Assignment 2
Due April 29, 2018 at 11.59 PM via ilearn Assignment-2 Submission folder

IMAGES. All images may be found in the TEST IMAGES folder on iLearn or in the folder uploaded specific to this assignment.

Independent Reading. Please read Ch. 7 of of Digital Image Processing by Gonzalez and Woods. It is on Wavelets and Multiresolution Processing.

Problem 1. [3 pts] From the above-mentioned book, solve 7.2, 7.3 and 7.15.

Problem 2. [1 pts]

(a) On the `house.tif` image, use the Haar wavelet to obtain a multi-resolution decomposition up to two levels of scale. Report and explain what you observe at the different resolutions. Reconstruct back the original image using all the multi-resolution decompositions, and by dropping some of the high frequency (detail) components. Report on the error in the reconstruction, especially on how you can choose to drop high-frequency (detail) components without sacrificing much in reconstruction accuracy.

Write MATLAB scripts for your solutions.

Problem 3. [2 pts]

(a) (0.5 pt) On the `house.tif` and `lena_gray_256.tif` images, perform edge detection using the a) Laplacian of the Gaussian and b) Canny edge detector. Report the results, especially the choice of thresholds in computing the edges.

(b) (1.5 pts) Using the output of the canny edge detector, implement a Hough transform to detect the lines in the `house.tif` image. You should not to use the built-in Hough transform related functions of Matlab.

Write MATLAB scripts for your solutions.

Problem 4. [6 pts]

(a) (2 pts) Implement the Shi-Tomasi corner detector as a function named `getCorners.m`, which is available with this package, but needs to be completed. Details of i/p and o/p can be obtained from the script. Use the following:

- Sobel operator to obtain the gradients
- Window size of 5×5 for w (see course slides for details)
- Suppress points which are not local maxima within a 5×5 window.

Run the corner detector on the `house.tif` image and display the top 50 corners according to the minimum eigen value criterion in the algorithm.

(b) This part of the problem is on HoG and SIFT features. `featurematching.m` is the main code for this problem. This code can be broken down into the following steps:

- The code loads the `blocks.png` image and then applies one of the two affine transforms `tform1` or `tform2` to obtain a transformed image. Then, it calls the `getCorners.m` function to obtain the corner points.
- (2 pts) The images along with their corresponding corners are used as inputs to the `getFeatures.m` function, which is provided with this package, but needs to be completed. Details about i/p and o/p to this function may be obtained inside it. This function should extract the Histogram of Gradient (HoG) features for 8×8 patches around each of the corner points. Use 16 bins to obtain the HoG features. You should not use the built-in function of HoG features in Matlab.

- (1 pt) The matches between the feature points of the original and transformed images are obtained using the `getMatches.m` function, which needs to be completed and its i/p and o/p details may be found inside the function. For each feature point, choose the nearest (according to HoG feature) as its match if the distance measure is greater than a certain threshold (hand picked). Use the normalized cross-correlation as a distance measure. The matches are then displayed.
- Finally, the `featurematching.m` code extracts the SIFT features using the vlfeat package (you do not need to implement this) and displays the matches.

(1 pt) Run the `featurematching.m` code on both the transformed images and analyze the displayed results. The HoG features perform better on which transformed image and why ? Which feature (HoG or SIFT) perform well on the transformed images, and why ?

DO NOT change the i/p and o/p arguments and their semantic meaning of the provided functions.

Submission Protocol. All codes should be written in MATLAB. Inbuilt functions can be used for problem 2. You should submit codes as well as explanations to each problem (if required). You should add comments to your codes to make them reader friendly. All coding related problems should be in separate scripts (.m files) named after the problem number. For e.g. Problem 3 has two parts (a) and (b). The codes corresponding to them should be in Problem3a.m and Problem3b.m. If you may require to call functions for a problem, you may do so, but include them in your submission. Keep all the images necessary to run a code in the same folder as the code, while you are submitting. You MUST also include a report written electronically (using the likes of LaTeXor MS Word). It should contain explanations, images, etc (as required).

Each student must do the assignment independently, although you may discuss prior to that. While discussion is allowed, we will be particularly careful about any plagiarism, whether from each other or from other sources.