ENGN4528 Computer Vision – 2020

Computer-Lab 2 (C-Lab2)

Objectives:

This is CLab-2 for ENGN4528. The goal of this lab is to help you get familiar with, and practice middle-level computer vision algorithms: feature point detection, matching, image region segmentation. Practise Eigen-Face based technique for face representation, detection, and recognition.

Note, however, in all the lab task descriptions given below, we by default use Matlab as the preferred language. You are however free to choose Python if you are more comfortable with Python. If you have not used Matlab or Python before, this lab is an opportunity to get you to quickly familiar with basic language usages of Matlab/Python for image processing and computer vision.

Special Notes:

1. Each computer lab has three weeks: session-A, session-B and session-C . Tutors/Lab instructor will provide basic supervision to all sessions.
2. Before you start to work on writing lab report, we strongly recommend you to watch the video '**how to write a good lab report**' in Computer Labs section on Wattle. Keep in mind, Lab markers would not be interested in a pure collection of your experiment results. (Imagine you are demonstrating your lab works to others, you need appropriate interpretations for experimental results.)
3. The requirement of Lab submission is attached in the last pages. Please ensure your submission meets the requirement.
4. Your Lab will be marked based on the overall quality of your lab report. The report is to be uploaded to Wattle site before the due time, which is usually on the Sunday evening of the third week after the announcing of computer lab tasks. (e.g. Sun of Week5 for Clab1). Note that for due dates at nighttime, the submission will close at 9am the following working day. Any submissions later than the 9am (e.g., 9:01am) will be considered a day late. The penalty for late submission is 10% per day. All student should consider the personal problems of Internet in advance.
5. It is normal if you cannot finish all the tasks within the two 2-hour sessions — these tasks are designed so that you will have to spend about 9 hours to finish all the tasks including finishing your lab report. This suggests that, before attending the second lab session, you must make sure that you have almost completed 80%.

Academic Integrity:

You are expected to comply with the University Policy on Academic Integrity and Plagiarism. You are allowed to talk with / work with other students on lab and project assignments. You can share ideas but not code, you should submit your own work. Your course instructors reserve the right to determine an appropriate penalty based on the violation

of academic dishonesty that occurs. Violations of the university policy can result in severe penalties.

CLab-2 Tasks

1 Tasks Harris Corner Detector. (5 marks)

For Matlab users:

1. Read and understand the corner detection code 'harris.m'.
2. Complete the missing parts, rewrite 'harris.m' into a Matlab function, and design appropriate function signature (1 mark).
3. Comment on line #13 and every line of your solution after line #20 (0.5 mark).
4. Test this function on the provided four test images (can be downloaded from Wattle). Display your results by marking the detected corners on the input images (using circles or crosses, etc) (0.5 mark for each image, 2 marks in total).
5. Compare your results with that from Matlab's built-in function *corner()* (0.5 mark), and discuss the factors that affect the performance of Harris corner detection (1 mark).

Listing 1: harris.m

```
1  %%
2  % CLAB2 Task-1: Harris Corner Detector
3  % Your name (Your uniID)
4  %
5  sigma = 2; thresh = 0.01; % Parameters, add more if needed
6  % Derivative masks
7  dx = [-1 0 1;-1 0 1; -1 0 1];
8  dy = dx'; % dx is the transpose matrix of dy
9  % compute x and y derivatives of image
10 Ix = conv2(bw,dx,'same');
11 Iy = conv2(bw,dy,'same');
12
13 g = fspecial('gaussian',max(1,fix(3*sigma)*2+1),sigma);
14 Ix2 = conv2(Ix.^2,g,'same'); % x and x
15 Iy2 = conv2(Iy.^2,g,'same'); % y and y
16 Ixy = conv2(Ix.*Iy,g,'same'); % x and y
17
18 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
19 % Task: Compute the Harris Cornerness                                    %
20 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
21
22 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
23 % Task: Perform non-maximum suppression and                             %
24 %       thresholding, return the N corner points                       %
25 %       as an Nx2 matrix of x and y coordinates                        %
26 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

For Python users:

In [1]:
```
"""
CLAB2 Task-1: Harris Corner Detector
Your name (Your uniID):
"""
import numpy as np
```

In [2]:
```
def conv2(img, conv_filter):
    # flip the filter
    f_siz_1, f_size_2 = conv_filter.shape
    conv_filter = conv_filter[range(f_siz_1 - 1, -1, -1), :][:, range(f_siz_1 - 1, -1, -1)]
    pad = (conv_filter.shape[0] - 1) // 2
    result = np.zeros((img.shape))
    img = np.pad(img, ((pad, pad), (pad, pad)), 'constant', constant_values=(0, 0))
    filter_size = conv_filter.shape[0]
    for r in np.arange(img.shape[0] - filter_size + 1):
        for c in np.arange(img.shape[1] - filter_size + 1):
            curr_region = img[r:r + filter_size, c:c + filter_size]
            curr_result = curr_region * conv_filter
            conv_sum = np.sum(curr_result)  # Summing the result of multiplication.
            result[r, c] = conv_sum  # Saving the summation in the convolution layer feature map.

    return result
```

In [3]:
```
def fspecial(shape=(3, 3), sigma=0.5):
    m, n = [(ss - 1.) / 2. for ss in shape]
    y, x = np.ogrid[-m:m + 1, -n:n + 1]
    h = np.exp(-(x * x + y * y) / (2. * sigma * sigma))
    h[h < np.finfo(h.dtype).eps * h.max()] = 0
    sumh = h.sum()
    if sumh != 0:
        h /= sumh
    return h
```

In [4]:
```
# Parameters, add more if needed
sigma = 2
thresh = 0.01

# Derivative masks
dx = np.array([[1, 0, -1], [1, 0, -1], [1, 0, -1]])
dy = dx.transpose()

# computer x and y derivatives of image
Ix = conv2(bw, dx)
Iy = conv2(bw, dy)
```

In [5]:
```
g = fspecial((max(1, np.floor(3 * sigma) * 2 + 1), max(1, np.floor(3 * sigma) * 2 + 1)), sigma)
```

In [6]:
```
Iy2 = conv2(np.power(Iy, 2), g)
Ix2 = conv2(np.power(Ix, 2), g)
Ixy = conv2(Ix * Iy, g)
```

In [7]:
```
######################################################################
# Task: Compute the Harris Cornerness
######################################################################


######################################################################
# Task: Perform non-maximum suppression and
#       thresholding, return the N corner points
#       as an Nx2 matrix of x and y coordinates
######################################################################
```

1. Read and understand the above corner detection code (page 3).
2. Complete the missing parts, rewrite them to 'harris.py' as a python script, and design appropriate function signature (1 mark).
3. Comment on block #5 (corresponding to line #13 in "harris.m") and every line of your solution in block #7 (0.5 mark).
4. Test this function on the provided four test images (can be downloaded from Wattle). Display your results by marking the detected corners on the input images (using circles or crosses, etc) (0.5 mark for each image, 2 marks in total).
5. Compare your results with that from python's built-in function *cv2.cornerHarris()* (0.5 mark), and discuss the factors that affect the performance of Harris corner detection (1 mark).

In your Lab Report, you need to list your complete source code with detailed comments and show corner detection results and their comparisons for each of the test images.

2 K-Means Clustering and Color Image Segmentation. (5 marks)

In this task, you are asked to implement your own K-means clustering algorithm for colour image segmentation, and test it on the following two images as shown in Fig.1. Please note that the PNG-type (Portable Network Graphics) images are in the 48-bit format, and you need to first convert them to 24-bit images.

1. Implement your own K-means function *my_kmeans()*.The input is the data points to be processed and the number of clusters, and the output is several clusters of points (you can use a cell array for clusters). Make sure each step in K-means is correct and clear, and comment on key code fragments (1.5 marks).
2. Apply your K-means function to color image segmentation. Each pixel should be represented as a 5-D vector that encodes: (1) L^* - lightness of the color; (2) a^* - the color position between red and green; (3) b^* - the position between yellow and blue; (4) x, y - pixel coordinates. Please compare segmentation results (1) using different numbers of clusters (1 mark), and (2) with and without pixel coordinates (1 mark).
3. The standard K-means algorithm is sensitive to initialization (e.g. initial cluster centres/seeds). One possible solution is to use K-means++, in which the initial seeds are forced to be far away from each other (to avoid local minimum). Please read the material http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf, summarize the key steps in the report (0.5 mark), and then implement K-means++ in your standard algorithm as a new initialization strategy (0.5 mark). Compare the image segmentation performance (*e.g.*, convergence speed and segmentation results) of this new strategy, with that of standard K-means, using different numbers of clusters and the same 5-D point representation from previous question (0.5 mark).

3 Face Recognition using Eigenface. (10 marks)

In this task, you are given the Yale face image dataset Yale−FaceA.zip. The dataset contains a *training_set* of totally 135 face images captured from 15 individuals (9 images from each individual). You are also given 10 test images in the *Test_set* directory.

1. Unzip the face images and get an idea what they look like. Then take 10 different frontal face images of yourself, convert them to grayscale, and align and resize them

to match the images in Yale-Face. Explain why alignment is necessary for Eigen-face (0.5 mark).

2. Train an Eigen-face recognition system. Specifically, at a minimum your face recognition system should do the following:

   (1) Read all the 135 training images from Yale-Face, represent each image as a single data point in a high dimensional space (the entire dataset is converted to a 2D map) and collect all the data points into a big data matrix.

   (2) Perform PCA on the data matrix (1 mark), and display the mean face (1 mark). Given the size of input image, direct eigen decomposition of covariance matrix would be slow. Read lecture notes and find a faster way to compute eigen values and vectors, explain the reason (1 mark) and implement it in your code (1 mark).

   (3) Determine the top k principal components and visualize the top-k eigen- faces in your report (1 mark). You can choose k=10 or k=15.

   (4) For each of the 10 test images in Yale-Face, read in the image, determine its projection onto the basis spanned by the top k eigenfaces. Use this projection for a nearest-neighbour search over all the 135 faces, and find out which three face images are the most similar. Show these top 3 faces next to the test image in your report (1.5 marks). Report and analyze the recognition accuracy of your method (1 mark).

   (5) Read in one of your own frontal face images. Then run your face recognizer on this new image. Display the top 3 faces in the training folder that are most similar to your own face (1 mark).

   (6) Repeat the previous experiment by pre-adding the other 9 of your face images into the training set (a total of 144 training images). Note that you should make sure that your test face image is different from those included in the training set. Display the top 3 faces that are the closest to your face (1 mark).

Hints:

(1) A simple way to do alignment is to manually crop (and rotate if necessary) the face region, resize the face image to a standard shape, and make sure the facial landmarks are aligned – e.g. centre of eyes, noses, mouths are roughly at the same positions in an image.
(2) In doing eigen-decomposition, always remember to subtract the mean face and, when reconstructing images based on the first k principal components, add the mean face back in at the end.
(3) You can use Matlab's (e.g., *eigs()*, *svd()*, *inv()*) and Python's (e.g., *numpy.linalg.eig(), numpy.linalg.svd(), numpy.linalg.inv() or other corresponding functions*) functions for matrix decomposition and inverse matrix to implement PCA. Other than these, you should not use Matlab's and Python's built-in PCA or eigenface function if there is one.

Lab Report Requirement

1 Files

Upload a single ZIP file by the due date. You must use the following file name: CLab- 1/2/3-Report-Uxxxxxx.zip, replacing Uxxxxx with your uni-ID.

(a) Required input and output images.

(b) A script name "code.py" or "code.m" includes all your *.m and *.py code implementations. Use comment lines to separate each task.

Your report file must contain the following contents:

A PDF of your Lab Report. The report generally contains sample results from all the Lab Tasks, along with necessary comments and descriptions, questions and answers. For more detail. Please refer to the following Template and General Instructions for Lab Report on the next page.

2 Lab Report

Kindly document different question under respective headings provided with the assignment. For example:

Task-1: The Question
1. Your first question under this theme

Documentation, observations, results, analysis etc.

2. Your second question under this theme

Documentation, observations, results, analysis etc.

3. Your third question under this theme

Documentation, observations, results, analysis etc.

2.1 General Formatting Instruction

Kindly use the same font single-spaced type for the entire document as much as possible, you may use the bold and italic version of the same font to highlight the important points. In the report, you need to use Times New Roman, which is quite widely used font to document projects and research papers.

• Kindly, use appropriate font size for sections heading and its contents accordingly. For example, 15 points Times, boldface type for heading and 12 points single-spaced type for the content is one of the widely used font sizes for documenting research papers.

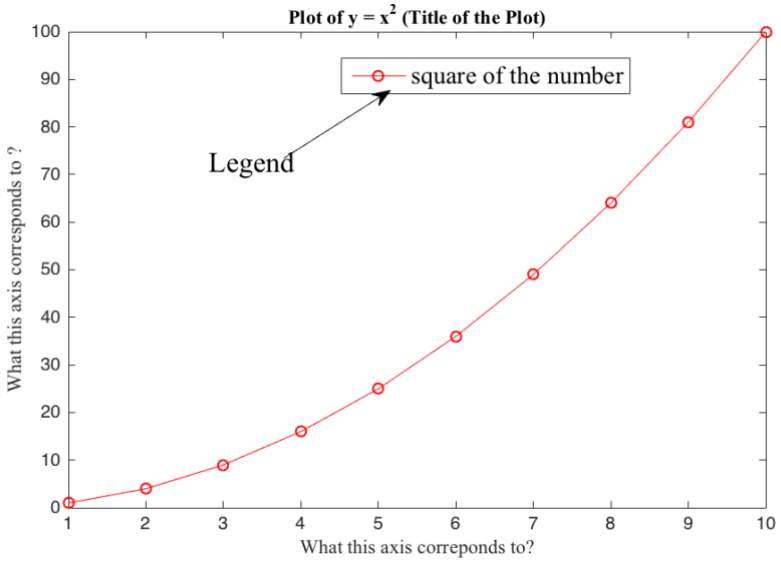
• Please number all your sections and subsections of the tasks as provided in the assignment.

• Please show the images mentioned in each task to make your answer clearer.

Brief explanations on how you solve the problem are expected.
• Please give your own answer following the question guidelines. ***Handwriting draft is not permitted***.
• Handwriting draft is not permitted for your submission.

2.2 Table, Figures and Plots

This is one of the important aspects of evaluating your report. Figures and the caption of the tables must be appropriately addressed. The figure should have an appropriate title if required. All the legends in the figure should be properly highlighted. The caption of the figure should explain your observation and understanding which may comprise of quantitative or qualitative evaluation to endorse your observation. Some of the widely used font to caption your figure, table and callouts are 10-11 points Roman type, 10-11 points Helvetica non-boldface type. Kindly, adjust the size of the figure in the document appropriately such that its clearly visible and perfectly eligible to illustrate your observation. We encourage you to look into the below example for reference. Note: You **cannot insist** we should zoom in or out to see tiny details on the graphs, plots, photographs, illustration, etc. Also, make sure the figures you include in your document is ***not a copyright image***.



Caption: Variation in the y-axis corresponding to the values in the x-axis and What does this mean, your observations?

For tables, graphs and others as well, kindly document the purpose of the statistical illustration which should include titles and proper labelling of the data and statistics.

Please follow the requirements to write your own Lab report.

========== END of C-Lab-2 report ==========