2011 3rd International Conference on Machine Learning and Computing (ICMLC 2011)

Fingerprint Re-alignment: A Solution Based on the True Fingerprint Center Point

Ishmael S. Msiza, Brain Leke-Betechuoh, and Tendani Malumedzha
Biometrics Research Group – Information Security
CSIR, Modelling & Digital Science Unit
Pretoria, Republic of South Africa
e-mail: {imsiza, blekebetechuoh, tmalumedzha}@csir.co.za

***Abstract*—Fingerprint re-alignment is an important aspect in the subject of fingerprint biometrics because a fingerprint that has been rotated, voluntarily or involuntarily, can have negative impact on the functionality of an automated fingerprint recognition system. A fingerprint recognition system that does not attempt to re-align an involuntarily rotated fingerprint can create opportunities for falsely rejecting a legitimate subject. It can also create opportunities for security attacks where an illegitimate subject voluntarily rotates their fingerprint, as a way of tempering with the functionality of the recognition system. This manuscript presents a new fingerprint re-alignment solution that uses the True Fingerprint Center Point (TFCP) as a reference, where the TFCP is introduced as the actual center of the fingerprint image foreground. Using the TFCP as a reference point proves – in general – to be fast, independent, and efficient for all types of fingerprints. It, in particular, is useful when dealing with the type of fingerprints that belong to the Plain Arch (PA) fingerprint class.**

Keywords*—*fingerprint rotation; fingerprint re-alignment; plain arch; true fingerprint center point

I. Introduction

The performance of an automated fingerprint recognition system, often quantified in terms of match/non-match rates and execution speed, can be negatively affected by a number of realities. The two main instances of these realities are fingerprint translation and fingerprint rotation. Fingerprint translation refers to the position of a captured fingerprint with respect to the fingerprint image background, that is, the capturing window of a fingerprint reader. For optimum functionality of an automated fingerprint recognition system, the fingerprint should be positioned in the central area of the fingerprint image background, as depicted in figure 1 (a). The fingerprint in figure 1 (b) is translated downwards and, because a very small area of its central part has been captured, could compromise the functionality of the system's matching module. This is because this module relies on the availability of a certain minimum number of features for its functionality, and this minimum number may not be met for a translated fingerprint image.

Fingerprint rotation refers to the alignment of a captured fingerprint when measured against the image background. For optimum functionality of a fingerprint recognition system, the captured image should sit at an angle of approximately 0 degrees (more or less upright) relative to the vertical axis, as depicted in figure 2 (a). The anti-clockwise rotated version of this fingerprint, as depicted in figure 2 (b), has lost some of the information that is captured in the upright fingerprint. For example, the upright fingerprint has both singular points [1] – the core (circle) and the delta (triangle) – captured, while the rotated fingerprint only has the core captured. An immediate implication of this reality is that the functionality of the singular point detection module of the fingerprint recognition system is compromised. A core is a fingerprint landmark located at the inner-most turning point of the fingerprint loop. A delta is a fingerprint landmark located at the point where the fingerprint ridges combine to form some triangulation.

Figure 1. The same fingerprint (a) centrally positioned, and (b) translated downwards

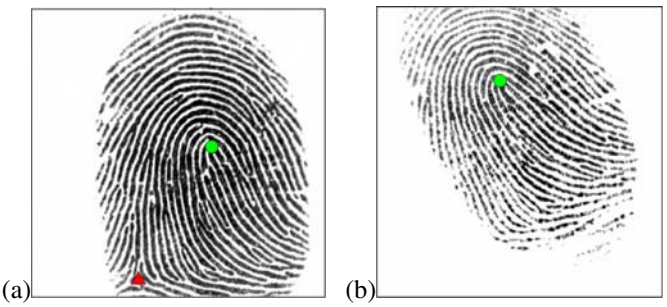

Figure 2. The same fingerprint (a) uprightly positioned, and (b) rotated anti-clockwise

A compromised fingerprint singular point detection module could, in turn, compromise the functionality of a fingerprint classification module that relies on the co-ordinate geometry of these singular points. In their 1996 manuscript, Karu and Jain [2] present an example of such a fingerprint classification scheme. Fingerprint translation and fingerprint rotation are equally problematic, however they cannot be fully exploited in the same discussion. This manuscript, therefore, only focuses on addressing the problem of fingerprint rotation. A rotated

978-1-4244-9253-4 /11/$26.00 ©2011 IEEE

fingerprint is normally dealt with by making use of the concept of fingerprint re-alignment, the subject of the next section.

II. The Theory of Fingerprint Re-alignment

Re-aligning a fingerprint involves the use of some landmark as a reference point. The re-alignment angle and the direction – clockwise or anti-clockwise relative to the vertical axis – can only be computed after the successful location of the chosen reference point. Maltoni *et al* [3] suggest that most fingerprint matching algorithms can withstand an involuntary rotation of up to 20 degrees, in either direction. The argument is that, rotation beyond 20 degrees is certainly not involuntary, hence the subject could be making an attempt to tamper with the functionality of the fingerprint recognition system. Such tampering should not be permitted and a message for re-capture should be issued the minute it is established that the captured fingerprint is more than 20 degrees out of line with the vertical axis.

This, therefore, implies that it is more ideal to compute the re-alignment angle as early as possible in the fingerprint analysis chain of processes, so as to prevent an unnecessary waste of computational resources on a fingerprint that would be rejected. Literature reveals that practitioners normally use two main landmarks as reference points for image re-alignment, the fingerprint core and the fingerprint center point. The following sub-sections exploit the disadvantages of using these landmarks as reference points for the fingerprint re-alignment problem.

A. The Fingerprint Core as a Reference Point

The 1982 manuscript that was presented by Wegstein [4] is one example of using a fingerprint core as a reference point. He then uses the core-delta segment, if the delta is present, to estimate the re-alignment angle. This approach – however – completely fails in one instance, and is likely to fail in a number of other instances. The instance where this approach completely fails is when faced with a pattern that belongs to the Plain Arch (PA) fingerprint class. Fingerprints that belong to the PA class have neither a core nor a delta, as depicted in figure 3 (a). Without a reference point, the core, it becomes impossible to re-align a rotated PA fingerprint.

One instance where this technique is likely to fail is when faced with a pattern that belongs to the Tented Arch (TA) fingerprint class. Fingerprints that belong to the TA class have a single core and a single delta, with the core located almost directly above the delta, as depicted in figure 3 (b). Many singular point detection techniques, however, find it relatively difficult to detect this delta and – in some instances – both the core and the delta. With only the core and not the delta detected, the reference point – the core – becomes useless because this method relies on a core-delta segment to re-align the fingerprint.

Another instance where this method is likely to fail is when faced with the task of re-aligning fingerprints that belong to the Left Loop (LL) and the Right Loop (RL) fingerprint classes. By definition, fingerprints that belong to these classes

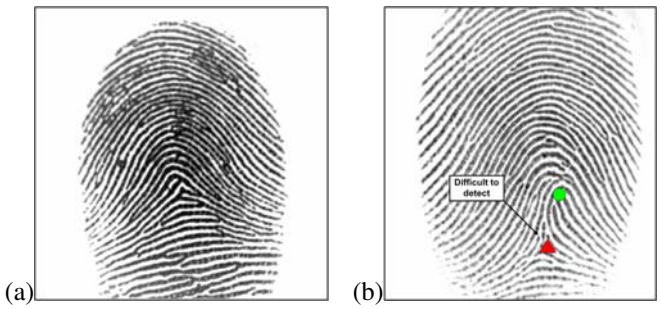

Figure 3. Centrally positioned and upright arch fingerprints (a) Plain Arch (PA), and (b) Tented Arch (TA)

should have both a core and a delta, with the only difference being the gradient of the line joining the core and the delta. However, fingerprints from these classes are, in most instances, impressed in such a way that the delta is not captured – even when the fingerprint is neither translated nor rotated. Figure 4 (a) depicts an upright and centrally located LL fingerprint missing a delta, while figure 4 (b) depicts an upright and centrally located RL fingerprint also missing a delta.

Figure 4. Centrally positioned and upright loop fingerprints, both, missing deltas (a) Left Loop (LL), and (b) Right Loop (RL)

B. The Fingerprint Center as a Reference Point

In their 1999 manuscript, Jain *et al* [5] define the fingerprint center point (FCP) as the point of maximum curvature of the concave ridges in a fingerprint image, as depicted in figure 5. The work of Chan and Abu-Bakar [6] is one example of the application of the FCP in the fingerprint re-alignment problem. As apparent in figure 5, the FCP is very close to the core, hence its position is estimated using the core detection algorithm [7]. Like the method that uses the fingerprint core as the reference point, this approach completely fails when faced with the task of re-aligning a fingerprint that belongs to the PA class.

C. The Current State of Affairs

Because – with the core or the FCP as a reference – practitioners generally find it difficult to deal with fingerprints that belong to the PA class, Lam *et al* [8] recently introduced a technique that can detect a reference point in fingerprints of this type. The only problem with this recent development is that this technique is dedicated only to those fingerprints that belong to the PA class. This is a problem because it implies that the practitioner is now faced with a challenge of,

Figure 5. Fingerprint center point (FCP) marked (×) at the point of maximum curvature

first, determining the class of a given fingerprint, then use the appropriate reference point location method, before re-aligning the fingerprint. This current solution is summarized in figure 6, where image preprocessing refers to contrast enhancement through – for instance – normalization [9], and ridge segmentation through – for instance – variance discrimination [10].

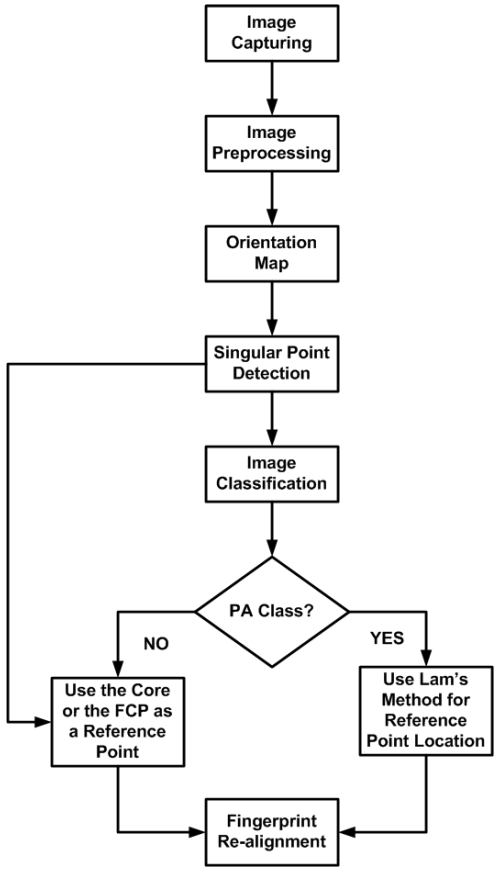

Figure 6. The current solution to the fingerprint re-alignment problem

The solution depicted in figure 6 does work, however inefficient it may be. It is inefficient because it involves a lot of image processing before the ultimate goal of re-aligning the fingerprint is realized. As an example, fingerprint classification is a process that is dependent on a number of other preliminary processes. Depending on how it is done, it may rely on a process such as singular point detection [11]. Singular point detection may, in turn, depend on a process such as orientation image computation [1]. If, for example, fingerprint re-alignment is an important requirement of a chosen quality control process, it means that the quality control module will have to be somewhere in the bottom order of the fingerprint analysis chain of processes. Any fingerprint recognition system that has a quality control module located in the bottom order of the fingerprint manipulation process, can be regarded as an inefficient system. Given this reality, there is a need for a more efficient solution – proposed in the next section – which determines a reference point and re-aligns the fingerprint in the top order of the fingerprint analysis chain of processes.

III. The Proposed Solution

The solution proposed in this manuscript is graphically summarized in figure 7. Looking at both figures 6 and 7, it is apparent that the solution proposed in this manuscript is simple, efficient, and computationally inexpensive. The following sub-sections discuss the key components of this newly proposed solution.

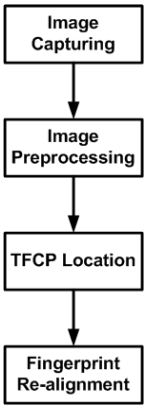

Figure 7. The new solution proposed for the fingerprint re-alignment problem

A. Image Preprocessing

An important image preprocessing operation is that of separating the fingerprint image ridge area – the foreground – from the image background, and it is known as fingerprint segmentation. From this operation, it is possible to generate an image known as the mask of the fingerprint and it contains two colors. A white color – binary one – is assigned to the image foreground, while a black color – binary zero – is assigned to the image background, as depicted in figure 8 (b). The method for obtaining this mask image is discussed by a number of practitioners, including Thai [10] and Hong *et al* [9]. Msiza *et al* [12] also discuss this approach, with an emphasis on the most suitable variable parameters to use for optimum segmentation.

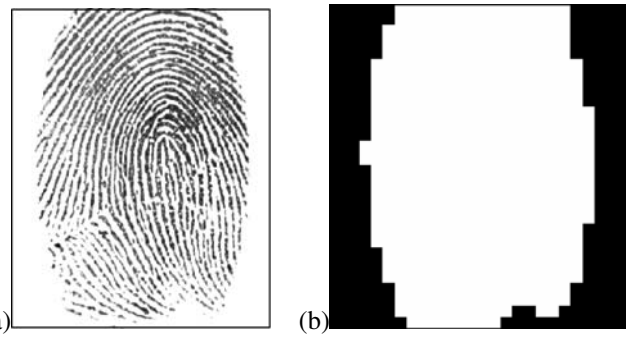

Figure 8. Fingerprint segmentation: (a) original image, and (b) mask image

B. True Fingerprint Center Point (TFCP) Location

The coordinates, $(x_f,\ y_f)$, of the TFCP are tracked through, both, horizontal and vertical navigation in the white area of the fingerprint mask. The x-coordinate, x_f, is tracked through the process of navigating horizontally, while the y-coordinate, y_f, is tracked through vertical navigation. An important question to answer is: where does this navigation process begin?

1) The Starting Point: The starting point of the navigation is informed by the fact that, under normal conditions, a participating subject will always impress their fingerprint in such a way that the center of the fingerprint image background, $(x_b,\ y_b)$, is always located somewhere within the image foreground – this is an involuntary action. For an image of width W and height H, the x-coordinate of the background center is given by:

$$x_b = 0.5 \times W, \tag{1}$$

while the y-coordinate is given by:

$$y_b = 0.5 \times H. \tag{2}$$

An experiment conducted on a set of 800 fingerprint images from database 1(a) of the year 2000 Fingerprint Verification Competition (FVC 2000), confirmed that the point $(x_b,\ y_b)$ is always located somewhere within the white area, under involuntary conditions. This affirmation was further vindicated by a series of three experiments, one conducted on 800 fingerprint images from database 1(a) of FVC 2002, and another conducted on 8000 fingerprint images from the NIST 4 special database. The implication is that this affirmation can, by far, be regarded as fact because it is backed by experiments conducted on three different and widely used fingerprint databases:

- FVC 2000 - a database containing real fingerprint images scanned from pieces of paper;
- FVC 2002 - a database containing synthetic fingerprint images; and
- NIST 4 - a database containing fingerprint images that have been endorsed by the National Institute of Standards and Technology (NIST), for algorithm development.

If the point $(x_b,\ y_b)$ is not located in the image foreground, then there is no image captured, or, a participating subject is voluntarily making an attempt to tamper with the functionality of a fingerprint recognition system. Because any possible tampering is a security threat, ensuring that the point $(x_b,\ y_b)$ lies within the white area can be added as one of the security layers in a fingerprint recognition system.

2) Horizontal Navigation: The horizontal navigation takes place in two stages, namely, a navigation towards the right, and a navigation towards the left. The right-ward navigation starts at the point $(x_b,\ y_b)$ and increases – in steps of 1 pixel – the value of the x-coordinate, while the y-coordinate remains unchanged. The value of this x-coordinate is increased up to the pixel that marks the interface between the mask image's white area and the black area, if it exists. Alternatively, it is increased up to the pixel that marks the right-most edge of the image background. This extreme-right x-coordinate is recorded and stored in x_r.

Similarly, the left-ward navigation starts at the point $(x_b,\ y_b)$ and decreases the value of the x-coordinate, while the y-coordinate remains constant. The value of the x-coordinate is decreased up to the pixel that marks the interface between the mask image's white area and the black area, if it exists. Alternatively, it is decreased up to the pixel that marks the left-most edge of the image background. This extreme-left x-value is recorded and stored in x_l. Following this two-stage navigation, it now becomes possible to compute the x-coordinate of the TFCP. It is given by:

$$x_f = \frac{(x_r + x_l)}{2}. \tag{3}$$

3) Vertical Navigation: Similarly, the vertical navigation occurs in two stages, namely, the upward navigation and the downward navigation. The upward navigation starts at the point $(x_b,\ y_b)$ and decreases – in steps of 1 pixel – the value of the y-coordinate, while the x-coordinate remains unchanged. The y-coordinate is decreased up to the pixel that marks the interface between the mask image's white area and the black area, if it exists. Alternatively, it is decreased up to the pixel that marks the upper-most edge of the image background. This extreme-top y-coordinate is recorded and stored in y_u.

The downward navigation, also, starts at the point $(x_b,\ y_b)$ and increases the value of the y-coordinate, while the x-coordinate remains constant. The value of the y-coordinate is increased up to the point that marks the interface between the mask image's white area and the black area, if it exists. Alternatively, it is increased up to the pixel that marks the lower-most edge of the image background. This extreme-bottom y-value is recorded and stored in y_l. Following this two-stage navigation, it now becomes possible to compute the y-coordinate of the TFCP. It is given by:

$$y_f = \frac{(y_u + y_l)}{2}. \tag{4}$$

4) Algorithmic Description: To further clarify it, the TFPC location procedure is described in algorithm 1, where the variables x_r, x_l, y_u, and y_l act as temporary holders of the x- and y-coordinates during the horizontal and the vertical navigations. The variable y_u is, however, useful for the ultimate re-alignment stage and, for this reason, should not be discarded. It is important to note that the check done in line

Algorithm 1: A procedure that locates the TFCP based on the fingerprint mask image

Input : Fingerprint mask image, M
Output : TFCP coordinates, $(x_f,\ y_f)$
Constants: W, the width of M; and H, the height of M
Variables : x_r; x_l; y_u; and y_l

```
Compute: (x_b, y_b) using equations 1 and 2;
if M(x_b, y_b) = 1 then
    Initialize: x_r = x_b;
    while M(x_r, y_b) = 1 AND x_r <= W do
        increment: x_r;
    end
    Initialize: x_l = x_b;
    while M(x_l, y_b) = 1 AND x_l >= 0 do
        decrement: x_l;
    end
    Initialize: y_u = y_b;
    while M(x_b, y_u) = 1 AND y_u >= 0 do
        decrement: y_u;
    end
    Initialize: y_l = y_b;
    while M(x_b, y_l) = 1 AND y_l <= H do
        increment: y_l;
    end
    Compute: (x_f, y_f) using equations 3 and 4;
end
Terminate;
```

2 is, as mentioned before, always satisfied under involuntary conditions.

C. Fingerprint Re-alignment

The re-alignment of a fingerprint is characterized by two main procedures. First, determine the re-alignment direction and, second, determine the re-alignment angle. If a fingerprint is rotated anti-clockwise, as depicted in figure 9, the re-alignment direction is clockwise. Figure 10 shows a fingerprint that is rotated clockwise and, in this instance, the re-alignment direction is anti-clockwise. An upright fingerprint, as expected, does not require any re-alignment.

Algorithm 2 summarizes the procedure for determining the re-alignment direction, while algorithm 3 summarizes the one for determining the re-alignment angle. The variable y_u, that serves as one of the inputs in algorithm 2, is the same variable that has been carried on from algorithm 1. The variables $TFCP$, $upperEdge$, C_1, C_2, C_3, C_4, and Q – as used in algorithm 1 and/or algorithm 2 and/or algorithm 3 – are the same ones that are shown as symbols in figures 9 and 10.

IV. Solution Assessment

A summarized assessment of the proposed solution, measured against the already existing solutions, is shown in table I. When using the fingerprint core or the fingerprint center point (FCP) as a reference, the re-alignment process becomes slow, inefficient, and dependent. It is slow because – after fingerprint capturing and pre-processing – it has to go through other fingerprint processing modules, as depicted earlier in figure 6. If the classification module indicates that the given fingerprint belongs to the Plain Arch (PA) class, this process then becomes solely dependent on Lam's method for reference point location. In the case where – after computing the re-alignment angle – it is established that the rotation was voluntary, the re-alignment module that uses the core or the FCP as a reference introduces a significant amount of inefficiencies in the fingerprint recognition system. As mentioned before, voluntary rotation is characterized by a re-alignment angle greater than 20 degrees in either direction. It is inefficient to let the image go through all the modules indicated in figure 6, only to find out that it has to be recaptured.

On the other hand, the newly introduced re-alignment technique is fast, independent, and efficient. It is fast because, as depicted in figure 7, it does not have to through all the fingerprint processing modules indicated in figure 6. It is independent because it does not rely on Lam's method, even when dealing with fingerprint that belong to the PA class.

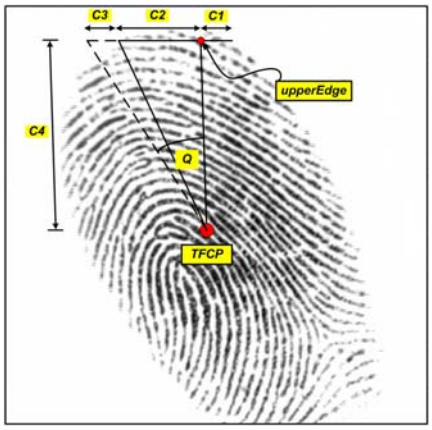

Figure 9. A fingerprint that has been rotated in the anti-clockwise direction, and needs to be re-aligned towards the clockwise direction

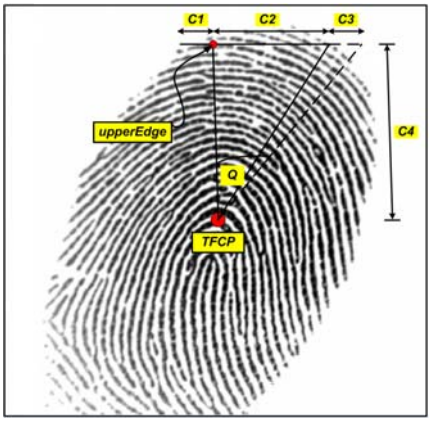

Figure 10. A fingerprint that has been rotated in the clockwise direction, and needs to be re-aligned towards the anti-clockwise direction

Algorithm 2: A procedure that determines the re-alignment direction of a rotated fingerprint

Input : Fingerprint mask image, M; x_f; and y_u
Output : Re-alignment direction, D
Constants: W, the width of M
Variables : x_u; C_1; and C_2

Initialize: $x_u = x_f$;
Start: at the point $(x_u,\ y_u)$, the $upperEdge$;
if $M(x_u,\ y_u) = 1$ **then**
  while $M(x_u,\ y_u) = 1$ *AND* $x_u <= W$ **do**
    increment: x_u;
  end
  Compute: $C_1 = x_u - x_f$;
  Re-initialize: $x_u = x_f$;
  while $M(x_u,\ y_u) = 1$ *AND* $x_u >= 0$ **do**
    decrement: x_u;
  end
  Compute: $C_2 = x_f - x_u$;
  if $C_1 < C_2$ **then**
    D is $clockwise$;
  else if $C_2 < C_1$ **then**
    D is $anti - clockwise$;
  else
    D is $upright$ (no re-alignment required);
  end
  Re-initialize: $x_u = x_f$;
end
Terminate;

Algorithm 3: A procedure that determines the re-alignment angle of a rotated fingerprint

Input : Re-alignment direction, D; C_1; C_2; y_f; & y_u
Output : Re-alignment angle, Q
Variables: C_3 and C_4

Initialize: $Q = 0$ degrees;
if D *is not* $upright$ **then**
  Compute: $C_4 = y_f - y_u$;
  if D *is* $clockwise$ **then**
    Add: an offset, $C_3 = C_1$, to the left of C_2;
  else if D *is* $anti - clockwise$ **then**
    Add: an offset, $C_3 = C_1$, to the right of C_2;
  end
  $Q = arctan(divide(sum(C_2, C_3), C_4))$ degrees;
end
Terminate;

It is efficient because, when subjected to voluntary rotation, the fingerprint recognition system can quickly detect it and ask for an image re-capture. This is because of the fact that, with this newly proposed solution, the fingerprint re-alignment procedure is one of the early procedures in the analysis chain of the fingerprint recognition process.

TABLE I
A SUMMARIZED ASSESSMENT

Reference	Procedure	Dependency	Efficiency
Core	Slow	High	Low
FCP	Slow	High	Low
TFCP	Fast	Low	High

V. CONCLUSIONS AND FUTURE WORK

This document presented a new method for computing, both, the re-alignment direction and angle of a rotated fingerprint. This method is new because it uses a reference point that has never been used before in literature, the true fingerprint center point (TFCP), which was introduced as the geometric center of the fingerprint image foreground. With fingerprint re-alignment being a topical subject – by far – for close to three decades, this newly introduced technique is the first to demonstrate fast and efficient execution, as well as independence from Lam's method, even when faced with fingerprint patterns that belong to the Plain Arch (PA) fingerprint class. Possible future work includes the application of this technique in fingerprint processing modules such as quality assessment, singularity detection, and matching; as another way of interrogating its credibility.

REFERENCES

[1] M. E. Mathekga and I. S. Msiza, "A Singular Point Detection Algorithm Based on the Transition Line of the Fingerprint Orientation Image," Proc. of the 20th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA), Dec. 2009, pp. 137-142.

[2] K. Karu and A. K. Jain, "Fingerprint Classification," Pattern Recognition, vol. 29, Mar. 1996, pp. 389-404, doi:10.1016/0031-3203(95)00106-9.

[3] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of Fingerprint Recognition. London: Springer, 2003.

[4] J. H. Wegstein, "An Automated Fingerprint Identification System," U.S. Government Publication, U.S. Department of Commerce, National Bureau of Standards, 1982.

[5] A. K. Jain, S. Prabhakar, and L. Hong, "A Multichannel Approach to Fingerprint Classification," IEEE Trans. on Pattern Anal and Machine Intelli., vol. 21, Apr. 1999, pp. 348-359, doi:10.1109/34.761265.

[6] Y. H. Chan and S. A. R. Abu-Bakar, "Fingerprint Center Point Location Using Directional Fields," Proc. of the Third Intl. Conf. on Image and Graphics (ICIG), Dec. 2004, pp. 286-289, 10.1109/ICIG.2004.67.

[7] L. Hong and A. K. Jain, "Classification of Fingerprint Images," MSU Technical Report MSUCPS:TR98-18, 1998.

[8] H. K. Lam, Z. Hou, W. Y. Yau, T. P. Chen, J. Li, and K. Y. Sim, "Reference Point Detection for Arch Type Fingerprints," Lecture Notes in Computer Science, vol. 5558, Jun. 2009, pp. 666-674, doi:10.1007/978-3-642-01793-3_68.

[9] L. Hong, Y. Wan, and A. K. Jain, "Fingerprint Image Enhancement: Algorithm and Performance Evaluation," IEEE Trans. on Pattern Anal. and Machine Intelli., vol. 20, Aug. 1998, pp. 777-789, doi:10.1109/34.709565.

[10] R. Thai, "Fingerprint Image Enhancement and Minutiae Extraction," Honors Project Report, School of Computer Science & Software Engineering, University of Western Australia, 2003.

[11] I. S. Msiza, B. Leke-Betechuoh, F. V. Nelwamondo, and N. Msimang, "A Fingerprint Pattern Classification Approach Based on the Coordinate Geometry of Singularities," Proc. of the IEEE Intl. Conf. on Syst., Man, and Cybern. (SMC), Oct. 2009, pp. 516-523, doi:10.1109/ICSMC.2009.5346860.

[12] I. S. Msiza, M. E. Mathekga, F. V. Nelwamondo, and T. Marwala, "Fingerprint Segmentation: A Investigation of Various Techniques and a Parameter Study of a Variance-Based Method," Intl. J. of Innovative Computing, Info., and Control, in press.