Assignment #2

Image Based Biometry
Faculty of Computer and Information Science
University of Ljubljana

Iris Recognition

You need to implement either Local Binary Patterns (LBP) [1] or some other method of your choosing/design, for iris code/feature extraction. Feel free to use modern LLM helpers, search engines and websites. Just make sure you understand each line of code produced and the logic behind it.

You **do not need to prepare a report for this assignment**. You need to implement some approach (LBP by default) that will perform as well as possible (which also means optimizing parameters, such as radius, sampling size, etc. as much as possible). The recognition pipeline is already prepared including segmentation, as shown in Fig. 1.

I. Installation and Preparation

For Iris Recognition we are using a Python project *Open Source Iris Recognition* (https://github.com/CVRL/OpenSourceIrisRecognition), developed by our colleagues at Notre Dame Computer Vision Research Lab.

INSTALLATION

(I) Download the predownloaded project with two placeholders for the functions you need to write from here: https://tinyurl.com/HDBIF-IBB.

(II) Create a new conda environment with all the required packages: `conda env create -f minimal_environment.yml -n [name]`

(III) After the installation, run `hdbif.py` to see that things are working.

(IV) Download the dataset from here: https://tinyurl.com/IBB-A2-Data into the `data/` folder. You may need to retry downloading it a few times because OneDrive is from Microsoft (imagine Adobe, SAP, Boeing levels of "quality").

Alternatively you can get the project from https://github.com/CVRL/OpenSourceIrisRecognition/tree/main/methods/HDBIF and follow instructions there.

II. Tasks

First steps (these are for your reference and will not be checked or graded):

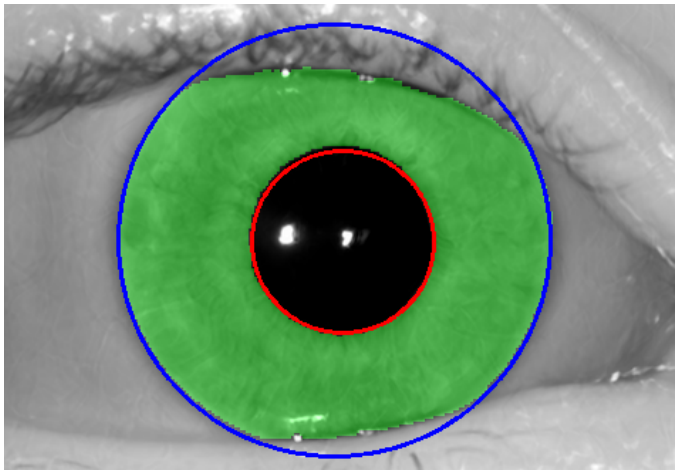

Fig. 1. Example of a iris segmentation marked with green.

- Run `hdbif.py` on the downloaded datasets and observe HDBIF performance. This will serve as a baseline for your own implementations. Although do not expect to beat HDBIF with LBP.
- Add a performance measure of your choice to assist during development (you can copy from the previous assignment) to evaluate your implementation.

For the core part, do the following:

1) Switch calls in `hdbif.py` from `extractCode` to `extractIBBCode` and `matchCodesEfficient` to `matchIBBCodes`.
2) Program some extraction method (by default that is LBP [1] based on the paper: https://ieeexplore.ieee.org/document/1017623) within IrisRecognition.py, function `extractIBBCode`. Make sure to find and set optimal parameters.
3) Program comparison of computed codes within `IrisRecognition.py`, function `matchIBBCodes`.
4) List the core (LBP) improvements (parts you implemented/optimized) in the comments on top of the `extractIBBCode` function.
5) Ensure the two function names are correct as your approaches will be evaluated on a sequestered test set.

III. Grading

Your will be evaluated with:

- **2.5pts** Ingenuity, code quality and your knowledge.
- **2.5pts** Recognition performance on the sequestered test set. Well implemented and optimized LBP can yield all the points here.

IV. Submission

Submit **ONLY the modified `irisRecognition.py`** on Eučilnica by the deadline (make sure that functions `extractIBBCode` and `matchIBBCodes` are spelled correctly). Oral defenses will be during that week. Have fun! (We mean it).

References

[1] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.