Fingerprint Recognition and Evaluation System

Sapienza University of Rome
Biometric Systems , 2016-2017 Term Project
Prof. Maria De Marsico

Students:
Huseyn Mammadov , 1772138
Farid Yusifli , 1772848
Tahmasib Asgarli , 1734428

Contents

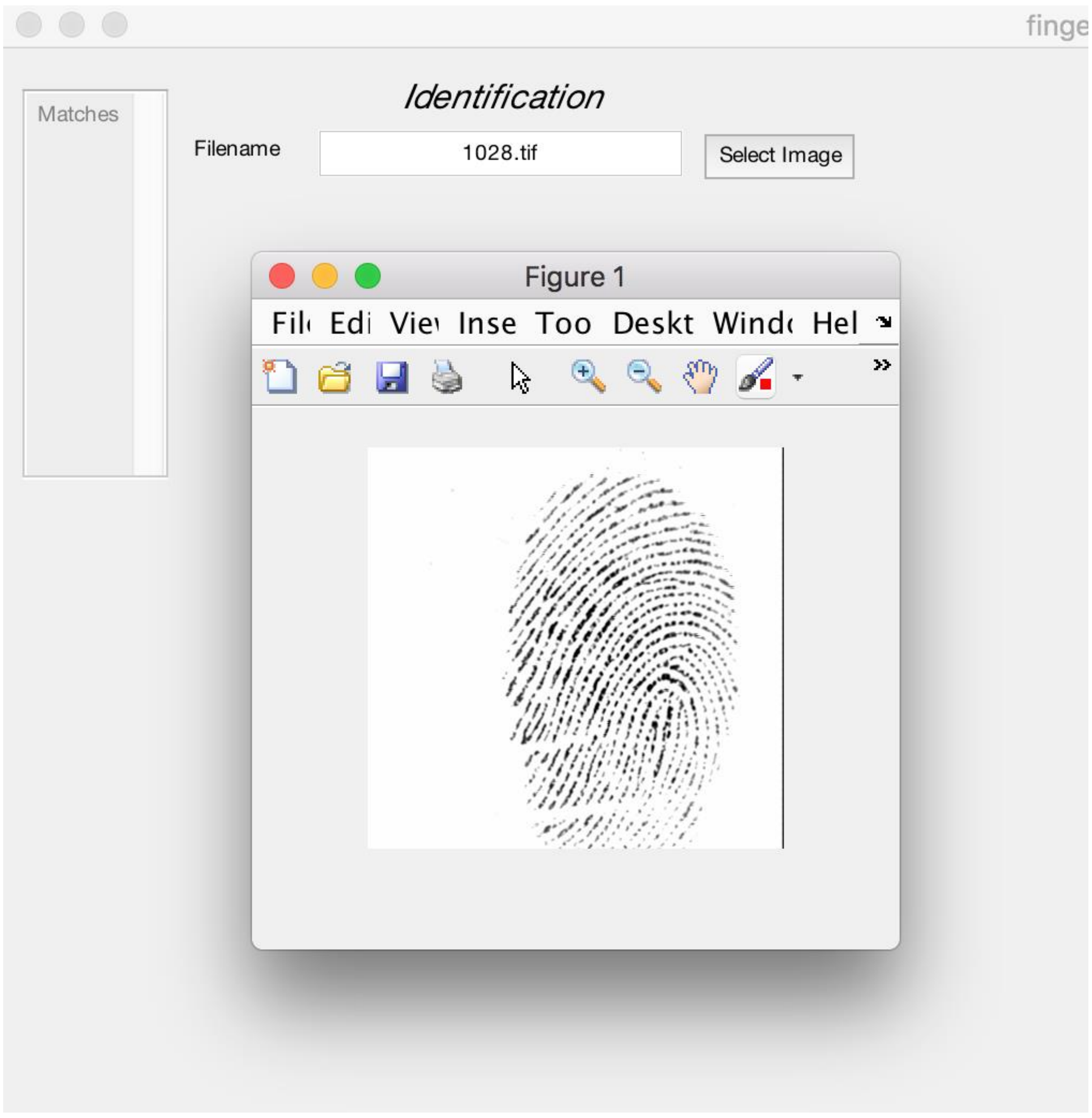
Introduction

Fingerprint recognition is one of the most well known biometrics, and it is used very often for security reasons and biometric solution for authentication on computerized systems. On the literature there are lots of methods to recognize fingerprint. The reasons for fingerprint recognition being so popular are the ease of obtaining and acceptance when compared to other biometrics. Fingerprint recognition has been present for a few hundred years. These days, the technology in this area has reached a point where the cost of purchasing a fingerprint scanner is very affordable. For this reason, these systems are becoming more widespread in a variety of applications. In this application we tried to recognize fingerprints with real finger image captured by camera. In order to do so we should transform our image to appropriate format. We start to apply filters to reach this fingerprint's features. At the end we will evaluate the performance of our application.

Gray Scale Image

After person's fingerprint taken, we need to convert original fingerprint in greyscale image which is also known as black and white image and each pixel has only one value which is the intensity of the pixel the value may range from 0 to 255 Greyscale images are composed exclusive of shades of gray, varying from black at the weakest intensity to white at the strongest intensity. In biometric systems greyscale image is used to identify, compare and match fingerprint. Because colour does not give any useful information for ridges. We need to reduce noisy unnecessary information to determine ridges. The gray scale levels of a fingerprint sensor are the number of gray shades produced for every pixel. 256 levels of gray is the de facto standard supported by most available fingerprint sensors today and results in using one byte per pixel. Other important aspects for gray scales are the uniformity and dynamic range. Uniformity assesses the gray level difference in a neighbouring pixel area while dynamic range measures the gray values actually used from the maximum of typically 256. Requirements depend on the application.

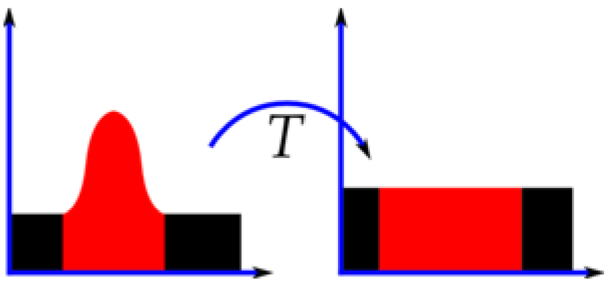
In our system we will input image and we will apply grayscale filter.



In the identification area of our application we input fingerprint picture, after that it will show us grayscaled fingerprint.

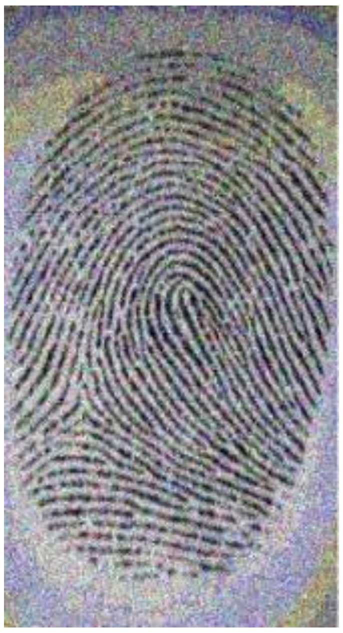
Histogram Equalization

Histogram equalization is a general process used to enhance the contrast of images by transforming its intensity values. As a secondary result, it can amplify the noise producing worse results than the original image for certain fingerprints.
If we want to adjust and transform this image matrix form, we should calculate some series of variables. Through this adjustment, the intensities can be better distributed on the histogram. This allows for areas of lower local contrast to gain a higher contrast.



Histograms of an image before and after equalization.

End of these calculations we obtain new transformed image matrix form. This matrix form also represent image which applied histogram Equalization

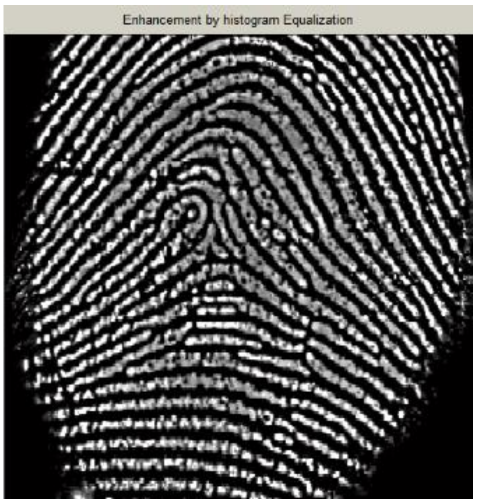


Original fingerprint **After histogram equalization**

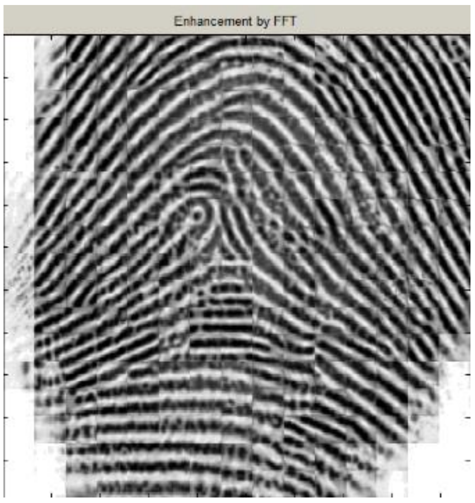
Enhancement Using Fourier Transform

We divided our fingerprint picture into small blocks (32x32 pixels) and perform Fourier Transform based on:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) \times \exp\left\{- j2\pi \times \left(\frac{ux}{M} + \frac{vy}{N}\right)\right\}$$



Enhancement by Histogram



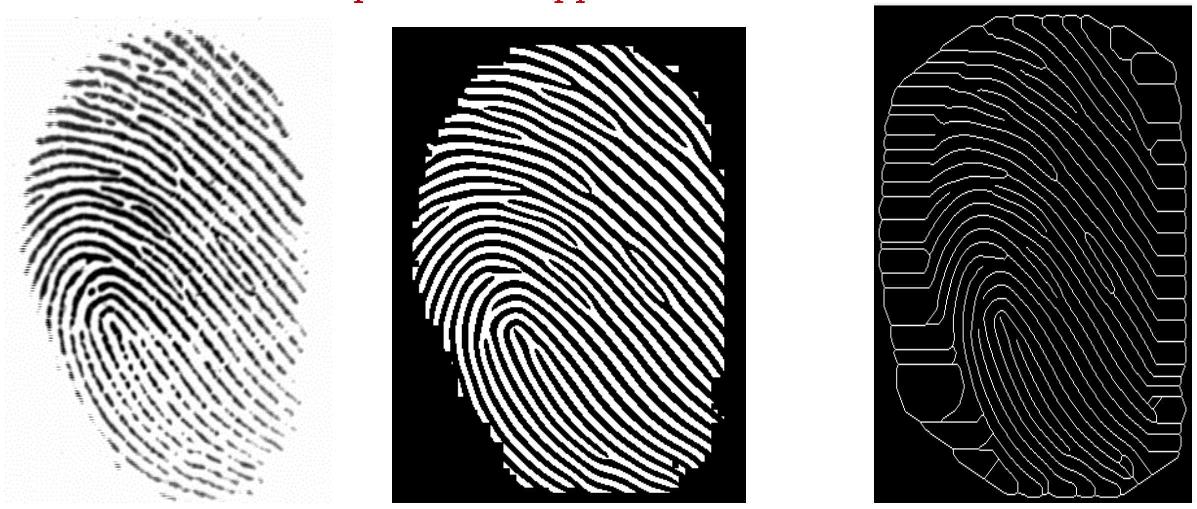
Enhancement by FFT

The image after FFT enhancement has the improvements to connect some falsely broken points on ridges and to remove some false connections between ridges.

Binarization and Thinning

Our image is ready to applying binary image to reach clearer image and more clear ridges to obtain features of fingerprint. It is a really important step in the process of ridge extracting, since the prints are taken as grayscale images, binarization transforms the image from a 256-level to a 2-level image. Typically, a finger pixel is a value of 1 while a background pixel is 0. We can use grayscale image to binary image with using thresholding method. On thresholding method we are considering only black or white if image value is above threshold it represent white and if image value is below the threshold, it's represent black. When a threshold is applied to an image, all pixel values are compared to the input threshold. In Matlab you can write ***B* = *I* > *t*** "t" is threshold value. According to this condition if it's true means any pixel values below the threshold are set to zero and it represent white, if its values false, it means greater than the threshold and we set to 1 and it represent black. By the end of this process, all pixel values within the image are either zero or one, and the image has been converted to binary format, here main difficulty is to choosing a correct value for the threshold. If we set threshold too low, then the resulting binary image will mostly containing white pixels. Conversely, if we set threshold too high, the image will contain a lot of undesired black pixels. So we should choose threshold value carefully.

Output of our application

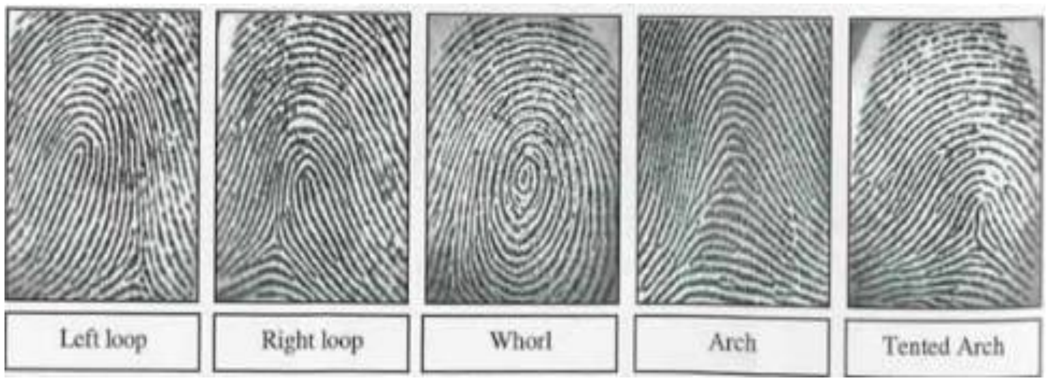


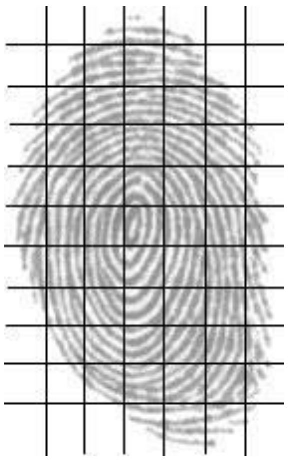
original image **binary image** **thinned image**

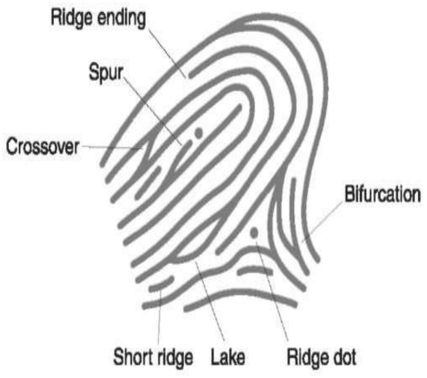
Thinning is elimination of the redundant pixels of ridges till the ridges are just one pixel wide. An iterative, parallel thinning algorithm is used. In each scan of the full fingerprint image, the algorithm marks down redundant pixels in each small image window and finally removes all those marked pixels after several scans. Following thinning, the location and orientation of the minutiae should still be the same as in the original image to ensure accurate estimation of their locations.

Useful Features to Identify Fingerprint Ridges

Fingerprints have some specific features that we use this features to recognize fingerprints.







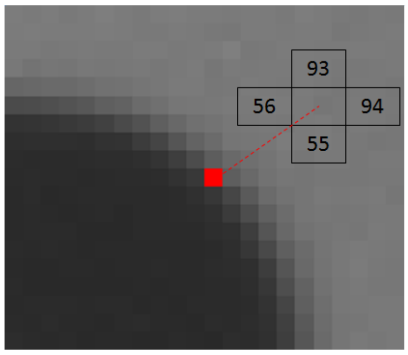
We used ridgeSegment function to identify ridge regions of fingerprint. Function identifies ridge regions of a fingerprint image and returns a mask identifying this region. It also normalizes the intensity values of the image so that the ridge regions have zero mean, unit standard deviation. This function breaks the image up into blocks of size block size x block size and evaluates the standard deviation in each region. If the standard deviation is above the threshold it is deemed part of the fingerprint. After we reached ridges masks, we can detect ridge of fingerprints according to this mask.

Ridge Orientation

Minutiae detection and matching are heavily dependent on ridge orientation estimation and image segmentation, since ridge orientation is inevitably used for detecting, describing and matching minutiae, and correct image segmentation helps to avoid detecting spurious minutiae in low quality image regions. There are many methods for ridge orientation estimation in literatures. Most of them are based on gray-scale relation- ship between pixels. Comparing each pixel block of size 2 × 2 with four types of edge templates to estimate the coarse orientation of the block, and then averages the orientation within a larger region. Computing gray consistency along 16 directions, respectively, at each pixel. And the orientation of the best consistency is taken as the ridge orientation for the corresponding pixel. Binaries the pixels into ridge pixels and non-ridge pixels, and evaluates ridge orientation by computing the consistency of pixel type, ridge and non-ridge, along a discrete number of orientations.

Ridge Orientation with Gradient Vector

The most popular method for ridge orientation estimation is the gradient-based method. It computes the gradient vector at each pixel. Along the direction of the gradient vector, the pixel-intensity changes most rapidly and the length of the vector indicates the speed of this change: larger length, faster change. It is the evidence for gradient-based method that the hypothetical edge, which is parallel to ridge orientation for good ridge structure, at an edge pixel is orthogonal to its gradient vector direction. A gradient vector can be computed for every pixel an image. It's simply a measure of the change in pixel values along the x-direction and the y-direction around each pixel. A gradient vector can be computed for every pixel an image.



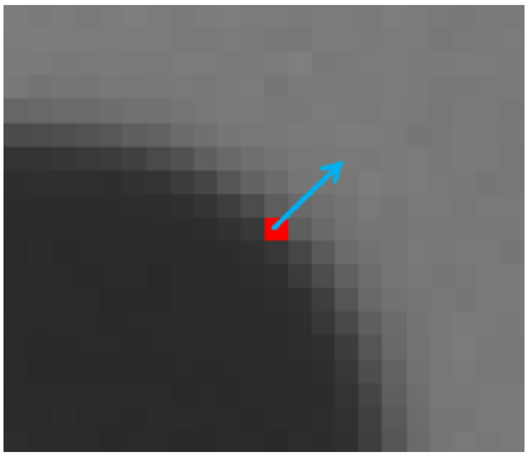
It's simply a measure of the change in pixel values along the x-direction and the y-direction around each pixel. 93 - 55 = 38 in the y-direction. Putting these two values together, we now have our gradient vector.

$$\text{Magnitude} = \sqrt{(38)^2+(38)^2} = 53.74$$

$$\text{Angle} = \arctan\left(\frac{38}{38}\right) = 0.785 \text{ rads}$$

$$= 45 \text{ degrees}$$

After this calculation We can now draw the gradient vector as an arrow on the image.

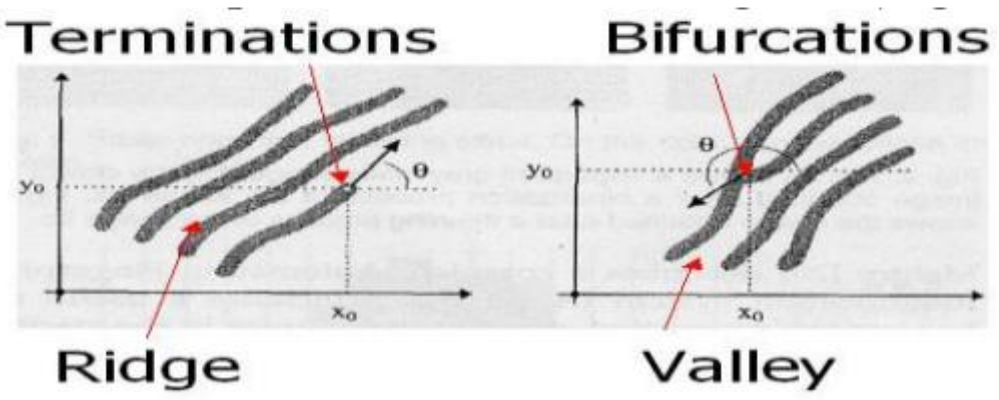


Ridge Frequency

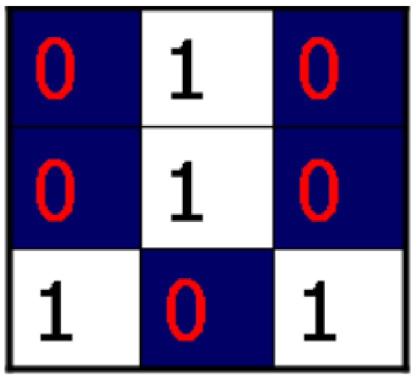
We need frequency value for each block in order to determine ridges of fingerprint and reach good result for ridge filer method. This is done by considering blocks of the image and determining a ridge count within each block by a call to FREQEST (in Matlab). The method calculating frequency for every block according to min and max wavelength if it cannot find satisfied frequency for block between min and max wavelength values, it set zero for this block.

We will use this frequency for ridge filter, we will look frequency of (rows cols) and we will filter according to this rule freq(row , column) > 0, It will be one of the condition while we are filtering the blocks as a ridges or not. Fingerprint identification and recognition methods are applied for laboratory conditions or for preconditions with many restrictions. One of these restrictions is the determination of average difference between finger print ridges. This distance is also called as ridges frequency. In addition, this parameter is an element of many fingerprint image quality enhancement methods. For this reason, an automatic method of given fingerprint ridges frequency recognition will be considered.

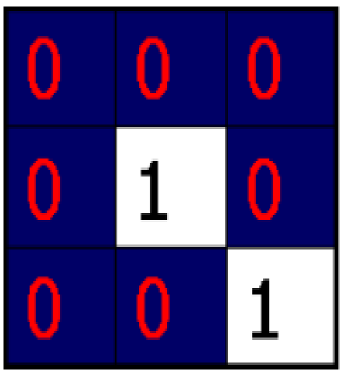
Minutiae Features



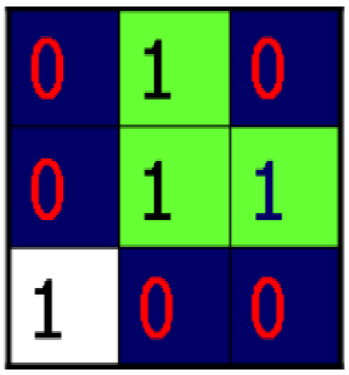
After the fingerprint ridge thinning, marking minutia points is relatively easy. In general, for each 3x3 window, if the central pixel is 1 and has exactly 3 one-value neighbours, then the central pixel is a ridge branch



0	1	0
0	1	0
1	0	1



0	0	0
0	1	0
0	0	1



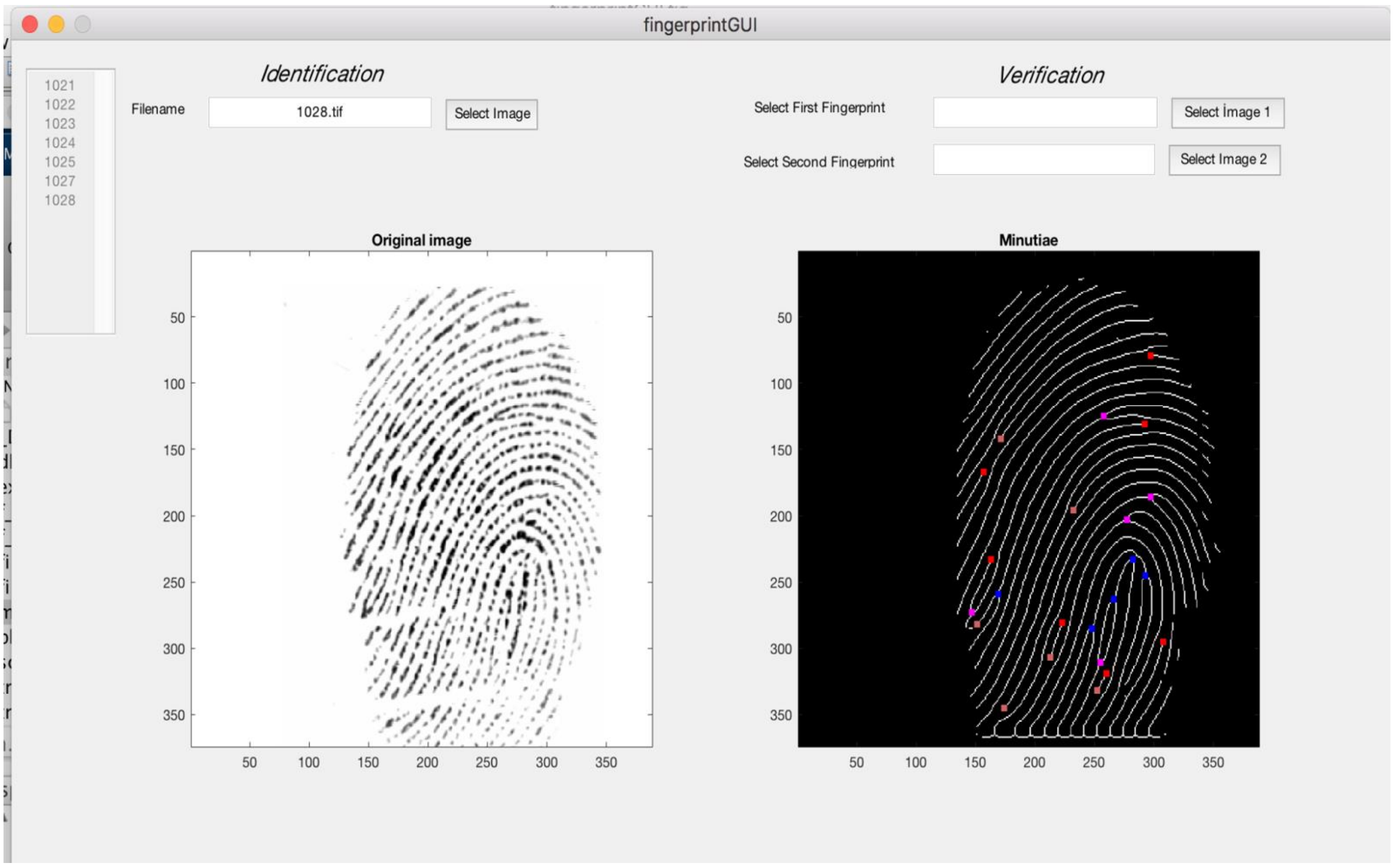
0	1	0
0	1	1
1	0	0

Bifurcation **Termination** **Triple counting branch**

If the central pixel is 1 and has only 1 one-value neighbour, then the central pixel is a ridge ending.

In order to achieve high accuracy minutiae with varied quality fingerprint images, segmentation algorithm needs to separate foreground from noisy background which includes all ridge-valley regions and not the background. Image enhancement algorithm needs to keep the original ridge flow pattern without altering the singularity, join broken ridges, clean artefacts between pseudo-parallel ridges, and not introduce false information. Finally minutiae detection algorithm needs to locate efficiently and accurately the minutiae points.
Most fingerprint minutia extraction methods are thinning based where the skeletonization process converts each ridge to one pixel wide. The end points are selected if they have a single neighbour and the bifurcation points are selected if they have more than two neighbours.

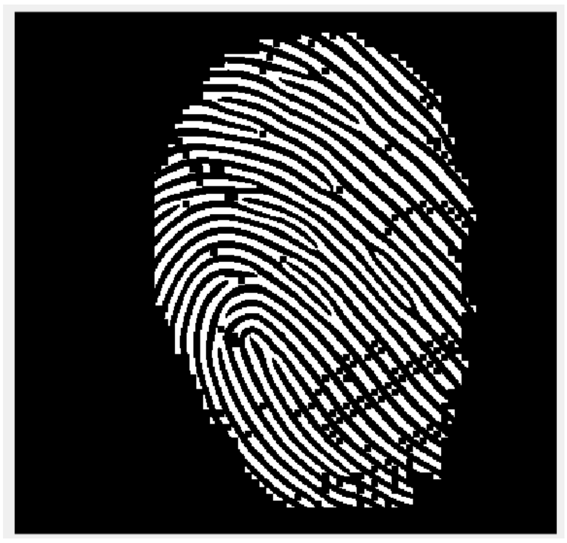
However, methods based on thinning are sensitive to noise and the skeleton structure does not conform to intuitive expectation. This category focuses on a binary image based technique of minutiae extraction without a thinning process. The main problem in the minutiae extraction method using thinning processes comes from the fact that minutiae in the skeleton image do not always correspond to true minutiae in the fingerprint image. In fact, a lot of spurious minutiae are observed because of undesired spikes, breaks, and holes.

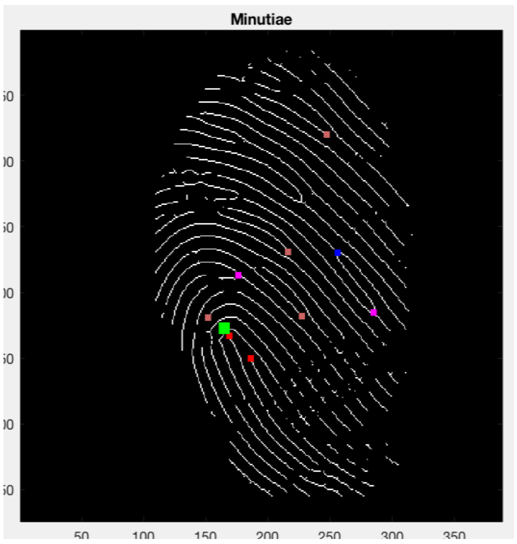


Appearance of our application after we input original fingerprint and apply filters which is described above.

Our Observations and Changes to Reach Better Result

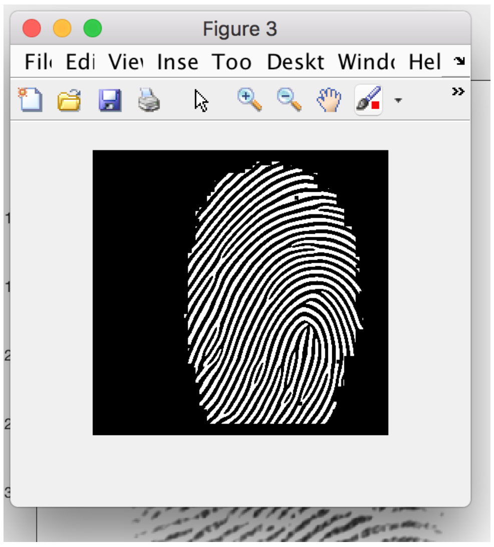
1) In ridge segmentation, fingerprint image divided into blocks. If we decrease block size we will consider smaller area and we can obtain more feature from entire image but this value quite depend on the our image and depend on our segmentation feature like ridges. Also we are using threshold to determine region (block x block) is part of the fingerprint or not. We decrease the block size because in our features we want to detect (ridges) are not consisting of big part.

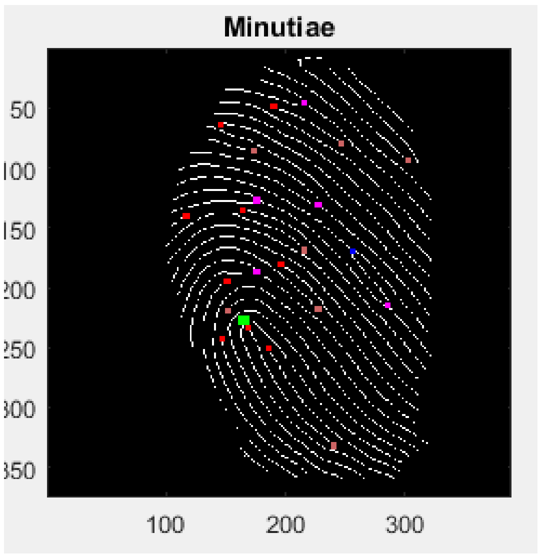




Binary Image **Thinned and Features Extracted image**

As It can be seen from the pictures they are not in good quality in order to find ridge features. So in order to obtain better result we modify block size and threshold, only after that we can get better result .





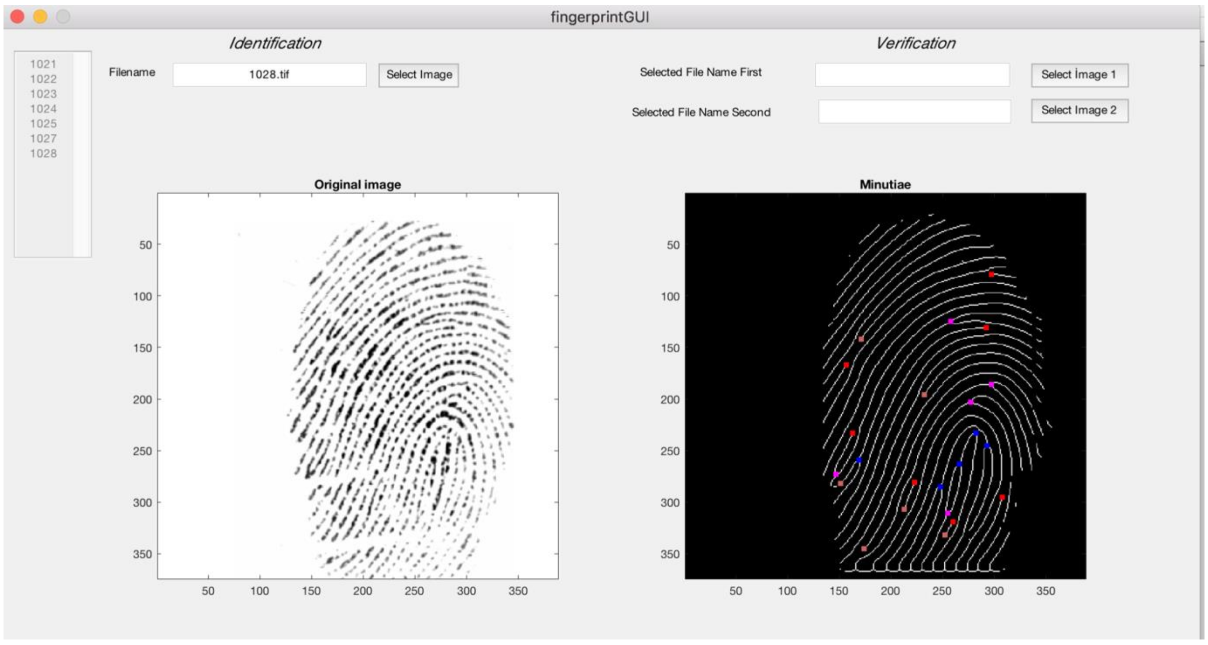
We used threshold to decide standard division of block is enough to be part of the fingerprint or not. After several tests we modify block size and threshold we obtain better result so as it can be seen from the picture after we modify variable our system recognize better the bifurcations and termination so our system now will output better result.

We tried and we reach good result with threshold=0.09 but also this with this threshold we lose some part of fingerprint image because of this threshold value. With some other threshold we loosed one bifurcation feature and it was problem because it is important feature for fingerprint recognition.

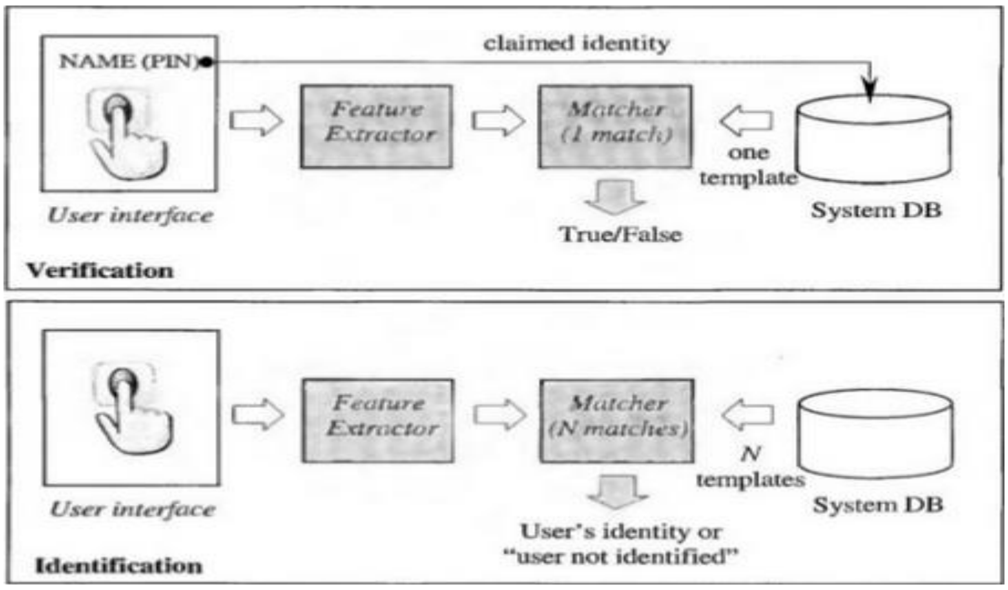
We noticed we determine bad result with block size more than 10 because our image and fingerprint ridges size is appropriate for this block size otherwise we are getting divided lines because they are not pass the threshold and its increase the false rejection rate for our segmentation.

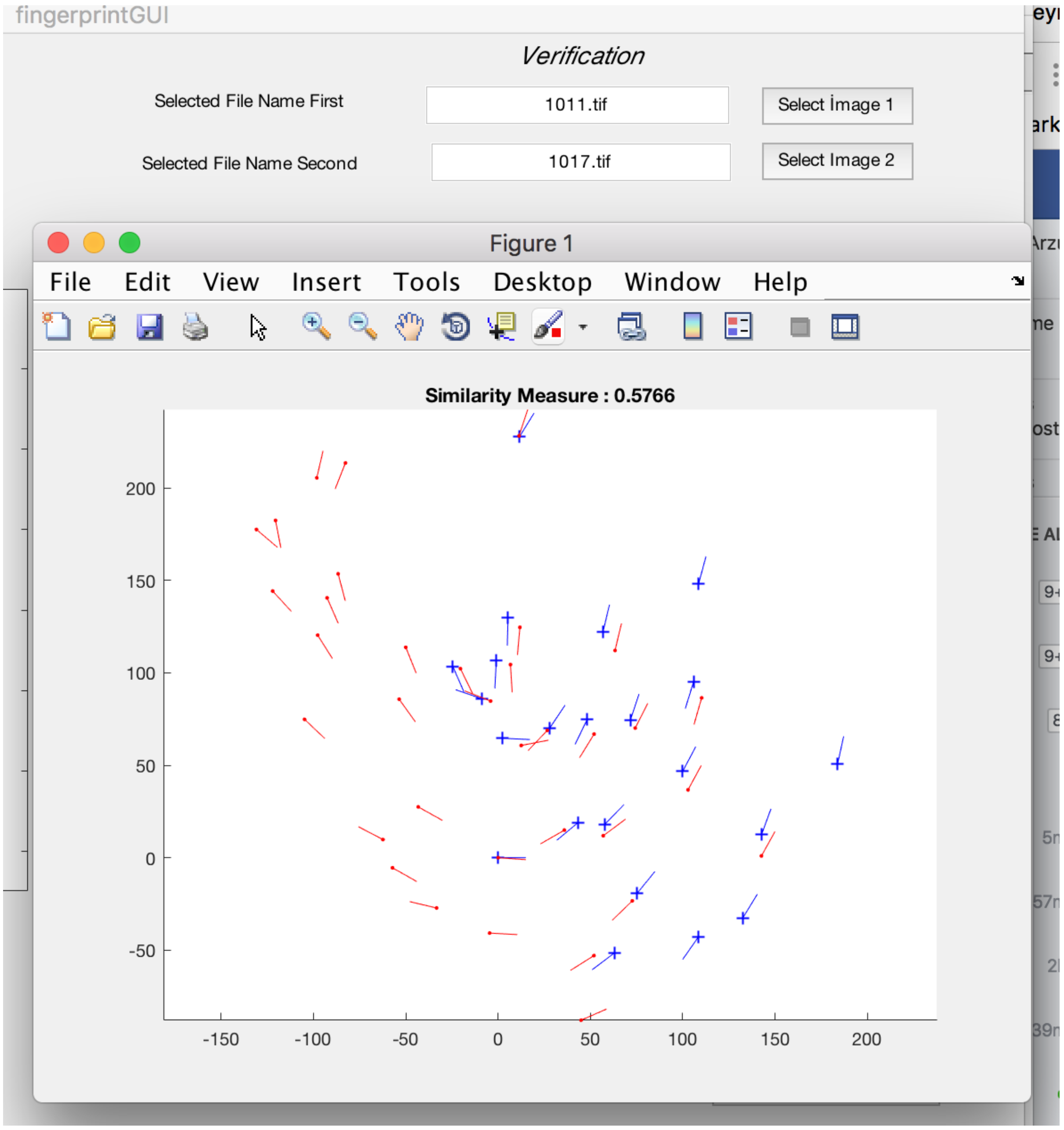
First picture represent the result with block size variable is 15 and threshold is 0.3. We can see from result is that if we reduce block size we can obtain more point but we should pay attention noise. We should find appropriate and sufficient one according to our fingerprint image. In our case we choose block size=5 and threshold 0.09.

2) After we found code for **identification** section of application and apply it in our project we create section for verification (one to one match) and we also add user interface.



Identification - Here it compare similarity of the one fingerprint through all fingerprints which is include in our database. And it will show fingerprint which is above the certain threshold.



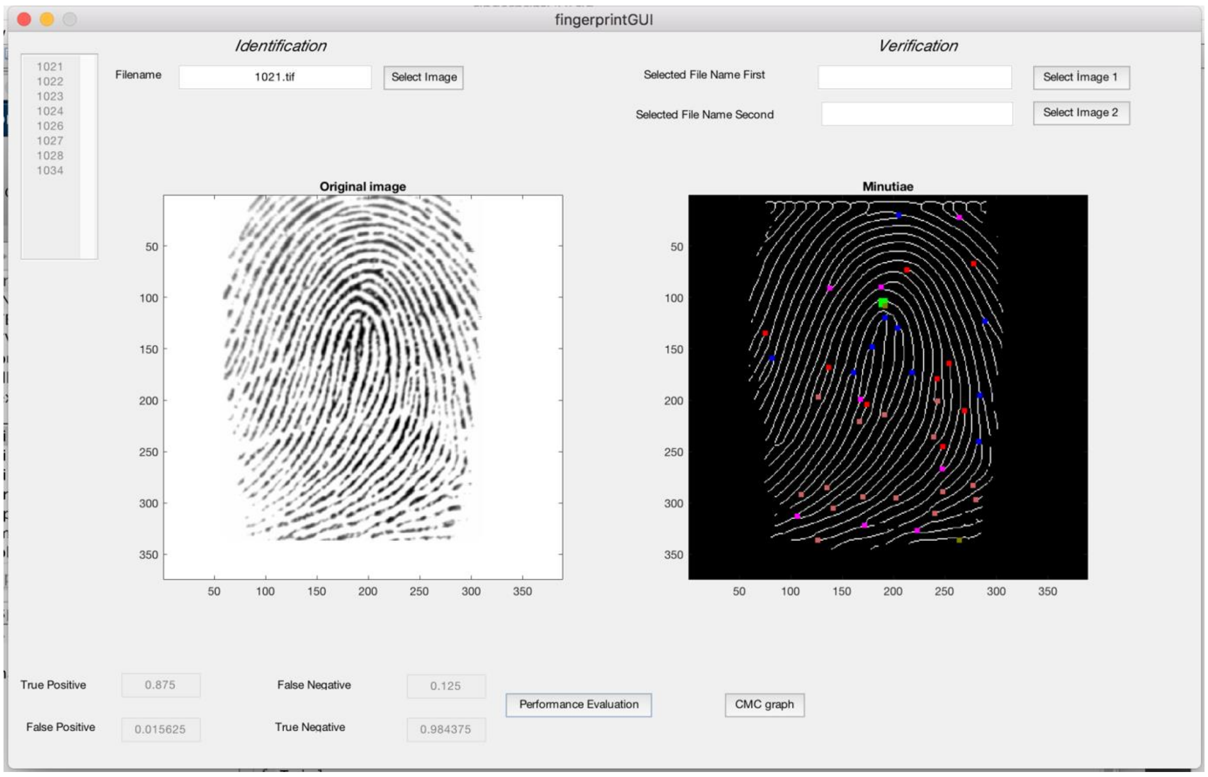


We add the feature of **verification**, so here from our set we chose two fingerprint image after we apply all feature extraction techniques which is described above we calculate the similarity measure. In this example similarity measure of fingerprint 1011.tif and 1017.tif is 0.5766.

A fingerprint matching module (identification and verification) computes a match score between two fingerprints, which should be high for fingerprints from the same finger and low for those from different fingers. Fingerprint matching is a difficult pattern-recognition problem due to large interclass variations (variations in fingerprint images of the same finger) and large interclass similarity (similarity between fingerprint images from different fingers). Interclass variations are caused by finger pressure and placement—rotation, translation, and contact area—with respect to the sensor and condition of the finger such as skin dryness and cuts. Meanwhile, interclass similarity can be large because there are only three types of major fingerprint patterns (arch, loop, and whorl).

3) We create **Performance Evaluation** in order to evaluate our application performance

A perfect biometric system would always make correct decisions, but in reality this is not possible. To evaluate how accurate a biometric system is to measure its biometric performance, many genuine (A single attempt by a user to match his/her own stored template.) and impostor(a user's template is matched against someone else's template) attempts are made with the system.


fingerprintGUI

Identification

Verification

1021
1022
1023
1024
1026
1027
1028
1034

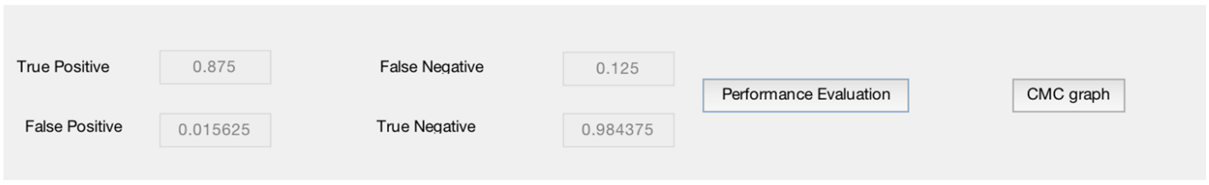
Filename: 1021.tif

Selected File Name First:

Selected File Name Second:

True Positive	0.875	False Negative	0.125
False Positive	0.015625	True Negative	0.984375

After we obtain result if we click the 'Performance Evaluation' button it will calculate True positive, False Positive, False Negative ,True Negative and show in the screen.



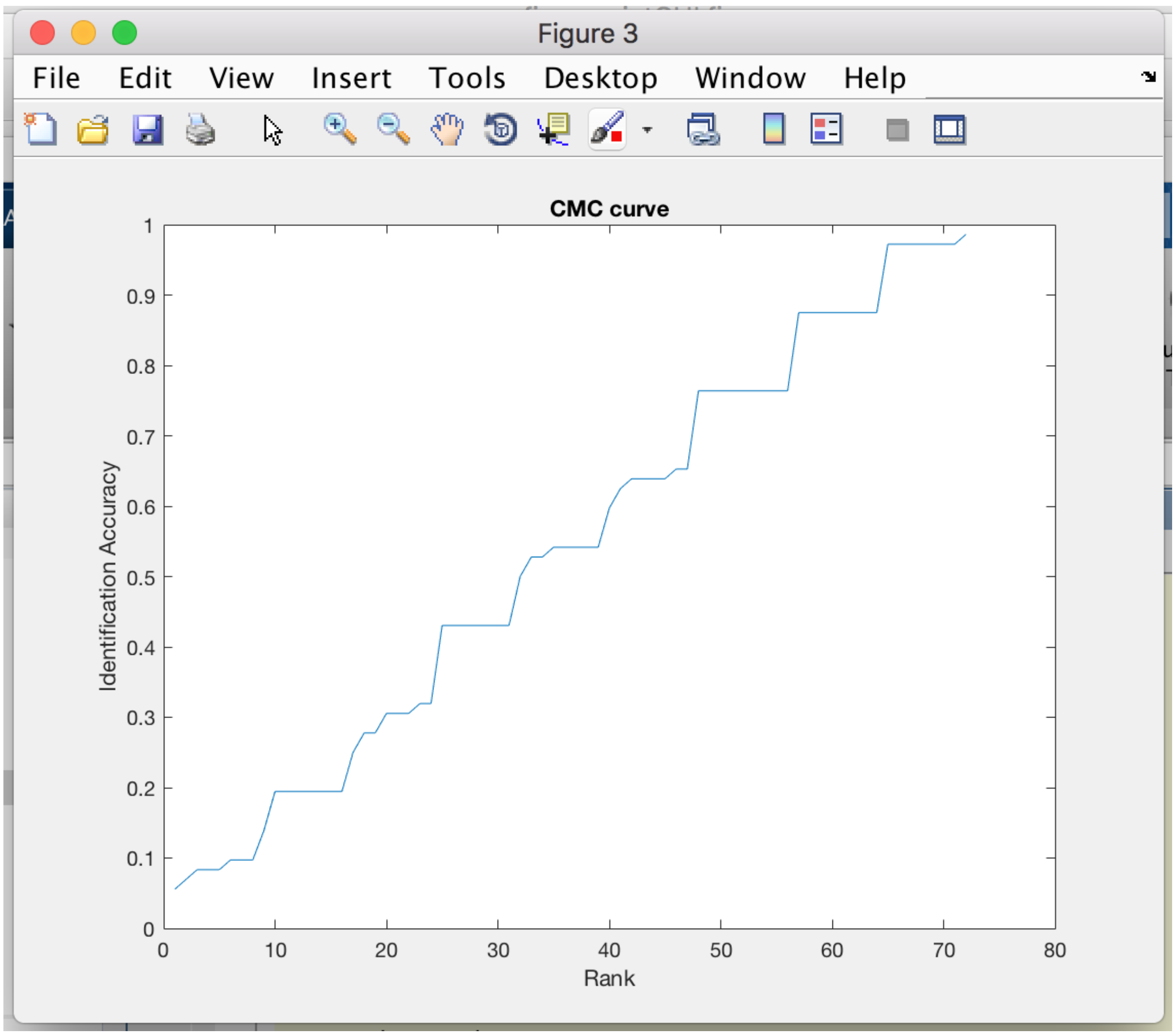
True Positive	0.875	False Negative	0.125
False Positive	0.015625	True Negative	0.984375

TP =Number of positive fingerprints of our application/Number of pictures' one fingerprint
TN = Negative result of application / exact number of negatives
FP=Number of negative that is shown as positive by application/exact number of negatives
FN = (Number of pictures' one fingerprint - Number of positive fingerprints of our application)/ Number of pictures' one fingerprint

4) In our system we focus on the closed-set problem, so for that reason we decided to use Cumulative Match Characteristic (CMC) graph in order to measure matching accuracy of our system . When you click to 'CMC graph' button it will show up the CMC graph of our application which is rank-based metric and used for summarizing performance of a closed-set identification system.

In this approach each probe biometric sample is compared against all gallery samples. The resulting scores are sorted and ranked. After that we only determine the rank at which a true match occurs.

Here we used True Positive Identification Rate (y axis) which is the probability of observing the correct fingerprint within the top K ranks (x axis)



Output of Cumulative Match Characteristic in our application

Conclusion

In the Industry Fingerprints are used very often for security reasons. Mostly systems collect fingerprint with scanners but in this project we try to collect fingerprint with camera but we notices in other to do this its required high quality camera. Otherwise it's impossible to apply filters appropriately and finding features of fingerprint. The reliability of any automatic fingerprint system strongly relies on the precision obtained in the minutia extraction process. A number of factors damage the correct location of minutia. Firstly we apply grayscale filter to the image in order to get one dimension matrix image according to intensity. We have only intensity factor of image. After that we apply histogram equalization filter in order to get better contrast from grayscale image and detect features of image. After these filters we applied binary filter with using threshold and after the binary filter we applied thinning method into image. So that the input image to the thinning stage could be made better, this could improve the future stages and the final outcome After this process, we can see the fingerprints details easier and more. Also we used Ridge Segment method to detect and recognize fingerprint features. RidgeSegment method divide image to blocks and according to threshold, it's provides us mask for features. After that we modify block size and threshold in order to get better result .We used this mask to apply ridge filter into image and we made fingerprints ridges more clearly. Minutiae detection and matching are heavily dependent on ridge orientation estimation and image segmentation. Our image is ready to detect features and matching. After choosing fingerprint and get output from our application we add performance evaluation in order to measure the accuracy. In our system we focus on the closed-set problem, so for that reason we decided to use Cumulative Match Characteristic (CMC) graph in order to measure matching accuracy of a system.

References

https://www.researchgate.net/publication/220603183_Systematic_methods_of_fingerprint_ridge_orientation_estimation_and_image_segmentation
https://www.youtube.com/watch?v=-FjHO55hvC0
http://www.cse.msu.edu/~rossarun/pubs/DeCannRossROC-CMC_BTAS2013.pdf
http://twiki.di.uniroma1.it/twiki/view/SB/CourseMaterial
https://arxiv.org/pdf/1503.01543.pdf
https://www.nist.gov/sites/default/files/documents/2016/12/06/12_ross_cmc-roc_ibpc2016.pdf
https://www.slideshare.net/sandeepkumarpanda/fingerprint-recognition-technique-35869944
https://ch.mathworks.com/
http://cdn.intechweb.org/pdfs/17747.pdf
https://cvtuts.wordpress.com/2014/04/27/gabor-filters-a-practical-overview/
https://www.researchgate.net/publication/245405636_A_new_fingerprint_ridges_frequency_determination_method
https://github.com/noureldien/FingerprintRecognition/blob/master/Presentation/COMP6206_Presentation.pdf
http://research.ijcaonline.org/volume61/number22/pxc3884809.pdf
https://www.researchgate.net/figure/221912828_fig12_Fig-15-Interpolation-step-a-the-minutiae-image-b-the-triangulated-image-c
https://precisebiometrics.com/wp-content/uploads/2014/11/White-Paper-Understanding-Biometric-Performance-Evaluation.pdf
http://www.intechopen.com/books/state-of-the-art-in-biometrics/fingerprint-matching-using-a-hybrid-shape-and-orientation-descriptor