A Fingerprint Biometric Cryptosystem in FPGA

Rosario Arjona and Iluminada Baturone
Instituto de Microelectrónica de Sevilla (IMSE-CNM)
Universidad de Sevilla – Consejo Superior de Investigaciones Científicas (CSIC)
Seville, Spain
{arjona, lumi}@imse-cnm.csic.es

***Abstract*—This paper presents the implementation of a complete fingerprint biometric cryptosystem in a Field Programmable Gate Array (FPGA). This is possible thanks to the use of a novel fingerprint feature, named QFingerMap, which is binary, length-fixed, and ordered. Security of Authentication on FPGA is further improved because information stored is protected due to the design of a cryptosystem based on Fuzzy Commitment. Several samples of fingers as well as passwords can be fused at feature level with codewords of an error correcting code to generate non-sensitive data. System performance is illustrated with experimental results corresponding to 560 fingerprints acquired in live by an optical sensor and processed by the system in a Xilinx Virtex 6 FPGA. Depending on the realization, more or less accuracy is obtained, being possible a perfect authentication (zero Equal Error Rate), with the advantages of real-time operation, low power consumption, and a very small device.**

Keywords—Fingerprint recognition, biometric cryptosystems, FPGA hardware design, CAD tools

I. INTRODUCTION

In the next years, the widespread use of biometric systems will lead to the massive storage of biometric data. If an individual is registered in different biometric systems, the same biometric data will be stored in several places. Let us consider a situation where a person is registered by means of his/her fingerprint and an impostor steals the fingerprint representation associated to the template. The fingerprint image can be reconstructed from the template and then used to attack successfully the fingerprint recognition system [1]-[2]. In this situation, the user has to cancel that fingerprint and uses another one. The problem is that a maximum of ten fingers are available for each individual.

From a security point of view, it is justified that biometric templates should be protected. Protection is required not only for template storage but also for operational and communication procedures of biometric data [3]. The realization of the complete biometric recognition system in the same hardware device (*Authentication on Card*) increases the security because the access to communication channels is more difficult. A further step to increase security is to employ biometric template protection schemes [4]. They provide interesting advantages such as *non-reversibility*, which means that it is computationally infeasible to recover the unprotected template from the protected template. At the same time, it can be possible to create different protected templates from the same template to be used in different applications. This property is known as *diversity* and leads to *revocability*, which means that as many protected templates as necessary can be generated when security is compromised. As drawback, the computational complexity of template protection schemes increases considerably and, in many cases, recognition accuracy decreases.

Template protection schemes are categorized commonly into *feature transformation systems* and *biometric cryptosystems*. An example of the first case is salting techniques, also known as Biohashing, which combine a password introduced by the user (named as salt) with biometric data [5]-[6]. In this way, different passwords generate different protected templates. However, both protected template and password have to be private because if one of them is known, all the information is public. In the other side, biometric cryptosystems are based on fusing the unprotected template with additional information to generate data, named as helper data. An advantage of biometric cryptosystems is that helper data do not have to be private because the additional information employed obfuscates the template information and the helper data does not give biometric information.

This paper focuses on implementing a complete fingerprint biometric cryptosystem in the same hardware device, in particular a FPGA (*Authentication on FPGA*). To the best of our knowledge, no protected biometric system has been implemented with dedicated hardware. The paper is structured as follows. Firstly, Section II reviews the main template protection approaches reported in literature. Section III presents a novel fingerprint feature named QFingerMap, which can be implemented in dedicated hardware with very few memory and computing resources. A cryptosystem that can fuse QFingerMaps from different samples of the same finger as well as passwords provided by the user is presented in Section IV. Section V summarizes the hardware implementation of the cryptosystem proposed and reports hardware implementation results in terms of timing and resource occupation. Finally, conclusions are given in Section VI.

II. BIOMETRIC CRYPTOSYSTEMS

The biometric cryptosystems based on *Fuzzy Commitment* [7] combine error correction and cryptographic techniques.

During the enrollment phase, helper data H is computed from the biometric template B and a codeword C generated by an Error Correction Code (ECC). Then, H and hash(C) are stored. At matching, the error correction scheme decodes the information from the input biometric data B' and the helper data stored H. Since the input biometric data B' is similar to the biometric template B, the word C' resulting from combining B' and H is similar to C, so that the error correction code applied to C' obtains C, ECC(C') = C. The authentication is successful when hash(C) and hash[ECC(C')] coincide. Any input biometric data B' similar to the biometric template B should be able to reconstruct C. The Fuzzy Commitment scheme is illustrated in Figure 1. This scheme requires that biometric representations are binary, length-fixed, sorted and aligned and that was the reason why the first practical approaches of fingerprint cryptosystems were applied to features such as *FingerCodes*. Feature vectors extracted from minutiae are not ordered or aligned features. Hence, they should be converted to suitable representations. If extraction of minutiae is already complex for dedicated hardware, its protection further complicates its implementation [8].

Error correction techniques can be categorized into two groups, depending on how errors are processed: bit-by-bit (which corrects random bit errors) or block-by-block (which corrects burst errors). Although errors are normally distributed as bursts, both types have been applied in biometric cryptosystems, particularly BCH and Reed-Solomon [8], [9]. Despite applying error correction codes, most existing biometric template protection methods cause degradation in biometric performance, in comparison to an unprotected system [10]-[11]. Most of biometric features are real-valued but template protection schemes require binary features so that discretization methods influence the performance of the biometric cryptosystem because there is loss of information. This can be seen in Table I, which shows results from *feature transformation systems* and *biometric cryptosystems*. FMR (False Match Rate) is the number of false matches for the impostor distribution and FNMR (False Non-Match Rate) is the number of false non-matches for the genuine distribution.

III. The Feature QFingerMap

A novel fingerprint feature based on textures is considered in this work [14]. The feature, named *QFingerMap*, is extracted from a window centered at the convex core point of the fingerprint, once the orientation or directional image (which contains the local ridge orientations of the pixels in the fingerprint image) has been segmented into homogeneous regions [15]. The complete extraction process is shown in Figure 2.

Let us consider a coarse directional image that assigns to each pixel 1 out of 8 possible direction intervals in the range from 0º to 180º: g0=[0º, 22.5º), g1=[22.5º, 45º), g2=[45º,67.5º), g3=[67.5º, 90º), g4=[90º, 112.5º), g5=[112.5º, 135º), g6=[135º, 157.5º), and g7=[157.5º, 180º). These intervals are represented by the following symbols (coded with 3 bits): 000, 001, 010, 011, 100, 101, 110, and 111. Each symbol is represented by a color in Figure 2. The selection of the interval (and symbol) for each pixel is determined by simple comparisons between horizontal and vertical gradient values calculated at each pixel. As in any technique that calculates orientation images, the next step after symbol assignation is a smoothing process because the objective is to obtain homogeneous regions with the same symbols. A nonlinear filter based on maximum operator has been employed. It considers the neighboring pixels inside an S x S window centered at the analyzed pixel and assigns it the symbol with the highest number of occurrences inside the window. This operation is in charge of removing isolated and noisy symbols. The window size depends on the sensor employed (its size, resolution, and technology) because the features of the fingerprint images acquired are different. If the window size is small, isolated and noisy symbols cannot be removed. In contrast, if the window size is too large, relevant information can be lost. A 27 x 27 window has been proven to provide good performance for different types of sensors [15].

The feature vector is generated from an N x N window within the smoothed segmented orientation image centered at the convex core point, being formed by N x N symbols in an ordered way. Each element of the feature vector is one symbol out of the 8 possible symbols. The selection of the window size N x N is also relevant for the recognition process. An adequate size depends in turn on the fingerprint image size acquired by the sensor. For most fingerprint sensors, which capture a fingerprint size of, approximately, 300 x 300, the several studies carried out have revealed that the most suitable option is a window size in the range of 129 x 129 because it gives the best tradeoff between distinctive capability and the fingerprint image size captured by the sensors.

The feature vector length is reduced by applying down-sampling to remove redundant information. The target is to generate a compact and distinctive representation of the fingerprint by selecting the most representative symbols. A simple way is to take 1 between d consecutive pixels (downsampling by a factor of d). A suitable performance is given by a factor of 8, which results a feature vector of 17 x 17 symbols. If symbols are coded by 3 bits, the 17 x 17 symbol vector requires 867 bits (17 x 17 x 3 bits), which is a considerable reduction with respect to the initial 129 x 129 symbol vector. Therefore, the feature vector QFingerMap is defined by a fixed-length vector composed of symbols distributed in a sorted way.

The matching operation between two QFingerMaps is done by computing the number of different symbols. From its conception, a QFingerMap has been thought for hardware implementations, so that the feature extraction and matching operations require a low computational cost. In addition, its translation to a binary representation is direct and does not need a quantization process (thus reducing the possible variations caused by the translation of continuous to discrete values). Hence, QFingerMaps are very suitable fingerprint features for the application of a Fuzzy Commitment scheme for the purpose of biometric template protection.

IV. A Biometric Cryptosystem Based on QFingerMaps

The helper data generation in a Fuzzy Commitment scheme is in charge of fusing the codeword and the biometric data (the QFingerMap) in a obfuscated way to create public information. In general, the fusion of the codeword and the biometric data is done by a XOR operator and so it has been employed in this

work. A one-way transformation function (a cryptographic hash function) protects the codeword information. The hash(C) value can be public because the hash function ensures that the codeword C is computationally infeasible to recover from the hash(C) value. *Keccak* has been selected as hash function [16].

The codeword length n is determined by the QFingerMap length L (which is associated to symbols or bits, depending on the representation) and has to satisfy $n \geq L$ to fuse the codeword with biometric information. For BCH and Reed-Solomon error correction codes, $n=2^m-1$ (n expressed as base 2 depending on m value). If $L<n$, a padding of $n-L$ is applied to QFingerMap to complete the n values. The creation of the codeword is performed from a random value with length k. Values of n and k establish the number of errors t, which the error correction code can correct as a maximum.

Since a QFingerMap is composed of 289 symbols or 867 bits (in total, 17 x 17 symbols coded with 3 bits), the selection of a BCH code, which corrects random bit errors, requires a minimum length for the codeword of 1023 bits (n=1023). In the case of Reed-Solomon codes, which correct symbol errors, the 289 symbols of a QFingerMap determine the parameter for the codeword length (n), whose value should be greater or equal than 511 (for $m \geq 9$ and, thus, $n \geq 2^9-1$). For Reed-Solomon codes, the maximum number of errors t that can be corrected is defined as $t=(n-k)/2$.

An unprotected biometric system based on QFingerMaps applies a matching operation which calculates the number of different symbols (codified by 3 bits) between two QFingerMaps. Hence, the number of different symbols (bits) used as threshold value to distinguish between genuine individuals and impostors is correlated to the number of symbols (bits) than the ECC can correct. The correction code capability also determines the values of FMR (False Match Rate) and FNMR (False Non-Match Rate) because as more errors are corrected, the FMR increases and the FNMR decreases, and vice versa.

The unprotected biometric system has been applied to 560 fingerprints captured in live by an optical sensor (the FS90 sensor from Futronic) and enhanced by applying the filtering proposed in [17]. In the genuine distribution, each sample is matched against the remaining samples of the same finger, and for the impostor distribution, the first sample of each finger is matched against the first sample of the remaining fingers. These distributions remove symmetric comparisons to avoid correlation according to the recommendations of Fingerprint Verification Competition (FVC). Table II shows a comparison of the EER (Equal Error Rate, that is, the rate where FMR and FNMR are equal) for the unprotected biometric system and the protected biometric system, considering BCH and Reed-Solomon codes. In general, there is not a clear relation between the unprotected biometric system and the cryptosystem in terms of FMR, FNMR, and EER. The cryptosystem usually performs worse but it cannot be estimated quantitatively by a simple analysis how much worse it is going to perform. In contrast, for QFingerMaps, there is a clear relation between the performance of the unprotected system and the performance of the cryptosystem. This is an advantage that eases the design of the cryptosystem because the threshold values of the unprotected biometric system indicate the number of errors to be corrected by the error correction code. Thus, the selection of the error correction code parameters depends on the FMR and FNMR values required by the application. This is illustrated in Figure 3, which shows how the results of the protected and the unprotected systems are quite similar. However, the minimum length for the BCH codewords is 1023 bits, and BCH codes are more suitable (offer more efficient implementations) for lengths of a hundred of bits [11]. Hence, a Reed-Solomon code has been chosen for the proposed cryptosystem.

A fingerprint recognition system usually employs several samples of a finger at enrollment and even at matching to increase recognition performance. Therefore, let us consider the acquisition of S_t samples at enrollment, so that S_t helper data values are computed as follows:

$$H = (S_1 \oplus C_1, \dots, S_t \oplus C_t) = (H_1, \dots, H_t) \quad (1)$$

where $(C_1, \dots, C_t)$ are codewords of an ECC.

At matching, S'_q samples are captured as inputs and S_t x S'_q values are calculated from the helper data values stored as follows:

$$\left(C'_1, \dots, C'_{S_t \times S'_q}\right) = (S'_1 \oplus H_1, \dots, S'_1 \oplus H_t, \dots, S'_q \oplus H_1, \dots, S'_q \oplus H_t) \quad (2)$$

The S_t x S'_q hash values, hash[ECC(C'_i)], are compared to the corresponding hash values stored, hash(C_j). The number of comparisons is S_t x S'_q and each comparison gives '1' if the compared values are equal, and '0' if they are different. The OR operator is applied to these values, and the result is '1' if an individual has been recognized, and '0' otherwise.

$$out = OR_{\substack{i=1,\dots,t \\ j=1,\dots,q}} [hash\left(ECC(S'_j \oplus H_i\right)] = hash(C_i) \quad (3)$$

Considering the fingerprint database commented above, the following realizations have been analyzed among others:

- Case 1: Multi-sample system composed of three samples at enrollment, and one sample at matching.

- Case 2: Multi-sample system composed of three samples at enrollment, and two samples at matching.

A two factor identification/authentication, which combines passwords and biometric information, increases the security because both information ('what you know' and 'who you are') are required to recognize an individual. Also, the fusion of biometric data and passwords is a simple way to obtain more accuracy in recognition by using a dual factor [18]. Hence, a bit operator (XOR) has been employed to fuse bits from passwords and bits from biometric data. Padding has been applied to the hash of the password to achieve the same QFingerMap length. The password obfuscates the biometric information and vice versa, so that both data are protected by the Fuzzy Commitment scheme. As a matter of fact, this case can be seen as a kind of salting transforms. The result of the fusion has the length of a QFingerMap. Hence, protection is applied similarly to equation (3), as follows:

$$out = OR_{\substack{i=1,\dots,t \\ j=1,\dots,q}} [hash\left(ECC(S'_j \oplus P' \oplus H_i\right)] = hash(C_i) \quad (4)$$

where the helper data H_i are associated to a sample *i* of the fingerprint, S_i, the hash of the password conveniently padded, P, and a codeword, C_i.

Table III shows the selection of ECC parameters for the cases 1 and 2 considered above, with and without passwords, using Reed-Solomon codes. Individual discrimination is improved. Figure 4 illustrates graphically the recognition performance (FMR and FNMR) for the multi-biometric case 2 when adding the passwords. Since the genuine and impostor distributions are separated, several Reed-Solomon codes are suitable for a perfect authentication.

V. Design of the FPGA System

The main blocks for the hardware implementation of the presented cryptosystem implement the QFingerMap extraction, the encoder to generate the codeword from a random value, the decoder to correct errors due to the variability in the biometric data acquisition, and the hash function to protect the codeword generated and to process the possible password introduced by the user. The enhancement filtering employed for the fingerprint database considered is not included in the system because the enhancement techniques depend on the type of the sensor and the captures. These blocks are shown in Figure 5.

The whole system has been designed with Matlab-Simulink. The tool HDL Coder from Simulink has been employed to generate synthesizable VHDL code. The tools from Xilinx ISE environment have been used to implement the design in the FPGA, verify the code at hardware level (with ISIM simulator), and verify the system at its context of operation (with FPGA-in-the-Loop functionality) [19].

The implementation has been performed in a Xilinx Virtex-6 (XC6VLX240T-1FFG1156) FPGA for 440 x 300 fingerprint images. The FPGA considered contains 37680 slices and 416 36-Kbit Block RAMs. More details about the architecture and data processing of the blocks that obtain the partitioned directional image, the smoothed partitioned directional image, and the singular points as well as the block that estimates the quality of the acquired fingerprint image can be seen in [15]. The realization also includes a 512-bit hash function based on the SHA-3 Keccak [20].

As discussed, Reed-Solomon is selected as the most suitable error correction technique for a cryptosystem based on QFingerMaps. Following the hardware design flow based on CAD tools, the Reed-Solomon encoder and decoder implemented have been obtained from the Matlab-Simulink library blockset for HDL Coder ([21] and [22], respectively). The selection of different parameters leads to different implementations and, thus, different occupation and resources needed. The results for a Reed-Solomon code with n=511, k=383 and t=64 are 505 slices (1.34%) for the encoder, and 25181 slices (66.83%) for the decoder. This code can correct 22.15 % of errors, which allows perfect authentication in the multi-biometric system of case 2 shown in Figure 4.

All blocks required for a fingerprint-based cryptosystem can be implemented in the same device, in this case a Xilinx Virtex-6 FPGA. It is a dual security level because recognition stages (feature extraction and matching) are performed in the same device and template protection lets store and match biometric data and passwords in a transformed domain, which reduces considerably the number of possible attacks. The occupation of all blocks is 29895 slices (79.34%). The Reed-Solomon Decoder occupies most of slices (66.83%) and limits the global maximum frequency to 58 MHz. Hence, the system generates the feature vector in 2.28 ms after processing serially, pixel by pixel, a 440 x 300 fingerprint image (440 x 300 / (58 MHz)). Once the hash of the query codeword is obtained from the query fingerprint (and possible password), the time to compute the comparison with the stored hash is negligible (it is a comparison of hash values).

The template memory is composed of 1023 bits from the helper data (511 bits) and the hash of the codeword used at enrollment (512 bits). Since 18.5 out of the 416 36-Kbit Block RAMs are used in a Xilinx Virtex-6 FPGA to extract a query codeword, the other 36-Kbit Block RAMs of the FPGA can store more than 14,323 helper data associated to samples and passwords.

VI. Conclusions

This work has presented an FPGA realization that protects a fingerprint template based on the feature QFingerMap. The proposal is a cryptosystem that applies a Fuzzy Commitment scheme because the feature QFingerMap is an ordered, binary and fixed-length vector. The main operations of the Fuzzy Commitment scheme are the error correction technique and the hash function. The error correction codes selected have been Reed-Solomon codes since they are more suitable for QFingerMaps, which are based on symbols instead of bits. Recognition performance results show that there are not significant differences between the performance of unprotected and protected systems, which is an interesting result that facilitates the design of the cryptosystems. Moreover, experimental results confirm that accuracy can be complete (both False Match Rate and False Non-Match Rate can be zero) and operation is performed in real time.

ACKNOWLEDGEMENTS

This work has been partially supported by TEC2011-24319 and IPT-2012-0695-390000 projects from Ministerio de Economía y Competitividad of the Spanish Government (with support from the PO FEDER-FSE).

References

[1] R. Cappelli, A. Lumini, D. Maio and D. Maltoni, "Fingerprint Image Reconstruction from Standard Templates". IEEE Transactions on Pattern Analysis and Machine Intelligence. 29(9), pp. 1489-1503. 2007.

[2] J. Feng and A. K. Jain, "Fingerprint Reconstruction: From Minutiae to Phase". IEEE Transactions on Pattern Analysis and Machine Intelligence. 33(2), pp. 209-223. 2011.

[3] N. K. Ratha, J. H. Connell and R. M. Bolle, "Enhancing Security and Privacy in Biometrics-based Authentication Systems". IBM Systems Journal - End-to-End Security. 40(3), pp. 614-634. 2001.

[4] D. Maltoni, D. Maio, A. K. Jain and S. Prabhakar, "Handbook of Fingerprint Recognition". 2nd ed. Springer. 2009.

[5] A. T. B. Jin, D. N. C. Ling and A. Goh, "Biohashing: Two Factor Authentication Featuring Fingerprint Data and Tokenised Random Number". Pattern Recognition. 37(11), pp. 2245-2255, Elsevier. 2004.

TABLE I. RESULTS FOR UNPROTECTED AND PROTECTED SYSTEMS

Proposal	Protection Scheme	Unprotected		Protected	
		FMR (%)	*FNMR (%)*	*FMR (%)*	*FNMR (%)*
Directional Image/Finger-Codes [11]	Fuzzy Commitment	1.4	1.4	4.5	4.5
Minutiae [12]	Cancelable Transformation	3.2	3.2	12.5	12.5
Minutiae [10]	Fuzzy Commitment	1.7	1.7	0.0	12.6
Minutiae [13]	Symmetric Hash Functions	1.7	1.7	3.0	3.0
Minutiae [12]	Biohashing Transformation	6.6	6.6	1.8	1.8

TABLE II. COMPARISON OF RECOGNITION RESULTS OBTAINED

ECC	Unprotected (threshold value)	Protected (n, k, t)
BCH code	FMR=5.63, FNMR=5.63 (146 different bits)	FMR=5.84, FNMR=5.44 (2047, 815, 146)
Reed-Solomon code	FMR=5.40, FNMR=5.40 (94 different symbols)	FMR=5.52, FNMR=5.35 (511, 323, 94)

TABLE III. EQUAL ERROR RATES AND PARAMETERS FOR REED-SOLOMON CODES DEPENDING ON THE REALIZATION

Realization	EER	ECC (n, k, t)
Case 1	2.52	(511, 363, 74)
Case 2	1.03	(511, 391, 60)
Case 1 with passwords	0.05	(511, 345, 83)
Case 2 with passwords	0.00	(511, 381, 65)

[6] A. B. J. Teoh, A. Goh and D. C. L. Ngo, "Random Multispace Quantization as an Analytic Mechanism for BioHashing of Biometric and Random Identity Inputs," in IEEE Transactions on Pattern Analysis and Machine Intelligence. 28(12), pp. 1892-1901. 2006.

[7] A. Juels and M. Wattenberg, "A Fuzzy Commitment Scheme," in Proceedings of the 6th ACM Conference on Computer and Communications Security. pp. 28-36. 1999.

[8] P. Li, X. Yang, H. Qiao, K. Cao, E. Liu and J. Tian, "An Effective Biometric Cryptosystem Combining Fingerprints with Error Correction Codes". Expert Systems with Applications. (39), pp. 6562-6574. Springer. 2012.

[9] Y. Imamverdiyev, A. B. J. Teoh and J. Kim, "Biometric Cryptosystem based on Discretized Fingerprint Texture Descriptors". Expert Systems with Applications. 40(5), pp. 1888-1901. Elsevier. 2013.

[10] K. Nandakumar, "A Fingerprint Cryptosystem based on Minutiae Phase Spectrum," in Proceedings of the 2010 IEEE International Workshop on Information Forensics and Security (WIFS). pp. 1-6. 2010.

[11] P. Tuyls, A. H. M. Akkermans, T. A. M. Kevenaar, G. Schrijen, A. M. Bazen and N. J. Veldhuis, "Practical Biometric Authentication with Template Protection," in Proceedings of the 5th Int. Conf. on Audio- and Video-based Biometric Person Authentication (AVBPA). 3546, pp. 436-446. 2005.

[12] A. Nagar, K. Nandakumar, and A. K. Jain, "Biometric Template Transformation: A Security Analysis," in Proceedings of the 2010 SPIE Media Forensics and Security. 7541, pp. 75410O-75410O-15. 2010.

[13] S. Tulyakov, F. Farooq, P. Mansukhani, and V. Govindaraju, "Symmetric Hash Functions for Secure Fingerprint Biometric Systems". Pattern Recognition Letters. 28(16), pp. 2427-2436. Elsevier. 2007.

[14] R. Arjona and I. Baturone, "Método de Identificación de Huellas Dactilares y Dispositivo que Hace Uso del Mismo". Patent Filing Number P201300721. Spain. August 2013.

[15] R. Arjona and I. Baturone, "A Hardware Solution for Real-Time Intelligent Fingerprint Acquisition". Journal of Real-Time Image Processing. 9(1), pp. 95-109, Springer. 2014.

[16] NIST selects the Winner of Secure Hash Algorithm (SHA-3) Competition (2012): http://www.nist.gov/itl/csd/sha-100212.cfm

[17] Fingerprint Recognition Software based on FingerCodes (2006): http://www.advancedsourcecode.com/fingerprint.asp

[18] A. A. Ross, K. Nandakumar, and A. K. Jain. Handbook of Multibiometrics. Springer. London, United Kingdom, 2006.

[19] R. Arjona and I. Baturone, "Model-based Design for Selecting Fingerprint Recognition Algorithms for Embedded Systems," in Proceedings of the 19th IEEE International Conference on Electronics, Circuits and Systems (ICECS). pp. 579-582. 2012.

[20] Implementation for the Hash Function Keccak (2013): http://keccak.noekeon.org/files.html

[21] Integer-Input RS Encoder HDL Optimized Matlab Simulink Block (2013): http://www.mathworks.es/es/help/comm/ref/integerinputrsencoderhdloptimized.html

[22] Integer-Output RS Decoder HDL Optimized Matlab Simulink Block (2013): http://www.mathworks.es/es/help/comm/ref/integeroutputrsdecoderhdloptimized.htm

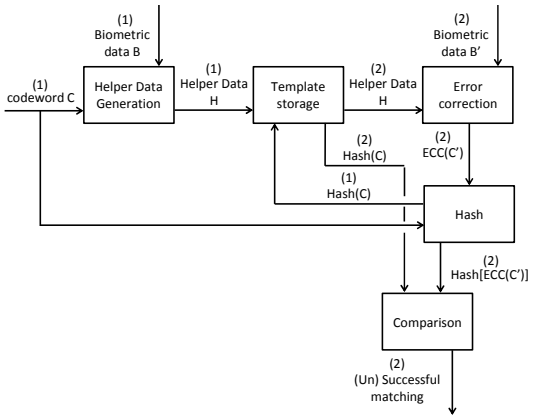


Fig. 1 Fuzzy Commitment scheme. (1) refers to the enrollment phase, and (2) refers to the matching phase.

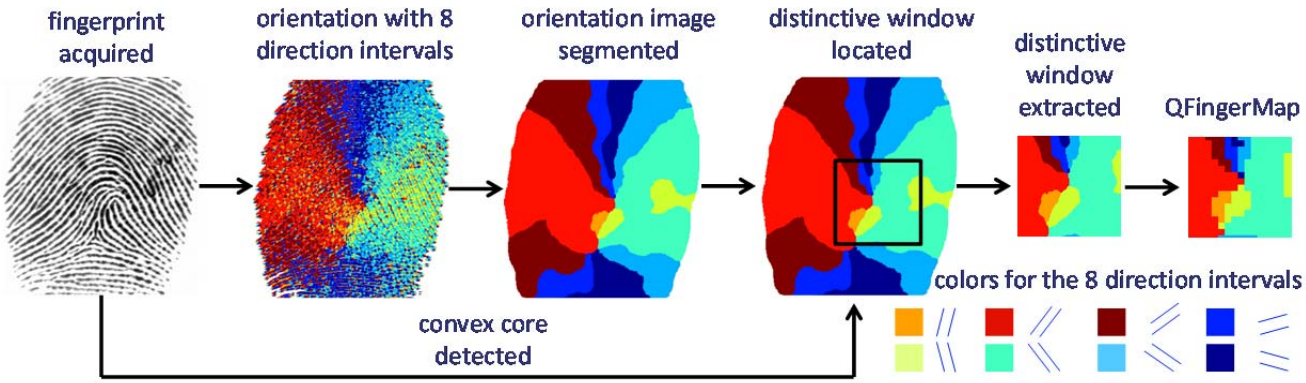
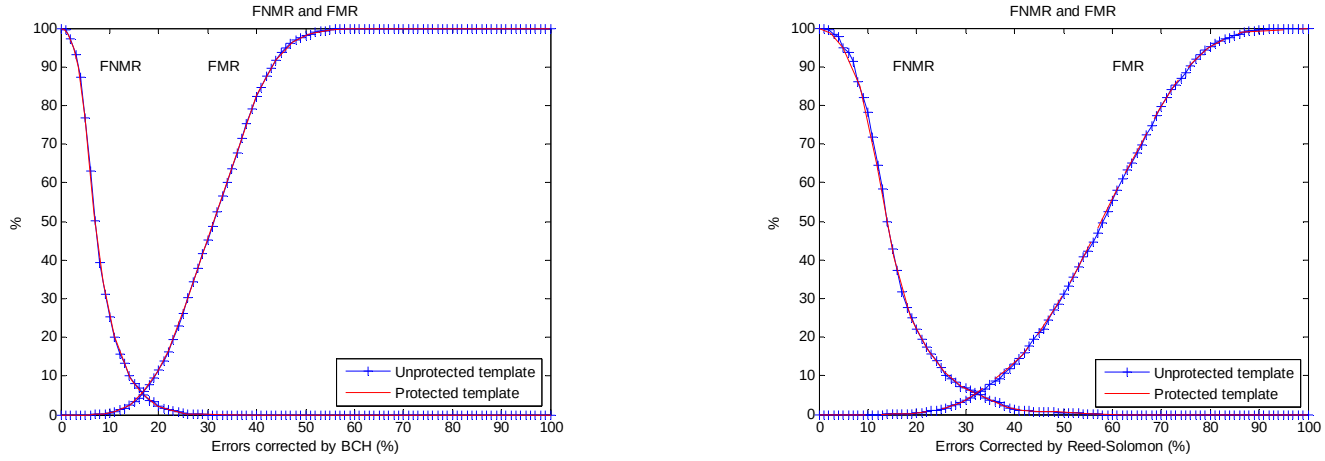


Fig. 2. Extraction process of the feature QFingerMap.



(a) (b)

Fig. 3. FNMR and FMR for QFingerMap-based unprotected and protected systems (a) using BCH and (b) Reed-Solomon codes.

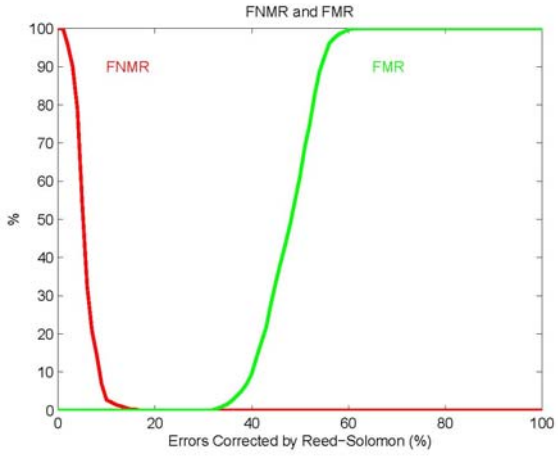


Fig. 4. FNMR and FMR for the case 2 with passwords depending on the number of errors corrected by a Reed-Solomon code.

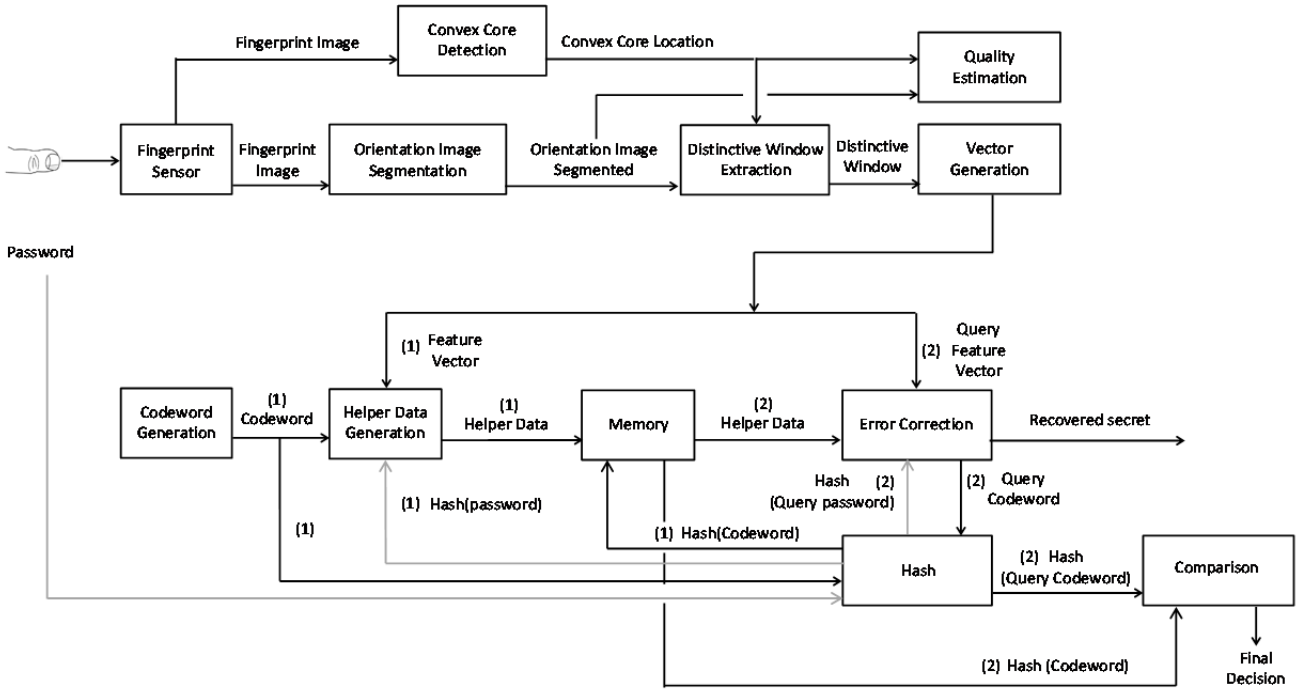


Fig. 5. Block diagram for the crypto-biometric identification/authentication process: (1) enrollment phase, and (2) matching phase. Paths without names are employed in both phases (enrollment and matching). Paths depicted with shaded lines are employed depending on the application.