A Novel Pattern Matching Approach for Fingerprint-based Authentication

Moudhi AL-Jamea, Tanver Athar, Costas S. Iliopoulos
Solon P. Pissis, M. Sohel Rahman

Department of Informatics
King's College London
London, UK
e-mail:{mudhi.aljamea,tanver.athar,costas.iliopoulos,solon.pissis,sohel.rahman}@kcl.ac.uk

Abstract—**In Biometrics, fingerprint is still the most reliable and used technique to identify individuals. This paper proposes a new fingerprint matching technique, which matches the fingerprint information by using algorithms for approximate circular string matching. The minutiae information is transformed into string information by using a series of circles, which intercepts the minutiae and that information into a string. This string fingerprint information is then matched against a database by using approximate string matching techniques.**

Keywords–Biometrics, fingerprints, matching, verification, orientation field.

I. INTRODUCTION

Recently, the need for automatic person identification has increased more and more in our daily activities, in general, and in the world of business and industry, in particular. To this end, the use of biometrics has become ubiquitous [1], [2]. Biometrics refers to metrics related to human characteristics and traits. Since biometric identifiers are unique to individuals, automatic person identification systems based on biometrics offer more reliable means of identification than the classical knowledge-based schemes such as password and personal identification number (PIN) and token based schemes such as magnetic card, passport and driving license. Among all the various forms of biometrics including face, hand and finger geometry, eye, voice and speech and fingerprint [3], the fingerprint-based identification is the most reliable and popular personal identification method.

Fingerprints offer an infallible means of personal identification and has been used for person authentication since long. Possibly, the earliest cataloguing of fingerprints dates back to 1891 when the fingerprints of criminals were collected in Argentina [4]. Now, it is used not only by police for law enforcement, but also in commercial applications, such as access control and financial transactions; and in recent times in mobile phones and computers.

In terms of applications, there are two kinds of fingerprint recognition systems, namely, verification and identification. In the former, the input is a query fingerprint with an identity (ID) and the system verifies whether the ID is consistent with the fingerprint and then outputs either a positive or a negative answer depending on the result. On the contrary, in identification, the input is only a query fingerprint and the system computes a list of fingerprints from the database that resemble the query fingerprint. Therefore, the output is a short (and possibly empty) list of fingerprints.

The majority of research in recent times has focused only on the fingerprint authentication, but not on the rotation of fingerprints. The majority of the state-of-the-art assumes that the fingerprint is aligned in the same direction as that of the stored fingerprint images. This is an important aspect of fingerprint matching, which various techniques have ignored, and only very few, in the literature [5], have considered. With the introduction of fingerprint detection in mobile devices, the rotation aspect of the fingerprint detection is an important area of research.

A. Our Contribution

In this paper, we revisit the fingerprint recognition problem that is the basis of any fingerprint based identification system. Despite a plethora of fingerprint matching algorithms there is still room for improvement [6]. Interestingly enough, in spite of similarities between the two domains, there has not been much work at the intersection of algorithms on strings and the study of fingerprint recognition. To the best of our knowledge, here we make a first attempt to employ string matching techniques to solve fingerprint recognition problem efficiently and accurately. Converting the fingerprint image into string results in a small string. Matching this string against other fingerprint images stored as strings can be done in time linear with respect to the total length of the strings. In our approach, we have formulated an algorithm to detect and verify a fingerprint regardless of its position and rotation in wide scanning surface area in a simple and efficient way.

B. Road Map

The organization of the rest of this paper is as follows. In Section II, we present some background related to fingerprints. Section III presents a very brief literature review. We present our approach in Section V after discussing some preliminaries in Section IV. Finally, we briefly conclude in Section VI.

II. BACKGROUND

Fingerprint pattern can be simply defined as the combination of ridges and grooves on the surface of a fingertip. The inside surfaces of the fingers contain minute ridges of skin with furrows between each ridge. The ridges and furrows run in parallel lines and curves to each other forming complicated patterns. The basic fingerprint (FP) patterns are whorl, loop, and arch [7]. However, the most common and widely used classification method is based on Henry's classification [8] [9] which contain 8 classes: Plain Arch, Tented Arch, Left Slant

Loop, Right Slant Loop, Plain Whorl, Double-Loop Whorl Central-Pocket Whorl, and Accidental Whorl (see Figure 1).

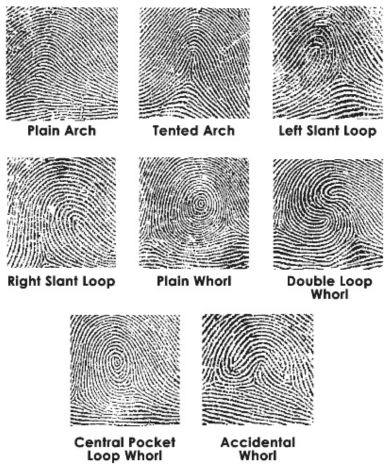

Figure 1. Classification of fingerprint patterns

Each fingerprint is highly stable and unique. This uniqueness is determined by global features like valleys and ridges, and by local features like ridge endings and ridge bifurcations, which are called minutiae. According to recent studies, the probability of two individuals having the same fingerprint is less than one in a billion [10].

Fingerprinting has been used historically to identify individuals using the so-called ink-technique [11], where the fingers are dabbed with ink to get an imprint on paper cards which are then scanned to produce the digital image. In this off-line fingerprint acquisition technique, the fingerprints are matched by using the scanned images produced above. This method is still very important and popular especially in the forensics field, where fingerprints are captured from crime scenes. However, this type of off-line methods are not feasible for biometric systems [12]. The other approach is of-course to scan and match fingerprints in real time.

III. Related works

Fingerprint recognition has been the centre of studies for a long time and as a result, many algorithms/approaches have been proposed to improve the accuracy and performance of fingerprint recognition systems. In the fingerprint recognition literature, a large body of work has been done based on the minutiae of fingerprints [13]–[16]. These works consider various issues including, but not limited to, compensating for some of the non-linear deformations and real word distortion in the fingerprint image. As a trade off with accuracy, the issue of memory and processor intensive computation has also been discussed in some of these works.

The minutiae-based matching are the most popular approach due to the popular belief that minutiae are the most discriminating and reliable features [17]. However, this approach faces some serious challenges related to the large distortions caused by matching fingerprints with different rotation (see Figure 2). As a result, researchers have also used other features for fingerprint matching. For example, the algorithm in [18] works on a sequence of points in the angle-curvature domain after transforming the fingerprint images into these points. A filter-based algorithm using a bank of Gabor filters to capture both local and global details in a fingerprint as a compact fixed-length finger code is presented in [19]. The combinations of different kind of features have also been studied in the literature [20], [21]. There exist various other works in the literature proposing different techniques for fingerprint detection based on different feature sets of fingerprints [22], [23], [15]. Due to brevity we do not discuss these works in this paper. Interested readers are referred to a very recent review by Unar et al. [2] and references therein.

Note that, in addition to a large body of scientific literature, a number of commercial and propitiatory systems are also in existence. In the related industry, such systems are popularly termed as Automatic Fingerprint Identification System (AFIS). One issue with the AFIS available in the market relates to the sensor used to capture the fingerprint image. In particular, the unrealistic assumption of the most biometric systems that the fingerprint images to be compared are obtained using the same sensor, restricts their ability to match or compare biometric data originating from different sensors [24]. Another major challenge of commercially available AFISs is to increase the speed of the matching process without substantially compromising accuracy in the application context of identification, especially, when the database is large [6]. This is why the quest for even better fingerprint recognition algorithms is still on particularly in the high-performance computing context [6].

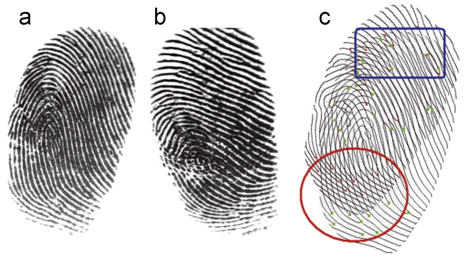

Figure 2. An example of large distortion from FVC2004 DB1 [25]

IV. Preliminaries

In order to provide an overview of our results and algorithms, we begin with a few definitions. We think of a string x of length n as an array $x[0..n-1]$, where every $x[i]$, $0 \le i < n$, is a letter drawn from some fixed alphabet Σ of size $\sigma = |\Sigma|$. The empty string of length 0 is denoted by ε. A string x is a factor of a string y if there exist two strings u and v, such that $y = uxv$. Let the strings x,y,u, and v, such that $y = uxv$. If $u = \varepsilon$, then x is a prefix of y. If $v = \varepsilon$, then x is a suffix of y. In the string $x = aceedf$, ac is a prefix, ee is a factor and df is suffix.

Let x be a non-empty string of length n and y be a string. We say that there exists an occurrence of x in y, or, more simply, that x occurs in y, when x is a factor of y. Every occurrence of x can be characterised by a position in y. Thus, we say that x occurs at the starting position i in y when $y[i..i+n-1] = x$.

A circular string of length n can be viewed as a traditional linear string, which has the left- and right-most symbols wrapped around and stuck together in some way [26]. Under this notion, the same circular string can be seen as n different linear strings, which would all be considered equivalent. Given a string x of length n, we denote by $x^i = x[i..n-1]x[0..i-1]$, $0 < i < n$, the i-th rotation of x and $x^0 = x$. Consider, for instance, the string $x = x^0 = abababbc$; this string has the following rotations: $x^1 = bababbca$, $x^2 = ababbcab$, $x^3 = babbcaba$, $x^4 = abbcabab$, $x^5 = bbcababa$, $x^6 = bcababab$, $x^7 = cabababb$. Here we consider the problem of finding occurrences of a pattern x of length m with circular structure in a text t of length n with linear structure. This is the problem of circular string matching.

The problem of exact circular string matching has been considered in [27], where an $O(n)$-time algorithm was presented. The approach presented in [27] consists of preprocessing x by constructing a suffix automaton of the string xx, by noting that every rotation of x is a factor of xx. Then, by feeding t into the automaton, the lengths of the longest factors of xx occurring in t can be found by the links followed in the automaton in time $\mathcal{O}(n)$. In [28], an average-case optimal algorithm for exact circular string matching was presented and it was also shown that the average-case lower bound for single string matching of $\Omega(n \log_\sigma m/m)$ also holds for circular string matching. Very recently, in [29], the authors presented two fast average-case algorithms based on word-level parallelism. The first algorithm requires average-case time $O(n \log_\sigma m/w)$, where w is the number of bits in the computer word. The second one is based on a mixture of word-level parallelism and q-grams. The authors showed that with the addition of q-grams, and by setting $q = \Theta(\log_\sigma m)$, an average-case optimal time of $O(n \log_\sigma m/m)$ is achieved.

The Approximate Circular String Matching via Filtering (ACSMF) algorithm [30] is used here in order to identify the orientation of the fingerprint. The basic principle of algorithm ACSMF is the partitioning scheme that splits the concatenation of the circular pattern string into $2d + 4$ fragments, where d is the maximum edit distance allowed. The Aho-Corasick automaton [31] is then used to search for the fragments against the text. Once a fragment is identified, the fragment is extended on both left and right directions to determine a valid match.

Theorem 1 ([30]): Given a pattern x of length m drawn from alphabet Σ, $\sigma = |\Sigma|$, a text t of length $n > m$ drawn from Σ, and an integer threshold $d = \mathcal{O}(m/\log_\sigma m)$, algorithm ACSMF requires average-case time $\mathcal{O}(n)$.

V. Our Approach

In this section we present our main contribution, i.e., a novel pattern matching approach to solve the fingerprint recognition problem. As has been discussed above, two main difficulties related to the fingerprint recognition problem are lack of a fixed orientation and the presence of errors in the scanned image due to various reasons (e.g., the presence of dust, oil and other impurities on the finger and on the scanning surface). We therefore employ a two-stage algorithm. We start with a brief overview of our algorithm below.

A. Algorithmic Overview

Our algorithm consists of two distinct stages, namely, the *Orientation Identification* stage and the *Matching and Verification* stage.

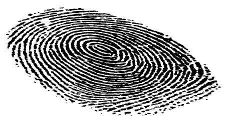
Figure 3. Left-oriented fingerprint

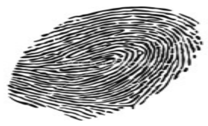
Figure 4. Right-oriented fingerprint

1) Stage 1 – Orientation Identification: When scanning a fingerprint, the user can place the finger on the scanning device at different angles. It could be aligned to left (see Figure 3) or right (see Figure 4). In fact, the position of the finger can be placed anywhere on the scanning surface. The scanning surface usually is somewhat larger compared to the fingerprint surface area. Hence, the first challenge is to exactly pinpoint the location and area of the fingerprint impression on the scanning surface.

The second challenge of course is to identify the orientation of the fingerprint. Without identifying the proper orientation, we can not properly compare it with the fingerprint(s) in the database and the recognition will no be possible. The task of this stage (i.e., Stage 1) is to identify and locate the fingerprint with its correct orientation.

2) Stage 2 – Verification and Matching: Like all other fingerprint recognition systems a database is maintained with fingerprint information against which the input fingerprint will be matched. In the database, we will store a black and white image. Once the orientation of the input fingerprint has been identified, we can easily reorient the fingerprint impression (according to the standard format stored in the database) and then the matching algorithm runs. Since finger print can contain dust, fudges, etc., the scanned information may contain errors which means that an exact match with the existing data is highly unlikely. So, in this stage (i.e., Stage 2) we perform an error tolerant matching in an effort to recognize the input fingerprint against the database of the system.

B. Details of Stage 1: Orientation Identification

In this stage we employ a novel approach based on circular templates as follows. Let us use f_i to denote the image of the input fingerprint. Let us assume that we know the appropriate center point, p of f_i. We then can convert f_i to a representation consisting of multiple circular bit streams by extracting circular segments of the image. This is achieved by constructing k concentric circles C_j of radius $r_j, 1 \leq j \leq k$, with center at point p. For each circle we obtain minutiae features of the image by storing 1 wherever the edge of a circle intersects with a ridge and a 0 if it intersects with a furrow [see Figure 5]. So, in this way, for f_i, we get k concentric circles, which can be transformed into k circular binary strings [see Figure 6]. Clearly, this procedure can be easily applied on a fingerprint data stored on the database. In what follows, we will use $Y_j, 1 \leq j \leq k$ to denote the k circular strings obtained after applying the above procedure on a fingerprint data stored in the database. In what follows, we may slightly abuse the notation and say the Y_j corresponds to the circle of radius r_j.

Now to identify the location and orientation of the input fingerprint we generalize the above approach to extract the minutiae feature and apply the approximate circular string matching algorithm of [30] as described below (please refer

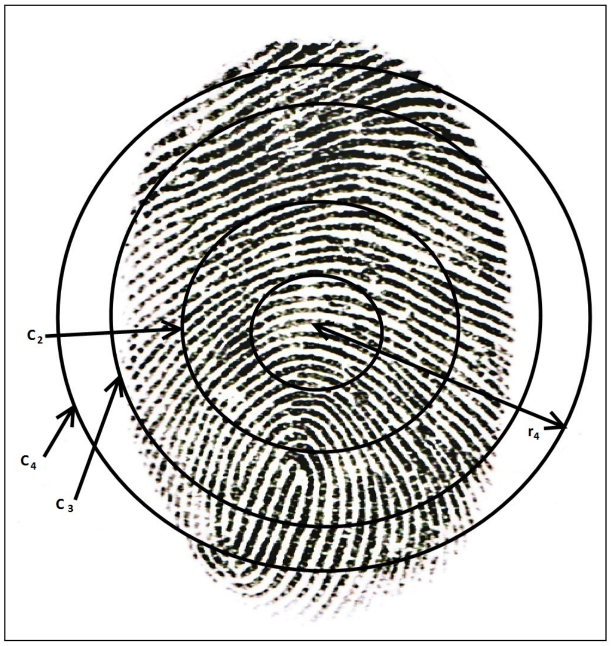

Figure 5. Fingerprint with scan circles

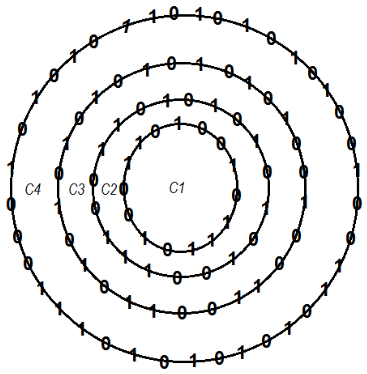

Figure 6. Intersection of a circle with the fingerprint

to Figure 7). What we do is as follows. For the input fingerprint, we cannot assume a particular center point to draw the concentric circle which is actually the main reason for difficulty in the process. So, instead, we take reference points at regular intervals across rows and columns of the entire frame of the image (i.e., the input scanning area) and at each point p_ℓ, concentric circles $C_{j\ell}$ of radius r_j are constructed. Like before, k is the number of circles at each reference point p_ℓ. So, from the above procedure, for each point p_ℓ we get k circular strings $X_{j\ell}, 1 \leq j \leq k$.

At this point the problem comes down to identifying the best match across the set of same radius circles. To do this we make use of the Approximate Circular String Matching via Filtering (ACSMF) algorithm, presented in [30], which is accurate and extremely fast in practice. To do this we take a particular $X_{j\ell}$, construct $X_{j\ell}.X_{j\ell}$ (to ensure that all conjugates of $X_{j\ell}$ are considered) and apply algorithm ACSMF on $X_{j\ell}.X_{j\ell}$ and Y_j. In other words, we try to match the circular string Y_j (corresponding to the circle of radius r_j) to all circular strings

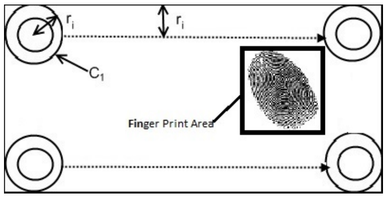

Figure 7. Identifying the orientation and surface area of the fingerprint impression

$X_{j\ell}$ (corresponding to the circle of radius r_j) generated at each point p_ℓ. Thus we can identify the best matched circular string, i.e., the best matched circles and thereby locate and identify the fingerprint impression with the correct orientation. Once the orientation has been identified, we can apply standard techniques to reorient the image to match with the image from the database in the next stage.

C. Details of Stage 2: Verification and Matching

Once Stage 1 of the algorithm is complete, we can assume that we have two images of the same size and orientation which we need to match and verify. We call this a verification process because in Stage 1 as well we have done a sort of matching already. However, we need to be certain and hence we proceed with the current stage as follows. Each image can now be seen as a two dimensional matrix of zero/one values, which can be easily converted to a (one dimensional) binary string. Now it simply comes down to pattern matching between two strings of the same length. However, note that, here as well we need to consider possibilities of errors. So, we simply compute the edit distance between the two binary strings and if the distance is within the tolerance level, we consider the fingerprint to be recognized. Otherwise, the authentication fails.

D. Accuracy and Speed

We have two parameters that determine the accuracy of our approach. In Stage 1, the accuracy depends on the number of concentric circles, k. The larger the value of k, the higher the accuracy of pinpointing the location with the correct orientation. However, as k increases the computational requirement and time also increases. In Stage 2, we have another parameter d which is the tolerance level, i.e., the (edit) distance allowed between the two strings.

At this point a brief discussion on the response time of our algorithm is in order. Note that, the bulk of the computational processing in our approach is required in Stage 1, where we apply algorithm ACSMF to identify the best matched circles. As has been shown in [30], on average, ACSMF works in linear time in the size of the input and is practically extremely fast. The size of the circles and hence the corresponding circular strings are very small and can be assumed to be constant for all practical purposes. As a result the running time of Stage 1 would be extremely fast. Again, since the size of the fingerprint image is very small, any efficient approximate string matching algorithm in Stage 2 would give us a very quick result. Overall, this promises us an excellent turn-around time.

E. Two Modes of Fingerprint Recognition System

As has been mentioned before, in terms of applications, there are two kinds of fingerprint recognition systems. So far we have only considered the mode where the input is a query fingerprint with an identity (ID) and the system verifies whether the ID is consistent with the fingerprint (i.e., verification mode). Here, the output is an answer of *Yes* or *No* and we need only match against one fingerprint from the database (i.e., the finger print coupled with the ID). To handle the other mode (identification mode), we need to match the query fingerprint against a list of fingerprints in the database. This can be done using an extension of algorithm ACSMF, namely Approximate Circular Dictionary Matching via Filtering algorithm (ACDMF) [32]. We omit the details here due to space constraints. Both ACSMF and ACDMF implementations are available at [33].

VI. Conclusion

This paper has proposed a new pattern matching based approach for quick and accurate recognition of fingerprints. One overlooked feature in fingerprint matching is that the rotation of the fingerprint is assumed to be in sync with the stored image; in this paper we have tackled this issue. The novel element of this paper is the process of using a series of circles to transform minutiae information into string information consisting of 0s and 1s, and then using the approximate circular string matching algorithm to identify the orientation. This technique is expected to improve the performance and the accuracy of the fingerprint verification system. The proposed approach is currently under implementation on different smart phone platforms.

Acknowledgement

M. Sohel Rahman is supported by a Commonwealth Academic Fellowship funded by the UK government and is currently on a sabbatical leave from Bangladesh University of Engineering and Technology (BUET).

References

[1] S. Sebastian, “Literature survey on automated person identification techniques,” International Journal of Computer Science and Mobile Computing, vol. 2, no. 5, May 2013, pp. 232–237.

[2] J. Unar, W. C. Seng, and A. Abbasi, “A review of biometric technology along with trends and prospects,” Pattern Recognition, vol. 47, no. 8, 2014, pp. 2673 – 2688.

[3] P. Szor, The art of computer virus research and defense. Addison-Wesley Professional, 2005.

[4] National Criminal Justice Reference Service, The Fingerprint Sourcebook, A. McRoberts, Ed. CreateSpace Independent Publishing Platform, 2014.

[5] A. Agarwal, A. K. Sharma, and S. Khandelwal, “Study of rotation oriented fingerprint authentication,” International Journal of Emerging Engineering Research and Technology, vol. 2, no. 7, October 2014, pp. 211–214.

[6] P. Gutierrez, M. Lastra, F. Herrera, and J. Benitez, “A high performance fingerprint matching system for large databases based on gpu,” IEEE Transactions on Information Forensics and Security, vol. 9, no. 1, 2014, pp. 62–71.

[7] K. H. Q. Zhang and H. Yan, “Fingerprint classification based on extraction and analysis of singularities and pseudoridges,” in Fingerprint Classification Based on Extraction and Analysis of Singularities and Pseudoridges, ser. VIP 2001. Sydney, Australia: VIP, 2001.

[8] E. R. Henry, Classification and Uses of Finger Prints. Routledge, 1900.

[9] H. C. Lee, R. Ramotowski, and R. E. Gaensslen, Eds., Advances in Fingerprint Technology, Second Edition. CRC Press, 2002.

[10] S. Sebastian, “Literature survey on automated person identification techniques,” International Journal of Computer Science and Mobile Computing, vol. 2, no. 5, 2013, pp. 232–237.

[11] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, Handbook of Fingerprint Recognition. Springer-Verlag, 2009.

[12] Griaule Biometrics. Online and offline acquisition. [Online]. Available: http://www.griaulebiometrics.com/en-us/book/ [retrieved: Nov., 2014]

[13] X. Tan and B. Bhanu, “Fingerprint matching by genetic algorithms,” Pattern Recognition, vol. 39, no. 3, 2006, pp. 465–477.

[14] A. K. Jain, L. Hong, S. Pankanti, and R. Bolle, “An identity-authentication system using fingerprints,” Proceedings of the IEEE, vol. 85, no. 9, 1997, pp. 1365–1388.

[15] Z. M. Kovacs-Vajna, “A fingerprint verification system based on triangular matching and dynamic time warping,” Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 22, no. 11, 2000, pp. 1266–1276.

[16] X. Tan and B. Bhanu, “Robust fingerprint identification,” in International Conference on Image Processing 2002, vol. 1. IEEE, 2002, pp. I–277.

[17] C. Kai, Y. Xin, C. Xinjian, Z. Yali, L. Jimin, and T. Jie, “A novel ant colony optimization algorithm for large-distorted fingerprint matching,” Pattern Recognition, vol. 45, no. 1, 2012, pp. 151–161.

[18] A. A. Saleh and R. R. Adhami, “Curvature-based matching approach for automatic fingerprint identification,” in System Theory, 2001. Proceedings of the 33rd Southeastern Symposium on. IEEE, 2001, pp. 171–175.

[19] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, “Filterbank-based fingerprint matching,” Image Processing, IEEE Transactions on, vol. 9, no. 5, 2000, pp. 846–859.

[20] A. Jain, A. Ross, and S. Prabhakar, “Fingerprint matching using minutiae and texture features,” in Image Processing, 2001. Proceedings. 2001 International Conference on, vol. 3. IEEE, 2001, pp. 282–285.

[21] A. V. Ceguerra and I. Koprinska, “Integrating local and global features in automatic fingerprint verification,” in Pattern Recognition, 2002. Proceedings. 16th International Conference on, vol. 3. IEEE, 2002, pp. 347–350.

[22] A. K. Jain, S. Prabhakar, and L. Hong, “A multichannel approach to fingerprint classification,” IEEE Transactions on Pattern Analysis and Machine Intelligence , vol. 21, no. 4, 1999, pp. 348–359.

[23] M. R. Girgis, A. A. Sewisy, and R. F. Mansour, “A robust method for partial deformed fingerprints verification using genetic algorithm,” Expert Systems with Applications, vol. 36, no. 2, 2009, pp. 2008–2016.

[24] A. Ross and A. Jain, “Biometric sensor interoperability: A case study in fingerprints,” in Biometric Authentication. Springer, 2004, pp. 134–145.

[25] C. Xinjian, T. Jie, Y. Xin, and Z. Yangyang, “An algorithm for distorted fingerprint matching based on local triangle feature set,” Information Forensics and Security, IEEE Transactions on, vol. 1, no. 2, 2006, pp. 169–177.

[26] B. Smyth, Computing Patterns in Strings. Pearson Addison-Wesley, 2003.

[27] M. Lothaire, Applied Combinatorics on Words. Cambridge University Press, 2005.

[28] K. Fredriksson and S. Grabowski, “Average-optimal string matching,” Journal of Discrete Algorithms, vol. 7, no. 4, 2009, pp. 579–594.

[29] K.-H. Chen, G.-S. Huang, and R. C.-T. Lee, “Bit-Parallel Algorithms for Exact Circular String Matching,” Computer Journal, 2013.

[30] C. Barton, C. S. Iliopoulos, and S. P. Pissis, “Fast algorithms for approximate circular string matching,” Algorithms for Molecular Biology, vol. 9, no. 9, 2014.

[31] A. V. Aho and M. J. Corasick, “Efficient string matching: an aid to bibliographic search,” Communication ACM, vol. 18, 1975, pp. 333–340.

[32] C. Barton, C. S. Iliopoulos, S. P. Pissis, and F. Vayani, “Accurate and efficient methods to improve multiple circular sequence alignment,” submitted.

[33] S. P. Pissis. ACSMF and ACDMF implementation. [Online]. Available: http://github.com/solonas13/bear/ [retrieved: Nov., 2014]