Homework 4
Image Analysis

Notes:

- Problem 1 is theoretical and the rest of the problems are practical.
- If a question asks you to run the code on images, please save your results in a folder. Or you can use jupyter notebooks and leave the results in the notebook.
- Most of the algorithms and modules of image processing that are mentioned in problems 2 to 4 are implemented both in opencv and scikit-image libraries (or even pillow!). You are free to use any of the libraries, but a good practice might be to use a library that you are unfamiliar with right now.

If you have any questions, please contact me at msdkhairi@gmail.com

Problem 1: Edge Detection (40 points)

(a) You have a 3×3 image which can be represented by a 9×1 vector. Construct a 9×9 matrix, which, when it operates on the image vector, produces another vector, each element of which is the estimate of the gradient component (Sobel) of the image along the i axis. (To deal with the boundary pixels, assume that the image is repeated periodically in all directions.)

(b) Using the matrix derived in (a), calculate the first derivative along the i axis of the following image:

$$\begin{pmatrix} 3 & 1 & 0 \\ 3 & 1 & 0 \\ 3 & 1 & 0 \end{pmatrix}$$

What is the component of the gradient along the i axis at the centre of the image?

(c) What is non-maxima suppression and why it is used in edge detection?

(d) Let's suppose we have these two masks:

-1	0	1
-3	0	3
-1	0	1

-1	-3	-1
0	0	0
1	3	1

Are these masks separable? How can we take advantage of the separability of a 2D mask to reduce the computational cost of convolution?

(e) Are Sobel masks appropriate for all images? explain.

(f) Can we use the optimal filters for edges to detect lines in an image in an optimal way?

(g) Show that the differentiation of the output of a convolution, of a signal $u(x)$ with a filter $f(x)$, may be achieved by convolving the signal with the derivative of the filter.

(h) Show that the second derivative of a Gaussianly smoothed signal may be obtained by convolving the signal with the second derivative of the Gaussian function.

Problem 2: Edge Detection (30 points)

(a) Extract the edges of the image "ames1.JPG" by using Sobel, Prewitt, and Roberts. Compare their results. (You can either use convolution to get the results yourself, or use libraries)

(b) Implement the Canny edge detector. You can refer to this wikipedia page to get familiar with the steps of the algorithm. For the smoothing gaussian filter use a 5×5 filter with $\sigma = 1$. In the last step, a high threshold and a low threshold should be specified. Run your algorithm with different thresholds and find the ones that work best.

(c) Canny edge detector is also implemented in the image processing libraries. Extract the edges with the implemntation of a library as well, then compare the results with your own implementation and the results in part (a).

Problem 3: Shape Representation and Matching (15 points)

Load the alphabet dataset from "Alphabet.zip". Rotate the images in the dataset such that for every image there are at least 2 rotated images. (for example you can rotate each image 45, 90, and 180 degrees). This webpage describes the steps of shape representation and matching using HuMoments implemented in opencv(hu-moments are present in skimage too). Match the rotated images with the dataset and see if HuMoments are good representation for matching rotated images.

Problem 4: Template Matching (15+5 points)

You are provided with 5 templates in "templates" folder. We are going to match the templates to images in "images".

(a) Write a function that specifies which template occurs in each image.

(b) (Extra)How long does it take for your code to process all the images? How can you make your code faster? (explain and implement the faster method)

Problem 5: Image Classification (15 points Extra)

In this problem we want to classify mnist dataset. To apply deep learning techniques you should install tensorflow or pytorch library. Download the mnist dataset.(both tensorflow and pytorch offer a dataset module which can be used to download the mnist dataset.)

(a) Using the SVM classifier, classify the mnist dataset and report the accuracy on test set.(sklearn library has a SVM implementation.)

(b) Now using a neural network do the classification and report the accuracy of test set. (It is recommended for your network to have three layers of convolution and one dense layer at the end. but you can have any other structure of your interest). This tensorflow page has a step by step instruction to classify a different dataset, but the main ideas and tools are basically the same.