Introducing an Experimental Framework in C# for Fingerprint Recognition

Miguel Angel Medina-Pérez[1], Octavio Loyola-González[1,2], Andres Eduardo Gutierrez-Rodríguez[1,2], Milton García-Borroto[3], and Leopoldo Altamirano-Robles[1]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica. Luis Enrique Erro No. 1, Sta. María Tonanzintla, Puebla, México, C.P. 72840
{migue,octavio,andres,robles}@inaoep.mx

[2] Centro de Bioplantas, Universidad de Ciego de Ávila. Carretera a Morón km 9, Ciego de Ávila, Cuba, C.P. 69450
{octavioloyola,andres}@bioplantas.cu

[3] Instituto Superior Politécnico José A. Echeverría. Calle 114, No. 11901, e/ Ciclovía y Rotonda, Apartado 6028. Marianao, Habana, Cuba, C.P. 11901
mgarciab@ceis.cujae.edu.cu

Abstract. In this paper, a framework for fingerprint recognition is introduced. The framework contains several algorithms for fingerprint matching and feature extraction, as well as the evaluation protocol for several fingerprint verification competitions. All the algorithms are implemented in C# providing the source code which researchers can review, reuse or modify. In order to show the framework relevance, an experimental comparison among different matching algorithms is presented.

Keywords: biometrics, fingerprint matching, experimental framework.

1 Introduction

When starting a new research project, an important challenge is to find a framework, SDK or library that could be reused to save coding time. While machine learning researchers count with established tools like Weka [13], KEEL [1] and PRTools [9], researchers in the area of fingerprint recognition [18] are more limited. Table 1 shows the limitations of available fingerprint recognition tools according to the following criteria:

a) The users must pay for using the libraries or they must use the tools with time limitations and/or limited amount of fingerprints. For example, "Free Fingerprint Verification SDK" [23] is limited to ten records in the database while "VeriFinger SDK" [23] is limited to a trial period of 30 days, after which users must pay for the SDK.
b) The tools do not contain multiple fingerprint matching and feature extraction algorithms. For instance, the users cannot test different combination schemes with "NIST Biometric Image Software" [29] because it only includes one fingerprint matching algorithm.

c) The users do not have free access to the source code so they cannot reuse any component of the algorithms. For example, users cannot reuse the segmentation algorithm included at "Fingerprint SDK" [12] without paying to access the source code.
d) The users do not have control over the fingerprint databases and they cannot create an experiment with a custom protocol for performance evaluation. For instance, latent fingerprint identification is a very active research topic at present but users cannot use their own databases with the "FVC-onGoing web-based automated evaluation system" [8] because the system does not allow the users to upload custom databases.
e) The tools do not contain any protocol for performance evaluation. For example, to test "SourceAFIS SDK" [31] in FVC2004 [17] databases, the users have to implement the performance evaluation protocol.

Table 1. This is a representative set of fingerprint recognition tools and their limitations

	Limitations				
Fingerprint recognition tools	**a**	**b**	**c**	**d**	**e**
NIST Biometric Image Software [29]		X			X
FVC-onGoing web-based automated evaluation system [8]	X		X	X	
SourceAFIS SDK [31]		X			X
MCC SDK [4, 5, 10]		X	X		X
VeriFinger SDK [23]	X	X	X		X
Fingerprint SDK [12]	X	X	X		X
BiometricSDK [27]		X			X
IDKit PC SDK [14]	X	X	X		X

To deal with all these limitations, this paper introduces a fingerprint recognition framework containing matching algorithms, feature extraction algorithms, and experimental protocols. This framework is implemented in .Net Framework using the C# language for the following reasons:

– .Net Framework and C# have proved to be effective for developing fingerprint recognition tools such as MCC SDK [4, 5, 10] and SourceAFIS SDK [31].
– The C# language is very easy to learn because of its similarities to C and C++ languages.
– .Net Framework includes a large pack of technologies, tools and class libraries that saves coding time. Algorithms in this framework or algorithms developed by researchers based on this framework can be easily included in systems developed for real clients. Take into account that faster implementations may be required for those applications that involve processing high amounts of fingerprints.

While designing the framework, the "high cohesion and low coupling" principle [2] was applied, so users can creatively reuse most components in their own applications without integrating the entire framework or modifying the existing architecture. The framework contains tools for fingerprint matching evaluation where new algorithms can be included using minimum effort without recompiling the framework.

This framework allows carrying out fingerprint matching experiments for several databases according to the evaluation protocols of the Fingerprint Verification Competitions (FVC) [6]. Additionally, users can include experiments with a custom evaluation protocol or different databases.

There is no charge for the use of this framework or its source code and there are no restrictions regarding the amount of time or number of fingerprint used. Source code and documentation are available at `http://www.codeproject.com/Articles/97590/A-Framework-in-C-for-Fingerprint-Verification`.

In order to show the framework relevance, an experimental evaluation of matching algorithms in the framework over eleven databases is presented. The results are summarized using *critical difference diagrams* (CD diagrams) [7]. This kind of experimental comparison is rarely found in previous research papers because of the limitations of previous experimental frameworks.

This framework was used to create a new version of M3gl [20] which is among the top ten algorithms according to the accuracy indicators in database FMISO-HARD of FVC-onGoing [8].

2 Structure of Our Framework

The framework includes the algorithms that we have implemented in our research projects. Some of these algorithms have hundreds of references according to Google Scholar and, as far as we know, their source code is not available on the web. This framework contains nine minutiae-based fingerprint matching algorithms distributed in the following assemblies:

– [`FR.Jiang2000`] includes the algorithms `JY` [15] and `MJY` [21].
– [`FR.Tico2003`] includes the algorithms `TK` [30] and `MTK` [22].
– [`FR.Parziale2004`] includes the algorithm `PN` [24].
– [`FR.Qi2005`] includes the algorithms `QYW` [25] and `MQYW` [21].
– [`FR.Medina2011`] includes the algorithm `MPN` [19].
– [`FR.Medina2012`] includes the algorithm `M3gl` [20].

The framework contains four algorithms to extract the basic features (orientation image, skeleton image and minutiae) distributed in the following assemblies:

– [`FR.Ratha1995`] includes the algorithms proposed in [26] to extract orientation image, skeleton image and minutiae.
– [`FR.Sherlock1994`] includes the algorithm proposed in [28] to extract the orientation image.

The rest of the tools contained in the framework are distributed in the following assemblies:

– [`FR.Core`] contains the architecture of the framework.
– [`FR.FMExperimenter`] contains an application that allows conducting fingerprint matching experiments.

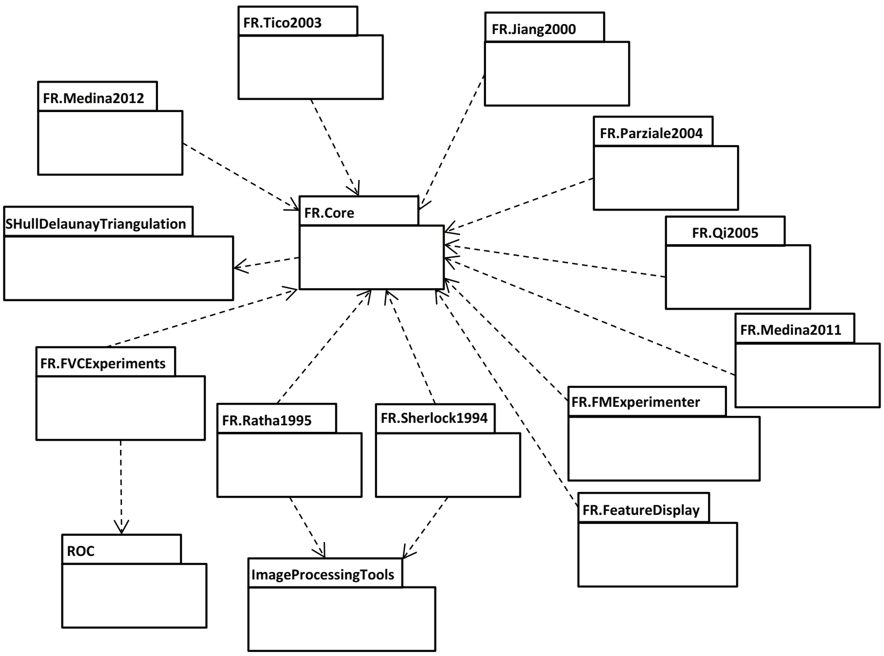

Fig. 1. The package diagram of the fingerprint recognition framework

- [`FR.FeatureDisplay`] contains an application that allows visualizing fingerprint features.
- [`FR.FVCExperiments`] contains the performance evaluation protocols of competitions FVC2000, FVC2002 and FVC2004 [6].
- [`ROC`] contains tools to build ROC curves.
- [`ImageProcessingTools`] contains tools to process images.
- [`SHullDelaunayTriangulation`] contains the algorithm available at `http://www.s-hull.org/` to compute Delaunay triangulation.

Figure 1 shows the dependency relationships among the assemblies that comprise the framework. It is worth mentioning, that assembly `ImageProcessingTool` is only used by assemblies `FR.Ratha1995` and `FR.Scherlock1994` because these are the only two assemblies that perform image processing.

3 Experimental Results

The algorithms are evaluated on databases DB1A, DB2A, DB3A, and DB4A of FVC2002 [16]; databases DB1_A, DB2_A, DB3_A, and DB4_A of FVC2004 [17]; and databases DB2_A, DB3_A, and DB4_A of FVC2006 [3]. The algorithms evaluated are: MCC [4, 5, 10] (SDK v1.4 available in `http://biolab.csr.unibo.it/mccsdk.html`), M3gl [20], JY [15], TK [30], PN [24], QYW [25], MJY [21], MTK [22], MPN [19], and MQYW [21].

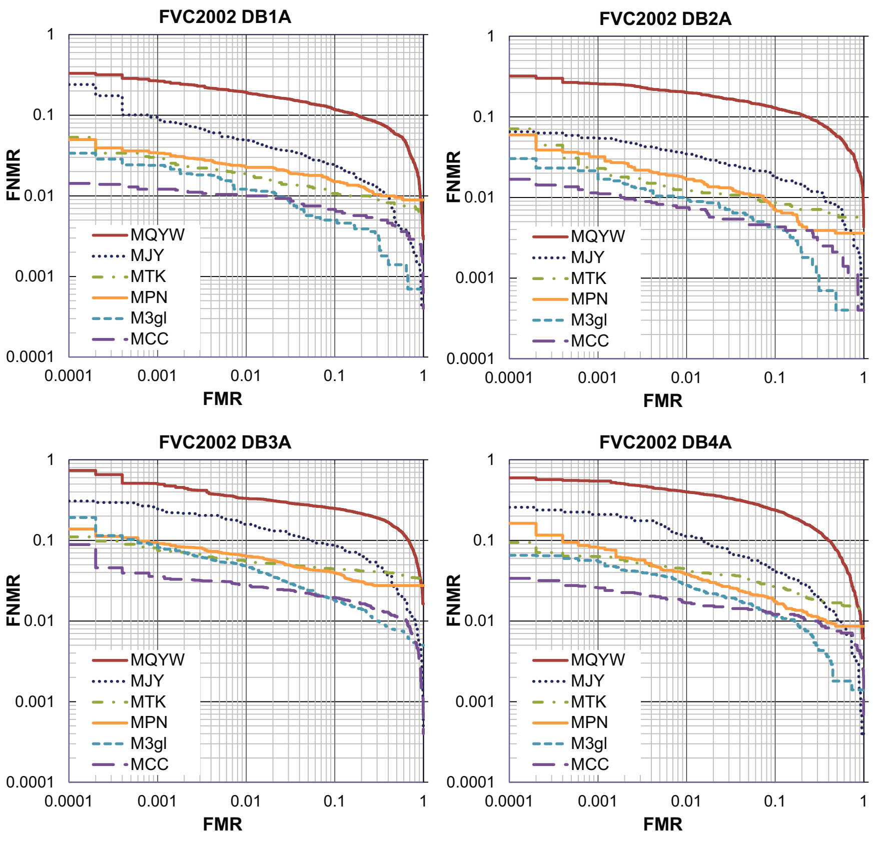

Fig. 2. ROC curves with the performance of the compared algorithms in FVC2002

The algorithms MJY, MTK, MPN, and MQYW are improved versions of JY, TK, PN, and QYW respectively; visualizing the results of JY, TK, PN, and QYW is avoided for the sake of clarity.

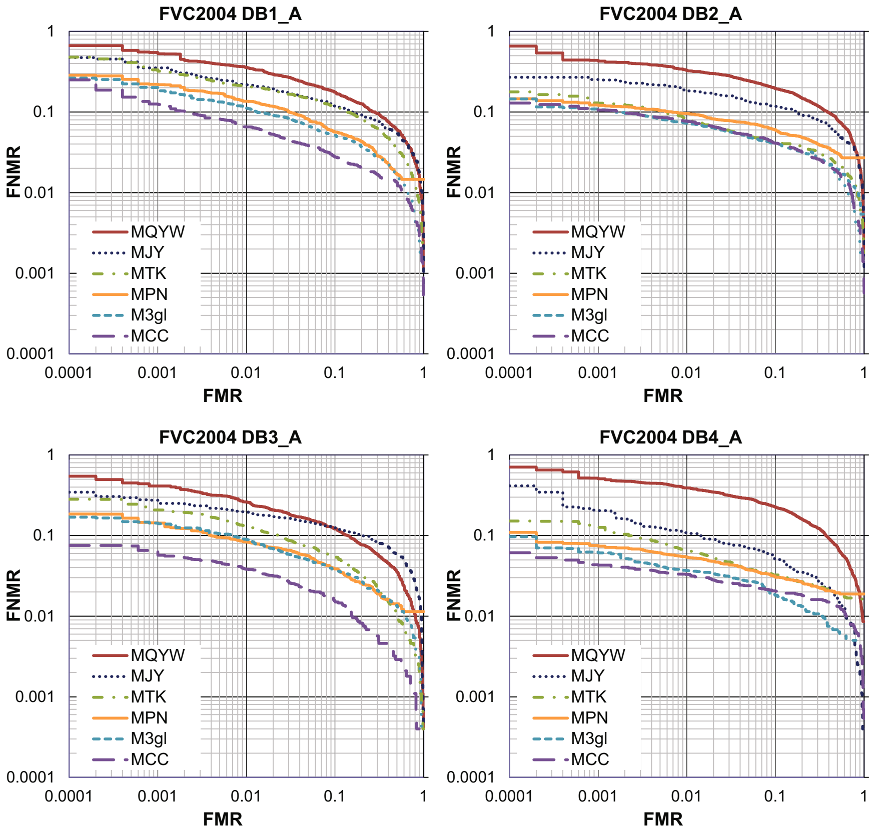

Fig. 3. ROC curves with the performance of the compared algorithms in FVC2004

All the algorithms are evaluated with their default parameters except for algorithm M3gl, whose parameter c (i.e. neighbor count) is changed to 7 (see reference [20]). The evaluation protocols proposed in [6] are used and the ROC curves with the performance of the compared algorithms are shown in Figure 2, Figure 3, and Figure 4.

Figure 2, Figure 3, and Figure 4 show that MCC tends to achieve the lowest FNMR for most of the values of FMR and it is only outperformed by M3gl for some values of FMR.

To further study the differences among the algorithms, the performance indicators proposed in [6] (*EER*, *FMR100*, *FMR1000*, *ZeroFMR*, and *matching time*) are used. 110 experiments (11 databases x 10 algorithms) are performed; so, there are 110 measures

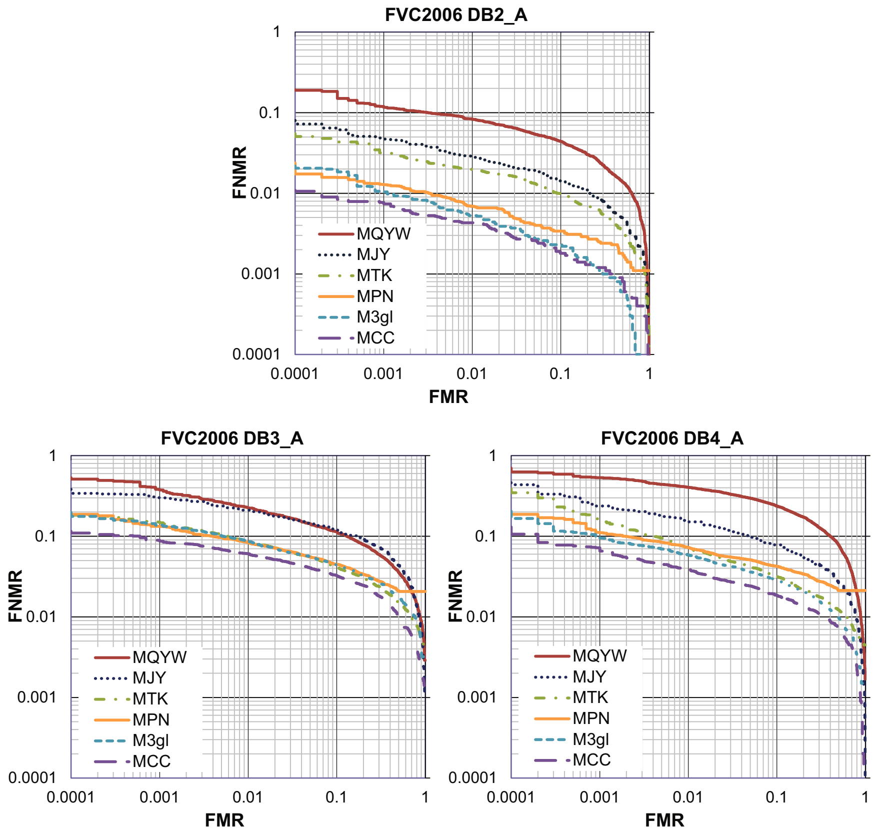

Fig. 4. ROC curves with the performance of the compared algorithms in FVC2006

of every performance indicator. Visualizing these results in a table of 5 columns (performance indicators) and 110 rows (11 databases x 10 algorithms) interferes with their interpretation. That is why the Friedman test [7] and Bergmann-Hommel dynamic post-hoc [11] are used to compare the results. *Critical difference diagrams* (CD diagrams) [7] are used to show the post-hoc results because CD diagrams compactly present the rank of the algorithm according to one performance indicator. CD diagrams also show the magnitude of differences between algorithms and the significance of the observed differences [7]. In a CD diagram, the rightmost algorithm is the best algorithm, while the algorithms sharing a thick line have statistically similar behaviors. Once again, visualizing the results of JY, TK, PN, and QYW is avoided for the sake of clarity.

Figure 5 shows that MCC achieves the best results according to EER, FMR100, FMR1000, and ZeroFMR; but there is no significant statistical difference with M3gl. This is a good result, taking into account that MCC is a patented algorithm for which the source code is not available. Because of the proposed framework, the research community has access to the source code of an algorithm (M3gl) which is not statistically different from MCC, which is one of the most accurate algorithms in state of the art papers.

Figure 5 shows that MCC is the slowest algorithm because its SDK was provided by its authors only for research purposes. A faster version of MCC is available but it is not free.

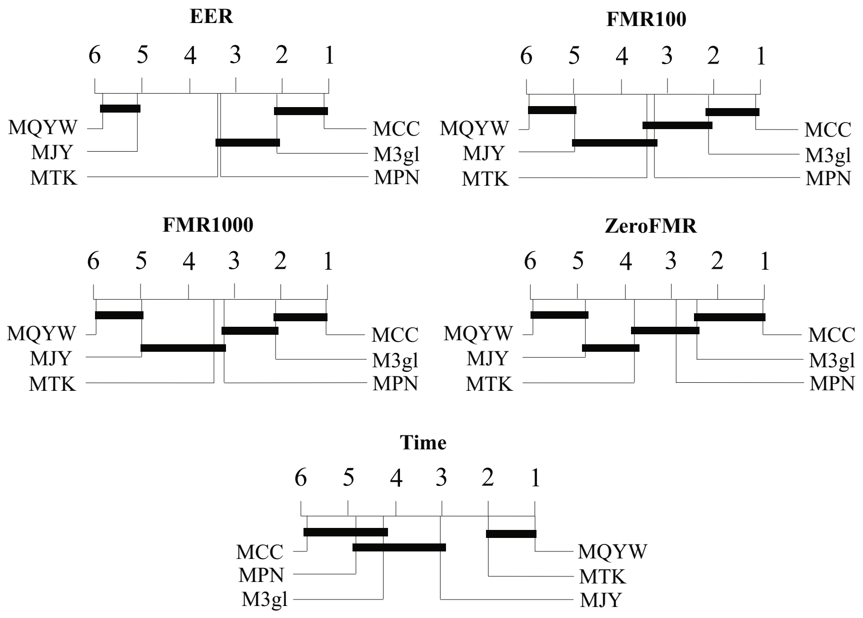

Fig. 5. CD diagrams with the statistical comparison of the algorithms according to EER, FMR100, FMR1000, ZeroFMR and matching time

4 Conclusions

The fingerprint recognition tools available on the web have several limitations. In this paper, a framework is proposed to overcome these limitations. Several matching algorithms are provided not only for experimental purposes, but also to create new applications. The source code for all of the algorithms is provided so users can reuse any part of the code as well as any package of the framework. 110 experiments are performed and the algorithms are compared according to the Friedman test and Bergmann-Hommel dynamic post-hoc. The experiments show that one of the algorithms included in the framework is not statistically different from a well-known patented algorithm. We hope this work motivates more people to collaborate in order to implement other algorithms and improve the fingerprint recognition framework for the benefit of programming and research communities.

Acknowledgment. The authors would like to thank Rebekah Hosse Clark and MSc. Dania Yudith Suárez Abreu for their valuable contributions improving the grammar and style of this paper.

References

1. Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S.: KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing 17(2-3), 255–287 (2011)
2. Booch, G., Maksimchuk, R.A., Engle, M.E., Young, B.J., Conallen, J., Houston, K.A.: Object-oriented analysis and design with applications, 3rd edn. Pearson Education, Inc. (2007)
3. Cappelli, R., Ferrara, M., Franco, A., Maltoni, D.: Fingerprint verification competition 2006. Biometric Technology Today 15(7-8), 7–9 (2007)
4. Cappelli, R., Ferrara, M., Maltoni, D.: Minutia cylinder-code: a new representation and matching technique for fingerprint recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(12), 2128–2141 (2010)
5. Cappelli, R., Ferrara, M., Maltoni, D.: Fingerprint indexing based on Minutia Cylinder-Code. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(5), 1051–1057 (2011)
6. Cappelli, R., Maio, D., Maltoni, D., Wayman, J.L., Jain, A.K.: Performance evaluation of fingerprint verification systems. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(1), 3–18 (2006)
7. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (2006)
8. Dorizzi, B., Cappelli, R., Ferrara, M., Maio, D., Maltoni, D., Houmani, N., Garcia-Salicetti, S., Mayoue, A.: Fingerprint and on-line signature verification competitions at ICB 2009. In: Tistarelli, M., Nixon, M.S. (eds.) ICB 2009. LNCS, vol. 5558, pp. 725–732. Springer, Heidelberg (2009)
9. Duin, R.P.W.: Prtools version 3.0: A matlab toolbox for pattern recognition. In: Proc. of SPIE, p. 1331 (2000)
10. Ferrara, M., Maltoni, D., Cappelli, R.: Noninvertible Minutia Cylinder-Code representation. IEEE Transactions on Information Forensics and Security 7(6), 1727–1737 (2012)

11. García, S., Herrera, F.: An extension on Statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. Journal of Machine Learning Research 9, 2677–2694 (2008)
12. Griaule Biometrics: Fingerprint sdk (2014), `http://www.griaulebiometrics.com/en-us/fingerprint_sdk` (accessed January 8, 2014)
13. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11(1), 10–18 (2009)
14. Innovatrics: Idkit pc sdk (2014), `http://innovatrics.com/products/fingerprint-identification-sdk` (accessed January 8, 2014)
15. Jiang, X., Yau, W.Y.: Fingerprint minutiae matching based on the local and global structures. In: 15th International Conference on Pattern Recognition, vol. 2, pp. 1038–1041 (2000)
16. Maio, D., Maltoni, D., Cappelli, R., Wayman, J.L., Jain, A.K.: Fvc2002: Second fingerprint verification competition. In: 16th International Conference on Pattern Recognition, vol. 3, pp. 811–814 (2002)
17. Maio, D., Maltoni, D., Cappelli, R., Wayman, J.L., Jain, A.K.: Fvc2004: Third fingerprint verification competition. In: Zhang, D., Jain, A.K. (eds.) ICBA 2004. LNCS, vol. 3072, pp. 1–7. Springer, Heidelberg (2004)
18. Maltoni, D., Maio, D., Jain, A.K., Prabhakar, S.: Handbook of fingerprint recognition, 2nd edn. Springer, Heidelberg (2009)
19. Medina-Pérez, M.A., García-Borroto, M., Gutierrez-Rodríguez, A., Altamirano-Robles, L.: Robust fingerprint verification using m-triplets. In: International Conference on Hand-Based Biometrics (ICHB 2011), Hong Kong, pp. 1–5 (2011)
20. Medina-Pérez, M.A., García-Borroto, M., Gutierrez-Rodríguez, A., Altamirano-Robles, L.: Improving fingerprint verification using minutiae triplets. Sensors 12, 3418–3437 (2012)
21. Medina-Pérez, M.A., García-Borroto, M., Gutierrez-Rodríguez, A., Altamirano-Robles, L.: Improving the multiple alignments strategy for fingerprint verification. In: Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F., Olvera López, J.A., Boyer, K.L. (eds.) MCPR 2012. LNCS, vol. 7329, pp. 147–154. Springer, Heidelberg (2012)
22. Medina-Pérez, M.A., Gutiérrez-Rodríguez, A., García-Borroto, M.: Improving fingerprint matching using an orientation-based minutia descriptor. In: Bayro-Corrochano, E., Eklundh, J.-O. (eds.) CIARP 2009. LNCS, vol. 5856, pp. 121–128. Springer, Heidelberg (2009)
23. NEUROtechnology Inc.: Verifinger (2014), `http://www.neurotechnology.com/` (accessed January 8, 2014)
24. Parziale, G., Niel, A.: A fingerprint matching using minutiae triangulation. In: Zhang, D., Jain, A.K. (eds.) ICBA 2004. LNCS, vol. 3072, pp. 241–248. Springer, Heidelberg (2004)
25. Qi, J., Yang, S., Wang, Y.: Fingerprint matching combining the global orientation field with minutia. Pattern Recognition Letters 26(15), 2424–2430 (2005)
26. Ratha, N., Chen, S., Jain, A.K.: Adaptive flow orientation-based feature extraction in fingerprint images. Pattern Recognition 28(11), 1657–1672 (1995)
27. Scott, R.J.: Biometricsdk (2013), `http://biometricsdk.sourceforge.net/` (accessed March 8, 2013)
28. Sherlock, B.G., Monro, D.M., Millard, K.: Fingerprint enhancement by directional fourier filtering. IEE Proceedings Vision Image and Signal Processing 141(2), 87–94 (1994)
29. The National Institute of Standards and Technology: NIST Biometric Image Software (2014), `http://www.nist.gov/itl/iad/ig/nbis.cfm` (accessed January 8, 2014)
30. Tico, M., Kuosmanen, P.: Fingerprint matching using an orientation-based minutia descriptor. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(8), 1009–1014 (2003)
31. Vazan, R.: SourceAFIS SDK (2014), `http://www.sourceafis.org/` (accessed January 18, 2014)