolgies for measuring performance of detectors.

   (c) You may notice that your detector returns a number of responses in the same area. This is a common occurrence for object detection algorithms. Design a non-maximum supression (NMS) routine to reduce these effects by only keeping the largest response in a local area. Plot the ROC with NMS on the same ROC curve from (b).

   (d) Design an image pyramid scheme and repeat your evaluation on the `TestImages_Scale` directory. Add the ROC curves to your plot.

2. Corner Detection

   (a) Consider the symmetric 2×2 matrix

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$

By finding the roots of the characteristic equation,

$$\det(\lambda I - A) = 0,$$

show that the eigenvalues of A are given by

$$\lambda = \frac{\mathrm{tr}(A) \pm \sqrt{\mathrm{tr}(A)^2 - 4\det(A)}}{2}.$$

The angle of the principle eigenvector of A is given by

$$\phi = \frac{1}{2}\arctan\left(\frac{2b}{a-c}\right).$$

(b) Compute the feature detection autocorrelation matrix A for the checkerboard image. Use a simple 3×3 box filter for the window function. Show the image with an overlay of the keypoint locations, defined as those points with $\lambda_{min} > \tau$ with τ 80% of the maximum λ_{min} value over the whole image. Also, draw a vector indicating the keypoint orientation (scaled by magnitude λ_{min}).

(c) Repeat for the fingerprint image.

3. SIFT Feature Matching

Read David Lowe's Sift papers found on his website

`http://www.cs.ubc.ca/~lowe/keypoints/`

(a) Use your corner detector from the previous problem to locate keypoints in the graffiti images (convert to grayscale). Overlay they keypoint locations (no angle) on the image.

(b) Write a function `sift_descriptor.m` that takes an image location and outputs the 128-d SIFT feature vector.

(c) Compare the keypoints in the two images. Plot the images side-by-side and connect matching keypoints by a line as done on Lowe's webpage.

(d) (Extra) Estimate the affine transform between the two images. You should compare your results using a robust estimator like RANSAC vs. the linear least squares.

4. Hough Transform

(a) Write code to implement the Hough transform for line detection. You may use `hough.m` as a guide, however, you should implment the version from the Szeliski book and not the traditional Hough transform.

(b) Compute the Hough transform of the city image. Display the Hough accumulator image and the original image with the top 5 lines overlayed.

(c) Compare your results with Matlab's implmentation.