International Journal of Computational Research and Development (IJCRD)
Impact Factor: 5.015, ISSN (Online): 2456 - 3137
(www.dvpublication.com) Volume 3, Issue 1, 2018

A STUDY ON FINGERPRINT HASH CODE GENERATION BASED ON MD5 ALGORITHM AND FREEMAN CHAIN CODE

K. Krishna Prasad* & P. S. Aithal**
* Research Scholar, College of Computer and Information Science, Srinivas University, Mangaluru, Karnataka
** College of Computer and Information Science, Srinivas University, Mangaluru, Karnataka

Cite This Article: K. Krishna Prasad & P. S. Aithal, "A Study on Fingerprint Hash Code Generation Based on MD5 Algorithm and Freeman Chain Code", International Journal of Computational Research and Development, Volume 3, Issue 1, Page Number 13-22, 2018.

Abstract:

The drastic changes in mobile and wireless based technologies and increasing number of applications and users demanded high-security concern, which leads to research on biometrics with a purpose to increase the security aspects and to minimize security threats. The current global mindset toward terrorism has influenced people and their governments to take some special actions and be extra proactive in protection or security problems. Fingerprint image and identification technology have been in life for hundreds of years. Archaeologists have exposed proof suggesting that interest in fingerprints dates to prehistory. But the modern study reveals that fingerprint is not so secured like secured passwords which consist of alphanumeric characters, number and special characters. Fingerprints are left at crime places, on materials or at the door which is usually class of latent fingerprints. We cannot keep fingerprint as secure like rigid passwords. In this paper, we discuss fingerprint image Hash code generation based on the MD5 Algorithm and Freeman Chain code calculated on the binary image. Freeman chain code extracts all possible boundaries for an image and which gives starting x and y positions as x0 and y0. Hashcode alone not sufficient for Verification or Authentication purpose, but can work along with Multifactor security model or it is half secured. To implement Hash code generation we use MATLAB2015a. This study shows how fingerprints Hash code uniquely identifies a user or acts as index-key or identity-key.

Key Words: Fingerprint Image, Fingerprint Hashcode, Authentication, Multifactor Authentication Model, Freeman Chain Code & MD5 Algorithm.

1. Introduction:

Biometrics is an investigation of checking and setting up the identity of an individual through physiological components or behavioral qualities. Despite the fact that biometric methods have been effectively connected in a vast number of true applications, outlining a decent and robust biometric system is still a testing issue. The four fundamental factors that expansion the many-sided quality furthermore, challenges of system configuration are accuracy, scalability, security, and protection [1-2]. Automatic Fingerprint Identification System (AFIS) consists of different steps like preprocessing, enhancement, segmentation, thinning, feature extraction, post-processing, minutiae orientation and alignment [3-9]. The distinctiveness of fingerprint is added forward by using ridge patterns and it has been proved that the information in small regions of friction ridges is in no way repeated. These friction ridges broaden in a human system all through the fetus level itself Fingerprint sensors or acquisition devices uses different types of sensors to take input or to get fingerprint image into the system [10].

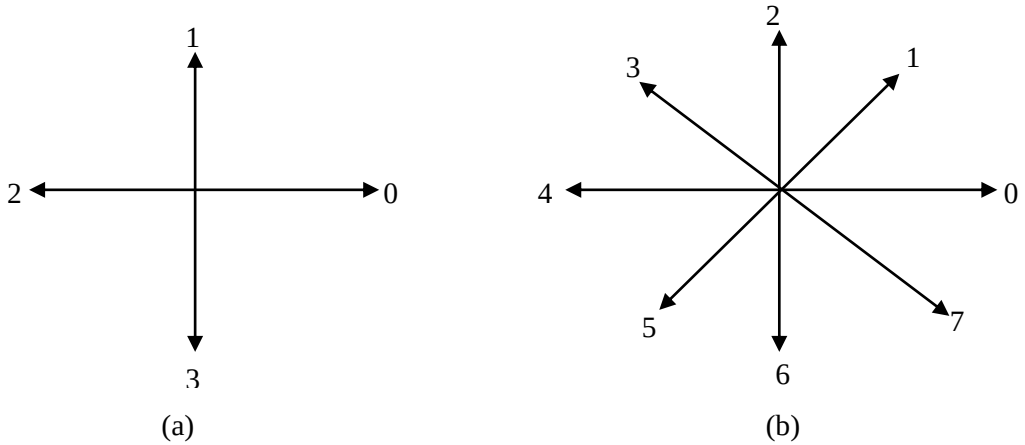

Figure 1: Neighbour Directions of Freeman Chain code

Freeman chain code is used to symbolize a boundary by means of a connected series of straight line segments of specific length and path in the predefined direction [11]. Typically this illustration is based on 4 or 8 connectivity of the segments [12]. The direction of each segment is coded or represented by the usage of a numbering format or scheme. A boundary code fashioned as a sequence of such directional numbers is called a

Freeman chain code. The chain code of a boundary relies upon at the place to begin. Running with code numbers gives a unified way to research or analyze the form of the boundary. Chain code follows the contour in a counter-clockwise manner and keeps tune of the directions as we go from one contour pixel to the following. Figure 1 (a) and 1 (b) represents, 4-connected and 8-connected neighbour of Freeman Chain code respectively.

The main drawback of 4-connectivity is that we lose the diagonal factors or pixels wherein those pixels are very useful in most of the image processing programs. So, in order to overcome the drawback of 4-connectivity here, we use 8-connectivity. In 8-connected neighbour, every code is taken into considerations. In 8-connected neighbour, the angular direction is in multiples of 450, that we have to pass or move from one contour pixel to the next. If we take into account figure 2, we can calculate freeman chain code for 4-connectivity as follows.

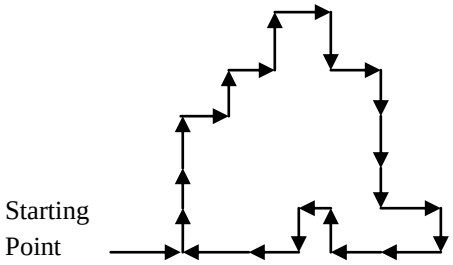

Figure 2: Freeman Chain code example for 4-connected neighbour

Freeman Chain code for Figure 2 is 1110101030333032212322. The first difference of Freeman Chain code for Figure 2 is 100313133130013303113O. The first difference of Freeman Chain coding for 4-connected neighbour is explained using Figure 3. The first difference value is calculated by counting a number of separating directions in an anti-clockwise direction from the starting point to ending point.

In this study, we make use of the MD5 Hash algorithm in order to generate Hash code. The process of the MD5 algorithm is disused below.

Input: Extracted Features
Output: Hash Code
Step-1: Attach the padded bits
Step-2: Append the length of the initial input to the result of the previous step-1
Step-3: Initialize MD buffer as A, B, C, D.
A four-word buffer (A, B, C, D) was used to evaluate the message digest. Here each of A, B, C, D is a 32- bit register
Step-4: Process message in 16-word blocks
Step-5: Finally, we get the 32-bit Hashcode as output

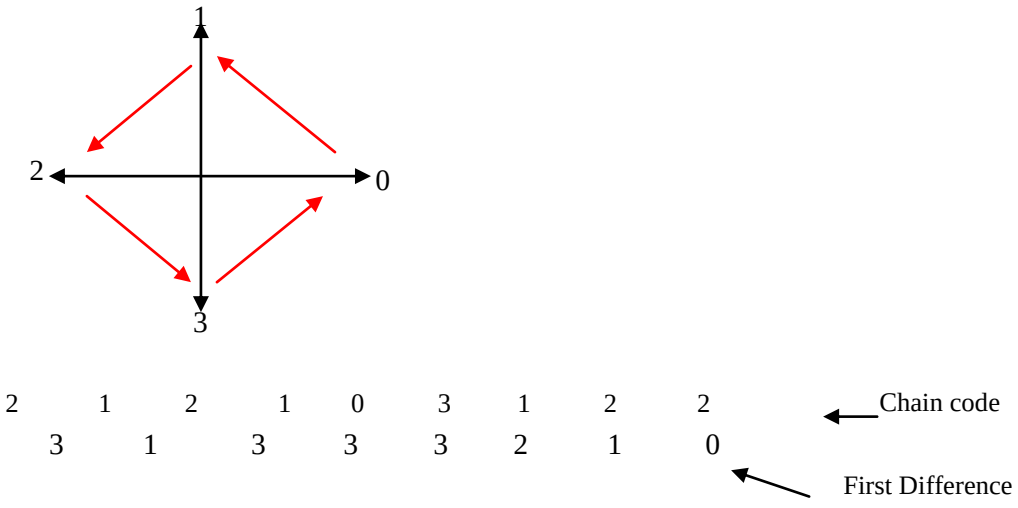

Figure 3: Freeman Chain code first Difference calculation-example

In 8-connectivity neighbour, we have code from 0 to 7. Consider a binary point as shown below.

0010
0100
0011

The Freeman Chain code first checks for non zero points, in a binary number which is 1. So in the first row starting from left 3 bit is 1. Next bit is 1 from this point three-bit position, which is in the 2nd row. Next non zero value is five-bit position from the second non-zero bit, which is in the third row. Next non zero bit is very next bit and which is also in the third row. If you observe 8 connected neighbour the direction from the first non zero bit to second is in the direction 5. From second non zero bit to third in the direction of 7. From third non zero bit to fourth in the direction of 0. So the 8- Commonly, all the profitable biometric systems shield the stored templates by using encrypting those using general cryptographic techniques. Either a public key cryptosystem like RSA (RSA laboratories, 1999) or a symmetric key cipher like AES (Advanced Encryption Standard, 2001) is usually used for template encryption. One of the important challenges in biometric identification or verification system is keeping the biometric data or template safe and secure. A hash function is usually transformed functions, which converts or transform data or features from one form to another. Always transform function should be a one-way function or another way it should not be invertible [10]. A number of template protection strategies like fuzzy commitment [13], fuzzy vault [13], protecting functions [14] and distributed supply coding [15] can be considered as the key binding biometric cryptosystem. Different schemes for securing biometric templates along with those positioned forth in [16-19] also fall under this class.

MD5 algorithm becomes robust and harder to decrypt for an intruder if we use salting process. Salting is adding extra strings to input arguments of the MD5 hash function. We use Freeman chain coding to extract features from the fingerprint image.

Fingerprints are half-secret, if passwords are leaked or hacked, it easily revocable using another password. But in a biometric security system, which uses only biometric features, is not easy to change fingerprint key or fingerprint are static biometric, which never change much throughout the lifespan. Fingerprints are left at the car, door or anyplace where every person goes and places his finger [20]]. Fingerprint Hash code is not used for full security or authentication purpose but it can be combined with other security mechanisms like password or OTP in order to enhance security. Fingerprint Hash code acts as the key, which can uniquely identify every person. So it can be replaceable with user-id or username and can work along with text-based or picture based or pattern based passwords. The fingerprint hash code is not constant with biometric sensors or reader [21-26].

This paper has six sections. Section-1 explains about introductory information of fingerprint and Freeman Chain coding, template protection and basic details of MD5 Hash function. Section-2, explains about objectives and methodology of the study. Section-3 explains about Algorithm of Hash code generation, Section-4 depicts a flowchart of Hash code generation. Section-5 explains Results and Discussions. Section-6 concludes the paper.

2. Objectives and Methodology:

There are many types of research are carried out translation and rotation invariant fingerprint hash code generation but even small or pixel changes cause a difference in Hash code. So this research does not concentrate on developing fingerprint hash code which is translation and rotation invariant. Fingerprint alone not gives full security, in order to improve the security of the system fingerprint acts one factor along with OTP, password, or any other biometric psychological or behavioral traits. The main objectives of this study are given below.

To Study a Fingerprint Hash code generation using Freeman Chain coding boundary starting x and y value, chain code value and the first difference value of the binary image. To verify the uniqueness of fingerprint Hash code using FVC ongoing 2002 benchmark dataset. Figure 4 explains the methodology used in this research work. Here initially FVC ongoing 2002 benchmark dataset is considered for testing the hash code. The benchmark dataset image is binarised and Freeman Chain code of the image is calculated for each pixel. Before calculating Freeman Chain code it finds boundary starting x and y value. Also, the first difference value of the chain code is calculated. The entire there parameters are considered and 32-bit length hash code is generated. Distinct Euclidean distance value summation, mean value and standard deviation values are considered for generating Hash code.

After converting fingerprint image to binary image we have to take one's complement of the binary image. So that all minutiae points will be represented using binary bit 1. Find all exterior, interior, parent and child boundary of the fingerprint binary image. Boundary pixel cell array returns all the pixel positions involved in a boundary. This function returns an array of boundary pixel values. Each element of the array is a matrix of size n × 2 dimensions. Where n represents a number of rows involved in forming boundary points and column are always 2, which represents x and y value of each point. Each boundary matrix is passed as an argument for Freeman Chain coding function.

Freeman Chain code function returns starting x and y position of the boundary, Freeman Chain code, the First difference value of the Freeman Chain code. Initially, if more than one argument is sent to Freeman Chain code function, then it returns an error message that too many arguments. Freeman Chain code Function then checks whether an argument matrix column value is 2. This means that each point should have two values corresponding to x and y pixel positions values respectively. If not this will return an error message that input dimension mismatched. Next, it checks for open contour or open boundary. If the first and last point of the boundary value is same then it is considered an open contour. In this case, Freeman Chain code boundary stating x and y value will be equal to first point pixel value. Both Freeman Chain code and First difference value will be zero.

If the first and last point of the boundary is not same, then find the difference x and y value, by subtracting the first point from the second point. Find difference for all the points which makes the boundary pixels. The transition from current pixel to the next pixel is computed from the connectivity diagram shown in figure 1 (b) and as shown in Table 1.

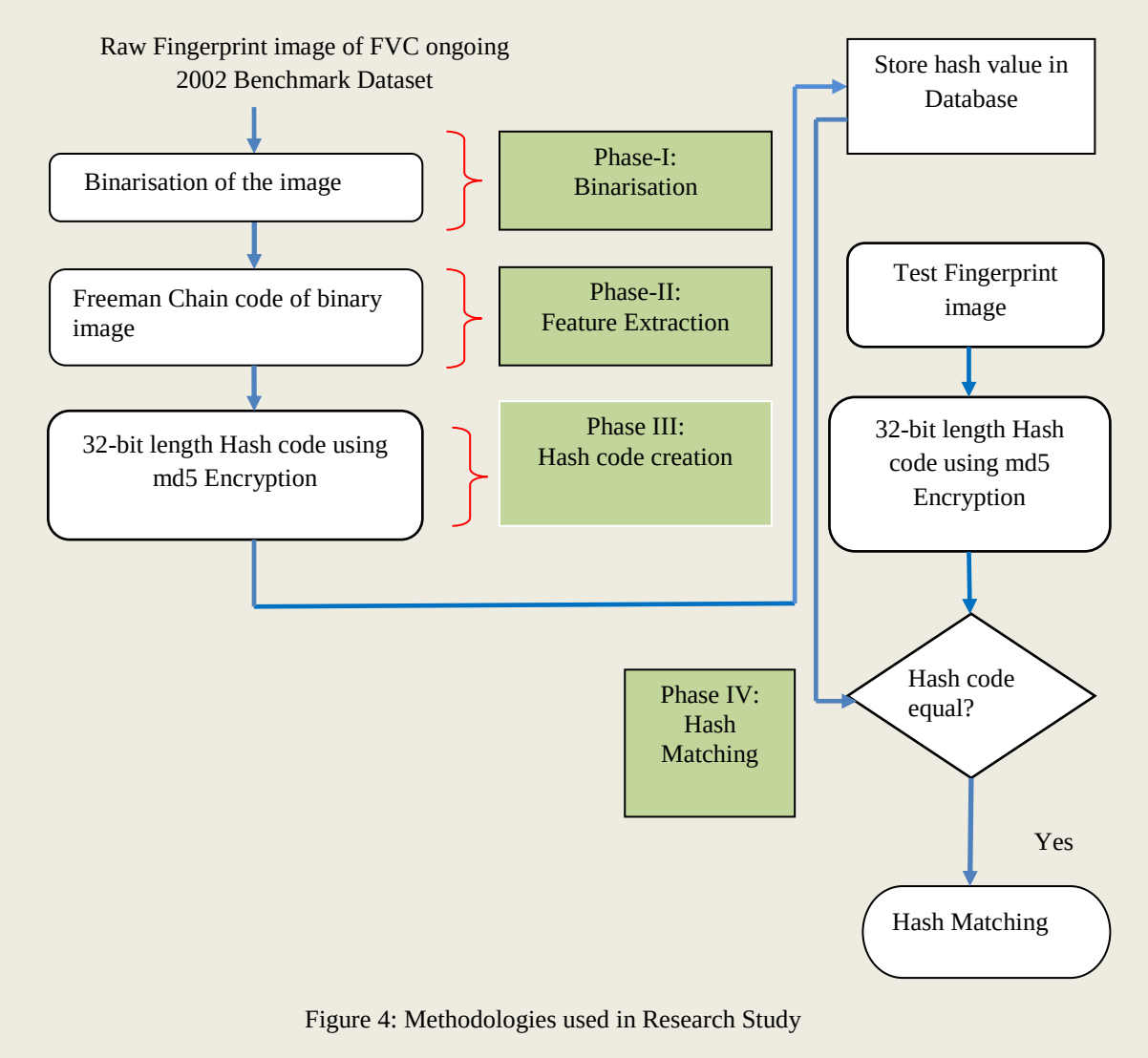

Figure 4: Methodologies used in Research Study

Table 1: Transition table of Freeman Chain code

Row Difference	Column Difference	Direction
0	+1	2
0	- 1	6
- 1	+1	3
- 1	- 1	5
+1	+1	1
+1	-1	7
-1	0	4
+1	0	0

Setting up mapping mechanisms between pixel value differences in 8-connectivity directions is controlled by the following statement

idx([1 2 3 5 7 9 10 11]) = [5 4 3 6 2 7 0 1]

idx corresponds index value of code value and unique value is obtained by doing a simple calculation as follows

diffB_map = 4*row_diff + col_diff + 6

Here diffB_map variable corresponds to each point pixel value difference that is involved in forming boundary pixels. These diffB_map values indexed into direction map using below statement and which produces chain code value.

chain_code_value = idx(diffB_map)

Finally, first difference value of chain code is obtained, by the following procedure

Step-1: Subtract each chain code value from Chain code end value
Step-2: Add number 8 to Step-1
Step 3: Take Modulus value of step-2 and 8 i.e. mod (Step-2, 8)

3. Algorithm of Hashcode Generation Using Freeman Chain Code:

This section explains step by step procedure to develop Hashcode by making use of Freeman Chain code on a binary fingerprint image. The steps of the algorithm are explained below. The algorithm also shows the pseudo code.

Step 1: Input Grayscale fingerprint image
read (input_image)
Step 2: Convert input image into 256 × 256 sized two-dimensional image
resized_image = image_resize (input_image, [256, 256])
Step 3: Convert 256 × 256 sized grayscale image into binary image
binary_image = convert_to_binary(resized_image)
Step 4: Perform One's complement of the binary_image
Binary_image = One's complement (binary_image)
Step 5: Find the Exterior boundaries of the binary image and boundaries of holes inside this exterior Boundary
boundary_pixel_array = boundary_pixel(binary_image)
Step 5: Find the row length of boundary_pixel_array
m = length (boundary_pixel_array) [where m is row length]
Step 6: Find Freeman Chain code for each element of the boundary_pixel_array
For i=1 to m
freeman_chain_code= freeman_chain_code (i)
end for
Step 7: Obtain Row and column coordinates for the starting pixel of the boundary from Step-6.
start_idx = freeman_chain_code. start_idx
Step 8: Obtain Freeman Chain code value for each boundary from Step-6
chain_code = freeman_chain_code. chain_code
Step 9: Obtain Freeman Chain code first difference value for each boundary from Step-6
firstdiff = freeman_chain_code. firstdiff
Step 10: Compute summation of Step-7, Step-8, and Step-9
Step 11: Divide all 3 values of Step-10 by m.
Step 12: Pass the value of Step-11 as parameter for MD5 Hash function
hash_value = MD5_DataHash(combine_value)

4. Flowchart of Hashcode Generation Using Freeman Chain Code:

The above algorithm is explained using a flowchart. The different process or workflow is listed below. With an intention to make the MD5 Hashcode more robust and to get the advantage of salting Freeman Chain code summation of boundary starting x and y position, Chain code value and first difference value are divided by a total number of boundary value and all values are combined as a string and passed to the MD5 algorithm. Figure 5 shows a flowchart of this. Converting input image to 256 × 256 sized grayscale image

- ✓ Converting to binary image
- ✓ Taking one's complement of each pixel of binary image
- ✓ Finding Freeman Chain code function
- ✓ Obtaining Row and column coordinates for the starting pixel of the each boundary
- ✓ Obtaining Freeman Chain code value for each boundary
- ✓ Obtaining Freeman Chain code first difference value for each boundary
- ✓ Generating a strong salting string for MD5 hash function-argument

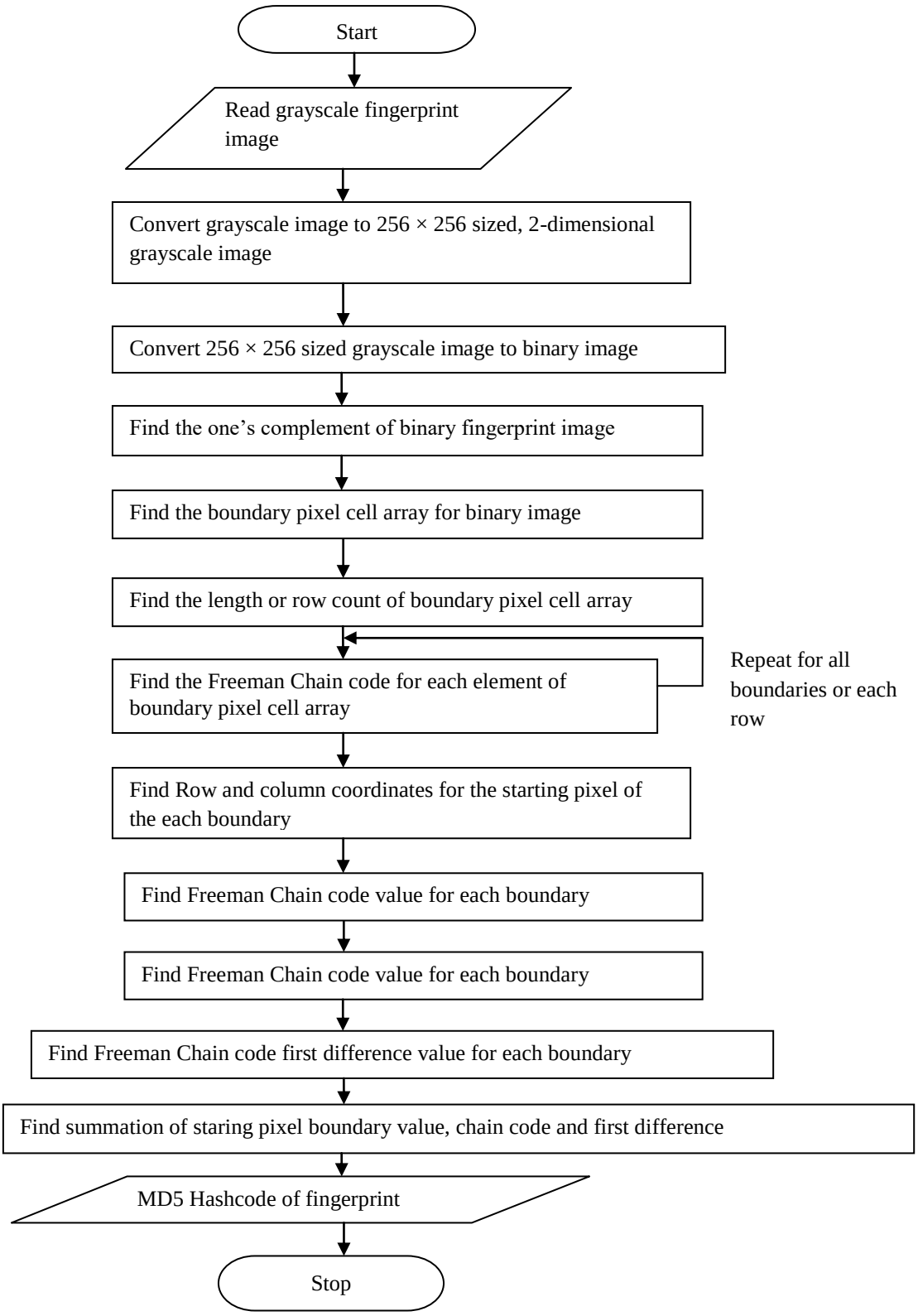

Figure 5: Flowchart of Hash code generation using Freeman Chain code

5. Results and Discussions:

In this study, WampServer is used to create a database. This database table contains two fields as id and Hashcode. The Hashcode generation using Euclidean distance is implemented using MATLAB2015a. The configuration of the system used to implement this study is given in Table 2.

Table 2: Configuration of System used for finding Execution Time

S.No	Parameters	System Details

1	Model	Compaq 435
2	Processor	AMD E-350 processor 1.60 GHz
3	Installed Memory	3 GB (2 GB usable)
4	System Type	32-bit Operating System
5	Operating System	Windows 7 Starter
6	Software	MATLAB 2015a 32-bit

The execution time for different randomly selected images of FVC ongoing 2002 dataset is shown in Table 3.

Table 3: Execution time of the training phase

Method Name	Image name	Execution Time (in seconds)	Average
Method-1	101_1	4.243991	1.672678
	101_5	2.343854	
	102_2	1.488027	
	103_3	1.447657	
	104_4	1.917586	
	104_7	1.280134	
	104_8	1.374160	
	105_8	0.775691	
	106_6	1.387195	
	109_3	1.275428	
	109_8	1.457092	
	110_3	1.343319	
	110_8	1.410684	

The average execution time of the fingerprint Hashcode generation using Freeman Chain code is very good and it is approximately on an average is 1.672678. Here we only consider the training phase. The testing phase includes around 0.44 seconds more than training phase. If the configuration of the system increases definitely execution time also increases. Table-4 shows the Hashcode generated based on an MD5 algorithm using Freeman Chain code.

Table 4: Hash code generated using Freeman Chain code

S.No	Hashcode
1	81981d7bd4cce1582d8b2b7504c26a50
2	76fc72bb5650bfb491117c933449936a
3	87ab0b35e67e28775f467e9b6e988fae
4	dcf4b2aac0a2819fc410fffbd4114105
5	a0dcdb0c511d48f4b8e9e8d80addc006
6	902db2784067d125c5c2d28120974428
7	ff1f7f440d3e81b119f38f764fb9af6f
8	c64a8793be56e573f95ff0b454864fb1
9	e71511ffd44da8a76b11fdb6f3f1b68f
10	8e9fc502886a69e39c0f6e773e470b33
11	9faf0422c4e5563331322e90937a99c4
12	f60da0277c5656560ac81d12ddd4c397
13	231f87480e44ee88f4a79932560833a8

The screenshots of the grayscale fingerprint image capture is shown using Figure 6.

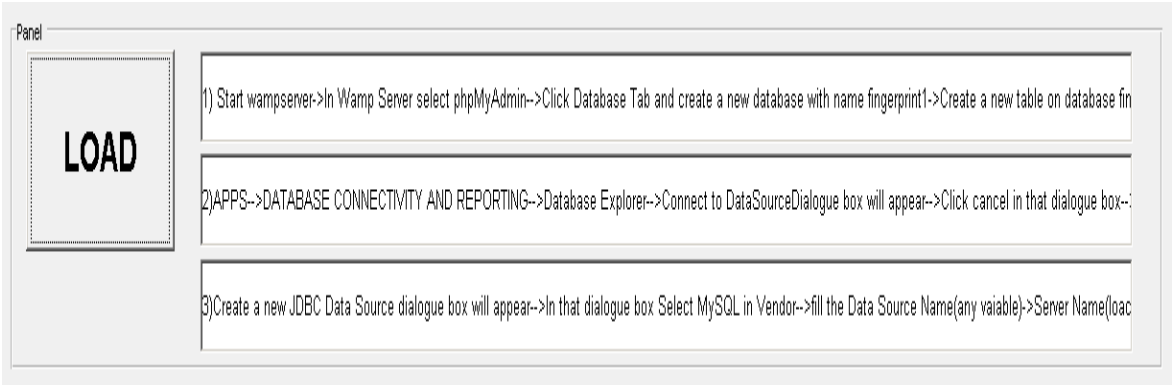

Figure 6: Screenshots of Fingerprint image capture

The screenshot or Figure 6 contains two push buttons. One push button is used to select grayscale fingerprint image. Another one gives instruction to create WampServer and to connect this from MATLAB2015a. Here we use static one time captured an image of FVC ongoing 2002 benchmark dataset. The screenshots of Database processing and status is shown using Figure 7.

The data processing control of figure 7 consists of five push buttons as Connect, Fetch, Data, Hash Generation, and Check. Connect button is used to connect to the database, Fetch button is used to fetch records from the database, Data button is used to show data in tables, hash generation is used to generate Hashcode for sample input fingerprint, and Check is used to check sample input Fingerprint image is matching or not matching with already stored hash code.

Advantages of Hash Value Produced Using Euclidean Distance:

- ✓ Hash code produced using Freeman Chain code is noninvertible
- ✓ Hash code takes very small amount of memory
- ✓ Hash code Hides original information of fingerprint image from the intruder
- ✓ The execution time of Hash code generation using Freeman Chain code is very good.
- ✓ It is unique for each fingerprint of the same person means ten fingerprints will be having ten different Hash codes.

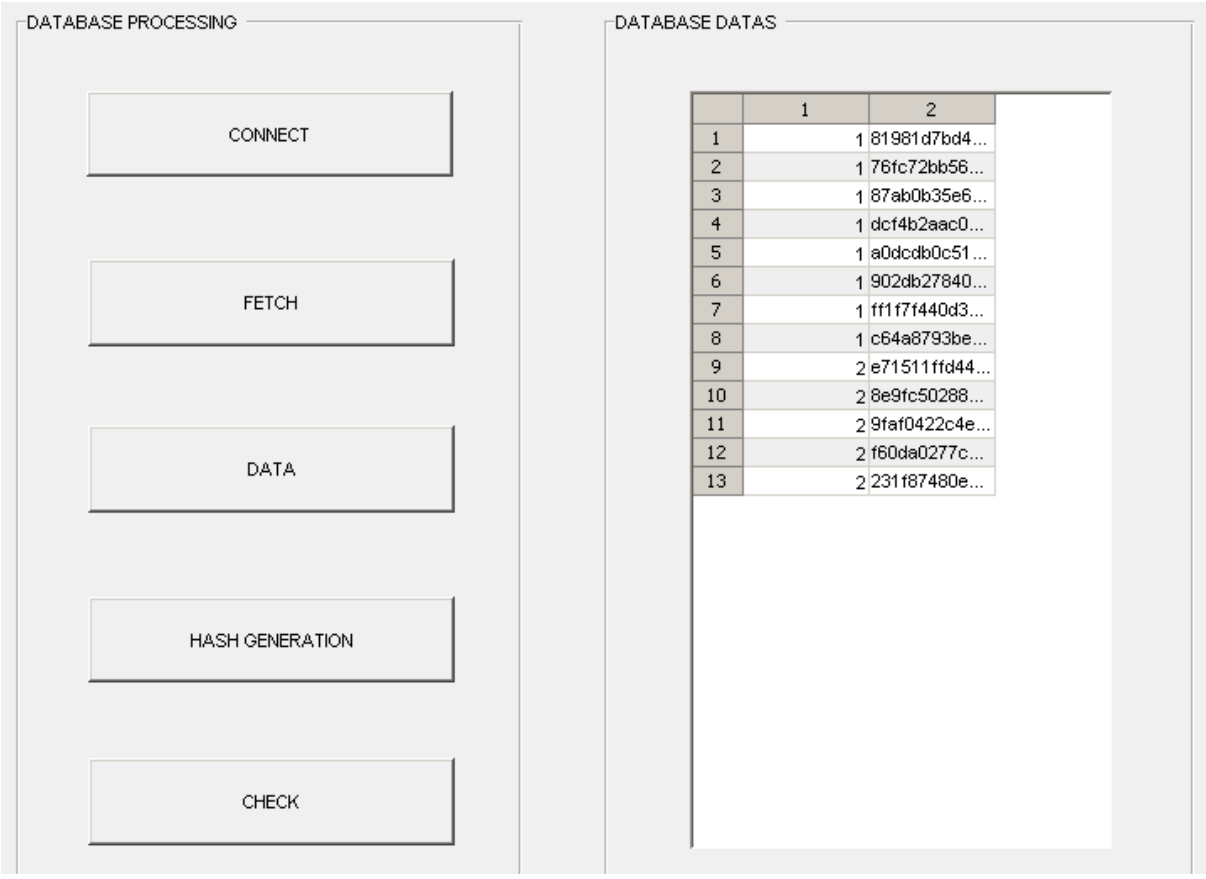

Figure 7: Screenshots of Database processing and Status

Benefits of Hash Value Produced Using Euclidean Distance:

- ✓ Hash code is used as identity-key or index-key for unique identification purpose of a user.
- ✓ Easily we can append salting in order to make the Hash code more robust.
- ✓ Fingerprint Hash code is a transformed function, which does not reveal original minutiae details.
- ✓ Fingerprint Hash code consumes very less time for training phase.
- ✓ Unlike another fingerprint matching, this study does not use scoring level. It uses only binary value either matching or not matching.
- ✓ This can be used for security or authentication purpose if the user takes some security measures to protect static one time captured fingerprint image like folder locking.

Constraints of Hash Value Produced Using Euclidean Distance:

- ✓ Small changes in fingerprint hash code make large differences.
- ✓ Fingerprint generations using Freeman Chain code are translation and rotation variant which is not having much scope when the fingerprint is used for identification purpose rather than security purpose.

Disadvantages of Hash Value Produced Using Euclidean Distance:

- ✓ Fingerprint hash code cannot be solely used for security or authentication purpose.
- ✓ If fingerprint image of same finger input is taken through any type of solid and robust sensors in consecutive two intervals, still fingerprint hash code generates different hash code.
- ✓ Even though developed fingerprint Hash code is invariant to translation and rotation, if the user presses hardly into one reader or sensor, or swipe the finger in a different orientation, or a cut in the finger, for a successive two capture, produces different Hash code.

6. Conclusion:

Fingerprint hashing is the new technique which combines biometrics and cryptography. The goal is to perform identification based on fingerprint simultaneously hiding or keeping the fingerprint information secretly or noninvertible way. Even though fingerprint is compromised intruder should not get original features of the fingerprint image. Fingerprint image having some drawbacks like which left by a human being at many places like door, wall, on the car and many more places are easily mimicked by fraud or intruder. The fingerprint does not get matched when the finger has some cut or wound and sensors are not able to recognize in some weather conditions like winter season. The fingerprint is effective as identity or index key and not as a full security feature. It works well with multifactor biometrics authentication as one major factor.

In this paper, we developed a Hash code based on an MD5 Hash function by making use of Freeman Chain code for a binary fingerprint image. This Hashcode can be effectively used as Index-key or identity-key. This method shows considerably better execution time. This method also gives 100% accurate matching as far as input fingerprint image is once captured and stored static digital fingerprint image. If we capture through sensor each time this gives different Hash code. So this method is not suitable for solely security purpose unless and until the user takes some security measure to protect static fingerprint image.

7. References:

1. Nandakumar, K., Jain, A. K., & Nagar, A. (2008). Biometric template security. Eurasip Journal on Advances in Signal Processing. https://doi.org/10.1155/2008/579416.
2. Nandakumar, K., & Jain, A. K. (2004). Local Correlation-based Fingerprint Matching. In ICVGIP (pp. 503-508).
3. Krishna Prasad, K. & Aithal, P.S. (2017). A Conceptual Study on Image Enhancement Techniques for Fingerprint Images. International Journal of Applied Engineering and Management Letters (IJAEML), 1(1), 63-72. DOI: http://dx.doi.org/10.5281/zenodo.831678.

4. Krishna Prasad, K. & Aithal, P.S. (2017). Literature Review on Fingerprint Level 1 and Level 2 Features Enhancement to Improve Quality of Image. International Journal of Management, Technology, and Social Sciences (IJMTS), 2(2), 8-19. DOI: http://dx.doi.org/10.5281/zenodo.835608.
5. Krishna Prasad, K. & Aithal, P.S. (2017). Fingerprint Image Segmentation: A Review of State of the Art Techniques. International Journal of Management, Technology, and Social Sciences (IJMTS), 2(2), 28-39. DOI: http://dx.doi.org/10.5281/zenodo.848191.
6. Krishna Prasad, K. & Aithal, P.S. (2017). A Novel Method to Contrast Dominating Gray Levels during Image contrast Adjustment using Modified Histogram Equalization. International Journal of Applied Engineering and Management Letters (IJAEML), 1(2), 27-39. DOI: http://dx.doi.org/10.5281/zenodo. 896653.
7. Krishna Prasad, K. & Aithal, P.S. (2017).Two Dimensional Clipping Based Segmentation Algorithm for Grayscale Fingerprint Images. International Journal of Applied Engineering and Management Letters (IJAEML), 1(2), 51-65. DOI: http://dx.doi.org/10.5281/zenodo.1037627.
8. Krishna Prasad, K. & Aithal, P.S. (2017). A conceptual Study on Fingerprint Thinning Process based on Edge Prediction. International Journal of Applied Engineering and Management Letters (IJAEML), 1(2), 98-111. DOI: http://dx.doi.org/10.5281/zenodo.1067110.
9. Krishna Prasad, K. (2017). A Critical Study on Fingerprint Image Sensing and Acquisition Technology. International Journal of Case Studies in Business, IT and Education (IJCSBE), 1(2), 86-92. DOI: http://dx.doi.org/10.5281/zenodo.1130581.
10. Tulyakov, S., Farooq, F., Mansukhani, P., & Govindaraju, V. (2007). Symmetric hash functions for secure fingerprint biometric systems. Pattern Recognition Letters, 28(16), 2427-2436.
11. Gonzalez, R. C., & Woods, R. E. (1992). Digital image processing.
12. Bernard, M., Fromont, E., Habrard, A., & Sebban, M. (2012, June). Handwritten digit recognition using edit distance-based KNN. In Teaching Machine Learning Workshop.
13. Juels, A. (2002). M. Sudan 'A fuzzy vault scheme'. In Proceedings of the 2002 IEEE International Symposium on Information Theory (Vol. 408).
14. Tuyls, P., Akkermans, A. H., Kevenaar, T. A., Schrijen, G. J., Bazen, A. M., & Veldhuis, R. N. (2005, July). Practical biometric authentication with template protection. In AVBPA (Vol. 3546, pp. 436-446).
15. Holst, J. C., & Draper, D. A. (1999). U.S. Patent No. 5,999,039. Washington, DC: U.S. Patent and Trademark Office.
16. Davida, G. I., Frankel, Y., Matt, B., & Peralta, R. (1999). On the relation of error correction and cryptography to an online biometric based identification scheme. In Workshop on coding and cryptography.
17. Hao, F, Anderson, R & Daugman, J 2006, Combining Crypto with Biometrics Effectively', IEEE Transactions on Computers, vol. 55, pp. 1081-1088.
18. Kelkboom, E. J., Gökberk, B., Kevenaar, T. A., Akkermans, A. H., & van der Veen, M. (2007, August). "3D face": biometric template protection for 3D face recognition. In International Conference on Biometrics (pp. 566-573). Springer, Berlin, Heidelberg.
19. Connie, T., Teoh, A., Goh, M., & Ngo, D. (2005). Palm Hashing: a novel approach for cancelable biometrics. Information processing letters, 93(1), 1-5.
20. https://hackaday.com/2015/11/10/your-unhashable-fingerprints-secure-nothing/, Last Accesses Date: 05-12-2017.
21. https://security.stackexchange.com/questions/42384/is-there-any-way-to-cryptographically-hash-a-human-thumbprint, Last Accesses Date: 05-12-2017.
22. Chikkerur, S. S. (2005). Online fingerprint verification system (p. 2005). State University of New York at Buffalo.
23. Bhuyan, M. H., Saharia, S., & Bhattacharyya, D. K. (2012). An effective method for fingerprint classification. arXiv preprint arXiv:1211.4658.
24. Meng, X. P., Wu, Z. G., & Zhao, Y. L. (2009). Algorithm of fingerprint identification based on fingerprint texture structure [J]. Computer Engineering and Design, 13, 031.
25. Aithal, P. S. (2016). A Review on Advanced Security Solutions in Online Banking Models, International Journal of Scientific Research and Modern Education (IJSRME), 1(1), 421-429. DOI: http://doi.org/10.5281/zenodo.160971.
26. Aithal, P. S. (2015). Biometric Authenticated Security Solution to Online Financial Transactions. International Journal of Management, IT and Engineering (IJMIE), 5(7), 455-464, DOI: http://doi.org/10.5281/zenodo.268875.