Francisco Javier Sáez Maldonado
franciscojavier.saez@estudiante.uam.es

Exercise 1

1.1 Copy here the two fingerprint images provided as examples example1_1 and example1_2

We are provided with the following to examples of fingerprints.

(a) Example1_1 (b) Example1_2

Figure 1: Examples of fingerprints.

As we can see, the pictures of the fingerprints have relatively high quality and its macro-singularities can be observed at first sight.

1.2 How many macro-singularities do you observe in each fingerprint?

As we already know, macro-singularities can be loops, deltas or whorls.

In the previous figure we can observe a single loop in each of the fingerprints. No deltas or whorls have been found.

1.3 Mark the macro-singularities in the images (deltas and loops).

We mark the previously mentioned loops using a red circle in Figure 2.

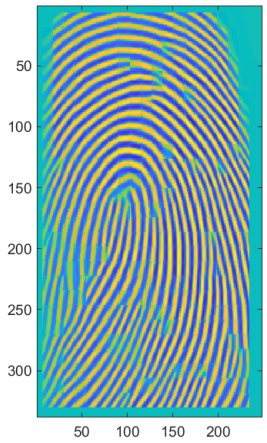
(a) Example1_1 with a marked loop. (b) Example1_2 with a marked loop.

Figure 2: Marked loops.

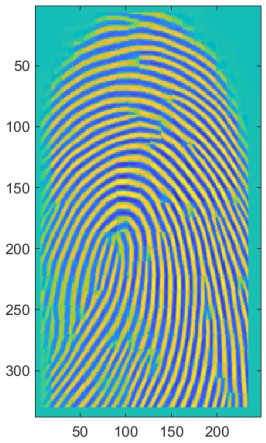
Exercise 2

2.1 Execute the provided code for Fingerprint Enhancement and paste the resulting image here

We execute the Matlab script `main.m` and stop it when the enhanced images are printed. The obtained enhanced fingerprints are shown in Figure 3.



(a) Example1_1 enhanced.



(b) Example1_2 enhanced.

Figure 3: Enhanced fingerprint examples.

2.2 What differences do you observe with respect to the original fingerprints?

The aim of the enhancement techniques is to improve the fingerprint image quality in order make the feature extraction task easier. Typically, we want to complete the ridge lines, deal with cuts or bruises on the finger or obtain a better separation between parallel ridges.

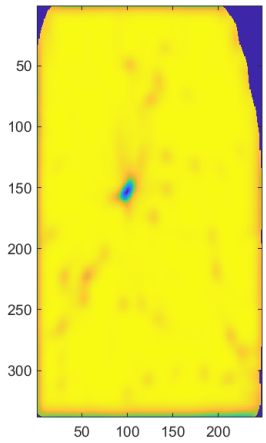
As we can see, the fingerprints in Figure 1 had incomplete ridge lines at the bottom part of the image which have been completed in the enhancement in Figure 3.

Also, the image has been *sharpened*, which means that now the contrast between the ridge lines and the *white spaces* has been increased, making ridge lines easier to identify.

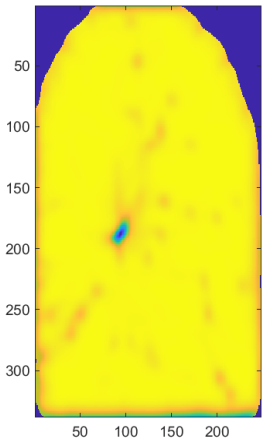
Exercise 3

3.1 Execute now the code for Quality Maps, and paste the resulting quality maps

If we keep executing the `main.m` script we obtain the following Quality Maps:



(a) Quality Map of the first fingerprint.



(b) Quality Map of the second fingerprint.

Figure 4: Quality maps.

3.2 What is the range of values for these quality maps?

To obtain this quantities, we have added the following code to the main script:

```
min(relI1,[],'all')
max(relI1,[],'all')

min(relI2,[],'all')
max(relI2,[],'all')
```

This code shows that:

- The minimum value for both Quality Maps is 0.0
- The maximum value for the first fingerprint Quality Map is 0.9991
- The maximum value for the second fingerprint Quality Map is 0.9988

Thus, we can say that both Quality Maps have all its values in the range $[0, 1]$.

3.3 What kind information (apart from the quality) can be inferred from such code?

We examine the file `testfin.m` to understand the result that we are obtaining. If we read the first commented lines, we find that the function we applied returns

```
% Returns:    newim - Ridge enhanced image.
%             binim - Binary version of enhanced image.
%             mask  - Ridge-like regions of the image
%             reliability - 'Reliability' of orientation data
```

Considering this and the code of the `main.m` file, what we just showed in Figure 4 is the **reliabiltiy** of the orientation data, that is, how secure we are of the orientation of the ridge-lines in each point of the image.

As we can see in the figure 4, we can say that we can not rely on the results *outside* the fingerprint (with outside meaning the white part with no ridge-lines), and that the algorithm has some problems in the zone where the loop is detected. This makes sense to us, since in the top of the loop the orientation of the ridge-line changes completely.

We also find that the process followed to obtain the quality maps is:

1. Identify the ridge-like regions and normalise the image. We can see that different blocksizes and thresholds have been used but finally using $blocksize = 16$ and $threshold = 0.1$ are the values used.
2. Determine the orientations of the found ridge-like regions.
3. Determine ridge frequency values across the image. Then, the median frequency is multiplied in the final mask that will be applied to the image (a comment says that the median frequency performed better than the actual frequency).
4. Lastly, the filter is applied to enhance the patterns

Exercise 4

4.1 Execute the code in order to show the Binarized Fingerprint and the Segmented Fingerprint. Apply different values of quality threshold (0.1, 0.3, 0.6, 0.9) and paste here the resulting images

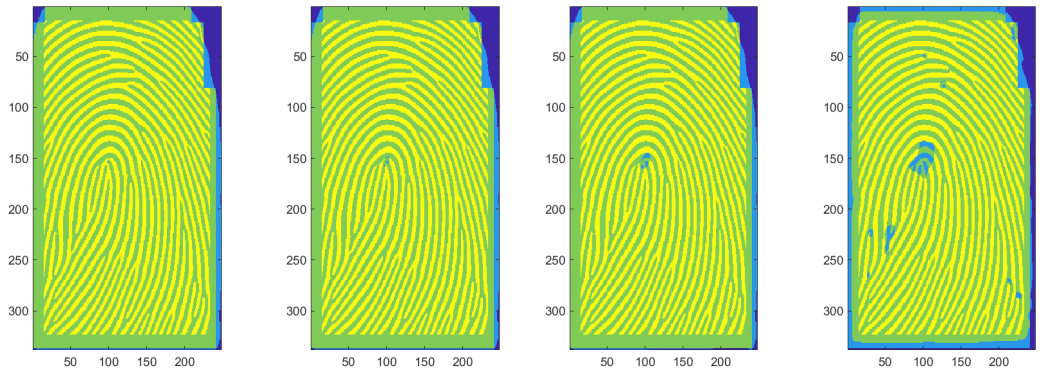
Let us first examine the code to see what we are doing in this section.

```
%Binarized and Segmented Fingerprints
threshold=0.1; %quality threshold
figure;
subplot(121)
imagesc(binI1+mask1+(relI1>threshold))
binI1(relI1<threshold)=0;
inv_binI1 = (binI1 == 0);
```

We can see that we stablish a threshold (which we are told to modify) and then plots an image that is an addition of the images:

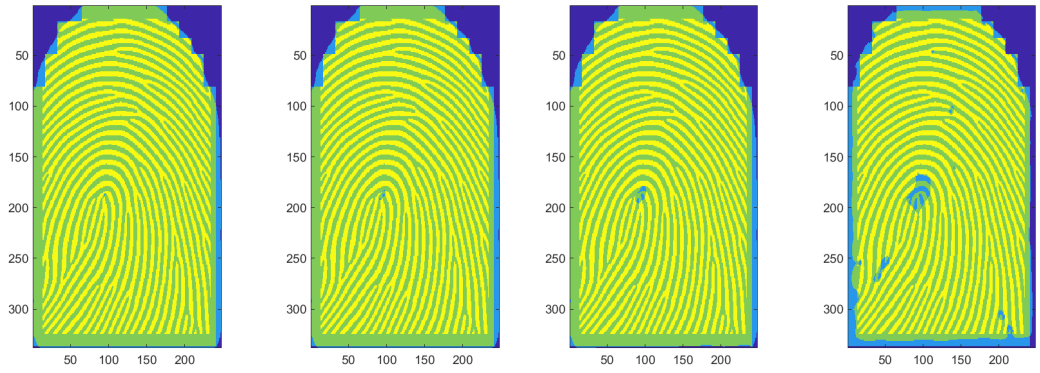
- Enhanced binarized image
- The mask that contains the ridge-like regions of the image (returned by `testfin`)
- The reliability of the orientation, selecting the pixels where the reliability is above the selected threshold

The result of the code is shown in Figures 5 and 6.



(a) 0.1 (b) 0.3 (c) 0.6 (d) 0.9

Figure 5: Different thresholds for the first fingerprint example.



(a) 0.1 (b) 0.3 (c) 0.6 (d) 0.9

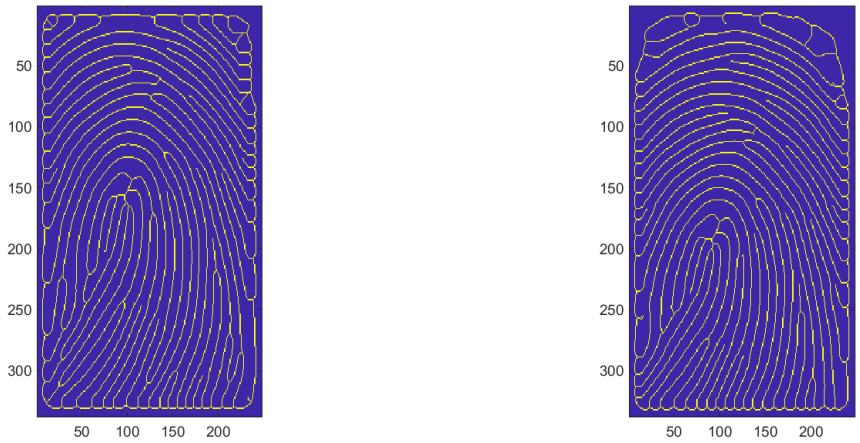
Figure 6: Different thresholds for the second fingerprint example.

We can observe in both cases that, as the threshold is increased, the reliability around the loops decreases and the yellow mask dissapear in both cases. This phenomenon also happens in other points that, until this point of the lab, we have not detected as important points.

Exercise 5

5.1 Execute the code for generating the Fingerprint Skeleton and the Minutiae Extractor. Paste the resulting images for the original values $window = 5$ and $margin = 5$.

Firstly, we show in Figure 7 the skeletons obtained by the function `bwmorph`.

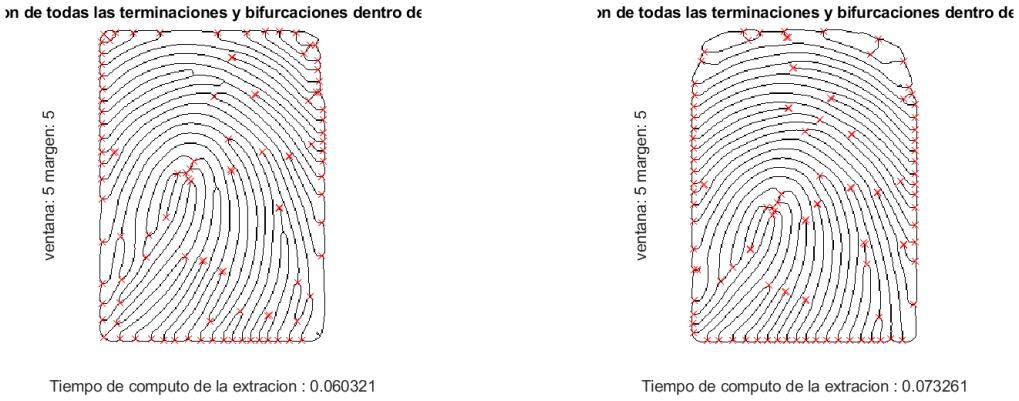


(a) Example1_1 (b) Example1_2

Figure 7: Skeletons of fingerprints.

We can observe that the fingerprint has been transformed in order to make its shape a bit more *squared* and we have only kept a think line for each ridge-line that we had in the beggining.

We now show in Figure 8 the minutias extracted for each of the skeletons, with the default parameters $window = 5$ and $margin = 5$.



(a) Example1_1 (b) Example1_2

Figure 8: Minutiae of example fingerprints.

As we can see, many *minutias* have been found at the edges of the skeleton.

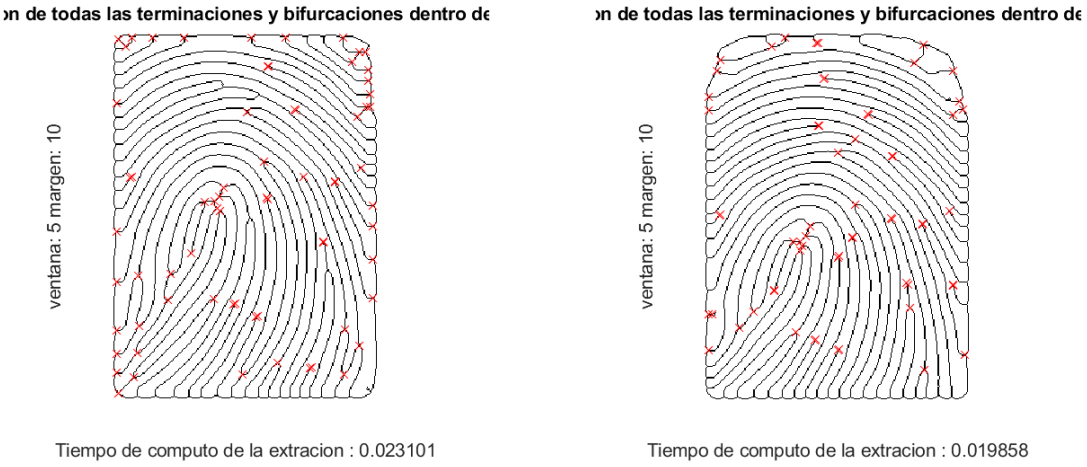
5.2 Search heuristically by looking at the images for the optimal values of parameters window and margin. Paste the resulting images with your optimal parameters and justify your decision.

Ideally, we would like to obtain parameters $window, margin$ such that:

1. There are not many (or none at all) minutias in the edge of the skeleton.
2. All the minutias are found inside the skeleton, not skipping any of them.

As we have mentioned,in the previous case we found many minutias at the edges, so initially we would like to **increase the margin**. Also, if we observe the images in Figure 8, many minutias have been ignored inside the edges of the skeleton, due to the big window size, so we will **decrease the window size**.

Firstly, we increase the margin to $margin = 10$, to check how the minutiae extraction works. We can see the result in Figure 9



(a) Example1_1 (b) Example1_2

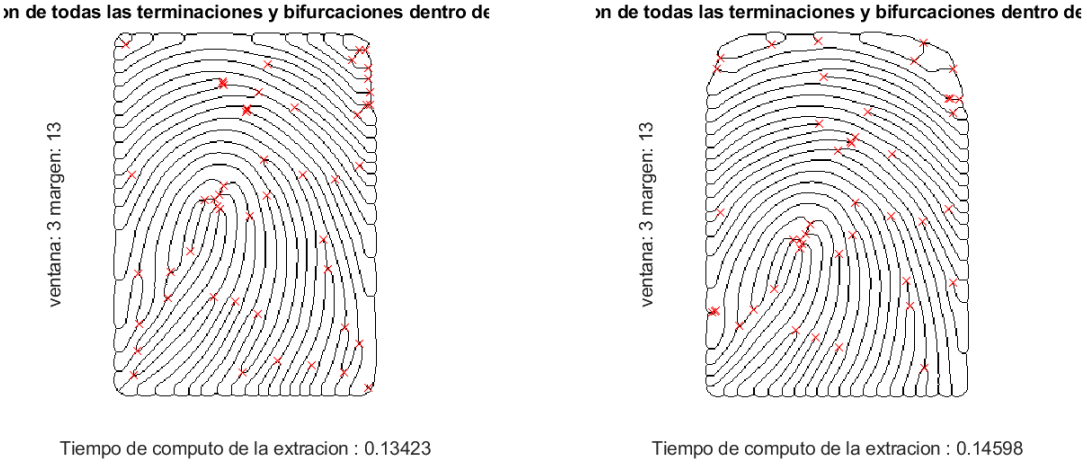
Figure 9: Minutiae of example fingerprints.

As we can see, there are still some minutias found at the edge of the skeleton. Thus, there is still capability to increase the margin and still capture possible minutias situated at the edges. It is needed to make a **balance** between how many *edge-minutias* you find and not ignoring any minutias that are close to the edges.

With this in mind, we decided to use a **margin of 13**, although other parameters in the range $\{11, 12, \ldots, 15\}$ could have been used as well.

Now, we try to **capture all the minutias** by reducing the window size. Since the initial window size is 5 and the window size must be **odd**, there is no other option than to use $window = 3$.

With these considerations, we decided to use a **window size of 3**. The results obtained with the selected hyperparameters (window,margin) are the ones presented in Figure 10.



(a) Example1_1 (b) Example1_2

Figure 10: Minutiae extraction using optimal parameters: $window = 3$, $margin = 13$.

As we can observe, all the minutias have been found (at least the ones that are easily visible) and there are not many minutias at/near the edges.

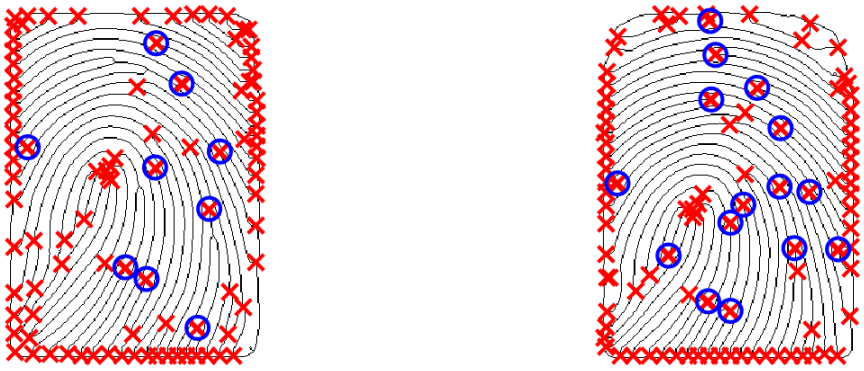
Exercise 6

6.1 Execute the code corresponding to the Minutiae Validation for window=5 and margin=5. Paste the resulting image including the minutiae extracted (red crosses) and validated (blue circles) of both fingerprints.

As we executed the code, we noticed that if we executed the given code, the images were printed complemented (black was white and vice-versa), so we added the following code to invert the images:

```
thin1_comp = imcomplement(thin1)
thin2_comp = imcomplement(thin2)
```

Now, we show in Figure 11 the result of the validation using the default parameters $window = 5, margin = 5$.



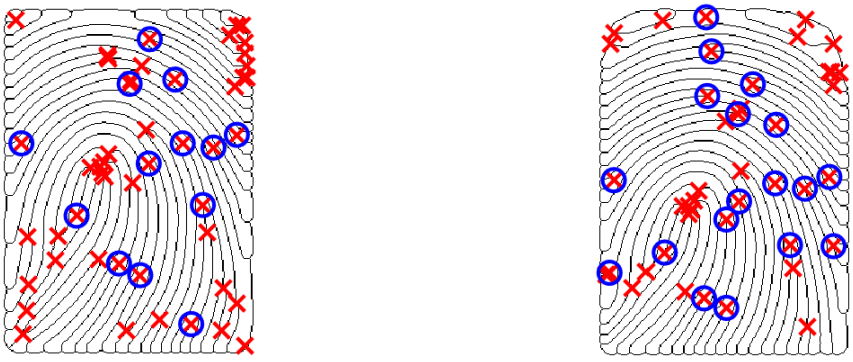
(a) Example1_1 (b) Example1_2

Figure 11: Validation of minutiae extraction using parameters: $window = 5$, $margin = 5$.

As we can observe, many minutias have not been validated. The parameters that we have chosen in this case are not the optimal determined in the previous exercise.

6.2 Execute the same code but with the optimal values of parameters window and margin. Paste the resulting image below.

The output obtained from the same code with the optimal values is the one presented in Figure 12.

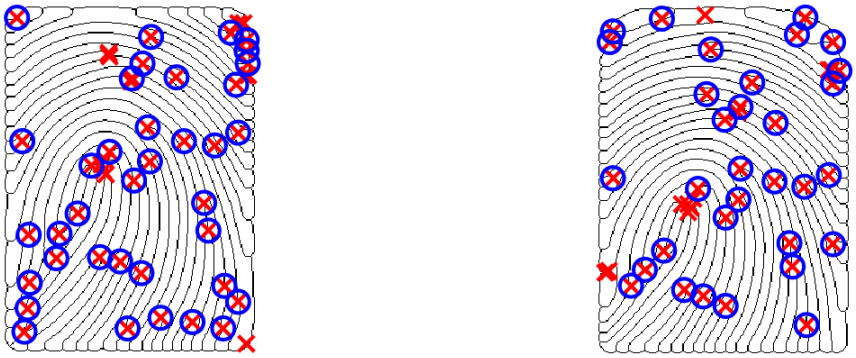


(a) Example1_1 (b) Example1_2

Figure 12: Validation of minutias extraction using the optimal parameters.

The result is similar to the previous one in terms of the validated minutias. There are a few more minutias validated in the case where the optimal $window$ and $margin$ are used.

As we did our experimentation with the parameters, we figured out that there is one more aspect that has to be taken into account: the **window size of the validation**. Until now, it was pre-set to one, but we can make it bigger to see what is the result.



(a) Example1_1 (b) Example1_2

Figure 13: Validation of minutias extraction using optimal parameters and validation window size equal to 7.

As we observe in Figure 13, when we increased the window size of the validation window, much more minutias were validated as we expected. This has its negative part too, since in a real problem it would have to be finetuned to obtain only the relevant minutias. However, we will postulate that the more information about valid minutias, the better, so for the **optimal case**, we now use the parameter $validation\_window = 3$.

6.3 Do you think it is a good idea to include the Minutiae Validation module? Justify your opinion.

In general, it **is good idea** to include the Minutiae Validation module, since sometimes all extracted minutias might not be relevant and we should be able to make a selection of the most important ones, which is performed in the validation selection. Furthermore, the extra window-size parameter in the validation stage all this preprocessing and extraction, gives us an extra degree of freedom in case we needed it for a Fingerprint detection problem.

Extra Exercise

With all the previous exercises done correctly you can obtain a mark up to 7 points out of 10. Extra work: If you want to obtain a mark up to 10 points out of 10 you should complete the following: In folder "/ddbb" you have 20 fingerprint images. 19 of them are labeled with the subject identity (e.g., H0001), and 1 is Unknown. Search for the identity of the Unknown fingerprint in the set of 19 labelled reference fingerprints. You can use the provided code "identification_1_19.m" as basis. Paste here the resulting ranked list of scores of the Unknown fingerprint with respect each one of the 19 reference fingerprints.

7.1 Solution

The first thing that we will do is to complete the file `extract_minutiae` in order to encapsulate all the useful code that we have done in `main.m`. We will set the name of many variables to `nil`, since they will be returned by the created functions but will not be used. The code of the created function is the following:

```
function [valid_x,valid_y]=extract_minutiae(I,minutiae_window,minutiae_margin,val_window)

  % Enhance
  [nil, nil,nil,nil,relI,enhI] =  fft_enhance_cubs(I, -1);
  % Quality map
  [nil, binI, nil, nil, nil] =  testfin(enhI);
  % Binarize and Segment Fingerprint
  threshold=0.9;
  binI(relI<threshold)=0;
  inv_binI = (binI == 0);
  %Fingerprint Skeleton
  thin =  bwmorph(inv_binI, 'thin',Inf);

  %Minutiae Extractor
  [minutiae, minutiae_x, minutiae_y,nil]=extraction(thin,minutiae_window,minutiae_margin);
```

```
    %Minutiae Validation
    [nil, valid_x, valid_y, nil]=validation(thin,minutiae,val_window);

end
```

Using this code, in this exercise we only have to extract minutias for each of the candidates and compare them with the unlabeled fingerprints to detect which one has the highest coincidence. The code that performs this action is too long to be included in this memory, so it is attached as a file: `identification_1_19.m`. Until this point, we have made several choices in the parameters $window, margin$ and $validation\_window$. In this last exercise, we test with a few possible combinations of those, to check if our previous decissions help in this problem.

7.1.1 Initially proposed parameters

Firstly, we test the **first parameters** that were proposed, which are $window = 5, margin = 5, validation\_margin = 1$ (we have selected this validation margin)

	1	2	3	4	5	6	7	8	9	10
Hough	0.21	0.16	0.18	0.16	0.12	0.12	0.17	0.22	0.16	0.17
	11	12	13	14	15	16	17	18	19	
	0.12	0.18	**0.53**	0.2	0.16	0.18	0.19	0.12	0.18	

Table 1: Results for the initial parameters.

As we can see, the selected subject is the **number thirteen**, with a coincidence of 53%, doubling the score of all the rest of the prints.

7.2 Optimal parameters with small validation window

We decided to firsly test the **optimal parameters** (which are $window = 3, margin = 13$) with a validation window size of 1. The results are the following:

	1	2	3	4	5	6	7	8	9	10
Hough	0.19	0.15	0.2	0.15	0.12	0.14	0.17	0.23	0.18	0.18
	11	12	13	14	15	16	17	18	19	
	0.12	0.2	**0.55**	0.2	0.16	0.2	0.2	0.13	0.2	

Table 2: Results for the optimal parameters and validation window equal to 1.

As we can see, using the optimal parameters gives us a little improvement in the percentage of coincidence using Hough in the correct case, and the rest of the cases vary: some increase its percentage of coincidence (by no more than 2%) and some of them decrease this Hough coefficient. However, it may be happening what we saw when we used no validation: there could be many minutias that are not very relevant that are coincident in all the fingerprints.

We still postulate that the unknown fingerprint belongs to subject 13.

7.2.1 Optimal parameters with bigger validation window

Now, we test the postulated **optimal parameters**, which are $window = 3, margin = 13, validation\_margin = 3$

	1	2	3	4	5	6	7	8	9	10
Hough	0.25	0.23	0.3	0.23	0.19	0.18	0.22	0.26	0.23	0.19
	11	12	13	14	15	16	17	18	19	
	0.2	0.26	**0.67**	0.24	0.2	0.23	0.24	0.18	0.24	

Table 3: Results for the optimal parameters.

This case increases the Hough scores in all cases, but really highlights a case: the number thirteen. We confirm in this case that the optimal parameters are providing us with good quality information about the suspects, increasing the percentage of coincidence.

7.2.2 Checking smaller margin

In a previous exercise, we chose a margin of 13 since using it removed many minutias from the edges that we believed to be probably not useful. We now check if reducing this margin to a middle point between the initial (5) and the chosen as optimal (13) turns into higher coincidence with the subjects or not. We execute the identifiation code with parameters $window = 3, margin = 8, val\_window = 3$ and obtain the following results.

	1	2	3	4	5	6	7	8	9	10
Hough	0.32	0.27	0.29	0.28	0.27	0.28	0.29	0.29	0.31	0.26
	11	12	13	14	15	16	17	18	19	
	0.27	0.29	**0.79**	0.33	0.29	0.26	0.29	0.27	0.28	

Table 4: Results using middlepoint window.

Again, we obtained that the subject with more coincidence is the **number 13**.

7.2.3 Conclusions

With all this cases tested, we can confirm with a high grade of confidence that the unknown fingerprint corresponds to **suspect 13**.

It has to be mentioned that as we can see in Figure 14 the hyperparameters:

$$window = 3, \quad margin = 8, \quad val\_window = 3$$

were the ones that resulted in higher coincidence in our number 13 candidate.

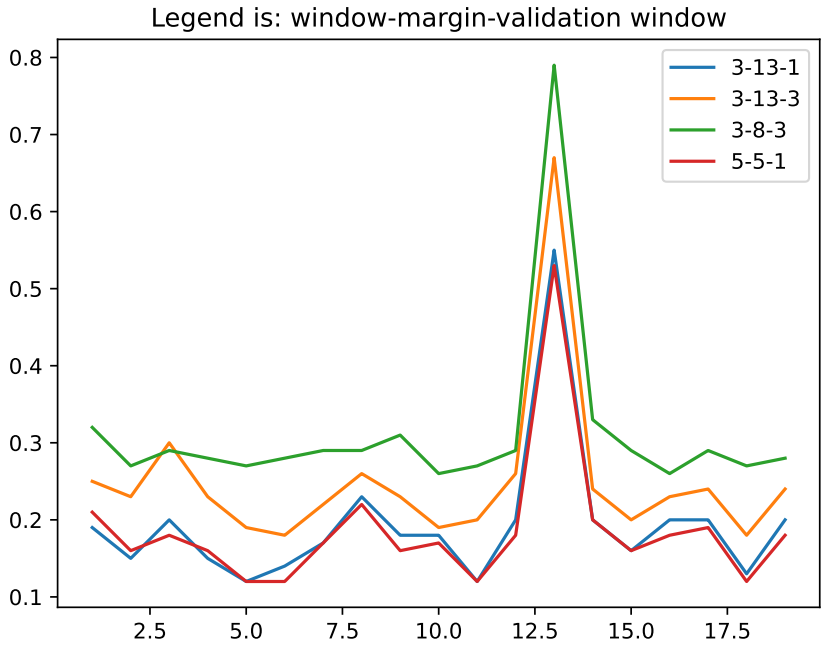


Figure 14: Comparison of Hough coefficients for the different hyperparameter configurations.

However, as we can see, when a set of parameters increases the coincidence percentage with our main subject, it *proportionally* does the same for all the subjects. In this case, is not very relevant since it is clear that the coincidence percentage is **quite higher** for subject number 13. However, in other identification

problems where the coincidences between the fingerprints or the minutias of the subjects are more similar, the set of parameters should be selected carefully.