Homework Assignment #4

Image Based Biometry 2022/23
Faculty of Computer and Information Science
University of Ljubljana

Answers to Questions

Luka Boljević

Code is worth 4 points (setting it up and changing necessary bits), 16 points are worth questions below:

1. Q — [**1 pt**] **Describe the dataset(s) you used for your experiments? Are there any issues with it and how can these issues be addressed?**

> A — The CASIA dataset consists of grayscale iris images, from 108 subjects, each having 7 iris images. The images were taken under near infrared illumination, hence the reason we can see "inside" of the iris. The conditions seem controlled, as for example the illumination is more or less the same on all images, and the pupil seems to have been artificially i.e. manually darkened/blackened to eliminate the light coming from the sensor taking the picture.
>
> Other than varying pupil sizes (across subjects and for the same subjects), eyelashes and eyelids, which are problems an iris recognition system should always tackle, most of the images in the provided CASIA dataset seem fine. Only potential issues I could find are the following:

- There are a few "lower quality" images, such as `007_1_3.jpg` or `056_2_3.jpg` for example.
- One subject (`062`) has a large pupil on all of his/her images, meaning

it might be harder to verify his/her identity when his/her pupil is of normal size.

- Even though it was stated that the pupil is artificially blackened, the pupil on image `067_1_1.jpg` is going over the eyelid.
- Pictures of subjects `091` and `101` aren't the greatest - the eyelashes and large pupil cover up a lot of the iris for both of them, meaning it could be hard to extract the iris code properly.

The "fix" is quite simple for all issues - either take new photos, or throw away the sample entirely.

2. Q — [**3 pts**] **Provide a diagram of your whole pipeline (code), including inputs and outputs in each step.**

A — Step 1 is always iris image acquisition. For that purpose, we have the CASIA database.

Step 2 is segmentation/localization. For a query image, we first segment out/localize the iris and the pupil - we obtain the center and radius of both. We also obtain a separate image, which marks where the noise (eyelashes, eyelids) is in the query image.

Step 3 is normalization. Using the image with marked noise, and the centers and radii of the iris and the pupil, we normalize the iris region by unwrapping it into a rectangle of constant dimensions. During this step, we also obtain a normalized form of the noise region.

Step 4 is feature extraction/encoding. From the normalized iris and noise regions, we generate the iris template/code and noise mask.

After step 4, we can either perform enrollment, or use the iris code (and mask) for matching against already enrolled templates (i.e. for verification).

For simplicity, to enroll a query image, we simply save the iris template/code and mask from step 4 in a MATLAB style `.mat` file.

To verify a query image, we compare the extracted iris code (and mask) from step 4 against all templates in the database. Based on the used distance function (Hamming) and a set threshold, we determine whether the query iris code matches any template in the database (i.e. distance is below threshold for some template) or not (distance is above threshold).

3. Q — [**2 pts**] **How did you tackle the problem of segmentation? Can you provide an example of a good and a poor segmentation? What do you think are the reasons for both?**

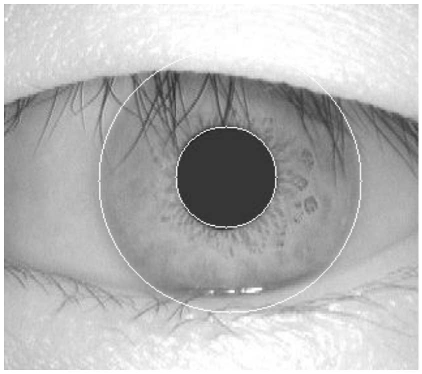
A — We used Daugman's integro-differential operator (or circular edge detector) to search for the boundary of the pupil (commonly called pupillary boundary), and then the of the iris (commonly called the limbus boundary). Both of these boundaries can be approximated using circles (of course this is now always the case, this is just the assumption of the integro-differential operator).

After we have successfully identifies the two boundaries, we can also detect/mark where the eyelids and eyelashes in the original image are. Pixels that contain eyelashes and/or eyelids are marked with black, the rest is white. For the CASIA dataset, eyelashes can be detected by comparing the pixel intensity of the (grayscale) query image to a threshold. If the pixel intensity is below the threshold (80), we "classify" this as an eyelash (mark those pixels black).

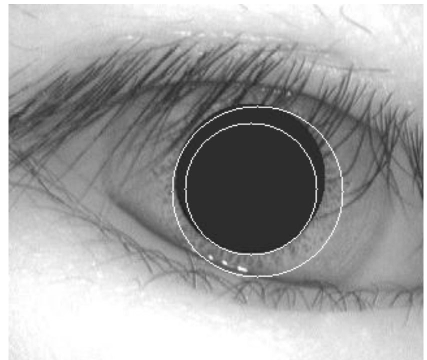
Eyelids were detected by searching for lines in a rectangle circumscribed around the limbus boundary. For this purpose, linear Hough transform and the Canny edge detector were used.

An example noise mask, for image `008_1_2.jpg`, is shown in Figure 1.

Figure 1: Example noise mask, for image `008_1_2.jpg`.



(a) Good segmentation - image `008_1_2.jpg`.



(b) Bad segmentation - image `091_1_2.jpg`.

Figure 2: Examples of good and bad segmentations.

Examples of good and bad segmentations are given in Figures 2a and 2b respectively. The good segmentation is good because there are little occlusions, the iris and the pupil are more or less clearly visible. The bad one is bad because the eyelashes are too thick and long, eyelids cover up quite a decent portion of the iris, and the pupil itself is quite big, at least in regard to the amount of visible iris.

4. Q — [**2 pts**] **Based on your overview of the literature, what are some of the state-of-the-art solutions to the problem of segmentation?**

A — While there are a lot of different robust, effective and efficient methods (those based on integro-differential operators, Hough transform and/or Canny edge detection, iterative algorithms, analysis of local features, active contours, etc.), most of the state of the art methods seem to incorporate neural networks and deep learning.

5. Q — [**2 pts**] **Describe your normalization part.**

A — Normalization can be performed after obtaining the results of the segmentation process.

The idea of normalization is to "unwrap" the iris region (I like to call it "the donut") into a rectangle of fixed dimensions, where the rows on that rectangle correspond to concentric bands of the iris region. This is the popular rubber sheet model from Daugman. In our case, we used a rectangle of size 240x20.

As the result, we obtain the normalized iris region, and the normalized noise region.

6. Q — [**2 pts**] **Describe your feature extraction part.**

A — After normalizing the iris (and noise) region, we can now extract the iris code/template, and the noise mask.

The normalized iris region, i.e. each row of it is first convolved with a 1D log-Gabor filter, with centre frequency $f_0 = 1/18$, and with $\sigma/f_0 = 0.5$, where σ is the bandwidth of the Gabor filter. The output of the Gabor filter is then demodulated to compress data - this is done by quantizing the phase information into two levels, since we have a 1D Gabor filter, which gives us the iris code. Noise can be determined by looking at the output of the Gabor filter too - if the amplitude is close to zero, this can be marked as noise.

As stated, the result of extracting phase data from the Gabor filter are the iris template/code and the noise mask.

7. Q — [**2 pts**] **Based on your overview of the literature, what are some of the state-of-the-art solutions to the problem of iris feature extraction?**

A — Again, similar to state-of-the-art segmentation, there exist a lot of different methods, but state-of-the-art feature extraction seems to be using neural networks and deep learning - something like [1] might be an example =).

8. Q — [**2 pts**] **Briefly report on rank-1 and rank-5 on the CASIA (on the images that you did not use for enrollment). What distance measure did you use?**

A — We used the normalized Hamming distance to calculate the distance between two iris codes. Of course, the noise masks are incorporated in order to disregard the noisy regions of the iris codes. The normalized Hamming distance between iris code A, and iris code B, is computed as

$$NHD = \frac{||(IrisCodeA \oplus IrisCodeB) \cap MaskA \cap MaskB||}{||MaskA \cap MaskB||}$$

The XOR operator detects bits that aren't the same, while the AND operator disregards the noisy regions. With the denominator, we normalize the HD between $[0, 1]$. The minimum value of 0 means there is absolutely no difference between the iris codes, while a value of 1 means the iris codes are completely different. The value of 0.5 means a random agreement between the iris codes - we want to avoid $NHD \approx 0.5$, and we generally consider $NHD < 0.5$ to mean a good match.

To account for the possibility of head tilts (i.e. eye/iris tilts), the NHD isn't calculated just between the original iris codes and noise masks. NHD is calculated for a range of left and right circular bitwise shifts. We used the range $[-8, 8]$, negative values meaning the iris code/noise mask is shifted to the left, positive meaning the same but to the right. After all 17 NHD values are calculated, the minimum is used as the distance between the two input iris codes.

Because we mentioned we consider $NHD < 0.5$ to mean a good match between the two iris codes, we tested the pipeline against various matching thresholds < 0.5, and calculated rank-1 and rank-5 accuracy for each. We used images with path `CASIA/*/1/*.jpg` for enrollment (324 images), while we tested the pipeline on images with path `CASIA/*/2/*.jpg` (432 images). The results are given in Table 1.

We were also curious what would happen if we switched the images we used for enrollment, and for testing - i.e., we use images `CASIA/*/2/*.jpg` for enrollment, and images `CASIA/*/1/*.jpg` for testing. The results of this test are given in Table 2.

We can see that accuracies in Table 2, when we enrolled images `CASIA/*/2/*.jpg` (432 of them), are slightly better than in Table 1, when we enrolled images `CASIA/*/1/*.jpg` (324 of them). The reason for that is probably because of the larger number of templates for each subject - 4 in the former case, and 3 in the latter case.

Threshold \ Rank-N	Rank-1	Rank-5
0.10	0.000	0.000
0.20	0.000	0.000
0.30	0.497	0.497
0.38	0.905	0.905
0.45	0.961	0.972

Table 1: Rank-1 and rank-5 accuracies for different matching thresholds. If NHD between two iris codes is less than a given threshold, the two codes are considered a match. Enrolled images were `CASIA/*/1/*.jpg`, while the pipeline was tested on images `CASIA/*/2/*.jpg`. 0.38 was determined to be the best overall threshold (based on F1 score).

Threshold \ Rank-N	Rank-1	Rank-5
0.10	0.000	0.000
0.20	0.000	0.000
0.30	0.488	0.488
0.38	0.932	0.932
0.45	0.972	0.975

Table 2: Rank-1 and rank-1 accuracies for different matching threshold, but now the enrolled images were `CASIA/*/2/*.jpg`, while the pipeline was tested on images `CASIA/*/1/*.jpg`.

9. Q — **(OPTIONAL – NOT GRADED) Did you encounter any difficulties while solving this assignment? Would you recommend something else or change something?**

   A — There weren't any real difficulties. Everything was simple and straight forward, and the code works without problems. In a way, the assignment was quite fun, because we got to see and understand how the entire pipeline is implemented, and how it works. It doesn't take long to finish. I wouldn't really change anything.

References

[1] Hafner, Andrej and Peer, Peter and Emeršič, Žiga and Vitek, Matej. Deep Iris Feature Extraction. In *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 258–262, 2021.