A Hardware Solution for Real-Time Intelligent Fingerprint Acquisition

Rosario Arjona · Iluminada Baturone

Abstract The first step in any fingerprint recognition system is the fingerprint acquisition. A well-acquired fingerprint image results in high resolution accuracy and low computational effort of processing. Hence, it is very useful for the recognition system to evaluate recognition confidence level, to request new fingerprint samples if the confidence level is low, and to facilitate recognition process if the confidence level is high. This paper presents a hardware solution to ensure a successful and friendly acquisition of the fingerprint image, which can be incorporated at low cost into an embedded fingerprint recognition system due to its small size and high speed. The solution implements a novel technique based on directional image processing that allows not only the estimation of fingerprint image quality but also the extraction of useful information (in particular, singular points). The digital architecture of the module is detailed and their features in terms of area occupation and processing speed are illustrated with implementation results on FPGAs from *Xilinx*. Performance of the solution has been verified with fingerprints from several standard databases that have been acquired with sensors of different sizes and technologies (optical, capacitive, and thermal sweeping).

Keywords

Fingerprint acquisition, fingerprint quality, biometric hardware, FPGA hardware design, CAD tools

1 Introduction

Fingerprint recognition refers to a type of biometric recognition which uses fingerprints for automatically recognizing individuals [Maltoni et al. (2009)]. A fingerprint recognition application requires the processing of fingerprint

(a) (b) (c) (d)

Fig. 1 Examples of fingerprint images from [FVC DB1 (2002)] with: (a) high quality, (b) faulty areas, (c) lack of recognition information, (d) wrong placement.

images to find distinctive features. The process is carried out in two different stages: (1) *Enrolment stage* is applied to storage identification data of a user. These data (known as template) are stored in the system database. (2) *Recognition* is applied to check if a live fingerprint sample matches with an authorized template in the database. If the database contains one or a few templates of the same person, recognition is known as *authentication* (one-to-one matching). If the database contains one or several templates of many persons, recognition is known as *identification* (one-to-many matching). Among biometric systems, fingerprint-based systems are one of the most extended because they offer good recognition rates, friendly environments for users, and low-cost accuracy sensors to acquire the images. Nevertheless, in order to increase the usability of these systems in access control applications or similar scenarios that imply the interaction with the user, it is very important to perform the enrolment and, mainly, authentication/identification process at real-time (i.e. at very few seconds).

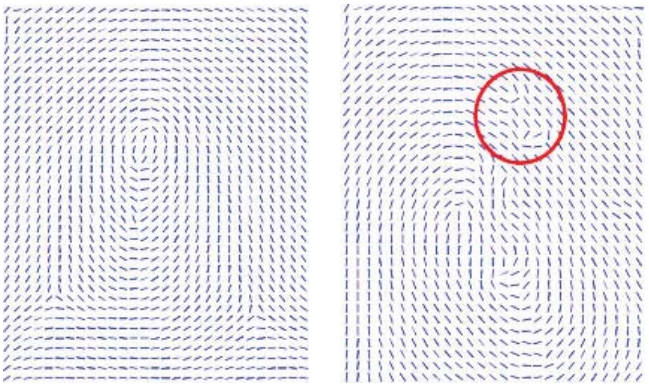
A *fingerprint image* is the reproduction of the exterior appearance of the epidermis and is formed by *ridges* (represented as dark colors) and *valleys* (bright colors). The first step in any fingerprint recognition system is the *fingerprint acquisition*. A well-acquired fingerprint image (Fig. 1a) contains well-defined ridge lines and the sufficient area of the finger so as to allow extracting a complete set of distinctive features. A good image acquisition is essential not only to store an adequate template but also to increase the performance of the recognition stage. Fingerprint images with good quality result in high recognition performance, while images of poor quality reduce the confidence of the biometric system. This is why fingerprint verification software usually provides specific quality measures as a way of predicting the recognition accuracy. Measuring the quality of fingerprint images is so relevant that the National Institute of Standards and Technology (NIST) developed the NIST Fingerprint Image Quality (NFIQ) algorithm in 2004 [Tabassi et al. (2004)], and, later, several other algorithms have been proposed [Alonso-Fernandez et al. (2007)].

Quality of fingerprint images can be affected by several causes. Each person has specific conditions such as scars, age, or type of skin. Moreover, depending on the user behavior (for example, the pressure on the sensor has been too hard or too weak) the image can be acquired with low contrast, low resolution, or with distortion. Finally, environmental conditions such as temperature, humidity, or unclean sensor surface can create faulty fingerprint images [Maltoni et al. (2009)]. Several enhancement algorithms have

been reported to improve the patterns of ridges and valleys in the images [Hong et al (1998)] and [Cheng et al. (2004)]. They usually require quite complex and compute-intensive image processing that have to be carried out by general-purpose microprocessor platforms working at high frequency or specific hardware processors to achieve real-time [Fons et al. (2011)]. Even using such algorithms, some fingerprint images can be unrecoverable. Fig. 1b is an example of low-quality fingerprint image that is difficult to recover. Other captures, which could be considered of high quality from the image processing point of view, should be rejected from the biometric point of view because they do not provide enough information. They are usually the consequence of a bad behavior of the user that does not use the whole sensing area. Two examples are shown in Fig. 1c - 1d. In these cases, it is not possible to recover information about the fingerprint because it has not been inserted. To cope with this concern, the recognition system should determine the quality level of a fingerprint image at real-time and request new samples whenever the acquisition presents low quality. If the quality is high enough, costly enhancement algorithms can be saved from being carried out, which is very interesting for real-time and low-cost embedded systems.

The measures proposed for fingerprint quality estimation can be grouped mainly in two categories: (a) those that evaluate each pixel or block of the image (local estimations), or (b) those that consider a quality value for the whole fingerprint image (global estimations) [Alonso-Fernandez et al. (2007)]. Among the first ones, some approaches evaluate the pixel intensities to estimate block-wise indexes based on statistics (mean, variance, gradients or histogram properties) of ridge frequencies. Other approaches evaluate the reliability of the local direction values of the ridges in the images by using Gabor filters or by processing blocks of the directional image. The *directional image* (also known as *orientation image*, *field* or *map*, or *directional field* or *map*) [Maltoni et al. (2009)] is a fingerprint representation that encodes the local directions of the ridges. Fig. 2 shows the directional images obtained from the fingerprint images in Fig. 1 with the algorithm reported in [Kovesi (2005)]. Fig. 2a is an example of directional image extracted from a high-quality fingerprint image where direction values are homogeneous except where ridges present discontinuities. Fig. 2b is the directional image extracted from a fingerprint capture with faulty image characteristics. Although the algorithm in [Kovesi (2005)] applies image enhancement, directional image shows discontinuities that should not have to appear. A quality measure that is obtained from directional images is the *Orientation Certainty Level* (OCL), which considers that a high-quality fingerprint image offers local directions clearly defined within each image block. Quality measurements based on *Gabor filters* also consider that blocks of the fingerprint image are good if they have a dominant direction. Another score is the *Local Orientation Quality*(LOQ), which compares the local direction of a block with the local directions of the surrounding blocks so as to measure the direction coherence of the ridges in the capture ([Tabassi et al. (2004)], [Alonso-Fernandez et al. (2007)], [Xie et al. (2010)], and [Xie et al. (2012)]).

Among the measures that consider the whole fingerprint image, some of them are also based on processing the information provided by the directional



(a) (b)

Fig. 2 Directional images extracted from fingerprint images in: (a) Fig. 1a, and (b) Fig. 1b.

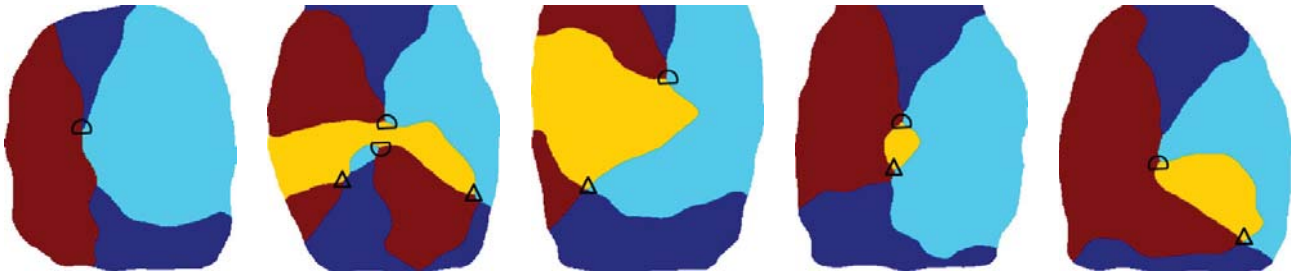
image. For example, a global score has been proposed that accumulates the changes of directions between blocks as a way to evaluate the continuity of the direction field. Direction changes between blocks should be continuous in high-quality images as is shown in Fig. 2a. A similar global score reported computes the standard deviation of the ridge-to-valley thickness ratio to evaluate the uniformity of the capture [Cheng et al. (2005)]. Other indexes at global level take into account the area and the position of the Region of Interest (ROI), which is the recognition area without background.

The detection and location of singular points have been also used to evaluate the global goodness of a fingerprint acquisition. *Singular points* are central features of fingerprint images created by the ridge lines [Maltoni et al. (2009)]. There are two types of singular points: *core points* or points where the ridge lines have maximum curvature (denoted by semi-circles in Fig. 3), and *delta points* or points where three ridge lines intersect (shown as a triangle in Fig. 3). Core points can be *convex* or *concave*, depending on the orientation of the ridges. Singular points are employed in quality estimations since singular areas (located around singular points) offer an important amount of recognition information. If singular points are detected and the core points are placed in the center of the image, the fingerprint has been acquired very well.

Once the fingerprint has been acquired and their features extracted, another essential task in the recognition stage is to compare them with the templates in the database. Comparison is effective if the fingerprints are aligned. However, fingerprints alignment is computationally costly since it usually performs several correlations of two-dimensional matrices to find the best possible alignment considering displacement, rotation and elastic deformation [Maltoni et al. (2009)]. This cost, which can be expensive in authentication applications, can be unaffordable in identification applications with large databases. Hence, some kind of indexing should be applied to reduce the number of comparisons in order to achieve real-time response. A usual practice is to split the fingerprint database into five classes (*arch*, *whorl*,

(a) (b) (c) (d) (e)

Fig. 3 (a) Classes of fingerprint images from FVC database FVC DB1 (2002) with singular points depicted (*convex core* as a semi-circle oriented at the top, *concave core* as a semi-circle oriented at the bottom, and *delta* as a triangle): (a) *arch*, (b) *whorl*, (c) *right loop*, (d) *tended arch*, and (e) *left loop*.



(a) (b) (c) (d) (e)

Fig. 4 Representations based on four homogeneous regions for fingerprint images in Fig. 3. Singular points are located where regions intersect and depend on fingerprint classes: (a) *arch*, (b) *whorl*, (c) *right loop*, (d) *tended arch*, and (e) *left loop*.

right loop, *tended arch*, *left loop*, as shown in Fig. 3), to classify the acquired fingerprint in one of those classes, and to perform matching with only the templates of the same class [Cappelli et al. (1999)].

The technique proposed in this work is a novel analysis of the directional image to extract the singular points and, at the same time, evaluate the quality of the acquired sample in terms of image features and biometric information. The technique has been developed to be efficiently implemented in hardware so as to obtain a hardware module that can be incorporated at low cost into an embedded fingerprint recognition system due to its small size and high speed. The module is able to detect cores and deltas, and can use them as a global measure that indicates well acquired fingerprints: the finger has been placed correctly on the sensor and the image has been acquired with enough quality. The singular points provided by the module can be employed as a reference system to accelerate fingerprint alignment. In addition, singular points allow classifying the acquired fingerprint into arch, whorl, right loop, tended arch, and left loop types. Hence, the module allows accelerating identification tasks. Right loop, tended arch, and left loop have one convex core and one delta. They are distinguished by delta position with

respect to convex core: on the left, aligned, and on the right, respectively. The whorl group has two deltas, one convex core and one concave core. The detection of such cores is enough to classify a fingerprint as a whorl. Finally, arch case does not have delta, but only a special core point. Moreover, the intermediate steps that are carried out by the module to obtain the singular points are exploited to provide other global indexes that evaluate the quality of the fingerprint image. Hence, this hardware solution allows a successful and friendly acquisition of the fingerprint image, so as to solve some of the problems stated above: it is very helpful for the recognition system to evaluate recognition confidence level, to request new samples if the confidence level is low or medium, and to accelerate recognition process if the confidence level is high.

The paper is structured as follows. Sect. 2 describes the hardware-friendly technique proposed. The functional blocks of the digital hardware architecture to implement the technique are explained in Sect. 3. The design flow employed, described in Sect. 4, starts with the high-level description of the algorithm (using *Matlab-Simulink*) and finishes with the hardware implementation of the solution in FPGAs from *Xilinx* (using *Xilinx ISE*). Performance of the hardware solution obtained is illustrated with fingerprint images taken from standard databases, and captured by sensors of different sizes and technologies (optical, capacitive, and thermal sweeping sensors). Implementation results in terms of area occupation and processing speed are also provided in Sect. 4. Finally, conclusions are given in Sect. 5.

2 A proposal for intelligent acquisition

The technique proposed in this paper is a hardware-friendly solution to extract simultaneously the possible singular points in the fingerprint and several indexes to evaluate the quality of the acquired image.

2.1 Singular point detection

The techniques reported in the literature to detect singular points follow mainly two approaches: (a) computation of measures that determine discontinuities in the directional image, and (b) partitioning the directional image into regions with homogeneous direction values [Ohtsuka et al. (2005)]. Examples of the first technique are: (a.1) Poincaré index [Kawagoe et al. (1984)], (a.2) curvature measurement [Mohammadi et al. (2009)], and (a.3) local histogram of the directional image [Srinivasan et al. (1992)]. They evaluate the change of directions around points in the directional image where there is not a dominant direction. Since it is possible to find false and isolated singular points in fingerprint images with low quality, the techniques based on discontinuity measures require an iterative process to ensure the decision. The second technique analyzes the behavior around intersections of homogeneous regions within the directional image: if direction regions converge, then there is a core point; if direction regions diverge, then there is a delta point. The proposal in [Lam et al. (2008)] employs two regions for core point

detection in arch classes and three regions for the rest of fingerprint classes. Singular points in non-arch fingerprints are located where the three areas intersect, while in arch fingerprints the points situated along the separation line between the two areas are analyzed to determine which one shows the maximum curvature in the separation line.

In order to reduce complexity, the approach considered herein to detect singular points explores also the intersections of regions with homogeneous directions within the directional image. Instead of applying a special procedure for arch fingerprints, the technique proposed looks for the intersections of four homogeneous regions considered within the directional image. As can be seen in Fig. 4, four regions allow extracting singular points in all the fingerprint classes (including arch classes). Hence, the first step of the proposed algorithm is to create a coarse directional image from the fingerprint that considers only four direction values.

The algorithms employed to calculate directional images apply gradients or masks [Chen et al. (2009)]. Gradients compute the direction values in a continuous domain while masks select the dominant direction considering a predefined set of directions. Hence, gradients provide more accurate results for the direction values but require higher computational cost. In the gradient-based technique, the direction value D for a pixel (i, j) is obtained as shown in Ec. 1, where G_x and G_y are the horizontal and vertical gradients at each pixel.

$$D(i,j) = \frac{\pi}{2} + tan^{-1}(\frac{G_y(i,j)}{G_x(i,j)}) \tag{1}$$

Horizontal and vertical gradients can be evaluated by using different filters or operators. Among them, the simplest solutions for hardware implementation are the *Sobel* operators (which are usually employed for edge detection). The values of G_x and G_y are obtained after the convolution of windows centered at each pixel of the image with the horizontal and vertical *Sobel* matrices, whose coefficients are integers and a few (unlike *Gaussian* operators, which include more coefficients and with decimal values less suitable for hardware implementations). Once obtained G_x and G_y, the hardware solutions that calculate directional images with gradient-based techniques usually employ CORDIC (COordinate Rotation DIgital Computer) processors to carry out the trigonometric operations ($tan^{-1}(\frac{G_y(i,j)}{G_x(i,j)})$ in Ec. 1).

Since the objective of the proposed technique is to distinguish four regions of directions in the directional image, direction values have to be clustered into four direction prototypes. There are numerous algorithms that can be employed to carry out clustering. The most typical one is *k-means* clustering, which aims at partitioning data into k clusters, finding the most representative element of each cluster. However, implementing *k-means* clustering in hardware would be costly. It is simpler, and equally effective, to select a set of equi-spaced values from 0 ° to 180 ° as the representative values of each cluster (0 ° or 180 °, 45 °, 90 °, and 135 °, considered herein as *c0*, *c1*, *c2* and *c3* clusters, respectively). These values work well with all types of sensors and offer a general solution. A way to obtain the directional image segmented into four homogeneous regions could be to calculate the

Table 1 Processing to obtain clusters from gradient values

IF	Gradient values	Previous quadrant	THEN
1)	G_x=0 OR G_y/G_x>=2.413	-	cluster c0
2)	G_y=0 OR G_y/G_x<0.414	-	cluster c2
3)	G_x=G_y OR (G_y/G_x>0.414 AND G_y/G_x<2.413)	1^{st} and 3^{rd}	cluster c3
4)		2^{nd} and 4^{th}	cluster c1

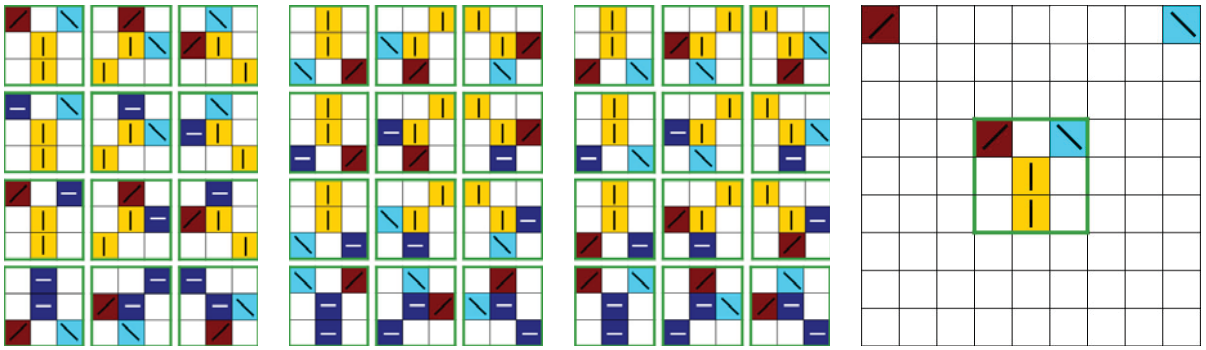
Firstly, G_x and G_y are converted to the first quadrant.

direction values (using Ec. 1) and subsequently replacing each value of the direction by the most similar predefined cluster value. The solution proposed herein is simpler. Once obtained G_x and G_y, the rules detailed in Table 1 can be carried out to determine directly which cluster is associated to that pixel. Hence, trigonometric operations and subsequent clustering is reduced to simple comparisons between G_x and G_y values. In fact, this processing can be seen as an already clustered computation of the arctangent function.

As in any technique that calculates directional images, smoothing process is required also in the proposed technique, particularly because the objective is to obtain homogeneous regions. Among the wide set of filters that can be used to perform smoothing, a non linear filter based on *maximum* operator has been selected. It considers the neighboring pixels inside a window centered at the analyzed pixel and assigns to the pixel the cluster value with the highest number of occurrences inside the window. The window size depends on the sensor employed: its size, resolution, and technology, because the features of the fingerprint images acquired are different. A smaller window may not delete isolated and noisy points and a bigger window may delete too much relevant information for recognition. For the optical, capacitive, and thermal sweeping sensors considered throughout this work, a 27x27 window has been proven to provide good performance.

The second step of the proposed technique is to evaluate if the pattern of a singular point is found at a window centered at the analyzed pixel. The patterns selected to represent the singular points consider a kernel window of 3x3 pixels and a confirmation window of 9x9 pixels. The 3x3 kernel windows considered are shown in Fig. 5. The twelve patterns in Fig. 5a correspond to a convex core or a core of an arch fingerprint, which does not include the vertical direction (in the case of the last row). The patterns in Fig. 5b evaluate the presence of a concave core, and those in Fig. 5c, the possible appearance of a delta. Since fingerprints can be acquired with a certain inclination, the patterns consider small rotations. If one of these 3x3 patterns is found, the decision is confirmed with the presence of specific cluster values in determined pixels of a 9x9 window. For example, if the pattern in the upper left corner of Fig. 5a is found, the decision is validated if the 9x9 pattern shown in Fig. 5d is also found.

Besides using confirmation windows, several rules are taken into account to filter false points and ensure the detection of correct points. Some of these



(a) (b) (c) (d)

Fig. 5 Patterns for detecting singular points on intersections of homogeneous regions. The 3x3 kernel windows for: (a) convex core, (b) concave core, and (c) delta. (d) An example of 9x9 kernel window to confirm the pattern in the upper left corner of (a).

rules evaluate the order in which the patterns are found, because the pixels of the fingerprint image are considered to be processed in a serial way (as will be shown in the following section). For example, if the pattern of a core corresponding to an arch fingerprint is found firstly and later the pattern of a convex core is found, the latter is registered and the former is removed. Another rule is that any delta or concave core is registered if the pattern of a convex core has not been found yet. Other rules evaluate distances to accept or reject points. For example, there should be a minimum distance between convex and concave cores and between convex core and delta.

2.2 Indexes for quality estimation

Simultaneously with the detection of possible singular points, the proposed technique allows computing several indexes to evaluate the quality of the acquired image as well as the reliability of the singular point detection. In order to carry out smoothing, the occurrences of the four representative direction values are counted within a window centered at the analyzed pixel. Considering a window size of 27x27 pixels, the number of occurrences can be 729 at most. The pixels that are assigned to the background instead of to the Region of Interest (ROI) are those whose smoothing windows contain a number of occurrences for the background direction (90 ° or c2) that is close to the maximum value (and are located in the boundaries of the image). Fig. 6 illustrates in black those pixels considered as background (in white, otherwise). Fig. 6 corresponds, respectively, to the fingerprints in Fig. 1. ROI gives information to estimate how fingerprints have been acquired. If the ROI area is wide, the fingerprint recognition area can contain all information needed to complete the recognition properly. Otherwise, the fingerprint acquired will not contain enough area of the finger. Hence, a global index provided by the proposed technique, herein referred to as *IB*, counts the number

(a) (b) (c) (d)

Fig. 6 Region of Interest (ROI) for fingerprint images in Fig. 1, respectively. ROI in white and background in black. ROI area is wide in (a), (b), and (d). (c) has lack of ROI information and, therefore recognition information is insufficient.

(a) (b) (c) (d)

Fig. 7 Smoothing reliability for fingerprint images in Fig. 1, respectively. Pixels whose smoothing is reliable are white, otherwise pixels are black. Smoothing reliability is specially low for (b).

of background pixels. A high value for IB corresponds to an acquisition of low quality.

Concerning smoothing process, a reliable smoothing result for a pixel is that the dominant number of occurrences is higher than a threshold. This is similar to the ideas applied by quality measurements based on local estimations, such as OCL and Gabor-filter techniques already commented in Sect. 1. In this case, the threshold has been chosen as half of the pixels in the window (i.e. 364). Fig. 7 illustrates in black those pixels whose windows do not satisfy that condition (in white, otherwise). As can be seen, if smoothing is reliable for many pixels, the quality of the image is high. Hence, a global index provided by the proposed technique, herein referred to as *IR*, counts the pixels whose smoothing has been reliable. A very lo value for IR corresponds to an acquisition of very low quality.

Those reliable pixels whose dominant directions are c0, c1, c2 (inside the ROI), and c3 are counted by other global indexes, respectively referred to as *I0*, *I1*, *I2*, and *I3*. A very low value for I0, I1, and I3 means that the capture is of low quality (I2 is not considered since it is low in arch-type fingerprints).

The information provided by these indexes can be completed with the results of the detection of singular points. In particular, the detection or not of a convex core should be also considered for quality estimation. If it is not detected, a singular area of the fingerprint has not been acquired and, hence, the capture is of low quality. The binary index with this information is referred to as *ICC*.

Table 2 illustrates all these indexes measured for the fingerprint images in Fig. 1. If the quality indexes do not satisfy certain constraints (imposed

Table 2 Results of quality indexes for fingerprint images in Fig. 1

Fingerprint image	IB	IR	I0	I1	I2	I3	ICC
Fig. 1a	30.6	79.4	15.9	16.5	27.8	19.3	1
Fig. 1b	41.6	38.1	2.6	2.7	25.5	7.3	1
Fig. 1c	74.3	69.4	3.9	0	20.9	44.6	0
Fig. 1d	43.8	70.1	11.8	13.9	41.3	2.9	0

Values for indexes (except ICC) are expresed in %.

Table 3 Results obtained by the proposed technique for the fingerprints in Fig. 3

Fingerprint image	I1	I3	CVC	CCC	D1	D2
Fig. 3a	16.2	24.1	183, 176	-	-	-
Fig. 3b	23.2	15.9	150, 182	180, 179	226, 134	245, 284
Fig. 3c	6.6	23.5	99, 208	-	235, 122	-
Fig. 3d	23.2	27.5	147, 171	-	211, 162	-
Fig. 3e	29.6	13.2	199, 185	-	299, 273	-

by the application), the acquired sample should be rejected and another one should be requested. In the other side, if the quality indexes show that the sample is of high quality, the singular point extracted would accelerate alignment stage and identification, because the relative values of the indexes can be combined with the information provided by the singular points to ensure the classification. In particular, the values of I1 and I3 should be similar for arch, tended arch, and whorl-type fingerprints; the value of I1 should be smaller than I3 for right loop fingerprints, and I1 should be greater than I3 for left loop fingerprints. In the case of high-quality images, the recognition system could even relax certain steps of the processing (for example, enhancement). Table 3 shows the quality indexes I1 and I3 (expressed in % of pixels in the image) obtained by the proposed technique for the fingerprints in Fig. 3. The results named as *CVC*, *CCC*, *D1*, and *D2* are the image locations (expressed in rows and columns) that have been extracted for convex core, concave core, delta1 and delta2 points, respectively.

3 Architecture of the hardware solution

The algorithm described above is very suitable to be implemented in hardware. The architecture developed is thought to receive the pixels of the fin-

gerprint image in a serial way, either as the sensor provides them or as they are read from a memory. A pipeline scheme is employed where each functional block applies several operations to the pixels of the window centered at each analyzed pixel. Processing of the pixels is performed basically in parallel, so that each functional block provides the result in one clock cycle. Hence, the throughput of the hardware module is one pixel per clock cycle while the latency depends on the windows sizes selected for the operations and the number of columns in the fingerprint image.

The hardware module developed carries out the following main steps of the algorithm: (1) generation of the directional image segmented into four regions with homogeneous directions, (2) detection of all types of possible singular points applying pattern recognition, and (3) extraction of quality indexes. The functional blocks considered in the architecture divide these global steps into less complex operations so as to accelerate the system execution. In this respect, the architecture contains five main blocks with the following functionality: (1.1) extraction of the four-cluster directional image and (1.2) smoothing to obtain a well-segmented image; (2.1) singular point detection (applying pattern recognition) and (2.2) singular point post-processing to remove false points and determine the location of the true singular points; and (3) estimation of image quality indexes based on operation results of the smoothing block.

Clustering block The block in charge of computing the four-cluster directional image applies horizontal and vertical Sobel masks to a 3x3 window centered in the current pixel in order to obtain gradient values (G_x and G_y). Mask convolutions involving the 9 pixels are performed in parallel once the 9 pixels are available. For this purpose, two line buffers and two delays are considered to maintain the 9 pixels of the window. After computation of gradient values, the cluster value is selected among the four representative directions accordingly to a combinational circuitry that implements the processing shown in Table 1. The cluster value obtained depends on G_x and G_y values.

As shown in Fig. 8a, two line buffers and two delays are required to provide the input in parallel to the clustering block (*2C+2* 8-bit registers considering values from 0 to 255 for the pixels of a fingerprint image and C columns in the image). The input-output interface of this block can be viewed in Fig. 8b. It receives nine pixels of the fingerprint image coded with 8 bits, computes gradient values with 14 bit-resolution, and results the cluster value coded with 2 bits.

Smoothing blocks The smoothing process considers 27x27 windows of cluster values obtained in the previous step. For each pixel, it evaluates the 729 cluster values inside the window to compute the cluster with the highest number of occurrences. The number of inputs considered within a window at the same time should be small in order to improve hardware implementation (reducing area and latency time). Hence, the 27x27 smoothing is carried out by the concatenation of 9x9 smoothing and 3x3 smoothing. The 3x3 smoothing is firstly performed in parallel because a 3x3 window is the window

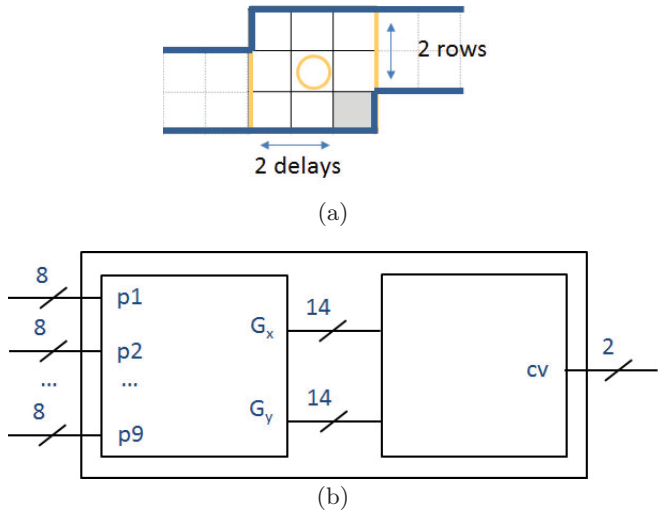


Fig. 8 For clustering block: (a) Number of registers necessary to parallelize the processing of the 3x3 window. The pixel in gray is the current pixel and the cluster direction is computed for the pixel indicated by a circle. (b) Architectural block where *p1-p9* are 8-bit image pixels, G_x and G_y are gradient values coded in 14 bits, and *cv* is the resulting cluster value coded with 2 bits.

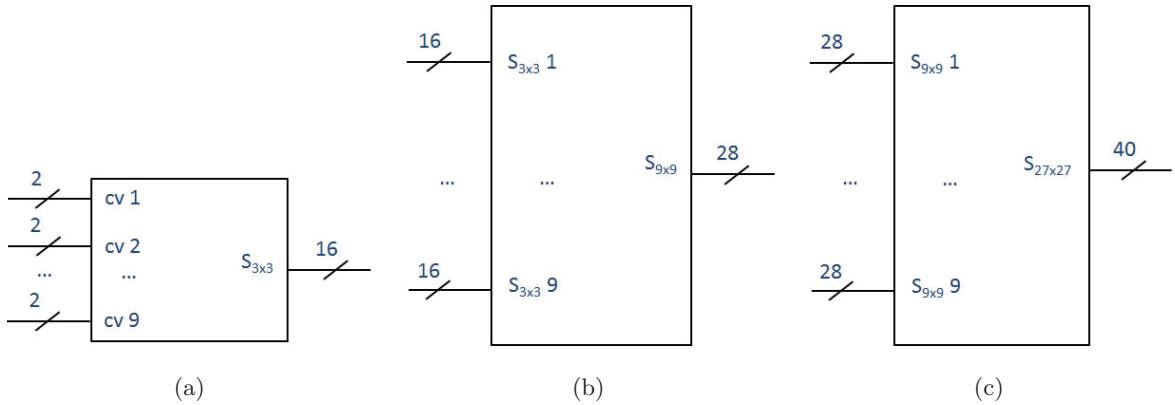


Fig. 9 Architectural blocks for: (a) 3x3 smoothing using cluster values (*cv 1-cv 9*) from clustering operation, (b) 9x9 smoothing using results from nine 3x3 smoothing blocks (S_{3x3} *1*-S_{3x3} *9*), and (c) 27x27 smoothing using results from nine 9x9 smoothing blocks (S_{9x9} *1*-S_{9x9} *9*).

size also processed in parallel by the previous block. The 3x3 smoothing block takes nine cluster values as input (coded with 2 bits) and returns the number of occurrences of the four direction values in that window (coded with 4 bits) as is displayed in Fig. 9a. The 9x9 smoothing block takes nine results from nine 3x3 smoothing blocks (nine counts of each cluster value) and returns the number of occurrences of the four clusters in the 9x9 window (coded with 7 bits) in Fig. 9b. Finally, the 27x27 smoothing block takes nine results from

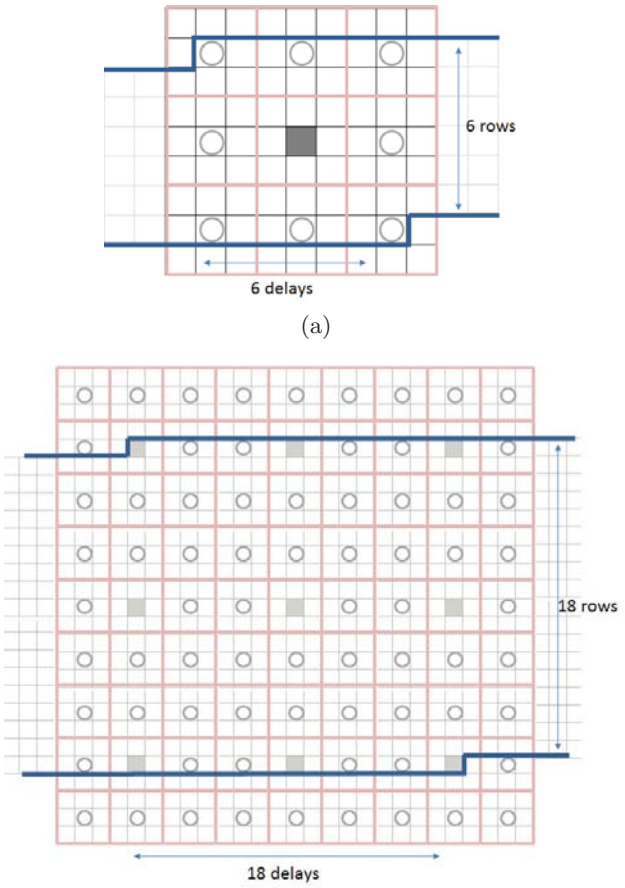


Fig. 10 (a) For 9x9 smoothing operation: number of registers necessary to parallelize a 9x9 window of cluster values from 3x3 smoothing. (b) For 27x27 smoothing operation: number of registers necessary to parallelize a 27x27 window of cluster values from 9x9 smoothing.

nine 9x9 smoothing blocks (nine counts of each cluster value) and returns the number of occurrences of the four clusters in the 27x27 window (coded with 10 bits) in Fig. 9c. In order to process the input data in parallel, two line buffers and two delays are required for the 3x3 smoothing, six line buffers and six delays are required for the 9x9 smoothing (Fig. 10a), and eighteen line buffers and eighteen delays are required for the 27x27 smoothing (Fig. 10b). In terms of registers, 3x3 smoothing block requires *2C+2* 2-bit registers (since the block inputs are cluster values), 9x9 smoothing block requires *6C+6* 16-bit registers (since the block inputs are the number of occurrences of cluster values from the 3x3 smoothing blocks), and 27x27 smoothing block requires an occupation of *18C+18* 28-bit registers (since the block inputs are the number of occurrences of cluster values from the 9x9 smoothing blocks).

Singular point detection block The block for singular point detection is in charge of recognizing the predefined patterns of singular points evaluating different conditions. Patterns are described within a 3x3 window and are verified within a 9x9 window, as indicated in Fig. 4. In order to carry out the process in parallel, eight line buffers and eight delays are required, which means *8C+8* 2-bit registers. To evaluate the appearance of each pattern, four pixels are analyzed in the 3x3 window and 2 pixels in the 9x9 window. The output of the singular point detection block is codified with five bits, each one determining the existence or not of: (a) a complete convex core (the three patterns at the top of Fig. 5a), (b) a partial convex core (the six patterns in the middle of Fig. 5a), (c) a core of arch fingerprints (the three patterns at the bottom of Fig. 5a), (d) a concave core (the twelve patterns in Fig. 5b), and (e) a delta (the twelve patterns in Fig. 5c).

Singular point post-processing block After detecting singular point patterns, it is necessary to apply a post-processing operation which removes false singular points and calculates the location of the true ones within the image. As commented in Sect. 2, this block implements the rules that analyze the order in which the points are found and their relative distances. The 5-bit output provided by the singular point detection block is analyzed by the post-processing block to store the location of the current pixel as a type of singular point or not. The outputs provided are the locations (rows and columns in the fingerprint image) of the four types of singular points that can be extracted: convex core, concave core, delta1, and delta2. The register sizes to store these locations depend on the number of rows and columns in the image, for example, 9-bit registers are required by a fingerprint image with 374 rows and 388 columns. In the case of the convex core, the register includes an additional bit that indicates if it is a core of an arch fingerprint or not.

Block for estimating quality indexes The approach to determine quality indexes computes the number of pixels that satisfy certain conditions. Hence, this block contains six counters, one for each quality index: IR, IB, I0, I1, I2, and I3. The inputs to the block are the number of occurrences for each cluster value computed by the 27x27 smoothing block (4 inputs of 10 bits). The bitsize provided for each index depends on the fingerprint image size, except for the index ICC, which has always 1 bit.

Latency, which indicates when the first valid value is obtained, depends mainly on the 27x27 smoothing, which is the slowest operation. If we consider a current pixel in the processing as in Fig. 11, it can be seen the operations completed in that point. Since all the blocks described above execute their operations in one clock cycle, the number of clock cycles between the time a pixel is introduced in the module and the time the module determines if it corresponds to a singular point is *18C +21*. To calculate the quality indexes, the whole image should be processed, which requires basically as many clock cycles as pixels in the image.

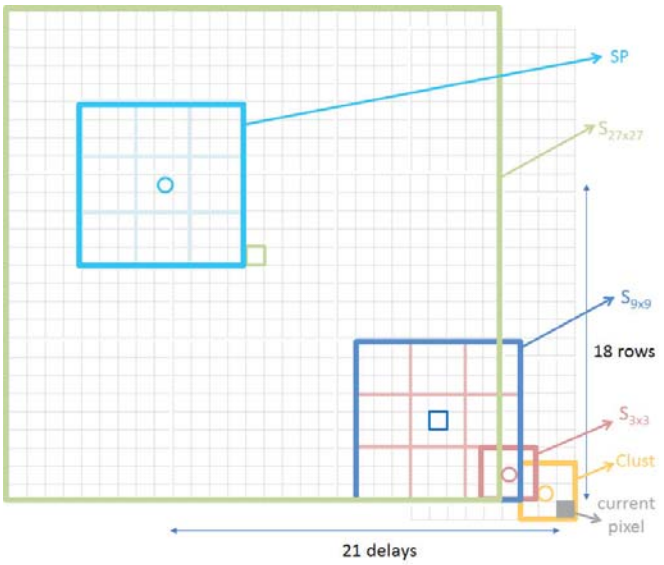


Fig. 11 Operations which have been completed in a fixed current input pixel. *Clust*, S_{3x3}, S_{9x9}, S_{27x27}, *SP* are clustering, 3x3 smoothing, 9x9 smoothing, 27x27 smoothing, and singular point detection blocks, respectively

4 Design methodology and digital implementation

A top-down design flow has been employed to implement this solution in hardware. *Matlab-Simulink* and *Xilinx ISE* CAD tools facilitate the hardware implementation following a low-cost design flow. It is composed by several stages at different levels: (1) high-level description, (2) hardware code generation, and (3) device implementation.

The functionality of the complete architecture has been verified at high level with *Matlab-Simulink*. A software implementation for the proposed technique has been developed in *Matlab* and this solution has been translated to a dynamic model description in *Simulink* that includes the functional blocks described in the previous section. In order to simulate the acquisition by a sensor, the fingerprint image is read as if it is captured and stored in a memory. A serialize block reads the data and provides pixels one by one to the next block (the clustering block). The *Simulink* model considers hardware parameters such as the number of bits (data are processed in fixed point), delays, and the line buffers employed by each block. The performance of each block as well as the whole system can be evaluated and compared with the software results for each fingerprint image considered.

Matlab-Simulink is provided with tools that generate HDL Code from high-level implementations in an automatic way. One of these tools is *Simulink HDL Coder*, which supports HDL code generation, including an associated *testbench*. *HDL Coder* eases the construction of DSP algorithms into FPGAs or ASICs and reduces the complexity of the design process. The generated VHDL or *Verilog* description is independent on the device. In this work,

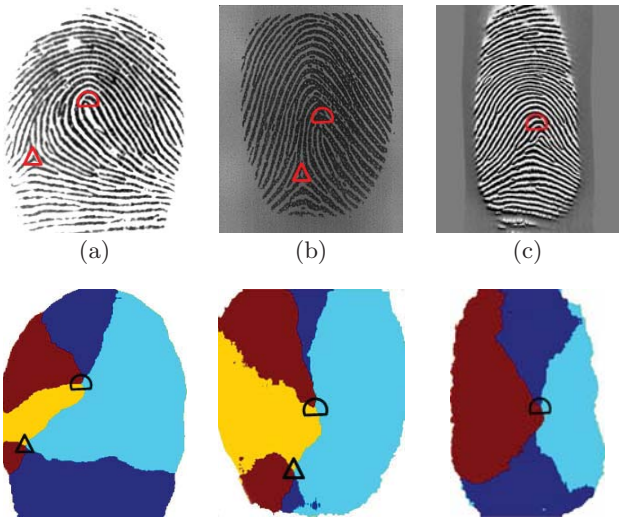


Fig. 12 Examples of fingerprint images from: (a) [FVC DB1 (2002)] DB1, (b) [FVC DB1 (2002)] DB3, and (c) [FVC DB1 (2006)] DB3. Singular points detected by our proposal are indicated. 27x27 smoothing representation for the corresponding fingerprint images with singular points detected where direction regions intersect.

VHDL description has been selected to generate code from the model described in *Simulink*. All the functional blocks are described in VHDL except the modules associated to the image reading and serializing, which are considered as signal conditioning blocks.

Since FPGAs have been selected as target devices, the CAD tools from *Xilinx ISE* have been employed to carry out the hardware implementation. They provide implementation details concerning resource utilization and timing. Moreover, *ISE Isim* simulator has been used to verify the circuit described in VHDL code at a hardware level. Therefore, system performance has been evaluated at different levels (at the software, model, and hardware levels).

The resulting hardware solution has been proven with fingerprint images from different types of sensors. In particular, with fingerprint images from the standard databases [FVC DB1 (2002)] DB1 and DB3, and [FVC DB1 (2006)] DB3 (corresponding to optical, capacitive and thermal sweeping sensors, respectively). When the quality of the image is high and the singular point detection is reliable, the singular points in the fingerprint are correctly detected by the hardware solution. Several examples are shown in Fig. 12. When the quality is low, the indexes IB to ICC already put on notice that confidence level of recognition is low. For example, for the fingerprints in Fig. 1, the indexes obtained by the hardware module coincide with those shown in Table 2 (obtained by software).

Implementation results in terms of timing and area depend on the image size because the size of line buffers that allow the parallel data processing is

Table 4 Block implementation results for different image sizes

Image size (RxC)	Implementation results	Clust	S_{3x3}	S_{9x9}	S_{27x27}	SP	SP Post	Quality indexes	Global result
96x96	Area (no. of slices)	126	64	304	830	111	161	191	1787
	Max. Frequency (MHz)	582	582	582	582	582	241	306	241
236x192	Area (no. of slices)	159	88	446	1441	196	113	96	2539
	Max. Frequency (MHz)	582	582	582	582	582	252	289	252
374x388	Area (no. of slices)	227	163	760	2672	370	203	109	4504
	Max. Frequency (MHz)	582	582	582	582	582	216	281	216

Table 5 Comparative of directional image implementations

Proposal	Spartan 3 [Cantó et al. (2009)]	Altera EPXA10 [Fons et al (2007)]	Spartan 3 [García et al (2007)]	Altera Byte-Blaster [Chao et al. (2005)]	Spartan 3A (our proposal)	Spartan 3A (our proposal with CORDIC)
Image size (RxC)	512x280	516x280	256x256	288x224	374x276	374x276
Frequency (MHz)	-	50	100	33	264	81
No. of slices	2468	9123	575	-	454	854
Execution time (ms)	12	25	262	1.2	0.39	1.27

directly proportional to the number of columns of the image. To illustrate this issue, the hardware implementation has been done into the same FPGA (a *Xilinx Virtex 5*) considering three image sizes (96x96, 236x192, and 374x388). 96x96 fingerprint images are captured by reduced area sensors suitable for low-cost and embedded systems, 236x 192 fingerprint images are typically used in smart cards, and 374x388 are standard sizes of fingerprint images [Neurotechnology (2012)]. The occupation and timing results for each block are shown in Table 4. Area is expressed by number of slices: 1787 slices (24.82% of slices of the FPGA), 2539 slices (35.26% of slices of the FPGA), and 4504 (62.56% of slices of the FPGA), for the three cases. The biggest block is the 27x27 smoothing block. Throughput is one clock cycle because operations within each block are performed in parallel. Maximum frequency depends on the slowest operation, which is singular point post-processing block: 241 MHz for 96x96 fingerprint images, 252 MHz for 236x192, and 216 MHz for 374x388

ngerprint images.

Results for maximum frequencies prove that this solution is suitable for working with either commercial or novel academic sensors proposed in the literature. For example, 10 MHz and 25 MHz ([Kim et al. (2008)] and [Fons et al. (2005)], respectively). Real-time constraints are satisfied since the total execution time for acquiring and processing a fingerprint image is low: 0.04 ms for 96x96 fingerprint images, 0.21 ms for 236x192 fingerprint images, and 0.67 ms for 374x388 fingerprint images (working at the maximum frequencies of

241 MHz, and 216 MHz for the two last cases). In contrast, 0.45 s, 1.78 s, and 6.51 s, respectively, are obtained for the software implementation executed into a Intel core i7-2620M processor (2.70 GHz) and 8 GB of RAM. Hence, this module could pave the way to include the whole recognition system into an embedded system.

At the best of our knowledge, there are not proposals for similar intelligent fingerprint acquisition systems in dedicated hardware. Comparisons between different approaches are difficult because most of biometric solutions are developed in software or employ HW/SW co-design and coprocessors. In [Militello et al. (2008)], the most typical technique for singular point detection (*Poincaré Index*) is implemented into a *Xilinx Virtex II* FPGA working at 25 MHz and invests a time of 31.20 ms to process an image of 560x296 pixels. With respect to directional image extraction, there are several implementations (without considering clustering), whose timing and area results are presented in Table 5. Since the proposed solution applies a coarse segmentation of a fingerprint image, it is not necessary to obtain a perfect directional image. The simplicity of our proposal to calculate the clustered directional image is based on applying an approximation for the arctangent function and assigning directly the cluster values associated to relations between gradient values. The typical process is to compute the direction values by using trigonometric functions (computed by a CORDIC module) and apply a posterior clustering. Obviously, the implementation using CORDIC module is worse in terms of area and speed. From the results in Table 5, it can be seen that the proposed solution provides a considerable reduction of complexity together with a high speed.

5 Conclusions

A hardware module has been designed that computes simultaneously the singular points of a fingerprint and a set of indexes that evaluate the quality of the sample acquired. The module implements a novel algorithm that is very suitable for hardware implementation. It has been designed by following a low-cost design flow supported by CAD tools from *Matlab-Simulink* and *Xilinx ISE*, which allows verifying the functionality at different abstraction levels (from software to hardware implementation in FPGAs). This has allowed testing its correct behaviour with fingerprints taken from several standard databases, which corresponds to different sensors. Its features concerning area occupation and processing speed are very competitive, making this module very suitable for embedded and real-time systems. In particular, it is able to work at frequencies higher than those employed by sensors of commercial and research scenarios. The use of this module in a fingerprint recognition system would increase the information taken from the fingerprint in the acquisition stage, which would improve the enrolment and recognition stages in terms of reliability and time response.

References

Alonso-Fernandez, F., Fierrez, J., Ortega-Garcia, J., Gonzalez-Rodriguez, J, Fronthaler, H., Kollreider, K., Bigun J. A Comparative Study of Fingerprint Image-Quality Estimation Methods. IEEE Transactions on Information Forensics and Security, 2. 734-743 (2007)

Cantó, E., Fons, M., López, M., Ramos, R. Acceleration of Complex Algorithms on a Fast Reconfigurable Embedded System on Spartan-3. International Conference on Field Programmable Logic and Applications (2009)

Cappelli, R., Lumini, A., Maio, D., Maltoni, D. Fingerprint Classification by Directional Image Partitioning. IEEE Transaction on Pattern Analysis and Machine Intelligence, 21. 402-421 (1999)

Chao, G., Lee S., Hornq, S. Embedded Fingerprint Verification System. Proceedings of the 11th International Conference on Parallel and Distributed Systems (2005)

Chen, D., Ji, X., Fan, F., Zhang, J., Guo, L., W. Meng. Comparative Analysis of Fingerprint Orientation Field Algorithms. Proceedings of the 5th International Conference on Image and Graphics (2009)

Cheng, J.G., Tian, J. Fingerprint Enhancement with Dyadic Scale-Space. Pattern Recognition Letters, 25. 1273-1284 (2004)

Chen, Y., Dass, S., Jain, A. Fingeprint Quality Indices For Predicting Authentication Performance. Proceedings of the 5th International Conference on Audio- and Video-based Biometric Person Authentication. Springer-Verlag (2005)

Fingerprint commercial scanners,
http://www.neurotechnology.com/cgi-bin/fingerprint-scanners.cgi

Fons, M., Fons, F., Canyellas, N., Cantó, E., López, M. Hardware-Software Codesign of an Automatic Fingerprint Acquisition System. Proceedings of the IEEE International Symposium on Industrial Electronics, (2005)

Fons, F., Fons, M., Cantó, E., López, M. Flexible hardware for fingerprint Image Processing. Research in Microelectronics and Electronics Conference, PRIME (2007)

Fons, F., Fons, M., Cantó, López, M. Real-time Embedded Systems Powered by FPGA Dynamic Partial Self-Reconfiguration. A Case Study Oriented to Biometric Recognition Applications. Journal of Real-Time Image Processing, Special Issue. 1-23 (2011)

FVC 2002 database,
http://bias.csr.unibo.it/fvc2002/

FVC 2006 database,
http://bias.csr.unibo.it/fvc2006/

García, M. L., Navarro, E. F. C. FPGA Implementation of a Ridge Extraction Fingerprint Algorithm Based on Microblaze and Hardware Coprocessor. International Conference on Field Programmable Logic and Applications (2006)

Hong, L., Wan, Y., Jain, A. Fingerprint Image Enhancement: Algorithm and Performance Evaluation. IEEE Transactioins on Pattern Analysis and Machine Intelligence, 20. 777-789 (1998)

Kawagoe, M., Tojo, A. Fingerprint Pattern Classification. Pattern Recognition, 17. 295-303 (1984).

Kim, S., Lee K., Han, S., Yoon E. A CMOS Fingerprint System-on-a-Chip With Adaptable Pixel Networks and Column-Parallel Processors for Image Enhancement and Recognition. IEEE Journal of Solid State Circuits, 43. 2558-2567 (2008)

Kovesi, P. Directional image algorithm,
http://www.csse.uwa.edu.au/p̃k/research/matlabfns/

Lam, H. K., Hou, Z., Yau, W. Y., Chen, T. P., Li, J. A Systematic Topological Method for Fingerprint Singular Point Detection. 10th International Conference on Control, Automation, Robotics and Vision (2008)

Maltoni, D., Maio, D., Jain, A. K., Prabhakar, S. Handbook of Fingerprint Recognition. 2nd ed., Springer (2009)

Militello, C. , Conti, V. , Sorbello, F., and Vitabile, S. A Novel Embedded Fingerprint Authentication System based on Singularity Points. International Con-

ference on Complex, Intelligent and Software Intensive Systems (2008)
Mohammadi, S., Farajzadeh, A. Fingerprint Reference Point Detection Using Orientation Field and Curvature Measurements. IEEE International Conference on Intelligent Computing and Intelligent Systems (2009)
Ohtsuka, T., Takahashi, T. A New Detection Approach for the Fingerprint Core Location Using Extended Relation Graph. IEICE Transactions on Information and Systems, 88. 2308-2312 (2005)
Srinivasan, V. S., Murthy, N. N. Detection of Singular Points in Fingerprint Images. Pattern Recognition, 25. 139-153 (1992)
Tabassi, E., Wilson, C. L., Watson, C. I. Fingerprint Image Quality. NIST Internal Report 7151, National Institute for Standards and Technology (2004)
Xie, S.J., Yoon, S., Shin, J., Park, D. S. Effective Fingerprint Quality Estimation for Diverse Capture Sensors. Sensors, 10. 7896-7912 (2010)
Xie, S.J., Yang, J., Gong, H., Yoon, S., Park, D. S. Intelligent Fingerprint Quality Analysis using Online Sequential Extreme Learning Machine. Soft Computing - A Fusion of Foundations, Methodologies and Applications, Springer (2012)