Exercises for
Introduction to Machine Learning
Linda-Sophie Schneider, Johannes Enk, Fengyu Han, Gia Long Nguyen, Lukas Huettner, Yangbin Peng
Task 3: deadline: 6th July 2024

Pattern Recognition Lab

Exercise 03: Palm Print Identification

Short notice

For this exercise your task will be very similar to what students often have to do during their Bachelor/Master/PhD thesis: program a method presented in a research paper. This is a task typically quite complicated because papers are limited in size (often 6-8 pages only!), and thus methods are only summarized. This paper is longer, and thus provides details which will be extremely helpful to solve this exercise. Its length is therefore not a difficulty, but a great help ;-)

Shorter notice

Read the whole exercise sheet before starting, especially the important hints on page 2.

Exercise 1 Implement a Palm Print Identification System

In this task, you are supposed to implement a palm print identification tool according to the paper of Li et al [1], provided on StudOn. It mainly consists of three major steps:

(a) The first step is the pre-processing step (Chapter 2).

(b) The second step is the feature extraction (Chapter 3).

(c) The third step is the feature matching and identification of the palm print (Chapter 4).

Three palm print images are provided, but, you are allowed (and even encouraged) to enlarge this database by your own palm prints[1].

For making your task a little easier, a code skeleton is provided consisting of the four following Python modules:

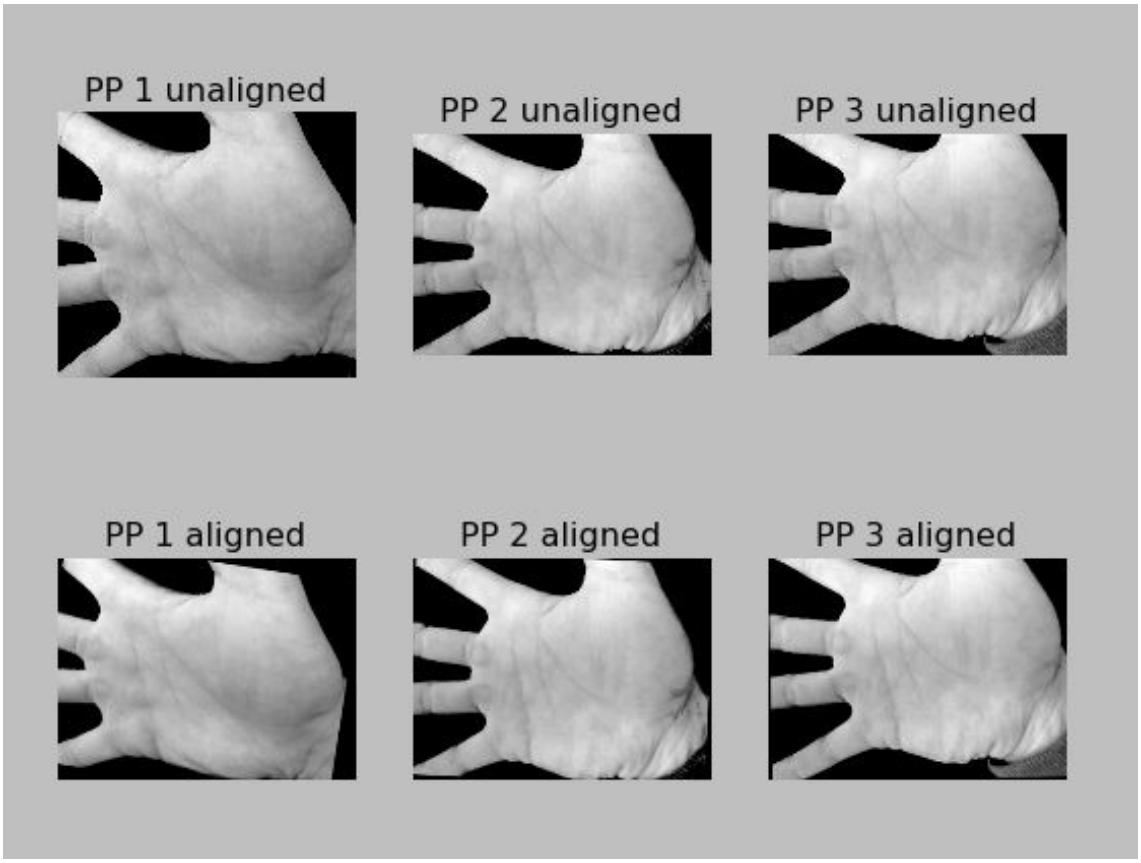
- 'mainPalmprint.py': Main module to call the sub-methods and visualize the results (DO NOT MODIFY!).
- 'PalmprintAlignment.py': Module pre-processing the original palmprint image and returning it (See Chapter 2).
- 'FourierTransform.py': Module calculating the Fourier Transform of the pre-processed image. (See Ch. 3).
- 'DistanceMeasure.py': Module calculating the final energies (See Ch. 4).

[1]This might make the exercise even more interesting for you. Make sure the photos look much like the provided ones, and have the same resolution. Please do not share them publicly (i.e., on the forum).

Important Hint 1: In Chapter 2 – Page 5/6 – it is mentioned to crop the image to an subimage with a fixed size. You are not supposed to do so! We can also work on the complete image! However, if you want to crop the image, you can, but this will not be evaluated during your submission.

Important Hint 2: You must not modify the main files, so if you want to try some variations of it, copy it to a new file and modify this new file.

Important Hint 3: Here's how the aligned palm print images should look like:



---

[1] Li, Wenxin, David Zhang, and Zhuoqun Xu. "Palmprint identification by Fourier transform." International Journal of Pattern Recognition and Artificial Intelligence 16.04 (2002): 417-432.

Exercise 2 Align Kaktovik symbols and compare them using MSE and θ-feature

This task focuses on spatial alignment (implemented in exercise 1) and image similarity using simple metrics. You will work with handwritten Kaktovik symbols and evaluate whether automatic alignment and matching produce meaningful similarity results. Your task consists of two parts:

Part 1 - Image Alignment:

Complete the file `kaktovikAlignmentSimple.py`, which includes the function `simpleAlignment(img)`. This function aligns a grayscale image by:

(a) Resizing the input image to a fixed size using OpenCV.

(b) Applying Otsu thresholding using OpenCV to obtain a binary mask.

(c) Finding the bounding box of the foreground using NumPy (no OpenCV allowed here).

(d) Extracting the corresponding region from the original (grayscale) image.

(e) Resizing the region to half the canvas size using OpenCV.

(f) Centering the result on a square canvas of defined size (using NumPy).

Part 2 - Distance Measure:

Extend the file `DistanceMeasure.py` with a function that computes the Mean Squared Error (MSE) between two grayscale images of the same size.

Evaluation:

Use the provided file `mainKaktovik.py` to test your implementation. It compares three aligned Kaktovik images using both the $\hat{I}_{\theta}$-feature and your MSE function. Interpret the similarity values - do the scores match your expectations?

Important Hint 4: Here's how the aligned Kaktovik images should look like: