Fingerprint Biometrics Lab - Report

APRENDIZAJE PROFUNDO PARA PROCESAMIENTO DE INFORMACIÓN BIOMÉTRICA

Luis Antonio Ortega Andrés
Antonio Coín Castro

Exercise 1

a) *Copy here the two fingerprint images provided as examples (`example1_1` and `example1_2`).*

(a) Example1_1 (b) Example1_2

Figure 1: Original fingerprints.

b) *How many macro-singularities do you observe in each fingerprint?*

We can see in Figure 1 that there is only one macro-singularity in each fingerprint. Specifically, we observe a **loop** in each one of them, around the centre of each image. There are no *deltas* or *whorls*.

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

The result of marking each of the loops in shown in Figure 2. To avoid cluttering the images, we only point out the zones in which the "U" shape is most notable.

(a) Example1_1

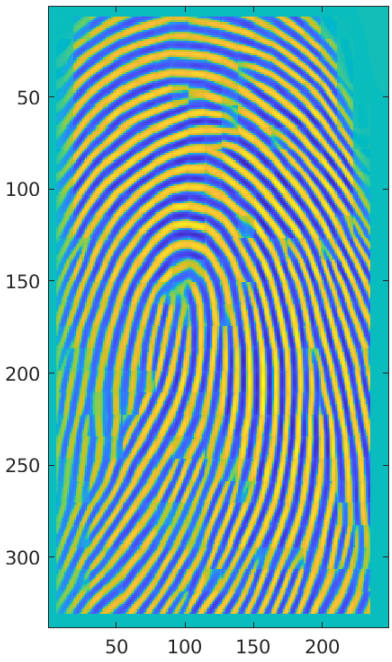
(b) Example1_2

Figure 2: Macro-singularities (loops).

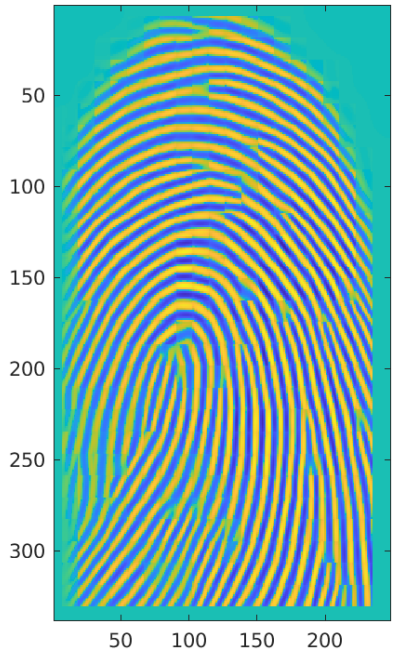
Exercise 2

a) *Execute the provided code for Fingerprint Enhancement and paste the resulting image here.*

The resulting image is shown in Figure 3.



(a) Example1_1



(b) Example1_2

Figure 3: Fingerprint Enhancement.

b) *What differences do you observe with respect to the original fingerprints?*

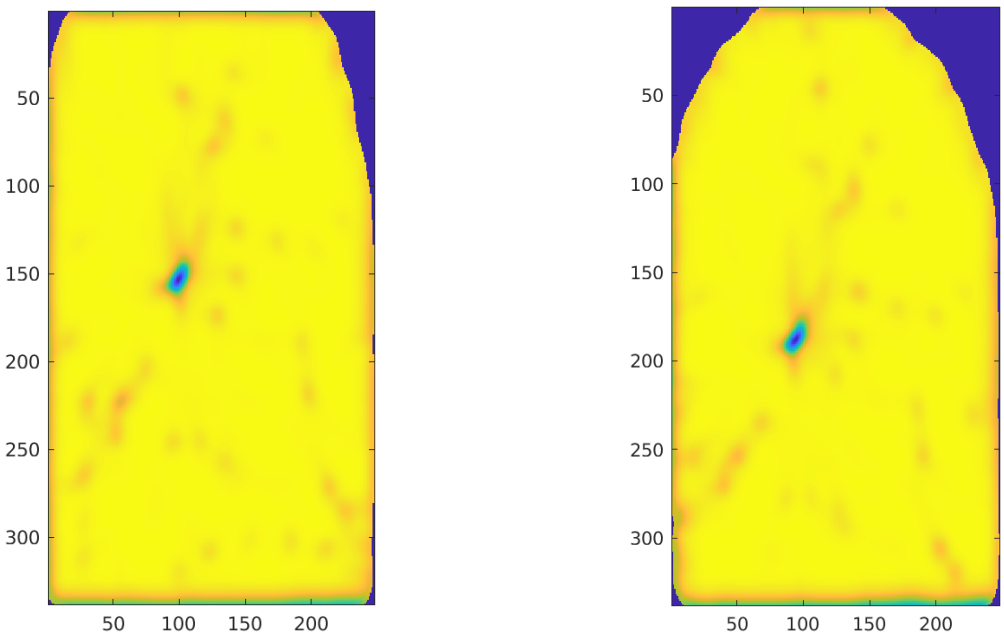
Fingerprint enhancement facilitates the identification of ridge-valley structures and hence the features of each fingerprint. More precisely, we obtain a representation where any intermittent stroke of a ridge

is replaced with a continuous one, and also contrast enhancement is performed. As we can see in Figure 3, the differences with respect to Figure 1 are that the ridge lines are continuous, smoother, and can be distinguished more clearly from the background.

Exercise 3

a) *Execute now the code for Quality Maps, and paste the resulting quality maps.*

The resulting quality maps are shown in Figure 4.



(a) Example1_1 (b) Example1_2

Figure 4: Fingerprint Quality Maps.

b) *What is the range of values for these quality maps?*

Inspecting each matrix by hand, we can see that for both images the minimum value (shown in dark blue) is 0.0, while the maximum values (shown in yellow) are 0.9991 and 0.9988 for `Example1_1` and `Example1_2`, respectively. Given this, we can assume that the range of values for the quality is $[0, 1]$.

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

Inspecting the given source code, we conclude that the heat map displayed in Figure 4 corresponds to a measure of the reliability of the computed orientation for each ridge (by means of the function `ridgeorient`). We can think of this quantity as a measure of how well the computed orientation resembles the real outline of the ridge curves.

Taking a look at our examples, there is one region in both fingerprints where the reliability is low (blue), which corresponds to the sharpest point of the loop. This phenomenon is to be expected, given that those points are precisely the ones where the orientation changes abruptly, leading to less accurate results.

Exercise 4

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.

The resulting images are shown in Figure 5. First, we plot the binarized fingerprint, that is, a representation where only black-and-white is used (to distinguish the ridge lines from the valleys). Then, we superimpose a mask that delimits the zone of what we can consider a fingerprint (versus the background). Finally, we plot the reliability mask only for the points that exceed a certain reliability threshold.

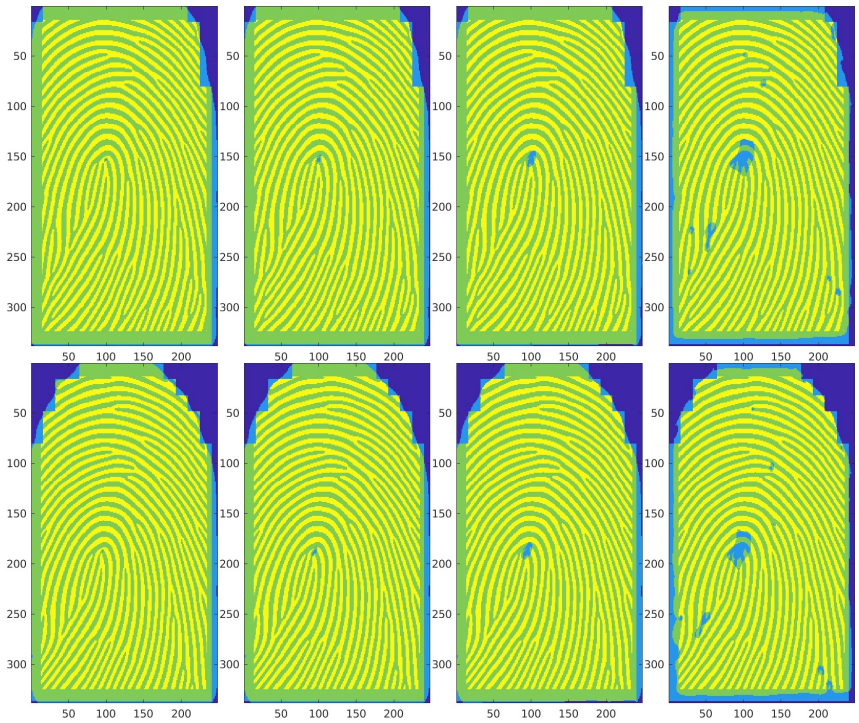


Figure 5: Binarized and Segmented fingerprints with thresholds $0.1, 0.3, 0.6$ and 0.9. The first row corresponds to `Example1_1` and the second one to `Example1_2`, while the threshold increases from left to right.

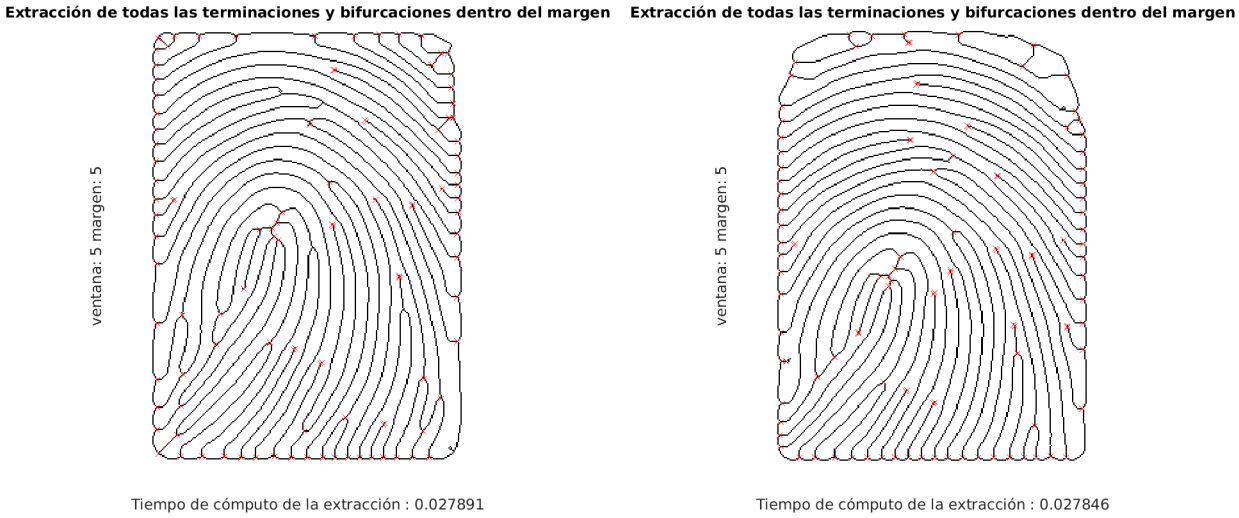
We can see that as we increase the threshold, more and more points are missing from the final mask (especially the ones around the loops, where the reliability is lower). In the final part of the code, we also "turn off" the points that don't pass the threshold test.

Exercise 5

a) *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.*

The Fingerprint Skeleton code aims to make the ridge lines thinner (1px), so that its analysis for extracting minutiae becomes easier. The Minutiae Extractor then tries to find all relevant minutiae, using a window of a specified size and ignoring the points within a certain distance of the margin of the image. The results are shown in Figure 6.

We notice that using the default parameters there are several mintuae that are not correctly extracted (in particular, some bifurcations in the first fingerprint are not detected). In addition, we can see that there are many marks in the border of each image, which can be suboptimal, since these minutiae are somewhat artificial and caused by the fact that we don't have a complete image (only a section of the frontal projection of the fingerprint).



(a) Example1_1 (b) Example1_2

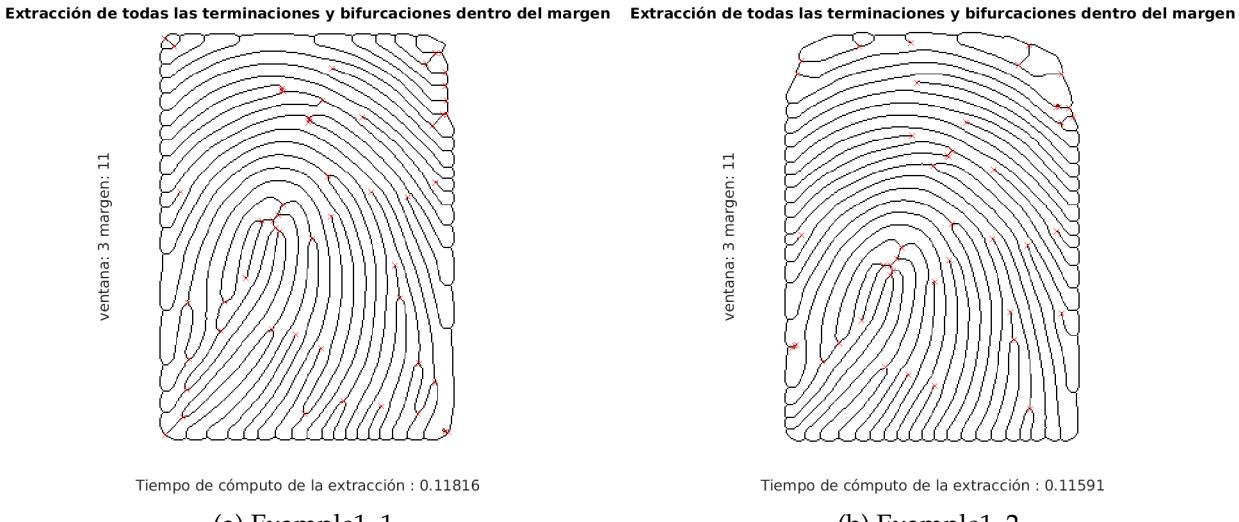
Figure 6: Fingerprint Skeleton and Minutiae Extractor output with default parameters.

b) *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.*

To solve the problems mentioned above, we search heuristically for the best combination of window and margin values. On the one hand, we found that decreasing the window to 3×3 provided better results (all visible minutiae were detected).

On the other hand, increasing the margin to a higher value alleviated the problem of wrongly detecting minutiae on the border, since the feasible region is moved away from the border. However, we have to be careful not to increase the margin too much, because then we could lose some genuine minutiae that are close to the border. We found that setting the margin to 11px was a good compromise.

The resulting images are shown in Figure 7.



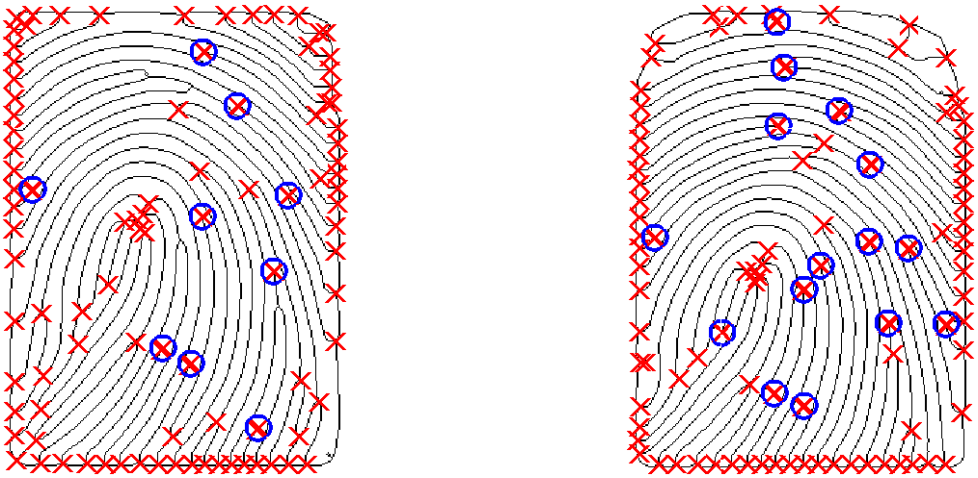
(a) Example1_1 (b) Example1_2

Figure 7: Fingerprint Skeleton and Minutiae Extractor output with optimal parameters.

Exercise 6

a) *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.*

The minutiae extracted and validated with the default extraction parameters are shown in Figure 8.

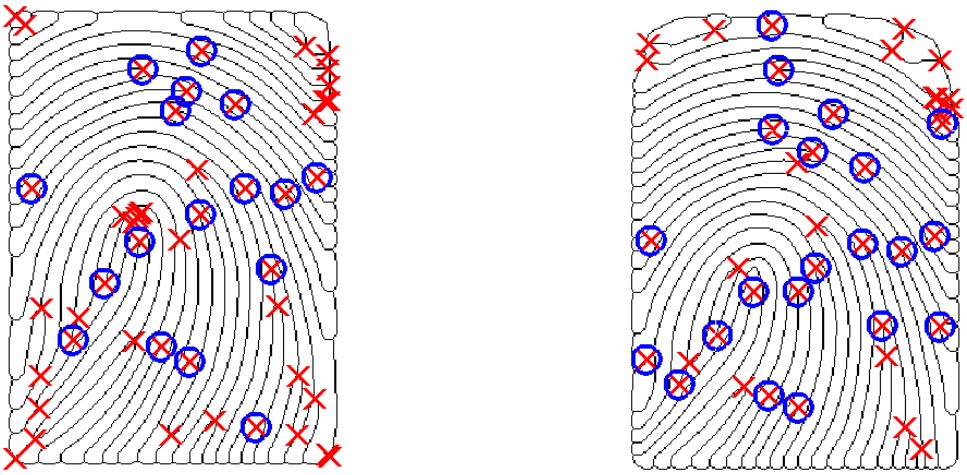


(a) Example1_1 (b) Example1_2

Figure 8: Minutiae Validation output with default parameters.

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

The minutiae extracted and validated with the optimal extraction parameters are shown in Figure 9.



(a) Example1_1 (b) Example1_2

Figure 9: Minutiae Validation output with optimal parameters.

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

It is a good idea, because it allows us to retain only those minutiae that are actually relevant for comparing two fingerprints. It isolates the minutiae so that they are easier to compare (reducing noise), and it also lets us correct the possible (non-)minutiae detected on the border.

Extra Exercise

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.

In order to compare each fingerprint in the database with the unknown print, we will extract (and validate) the minutiae of each fingerprint and apply the Hough transform to compute the scores between each one of them and the unknown one. To this end, we use the code already implemented in `Hough.m` and we write a function called `extract_minutiae`:

```
function [valid_x,valid_y]=extract_minutiae(I, ew, em, vw)
  ext_window=ew;
  ext_margin=em;
  val_window=vw;
  [~,~,~,~,~,enhI] = fft_enhance_cubs(I, -1);
  [~, binI,~,~, I1_enhaced] = testfin(enhI);
  inv_binI = (binI == 0);
  thin = bwmorph(inv_binI, 'thin', Inf);
  [minutiae, minutiae_x, minutiae_y,~]=extraction(thin,ext_window,ext_margin);
  [valid, valid_x, valid_y,~]=validation(thin,minutiae,val_window);
end
```

We have tried a total of 4 combinations of the parameters `window` and `margin` for extraction, and `window` for validation. We decided to modify this last parameter to see how it would influence the result.

First attempt. The first combination of parameters chosen was the same as in Exercise 6 (the optimal extraction parameters and the default validation window):

$$\{\texttt{extract\_window} = 3,\ \texttt{extract\_margin} = 11,\ \texttt{validation\_window} = 1\}$$

We obtained the following scores for each of the 19 fingerprints on the database, shown in Table 1.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Score	0.1860	0.1525	0.2045	0.1538	0.1220	0.1361	0.1739	0.2318	0.1758	0.1764
	11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	
Score	0.1333	0.2000	**0.5500**	0.1978	0.1587	0.1975	0.2040	0.1269	0.2022	

Table 1: Hough scores for each fingerprint in the first attempt.

As we can see, the fingerprint number 13 is the winner, obtaining a coincidence score of 55%, way ahead of the rest of the prints.

Second attempt. Next, we decided to slightly increase the validation window:

$$\{\texttt{extract\_window} = 3,\ \texttt{extract\_margin} = 11,\ \texttt{validation\_window} = 3\}$$

The scores obtained are displayed on Table 2. In this case, the score of fingerprint 13 increases to 65%, while the scores for the rest of the prints remain relatively low.

	1	2	3	4	5	6	7	8	9	10
Score	0.2598	0.2295	0.2645	0.2153	0.1964	0.1951	0.2237	0.2614	0.2241	0.2033
	11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	
Score	0.2044	0.2352	**0.6542**	0.2362	0.2039	0.2289	0.2311	0.2000	0.2393	

Table 2: Hough scores for each fingerprint in the second attempt.

Third attempt. Finally, we thought we would try to decrease the value of the extraction margin, since we had increased it quite a bit from the default value:

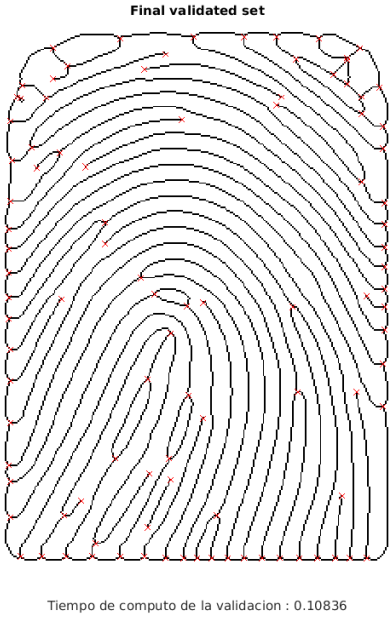
$$\{\texttt{extract\_window} = 3,\ \texttt{extract\_margin} = 7,\ \texttt{validation\_window} = 3\}$$

The scores obtained are shown in Table 3. As we can see, with this combination of parameters we obtain an even higher score for fingerprint 13, which reaches 76% of coincidence. The downside of this approach is that the scores for the rest of the fingerprints are also higher, but they still remain low enough in comparison to the winner.

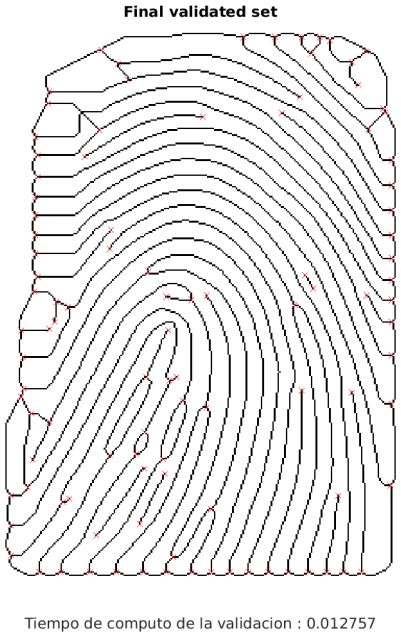
	1	2	3	4	5	6	7	8	9	10
Score	0.3237	0.2616	0.2857	0.2816	0.2636	0.2800	0.2872	0.2978	0.3061	0.2515
	11	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	
Score	0.2735	0.2857	**0.7666**	0.3377	0.2827	0.2685	0.2876	0.2649	0.2789	

Table 3: Hough scores for each fingerprint in the third attempt.

For this third attempt (which was the best), we show in Figure 10 the validated minutiae for both the unkown fingerprint and image number 13, where we can see the coincidences. Based on the previous discussion, we are quite sure that **the unkown fingerprint corresponds to the 13th fingerprint on the database**.



(a) Unknown fingerprint



(b) Fingerprint 13

Figure 10: Extracted and validated minutiae in the third attempt.