IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 27, 2021, accepted June 4, 2021, date of publication June 14, 2021, date of current version June 21, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3088314

An Indexing Algorithm Based on Clustering of Minutia Cylinder Codes for Fast Latent Fingerprint Identification

ISMAY PÉREZ-SÁNCHEZ[1], BÁRBARA CERVANTES[1], MIGUEL ANGEL MEDINA-PÉREZ[1], RAÚL MONROY[1], OCTAVIO LOYOLA-GONZÁLEZ[2], SALVADOR GARCÍA[3], AND FRANCISCO HERRERA[3,4], (Senior Member, IEEE)

[1]School of Engineering and Sciences, Tecnologico de Monterrey, Ciudad López Mateos 52926, Mexico
[2]Altair Management Consultants Corporation, Waltham, MA 02451, USA
[3]Department of Computer Science and Artificial Intelligence, Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI), University of Granada, 18071 Granada, Spain
[4]Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

Corresponding author: Miguel Angel Medina-Pérez (migue@tec.mx)

This work was supported in part by National Council of Science and Technology of Mexico (CONACyT), Mexico, through the Scholarship under Grant 492968.

ABSTRACT Latent fingerprint identification is one of the leading forensic activities to clarify criminal acts. However, its computational cost hinders the rapid decision making in the identification of an individual when large databases are involved. To reduce the search time used to generate the fingerprint candidates' order to be compared, fingerprint indexing algorithms that reduce the search space while minimizing the increase in the error rate (compared to the identification) are developed. In the present research, we propose an algorithm for indexing latent fingerprints based on minutia cylinder codes (MCC). This type of minutiae descriptor presents a fixed structure, which brings advantages in terms of efficiency. Besides, in recent studies, this descriptor has shown an identification error rate, at the local level, lower than the other descriptors reported in the literature. Our indexing proposal requires an initial step to construct the indices, in which it uses k-means++ clustering algorithm to create groups of similar minutia cylinder codes corresponding to the impressions of a set of databases. K-means++ allows for a better outcome over other clustering algorithms because of the selection of the proper centroids. The buckets associated with each index are populated with the background databases. Then, given a latent fingerprint, the algorithm extracts the minutia cylinder codes associated with the clusters' indices with the lowest distance respect to each descriptor of this latent fingerprint. Finally, it integrates the votes represented by the fingerprints obtained to select the candidate impressions. We conduct a set of experiments in which our proposal outperforms current rival algorithms in presence of different databases and descriptors. Also, the primary experiment reduces the search space by four orders of magnitude when the background database contains more than one million impressions.

INDEX TERMS Fingerprint indexing, latent fingerprint, K-means clustering, minutia cylinder code.

I. INTRODUCTION

Biometric traits are key in computer systems that are used to identify individuals. Latent fingerprints are very popular in such identification systems, for they have been used to determine authorship of criminal acts [1]–[8]. A latent fingerprint is a print that is unintentionally left on the surface of an object. It stems from the oily nature of the skin [9]. A latent fingerprint, unfortunately, often leads to a bad quality image, which furthermore contains only partial information of the actual individual's fingerprint. This is in contrast with a fingerprint impression, which is obtained in a controlled environment [4], [10]. Given both objects, a latent fingerprint (known as *the query*) and a background database of fingerprint impressions (each associated with an individual) an Automated Fingerprint Identification System (AFIS) aims to pinpoint a (small) collection of impressions that the query most likely match. Due to the noisy nature of latent fingerprints, identification is quite challenging and error prone.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhe Jin.

Clearly, an AFIS must be accurate, but also efficient. This is because the query has to be attempted to be matched many times (in principle, just as many times as appropriate impressions are in the database, which often grows to the order of millions). The problem becomes apparent if one considers that the cost of comparing two fingerprints in an AFIS is in the order of milliseconds (considering a standard 2GHz processor personal computer). To get around this burden, an AFIS may use an indexing mechanism [11]. However, existing indexing mechanisms perform poorly when used in the context of latent fingerprint identification. This is because they decay identification accuracy or because they are not up to a fingerprint database of even a moderate size. This could be explained by the fact that most indexing mechanism have been designed and tested considering only the fingerprint verification (see, for example, [12]–[22]); further, they have been tested using rather small databases (see, for example, [23]–[29].)

In this paper, we present a clustering-based indexing algorithm, especially designed to work with a latent fingerprint identification algorithm built upon minutia cylinder codes (a minutiae-based descriptor). To reduce the identification error rate, our algorithm uses clustering in a way that each cluster holds fingerprints with similar minutia cylinder codes. Clustering could still be expensive, in particular when the background database is in the order of million fingerprints. To get around this problem, we followed a two-step approach. In the first step (see index construction in Section III), we used standard small fingerprint databases and a moderate-size latent fingerprint database, in order to identify cluster centroids via k-means++ [30]. Then, in the second step, we used these centroids to populate as many clusters as centroids but now using our background database, which contains more than 1.1M fingerprint impressions. Thanks to this approach, we control both the number of clusters and the homogeneity in the size of the clusters. With this we finish building the index structure, ready to be used for impression retrieval (also a two step approach, see Section III).

From our experimentation, we have found that, when compared to other indexing algorithms that also use minutia cylinder codes, our algorithm decreases the error rate when it is asked to return a significantly smaller set of fingerprints the query has to be matched against with. This is paramount, because an expert examiner finds it easily overwhelming to study even a handful tens of impressions. Furthermore, in such cases, our indexing algorithm yields a barely visible increment in the retrieval time, of less than one order of magnitude. Our indexing algorithm has been successfully tested against others, using our background database, of a size of over 1.1M fingerprints. To assess the relative performance of the various indexing algorithms under consideration, in our experimentation we used the protocol put forward in the Handbook of Fingerprint Recognition [11]. This protocol stresses the balance between error rate and the percentage of the database that the associated AFIS has to search.

From our experimental results, we conclude that our indexing algorithm outperforms that embedded in Cappelli *et al.*'s SDK [14], and other more recently reported algorithms, like the one in Khodadoust and Khodadoust [31], Su *et al.* [32], Cappelli [33], Muñoz-Briseño *et al.* [21], Wang *et al.* [15], and Deblonde & Morpoho [34]. Given that the indexing algorithm used in the SDK of Cappelli *et al.* [14] has been used as part of an ensemble of indexing mechanisms in the work of [35], it follows that our indexing mechanism could be used to improve on the results reported therein too.

In summary, the contributions of this paper are the following:

- An indexing structure based on clustering of minutia cylinder codes that ensures the proximity of these descriptors while maximizing the retrieval of an impression that matches with the latent fingerprint used as the query. This indexing structure achieves homogeneous indexing without losing the descriptors' proximity, even when the descriptors have a heterogeneous distribution.
- An indexing algorithm for latent fingerprints that reduces the error rate by using only minutia cylinder codes.

Our indexing mechanism has been validated on a database with more than one million impressions, which, according to our literature review, has not been attempted before.

The rest of this paper is structured as follows. Section II gives an overview of the descriptor we use, namely: minutia cylinder codes (II-A) and of indexing techniques (II-B). Section III presents the proposed algorithm. The experimental setup is presented in section IV: databases used to evaluate this algorithm are presented in subsection IV-A; whereas subsection IV-B describes the evaluation protocol. Section V includes our experimentation details and the results obtained. Finally, in Section VI, we state our conclusions.

II. PRELIMINARIES

A. MINUTIA CYLINDER CODES (MCC)

Minutiae-based descriptors [36], [37] in the literature include minutiae singles [38], [39], minutiae pairs [29], minutiae triplets using Delaunay triangulation [28], [40], convex structures [41], minutiae k-plets [12], [26], and minutia cylinder codes (MCC) [14], [15]. Minutia cylinder codes [42] have gained increasing attention in the recent years [16]–[18]. They have properties that bring advantages to latent fingerprint identification performance when used alone, as experimentally shown by Valdes-Ramirez *et al.* [4]. We elaborate on these advantages in the following paragraphs.

Minutia cylinder codes consider the information of minutiae within a radius r (see Fig. 1). A vector v, of size l, is obtained using the information of the neighbourhood of minutia m. This vector is formed by the binary representation of q cylindrical sections. Each section is assigned a range of angular differences, of the same size, which covers 2π radians. So, given q sections, the covered ranges are: $\left[0, \frac{1}{q}2\pi\right)$, $\left[\frac{1}{q}2\pi, \frac{2}{q}2\pi\right) \ldots \left[\frac{q-1}{q}2\pi, 2\pi\right)$.

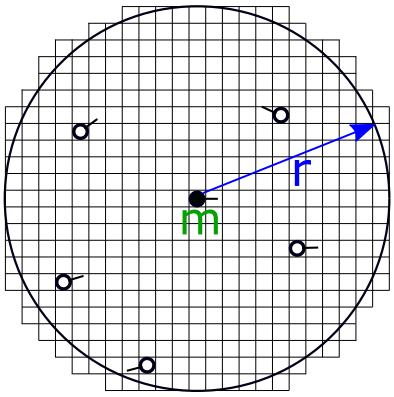


FIGURE 1. The neighbourhood of minutia (*m*), used to extract its MCCs.

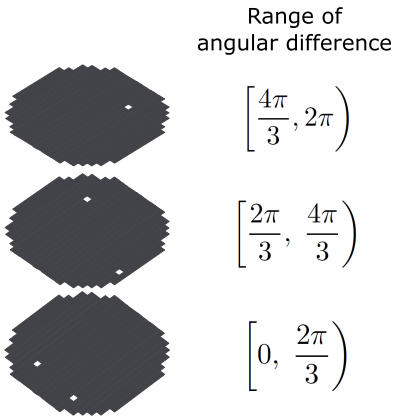


FIGURE 2. Three slices of an MCC.

For their binary representation, each cylindrical section takes into account only the information contributed by those minutiae with a directional difference (concerning the minutia m) that is within its range of angular difference. Fig. 2 shows the graphical representation of the cylinder code that belongs to the example minutia in Fig. 1, using three cylindrical sections. Note that each minutia m_n within the neighborhood of m, contributes to one cylindrical section, that whose range includes the value of the directional difference between the minutia m_n and m.

Minutia cylinder codes have the following characteristics, which help mitigate the challenges raised by latent fingerprint analysis, and make them a suitable choice for a descriptor:

- They use a structure based on a fixed radius, and the extracted vector is of fixed size, which makes computing less complicated [42].
- Minutiae near the image borders are treated as any other minutia, so there is no additional processing at the identification or verification stage [42].
- Having a binary representation allows optimizing the comparison of descriptors in lower-level instructions on a processor [42], [43].

B. INDEXING FOR LATENT FINGERPRINT

Indexing algorithms use a numeric vector to represent the features of a fingerprint. These feature vectors allow the measurement of the similarity between fingerprints; similar fingerprints should be mapped close to each other. For latent fingerprint identification, the query should be compared against those impressions that, according to their feature vector, are close one another [11]. Many descriptors and techniques have been used by indexing algorithms [12], [28], [31], [38], [42], [44]. Among them, we have indexing based on spatial data structures, singular points, ridge-line frequency, Galton-Henry classification scheme, and minutiae-based descriptors [11].

Indexing algorithms are suitable for latent fingerprint identification. This is because most of them are based on minutiae descriptors, and so they can exploit partial fingerprint information [45]. Further, minutiae-based indexing algorithms are invariant to translation and rotation of the fingerprint [11], they are tolerant to low-quality images [46], and they are robust to feature extraction errors, such as the absence of existing minutia or inclusion of non-existent minutia [11], [47]. The main challenge indexing algorithms face nowadays is that they do not scale up very well for big databases.

Existing indexing mechanisms can be split into two classes, according to whether they have been tested using latent fingerprints or not. Table 1 displays a collection of papers, each of which introduces an indexing algorithm the performance of which was tested using a fingerprint impression as a query. Papers are all grouped in terms of the type of descriptor the associated mechanisms use. For each paper, we provide the size of the database the corresponding authors used in their experimentation, together with the results reported therein (for a description of the performance measurements, see IV-B). Looking closely into Table 1, we notice that mechanisms' performance is, in general, pretty good. This can be explained by that the input query is an impression, which greatly enhances the chances of identifying the block that contains the fingerprints that are most similar to the query. We also notice that the size of the background databases is relatively small, lesser than 30K impressions. Thus, the low error rates reported on in Table 1 result from the use of high-quality images and a small database.

Table 2 displays a collection of papers, which also introduce a new indexing mechanism, but where mechanism performance was tested using a latent fingerprint as a query (which is in contrast with those shown in Table 1). The information portrayed in Table 2 is similar to that of Table 1), except for the column named "Features", the rationale of which is as follows. Feng and Jain [48] and Paulino *et al.* [35] have independently shown that using one descriptor is not enough for latent fingerprint analysis, due to the limited information available in a latent fingerprint. Instead, they suggest one should use multiple descriptors. A query fingerprint is now indexed one time per descriptor; then an integration function is used to obtain a single indexing result. This integration function is a linear combination of the indexes computed per descriptor. Looking closely into Table 2, we first notice that the list of papers is considerably shorter than that of Table 1. We also notice that mechanisms' performance is no longer

TABLE 1. Summary of the most relevant minutiae-based algorithms in the context of indexing impressions. Abbreviations: PR = Penetration Rate, ER = Error rate, HR = Hit Rate, CIP = Correct Index Power, AAcc = Average Accuracy, EER = Equal Error Rate, PER = Pre-selection error rate, FRR = False Rejection Rate, and FAR = False Acceptance Rate.

Authors	Descriptor	Impression count	Results
Jayaraman et al. (2014) [38]	MBP	800	95% HR in 2.24% PR
Tiwari et al. (2015) [39]	CGTC	800	100% HR in 1.32% PR
Wang et al. (2012) [29]	DITOM	800	7.5% EER
Bebis et al. (1999) [28]	triplets	300	94.2% AAcc
Bhanu et al. (2003) [19]	triplets	5000	72.4% CIP
Choi et al. (2003) [20]	triplets	1360	99.2% PR in 10% Rank
Liang et al. (2007) [27]	triplets	880	100% HR in 20.9% PR
Uz et al. (2009) [40]	triplets	800	1.53% FRR in 0% FAR
Muñoz-Briseño et al. (2013) [21]	triplets	24,000	less than 98% CIP in 30% PR
Gago et al. (2013) [22]	triplets	24,000	98% CIP in 30% PR
Khodadoust and Khodadoust (2017) [31]	triplets	2,000 2,700	95.2% HR in 10% PR 93.8% HR in 10% PR
Iloanusi et al. (2011) [23]	quadruplets	800	100% HR in 5.2% PR
Iloanusi et al. (2014) [24]	quadruplets	800	100% HR in 5% PR
Khodadoust et al. (2020) [13]	quadruplets in 3D space	1,350	1.34 EER
Chikkerur et al. (2006) [25]	k-plets	800	1.5% EER
Bai et al. (2015) [12]	k-plets	2,000	less than 94% CIP in 10% PR
Mansukhani et al. (2010) [26]	graph	800	0.95% AAcc
Cappelli et al. (2011) [14]	MCC	24,000	9% ER in 1% PR
Wang et al. (2015) [15]	compact MCC	20,000	92% HR in 10% PR
Bai et al. (2018) [16]	compact MCC	27,000	4% ER in 10% PR
Bai et al. (2018) [17]	compact MCC	27,000	3% ER in 10% PR
Anand and Kanhangad (2020) [18]	pore and MCC	1,984	less than 10% PER

outstanding. This time, except from the one used in the work of Paulino *et al.* [35], the size of the background databases used in their experimentation is rather small.

III. NEW INDEXING ALGORITHM BASED ON CLUSTERING OF MINUTIA CYLINDER CODES

We propose to improve the results of MCC by following an algorithm with four stages, as shown in Fig. 3. In the following subsections, we describe each of the stages of our proposed algorithm.

A. INDEX CONSTRUCTION

The goal of the index construction step in our algorithm is to obtain an indexing structure that assigns descriptors homogeneously to the indices. This first step includes extracting the MCCs of impressions in databases, clustering these descriptors, and computing each cluster's center to obtain an index (see Fig. 4).

The MCCs are clustered in k clusters using Euclidean distance. For each cluster, a center c is computed, according to the geometric center of the vectors in the cluster. In our experiments, l, the dimension of the MCCs, is 1280.

These centers will serve as the indices in our indexing structure. We assume that the databases used to compute the centers are representative of the set of fingerprints used in the following experiments. So, we estimate that the groups formed in the next stage will follow the same distribution of groups used to obtain the centers.

B. INDEXING STEP

In the indexing step, the MCCs are inserted into the indexing structure (Fig. 3, step 2). For each minutia cylinder code cc, we computed the euclidean distance to each center c_j (obtained in the index construction step). A tuple formed by the cylinder code, cc, and the index of the impression to which it belongs, i, is associated with the closest center. This process is described in Algorithm 1.

Algorithm 1 Indexing Stage

Function `buildIndexingStructure`(T, C)
Input:

- $T = \{T_i : i \in [1, n]\}$, where T_i is the set of MCCs of the i-th impression, and n is the number of impressions.
- $C = [c_1, c_2, \ldots, c_k]$, where c_j is the j-th center of the clusters computed in the preprocessing stage, and k is the number of clusters.

Output: Indexing structure H.

begin
  Let d be the euclidean distance function.
  $H \leftarrow \{\}$
  foreach $T_i \in T$ **do**
    foreach $cc \in T_i$ **do**
      $c \leftarrow argmin_{c_j \in C}\ d(cc, c_j)$
      $H \leftarrow H \cup \{\langle c, \langle cc, i\rangle\rangle\}$
  return H

The selection of euclidean distance to find the closest center to a cylinder code was determined empirically. We experimented with a variant of the comparison function in Cappelli *et al.* [14] and obtained the best results when the euclidean distance was used.

TABLE 2. Summary of the algorithms in the context of indexing latent fingerprints. Abbreviations: PR = Penetration Rate, ER = Error rate, HR = Hit Rate, and Rank-1 = First position of rank.

Authors	Features	Descriptors	Impression count	Results
Feng and Jain (2008) [48]	Fingerprint class, singular points, ridges	Fingerprint class; delta core and orientation; field orientation and curvature, single minutiae	10,258	97.3% HR in 10% PR
Paulino et al. (2013) [35]	Minutia, ridge, singular points	Minutia triplets and MCC; field orientation and ridge frequency; core and deltas	267,258	81.8% HR in 10% PR
Krish et al. (2014) [49]	Ridges	Orientation tensor	88	near 68% HR for rank-1
Krish et al. (2015) [50]	Ridges	Orientation tensor	258	80.62% HR for rank-1

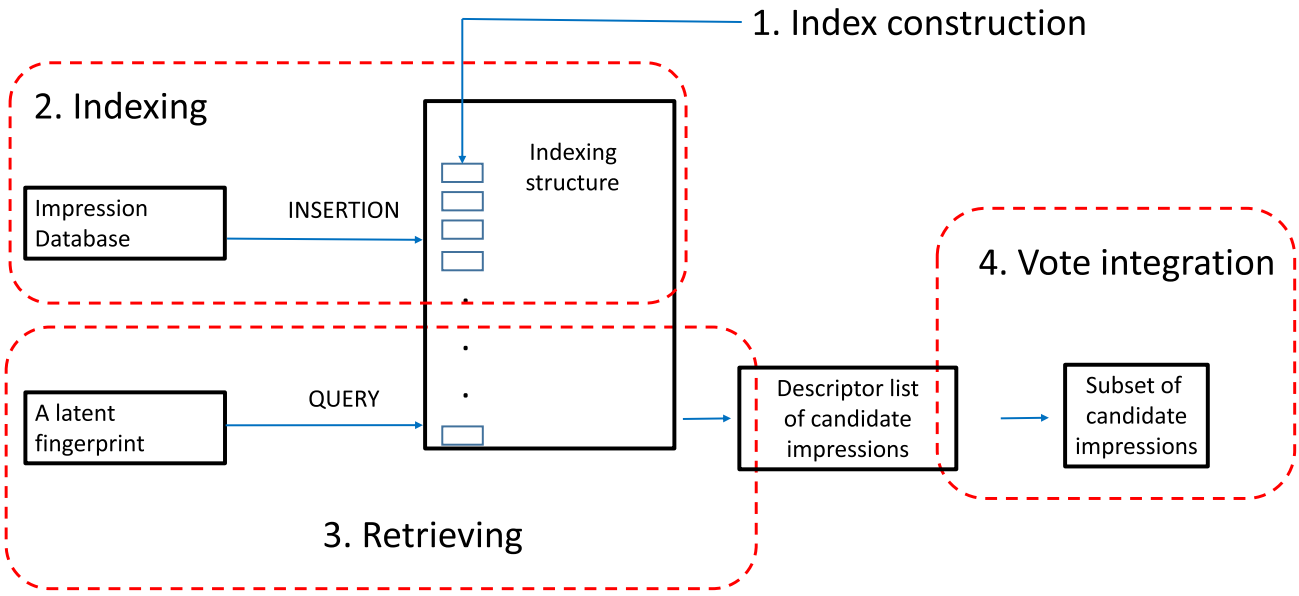


FIGURE 3. General scheme of the proposed algorithm. 1. First, the descriptors of the fingerprint impressions in the database are clustered to obtain the indexing structure's indices. 2. The descriptors of the fingerprint impressions in the database are inserted into the indexing structure. 3. Given a latent fingerprint, we obtain the subset of impression descriptors associated with its index. 4. Vote integration stage, by impression, to obtain a subset of candidate impressions that may match the latent fingerprint query.

The indexing structure H, obtained from the Algorithm 1, is used to identify a latent fingerprint query. The next sections describe two further steps needed to complete the indexing process.

C. RETRIEVING STEP

The retrieving step's purpose is to obtain a set of impressions that leads to the fast identification of a latent fingerprint query. Assuming that there is an impression that matches the query, to reduce the identification time, the probability of the matching impression being in the obtained set should be maximized.

We follow with the application of the Algorithm 2 for each cylinder code, qcc, obtained from the query fingerprint. First, the centers of the groups are sorted based on their distance to qcc. Starting from the closest center, we take the tuples in H, associated with that center. These tuples, $\langle cc, i \rangle$, contain a cylinder code and the index of the impression to which it belongs. Next, the similarity s, between cc and qcc is computed using the formula by Cappelli *et al.* [42]. This similarity value compounds another set of tuples of the form $\langle s(qcc, cc), i \rangle$, which are added to the results set used in the next stage. We refer to this set as L_{qcc}. We continue this process with other centers until L_{qcc} has at least u tuples.

Unless the indexed descriptors are distributed homogeneously, the size of the impressions set obtained for a query fingerprint is diverse. The algorithm we propose ensures a homogeneous search space in every query ($u = a/k$), where a is the number of indexed cylinder codes and k is the number of indices.

The Algorithm 2 is applied to every cylinder code extracted from the latent query fingerprint. Each of these codes will have its own set L_{qcc}. The following stage unifies the results per impression.

D. VOTE INTEGRATION STEP

As we have mentioned, for a given query latent fingerprint, we have several sets of tuples, one per cylinder code.

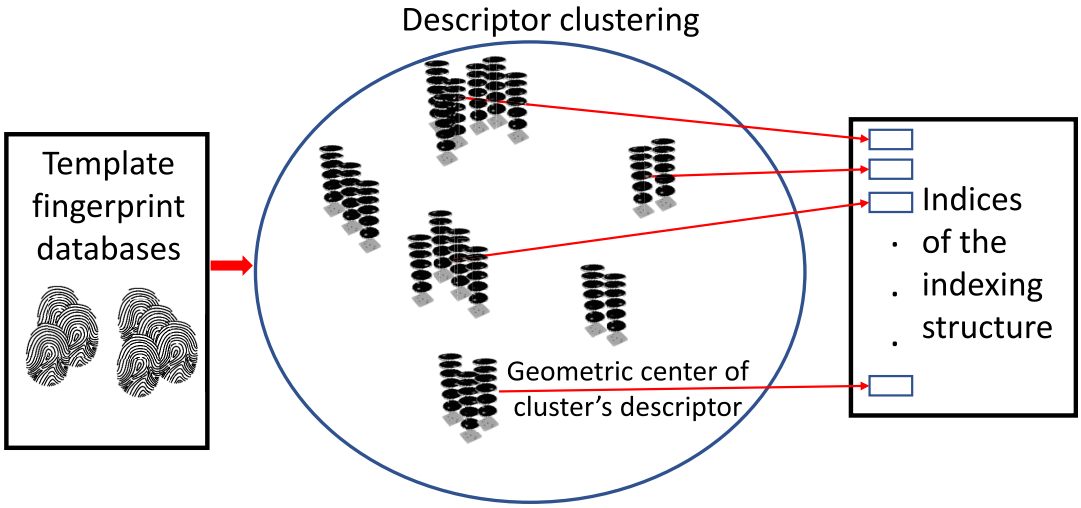


FIGURE 4. Index construction step. Minutiae descriptors are extracted from the database and grouped. The center of each cluster is computed and added as an index in the indexing structure.

An impression may appear more than one time in a tuple set, or it may appear in more than one of these tuple sets. The vote integration stage is in charge of obtaining a unique set of candidate impressions.

In this context, each tuple, $\langle s, i \rangle$, represents a vote in favor of the ith impression. To complete this stage, we developed a variant of the Borda count [51], as the vote integration method. This variant (Algorithm 3) is adapted to create a unique set of candidate impressions.

First, each tuple set L_c is sorted so that the tuples with a higher similarity value come first. Any repeated value of i in L_c is removed, keeping the tuple with the higher similarity value s. Next, a list of rankings R is created with the information of the sorted and filtered sets. In this case, we use a tuple to represent the position at which the impression appears in the corresponding L_c and the identifier of the impression. We iterate over R to obtain a single score per impression. This score is the accumulated sum of the inverted order. Finally, the impressions are sorted according to this score, in descending order, which gives more weight to those impressions found more times in the top positions of the ranking.

IV. EXPERIMENTAL SETUP

In this section, we describe the databases we used in our experimentation and the evaluation protocol we followed.

A. FINGERPRINT DATABASES

The databases come from different sources, including public data, proprietary data, and synthetic data. We use them in different stages of the proposed algorithm, as described in Section V.

We used the following three datasets from the National Institute of Standards and Technology (NIST [52] of the United States of America:

- SD4 [53]: Contains 4,000 rolled impressions (2,000 pairs) with a resolution of 500dpi and 480 × 512 pixels.
- SD14 [54]: Contains 54,000 rolled impressions (27,000 pairs) with a resolution of 500dpi and 832 × 768 pixels.
- SD27 [55]: Contains 258 latent fingerprints and their paired impressions, with a resolution of 500dpi and 800 × 768 pixels.

Note that each pair of fingerprints corresponds to two instances of the same finger. Additionally, we use a proprietary database with 106,921 impressions and 284 latent fingerprints. In this database, the size of the fingerprints varies from 377 × 483 pixels to 784 × 766 pixels of resolutions. Finally, a synthetic database with 997,377 impressions of which, for a matter of storage space, we only keep information of the minutia extracted, making the resolution irrelevant.

The synthetic database was generated with SFinGe version 4.1 (build 1746) Demo [11] and is used to observe how the proposed algorithm performs in a database with more than a million impressions. Artificially generated impressions by SFinGe have been used before in verification competitions, where they have obtained results similar to those of real databases [56].

B. EVALUATION PROTOCOL

We followed the evaluation protocol for indexing algorithms presented in the Handbook of Fingerprint Recognition [11]. This protocol evaluates fingerprint indexing algorithms by looking at the trade-off between the error rate (ER) and the penetration rate (PR).

The error rate is defined as the percentage of fingerprints not found concerning the total number of queries in the test set. The penetration rate is the percentage of the database that the system has to search. The balance between ER and PR is computed for all values of $max_{PR} \in [1, 100]$. This value determines the maximum number of candidate fingerprints

Algorithm 2 Retrieving Impressions

Function `retrieveImpressions`(*qcc, C, H*)
Input:

- *qcc*, query's MCC .
- $C = [c_1, c_2, \ldots, c_k]$, where c_j is the j-th center of the clusters computed in the preprocessing stage, and k is the number of clusters.
- H, indexing structure obtained in Algorithm 1.

Output: L_{qcc}, a set of tuples $\langle s, i \rangle$, where s is the similarity between *qcc* and the MCCs of the i-th impression.

begin
  Let d be the euclidean distance function.
  Let s be the similarity function defined in Cappelli *et al.* [42].
  $u \leftarrow a/k$, where a is the total number of MCCs indexed in H and k is the number of clusters in C.
  $D \leftarrow$ list of $c_j \in C$ sorted in ascending order according to $d(qcc, c_j)$.
  $indice \leftarrow 1$
  $L_{qcc} \leftarrow \{\}$
  repeat
    foreach $\{\langle cc, i \rangle : \langle c_{indice}, \langle cc, i \rangle\rangle \in H \wedge c_{indice} \in D\}$ **do**
      $L_{qcc} \leftarrow L_{qcc} \cup \{\langle s(qcc, c_j), i \rangle\}$
    $indice \leftarrow indice + 1$
  until $\|L\| >= u$
  return L_{qcc}

Algorithm 3 Vote Integration

Function `integrateVotes`(*L*)
Input:

- $L = [L_1, \ldots, L_m]$, where m is the number of MCCs extracted from the query latent fingerprint, and $L_c = \{\langle s, i \rangle_t, t \in [1, p_c]\}$ where p_c is the number of impressions retrieved by the c-th cylinder code of the query latent fingerprint.

Output: List if candidate impressions I.

begin
  $R \leftarrow \{\}$
  foreach $L_c \in L$ **do**
    Sort L_c by similarity in descending order.
    $P \leftarrow \{\}$
    $v \leftarrow 1$
    foreach $\langle s, i \rangle \in L_c$ **do**
      if $i \notin P$ **then**
        $R \leftarrow R \cup \{\langle v, i \rangle\}$
        $v \leftarrow v + 1$
        $P \leftarrow P \cup \{i\}$
  Let n be the number of indexed impressions.
  foreach $\langle v, i \rangle \in R$ **do**
    $score_i \leftarrow score_i + (n - v)$
  Sort impressions identifiers (i values) by their $score_i$ in descending order and store in I.
  return I

$max_C = max_{PR} * n/100$, where n is the total number of impressions in the database. If the candidate list contains more than max_C elements, only the first max_C candidates are used to compute the corresponding ER and PR. A detailed description of this step can be found in [57].

To compare indexing algorithms, we can look to the curves these algorithms generate, such as the one presented in Fig. 5. In this figure, the curve of one algorithm can dominate another curve through all PR values; this means that for every value of PR (x-axis), the dominant curve has a lower value of ER (y-axis). Therefore, the dominant curve is closer to the horizontal axis. In this example, algorithm B has a better performance than algorithm C.

It is possible to obtain curves that intersect each other. In this case, there is no better algorithm for any given value, so it is essential to establish which values of PR are of interest. It is also possible to obtain incomplete curves (examples can be found in [14], [22], [38]); in this case, the comparison is made in the range of values where all algorithm curves are defined.

V. RESULTS AND DISCUSSION

In this section, we describe the databases used in the index construction and the results of this step, followed by the description and results of two different experiment configurations. Additionally, we include a complementary experiment to test the proposed algorithm in the context of impressions. Finally, we include the results of our execution time analysis.

A. INDEX CONSTRUCTION

For the index construction step, we used the impressions of the NIST databases (SD4, SD14, and SD27) and the proprietary database. We excluded the latent fingerprints in SD27 and the proprietary database. In the cases of SD4 and SD14 databases, all the fingerprints were included in the clustering because they are impressions.

To cluster the cylinder codes we used the implementation of k-means++ [30] available in Weka [58]. K-means++ is an extension of the original k-means [59] that improve the initialization of the cluster centers to obtain clusters close to the optimal [30].

The parameter k was chosen, so it eases the reduction of the number of retrieving impressions by two or more orders of magnitude. We experimented with several values of k, before arriving at the selected value $k = 100$. With values greater than 100, we obtained worse results, whereas smaller values do not reduce two orders of magnitude.

During the index construction step, we noticed that one of the clusters was conformed by more than three times the average number of elements by clusters. This distribution causes

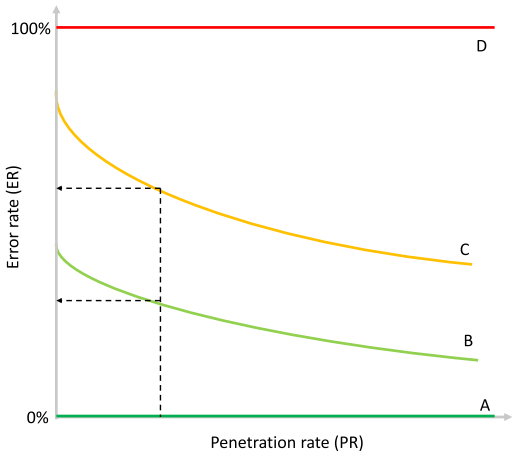


FIGURE 5. Evaluation of indexing algorithms [11]. Error rate and penetration rate are the used metrics. Letters from A to D represent four hypothetical algorithms. Dashed lines indicate how to evaluate algorithms B and C given a specific penetration rate; in this case, B is better, which yields a lower error. Denoted by the dark green line overlapping the x-axis, algorithm A achieved the ideal performance, making no mistakes. Conversely, the red line representing algorithm D shows the worst performance, with the maximum error in all values.

unfortunate consequences in our algorithm's performance because the number of descriptors needed to be filtered is high in this cluster. Furthermore, this would affect not only queries closest to this large cluster but also those closest to a cluster with not enough elements to complete the desired search space in the retrieving stage, and that would use the larger cluster to complete this set. For this reason, we decided to distribute the descriptors assigned to this cluster among the other clusters, keeping a total of 99 clusters.

B. RESULTS IN IMPRESSIONS DATASETS

Even though our main focus is latent fingerprints, we also tested our algorithm in the context of impressions. To evaluate this context, we take as a reference the work of Khodadoust and Khodadoust [31] given that it uses only minutiae information (triplets) and makes an extensive analysis of 31 algorithms that use only one descriptor and 17 that combine several descriptors. We selected the top four algorithms in Khodadoust and Khodadoust [31] for the NIST SD4 and NIST SD14 databases. Note that most of these works are not covered in the literature review because they are not limited to minutiae descriptors. Also, these results belong to a variant of the NIST SD14, which is reduced to the last 2,700 pairs.

Fig. 6 shows the results of our algorithm (MCCCluster4FI) compared to the top four algorithms in the NIST SD4 database: Khodadoust and Khodadoust [31], Su *et al.* [32], Cappelli [33], and Muñoz-Briseño *et al.* [21]. It is important to note that the algorithm of Cappelli [33] in Fig. 6 is based on the frequency and orientation of ridges and it is not the same as the baseline for the previous experiments (Cappelli *et al.* [14]).

Fig. 7 shows the results of our algorithm (MCCCluster4FI) compared to the top four algorithms in the NIST

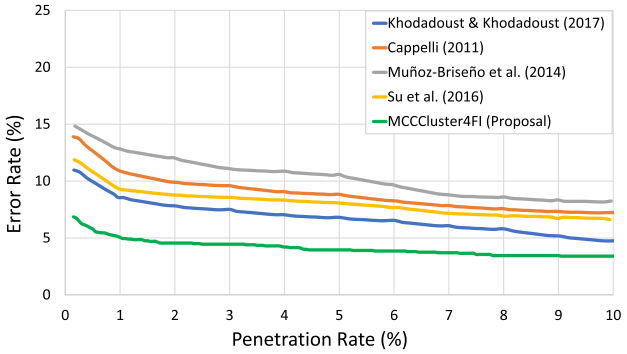


FIGURE 6. Comparison of the proposed algorithm against top algorithms reported in Khodadoust and Khodadoust [31] for the NIST SD4 dataset (2,000 impression pairs) up to 10% penetration rate.

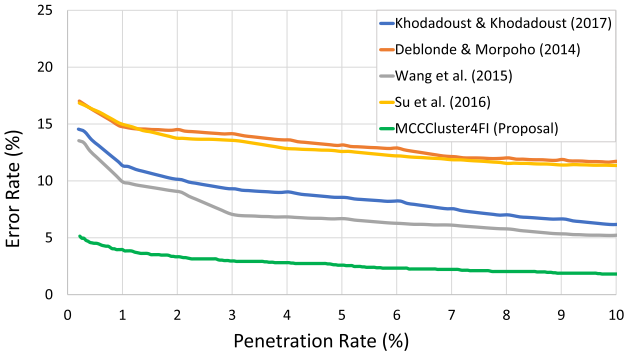


FIGURE 7. Comparison of the proposed algorithm against top algorithms reported in Khodadoust and Khodadoust [31] for the NIST SD14 dataset (last 2,700 impression pairs) up to 10% penetration rate.

SD14 database: Khodadoust and Khodadoust [31], Su *et al.* [32], Wang *et al.* [15], and Deblonde & Morpoho [34].

The proposed algorithm achieves an improvement compared to all the results reported in NIST SD4 (Fig. 6) and NIST SD14 (Fig. 7) by a margin of at least 1% and 3%, respectively, considering a 10% penetration rate. At lower penetration rates, the improvement becomes more notorious, a margin of at least 3% and 5%, respectively, at a 1% penetration rate.

C. RESULTS IN THE PROPRIETARY DATABASE

In this experiment, we applied our algorithm to the proprietary database as follows. The impressions in the database were used to build the structure in the indexing stage, and the latent fingerprints were used as queries.

The evaluation results are presented in Fig. 8. They show how the proposed algorithm (MCCCluster4FI) reduces the error consistently up to a 10% penetration rate than the baseline (Cappelli *et al.* [14]).

In this evaluation protocol, the lower the penetration rate, the lower the time it takes to process the list of candidate fingerprints returned by the algorithm. For this reason, we decided to analyze the results at crucial low penetration rates (0.01%, 0.1%, 0.5% y 1%). The table 3 shows the error rate at these penetration rates.

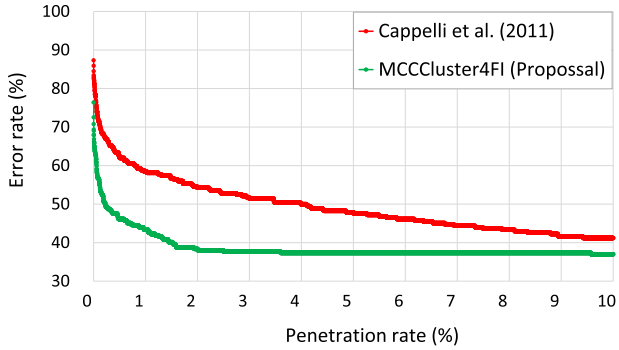


FIGURE 8. Results in the proprietary database up to 10% penetration rate for 284 latent fingerprints and 106,921 impressions.

TABLE 3. Error rate at key penetration rates for a proprietary database with 284 latent fingerprints and 106,921 impressions. The best values of Penetration Rate appear in bold.

Algorithm	Penetration rate 0.01%	0.1%	0.5%	1%
Cappelli et al. [14]	82.0%	71.5%	61.9%	58.4%
MCCCluster4FI	**65.8%**	**56.7%**	**46.1%**	**43.3%**

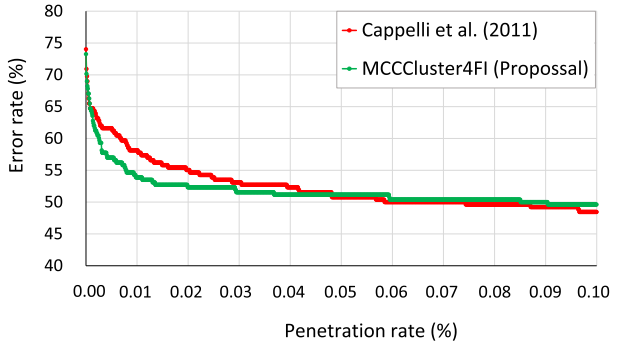


FIGURE 9. Results in the NIST SD27 experiment up to 0.1% for 258 latent fingerprints and 1,104,556 impressions.

TABLE 4. Error rate at crucial penetration rates for a database with 258 latent fingerprints (NIST SD27) and 1,104,556 impressions. The best values of Penetration Rate appear in bold.

Algorithm	Penetration rate 0.01%	0.1%	0.5%	1%
Cappelli et al. [14]	58.1%	**48.4%**	**39.9%**	**32.2%**
MCCCluster4FI	**53.9%**	49.6%	42.6%	39.5%

D. RESULTS IN THE NIST SD27 DATABASE

In this experiment, we applied our algorithm to the NIST SD27 database as follows. The impressions in the NIST SD27 database, the proprietary database, and the synthetic database was used to build the structure in the indexing stage, and the latent fingerprints in the NIST SD27 database were used as queries.

The evaluation results are presented in Fig. 9. They show how the proposed algorithm (MCCCluster4FI) has a lower error of up to 0.043% penetration rate. The performance comparison at key penetration rates is shown in Table 4.

In other words, our algorithm obtains better results when returning up to 475 candidate fingerprints (0.043%) of a total of 1,104,556 impressions. This number is already large to be reviewed manually by an expert to determine if the fingerprint matches one of these candidates.

TABLE 5. Average search time of MCCs, measured in seconds. These results correspond to the experiments in sections V-C and V-D. Abbreviations: lf = latent fingerprints and imp = impressions.

Algorithm	Experiments: proprietary database lf: 284 imp: 106,921	sd27+proprietary +synthetic lf: 258 imp: 1,104,556
Cappelli et al. [14]	0.627	2.626
MCCCluster4FI	4.483	23.218
Proportion of execution time	7.14	8.84

TABLE 6. Total time for indexing the impression databases, measured in seconds. These results correspond to the experiments in sections V-C and V-D. Abbreviations: lf = latent fingerprints and imp = impressions.

Algorithm	Experiments: proprietary database lf: 284 imp: 106,921	sd27+proprietary +synthetic lf: 258 imp: 1,104,556
Cappelli et al. [14]	7,242	15,260
MCCCluster4FI	3,695	5,830
Proportion of execution time	0.51	0.38

E. EXECUTION TIME ANALYSIS

A secondary goal of these experiments is related to the time it takes to obtain the results. The time to measure is the average search time of minutiae descriptors to form the list of candidate impressions, for each latent query fingerprint. The table 5 shows these results.

The aim is for this time to be reduced or remain at most one order of magnitude above the time reported by the implementation by Cappelli *et al.* [14]. As can be seen in Table 5, the proportion of time is in the established threshold. We made this distinction about the average search time to retrieve the list of MCC because the Algorithm 2 has a dynamic behavior depending on the number of MCCs in each index. Therefore, whenever this algorithm selects an index with a number of MCCs associated lower than the mean of MCCs per index, the time consumed grows due to the additional search to guarantee a homogeneous output. Hence, it was expected the algorithm of Cappelli *et al.* [14] outperforms ours in this aspect because it retrieves only the information associated through a family of hash functions and no further steps are done.

Nevertheless, between the two experiments with latent fingerprints (Fig. 8 and Fig. 9) there is an increase of 10.33 times of the search space, while our algorithm just increases the average search time by a factor of 5.18. It means that, though not perfect, our algorithm scales well. Also, the number of indices can be computed accordingly to keep a certain average of MCCs per index, and thus, reduce the effect of big databases.

Indexing is another step of the algorithm that can be analyzed separately. Fig. 6 shows the amount of time spent by both algorithms in the creation of the indexing structure. This time MCCCluster4FI algorithm outperforms the MCC SDK. It is possible because our indexing step processes every MCC only once, while the MCC SDK index the minutia descriptors as many times as the number of hash functions for which it is configured to work with. Although, this is a step performed infinitesimally fewer times than a search for an MCC in the indexing structure. That is why we focused more on the search step.

VI. CONCLUSION

We have proposed an indexing algorithm based on MCCs that finds clusters using k-means++. The proposed retrieval algorithm returns a homogeneous number of impressions per query, allowing a regular performance for each MCC. This algorithm is important because the number of MCCs in a given neighborhood can vary greatly. Finally, we have implemented a version of Borda's counting algorithm and used it to integrate the votes per impression to create a set of candidate fingerprints.

We tested the new algorithm in three different scenarios, considering only the MCCs and showing better results in all the cases. Without the elaboration, we also consider the problem of indexing fingerprints in the context of fingerprint verification using all the databases available. We observe that our algorithm outperforms other algorithms in that context too.

Most importantly, we used a background database with more than one million impressions, where the proposed algorithm achieves a lower error rate up to 0.043% penetration rate, a value that represents a reduction of four orders of magnitude in the search space.

This experimentation is more extensive than others reported on the literature, in both the number of impressions and latent fingerprints. The average search time per query fingerprint takes longer than those reported by the MCC SDK, but without increasing by one order of magnitude.

ACKNOWLEDGMENT

The authors would like to thank the Tecnologico de Monterrey, Mexico, where most of the research was performed as part of the master's degree in computer science. They also like to thank the professors and researchers at GIEE-ML (Machine Learning) group for their support during the seminars and the Department of Computer Science and Artificial Intelligence at Universidad de Granada, Spain, where part of the results was obtained. They also thank the Project PN-720 of CONACyT, titled Algoritmos basados en minucias para la identificación de huellas latentes dactilares y palmares. They also thank the collaboration and resources given by several institutions.

REFERENCES

[1] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, "Preface, overview," in *Handbook of Fingerprint Recognition*, 2nd ed. London, U.K.: Springer, 2009, p. 11.

[2] L. J. González-Soler, L. Chang, J. Hernández-Palancar, A. Pérez-Suárez, and M. Gomez-Barrero, "Fingerprint presentation attack detection method based on a bag-of-words approach," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, M. Mendoza and S. Velastín, Eds. Cham, Switzerland: Springer, 2018, pp. 263–271.

[3] L. J. González-Soler, M. Gomez-Barrero, L. Chang, A. P. Suárez, and C. Busch, "On the impact of different fabrication materials on fingerprint presentation attack detection," in *Proc. Int. Conf. Biometrics (ICB)*, 2019, pp. 1–6.

[4] D. Valdes-Ramirez, M. A. Medina-Pérez, R. Monroy, O. Loyola-González, J. Rodríguez, A. Morales, and F. Herrera, "A review of fingerprint feature representations and their applications for latent fingerprint identification: Trends and evaluation," *IEEE Access*, vol. 7, pp. 48484–48499, 2019.

[5] E. Ramírez-Sáyago, O. Loyola-González, and M. A. Medina-Pérez, "Towards inpainting and denoising latent fingerprints: A study on the impact in latent fingerprint identification," in *Pattern Recognition*. Cham, Switzerland: Springer, 2020, pp. 76–86.

[6] M. A. Medina-Pérez, A. M. Moreno, M. Á. F. Ballester, M. García-Borroto, O. Loyola-González, and L. Altamirano-Robles, "Latent fingerprint identification using deformable minutiae clustering," *Neurocomputing*, vol. 175, pp. 851–865, Jan. 2016.

[7] O. Loyola-Gonzalez, "Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view," *IEEE Access*, vol. 7, pp. 154096–154113, 2019.

[8] K. N. Win, K. Li, J. Chen, P. F. Viger, and K. Li, "Fingerprint classification and identification algorithms for criminal investigation: A survey," *Future Gener. Comput. Syst.*, vol. 110, pp. 758–771, Sep. 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X19315109

[9] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, "Introduction, fingerprint sensing and storage," in *Handbook of Fingerprint Recognition*, 2nd ed. London, U.K.: Springer, 2009, pp. 36–38.

[10] J. Rodríguez-Ruiz, M. A. Medina-Pérez, R. Monroy, and O. Loyola-González, "A survey on minutiae-based palmprint feature representations, and a full analysis of palmprint feature representation role in latent identification performance," *Expert Syst. Appl.*, vol. 131, pp. 30–44, Oct. 2019.

[11] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, 2nd ed. London, U.K.: Springer, 2009.

[12] C. Bai, T. Zhao, W. Wang, and M. Wu, "An efficient indexing scheme based on k-plet representation for fingerprint database," in *Proc. Int. Conf. Intell. Comput.* Cham, Switzerland: Springer, 2015, pp. 247–257. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-22180-9_25

[13] J. Khodadoust, A. M. Khodadoust, S. S. Mirkamali, and S. Ayat, "Fingerprint indexing for wrinkled fingertips immersed in liquids," *Expert Syst. Appl.*, vol. 146, May 2020, Art. no. 113153.

[14] R. Cappelli, M. Ferrara, and D. Maltoni, "Fingerprint indexing based on minutia cylinder-code," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 1051–1057, May 2011.

[15] Y. Wang, L. Wang, Y.-M. Cheung, and P. C. Yuen, "Learning compact binary codes for hash-based fingerprint indexing," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 8, pp. 1603–1616, Aug. 2015.

[16] C. Bai, W. Wang, T. Zhao, and M. Li, "Fast exact fingerprint indexing based on compact binary minutia cylinder codes," *Neurocomputing*, vol. 275, pp. 1711–1724, Jan. 2018.

[17] C.-C. Bai, W.-Q. Wang, T. Zhao, R.-X. Wang, and M.-Q. Li, "Deep learning compact binary codes for fingerprint indexing," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 9, pp. 1112–1123, Sep. 2018.

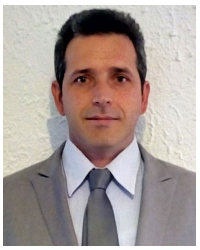
[18] V. Anand and V. Kanhangad, "Pore-based indexing for fingerprints acquired using high-resolution sensors," *Pattern Anal. Appl.*, vol. 23, no. 1, pp. 429–441, Feb. 2020.

[19] B. Bhanu and X. Tan, "Fingerprint indexing based on novel features of minutiae triplets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 616–622, May 2003.

[20] K. Choi, D. Lee, S. Lee, and J. Kim, "An improved fingerprint indexing algorithm based on the triplet approach," in *Proc. 5th Int. Conf. Audio-Video-Based Biometric Person Authentication*. Berlin, Germany: Springer, 2003, pp. 584–591.

[21] A. Muñoz-Briseño, A. Gago-Alonso, and J. Hernández-Palancar, "Fingerprint indexing with bad quality areas," *Expert Syst. Appl.*, vol. 40, no. 5, pp. 1839–1846, Apr. 2013.

[22] A. Gago-Alonso, J. Hernández-Palancar, E. Rodríguez-Reina, and A. Muñoz-Briseño, "Indexing and retrieving in fingerprint databases under structural distortions," *Expert Syst. Appl.*, vol. 40, no. 8, pp. 2858–2871, Jun. 2013.
[23] O. Iloanusi, A. Gyaourova, and A. Ross, "Indexing fingerprints using minutiae quadruplets," in *Proc. Workshop Biometrics Conf. Comput. Vis. Pattern Recognit.* Colorado Springs, CO, USA: IEEE, Jun. 2011, pp. 127–133.
[24] O. N. Iloanusi, "Fusion of finger types for fingerprint indexing using minutiae quadruplets," *Pattern Recognit. Lett.*, vol. 38, pp. 8–14, Mar. 2014.
[25] S. Chikkerur, A. N. Cartwright, and V. Govindaraju, "K-plet and coupled BFS: A graph based fingerprint representation and matching algorithm," in *Proc. Int. Conf. Biometrics*. Berlin, Germany: Springer, 2006, pp. 309–315.
[26] P. Mansukhani, S. Tulyakov, and V. Govindaraju, "A framework for efficient fingerprint identification using a minutiae tree," *IEEE Syst. J.*, vol. 4, no. 2, pp. 126–137, Jun. 2010.
[27] X. Liang, A. Bishnu, and T. Asano, "A robust fingerprint indexing scheme using minutia neighborhood structure and low-order delaunay triangles," *IEEE Trans. Inf. Forensics Security*, vol. 2, no. 4, pp. 721–733, Dec. 2007.
[28] G. Bebis, T. Deaconu, and M. Georgiopoulos, "Fingerprint identification using delaunay triangulation," in *Proc. Int. Conf. Inf. Intell. Syst.*, 1999, pp. 452–459.
[29] S. Wang and J. Hu, "Alignment-free cancelable fingerprint template design: A densely infinite-to-one mapping (DITOM) approach," *Pattern Recognit.*, vol. 45, no. 12, pp. 4129–4137, Dec. 2012.
[30] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*. Philadelphia, PA, USA: SIAM, 2007, pp. 1027–1035.
[31] J. Khodadoust and A. M. Khodadoust, "Fingerprint indexing based on expanded delaunay triangulation," *Expert Syst. Appl.*, vol. 81, pp. 251–267, Sep. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095741741730204X
[32] Y. Su, J. Feng, and J. Zhou, "Fingerprint indexing with pose constraint," *Pattern Recognit.*, vol. 54, pp. 1–13, Jun. 2016.
[33] R. Cappelli, "Fast and accurate fingerprint indexing based on ridge orientation and frequency," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 6, pp. 1511–1521, Dec. 2011.
[34] A. Deblonde, "Fingerprint indexing through sparse decomposition of ridge flow patches," in *Proc. IEEE Symp. Comput. Intell. Biometrics Identity Manage. (CIBIM)*, Dec. 2014, pp. 202–208.
[35] A. A. Paulino, E. Liu, K. Cao, and A. K. Jain, "Latent fingerprint indexing: Fusion of level 1 and level 2 features," in *Proc. IEEE 6th Int. Conf. Biometrics: Theory, Appl. Syst. (BTAS)*, Sep. 2013, pp. 1–8.
[36] D. Peralta, S. García, J. M. Benitez, and F. Herrera, "Minutiae-based fingerprint matching decomposition: Methodology for big data frameworks," *Inf. Sci.*, vol. 408, pp. 198–212, Oct. 2017, doi: 10.1016/j.ins.2017.05.001.
[37] D. Peralta, M. Galar, I. Triguero, D. Paternain, S. García, E. Barrenechea, J. M. Benítez, H. Bustince, and F. Herrera, "A survey on fingerprint minutiae-based local matching for verification and identification: Taxonomy and experimental evaluation," *Inf. Sci.*, vol. 315, pp. 67–87, Sep. 2015, doi: 10.1016/j.ins.2015.04.013.
[38] U. Jayaraman, A. K. Gupta, and P. Gupta, "An efficient minutiae based geometric hashing for fingerprint database," *Neurocomputing*, vol. 137, pp. 115–126, Aug. 2014.
[39] K. Tiwari and P. Gupta, "Indexing fingerprint database with minutiae based coaxial Gaussian track code and quantized lookup table," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 4773–4777.
[40] T. Uz, G. Bebis, A. Erol, and S. Prabhakar, "Minutiae-based template synthesis and matching for fingerprint authentication," *Comput. Vis. Image Understand.*, vol. 113, no. 9, pp. 979–992, Sep. 2009.
[41] J. Khodadoust and A. M. Khodadoust, "Fingerprint indexing based on minutiae pairs and convex core point," *Pattern Recognit.*, vol. 67, pp. 110–126, Jul. 2017.
[42] R. Cappelli, M. Ferrara, and D. Maltoni, "Minutia cylinder-code: A new representation and matching technique for fingerprint recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2128–2141, Dec. 2010.
[43] A. J. Sanchez-Fernandez, L. F. Romero, D. Peralta, M. A. Medina-Perez, Y. Saeys, F. Herrera, and S. Tabik, "Asynchronous processing for latent fingerprint identification on heterogeneous CPU-GPU systems," *IEEE Access*, vol. 8, pp. 124236–124253, 2020.
[44] L. Wang and M. Dai, "Application of a new type of singular points in fingerprint classification," *Pattern Recognit. Lett.*, vol. 28, no. 13, pp. 1640–1650, Oct. 2007.
[45] J. Khodadoust and A. M. Khodadoust, "Partial fingerprint identification for large databases," *Pattern Anal. Appl.*, vol. 21, no. 1, pp. 19–34, Feb. 2018.
[46] A. K. Jain and J. Feng, "Latent fingerprint matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 88–100, Jan. 2011.
[47] T.-Y. Jea, "Minutiae-based partial fingerprint recognition," Ph.D. dissertation, Dept. Comput. Sci. Eng., State Univ. New York Buffalo, Buffalo, NY, USA, 2005, Art. no. aAI3203911.
[48] J. Feng and A. K. Jain, "Filtering large fingerprint database for latent matching," in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.
[49] R. P. Krish, J. Fierrez, D. Ramos, J. Ortega-Garcia, and J. Bigun, "Pre-registration for improved latent fingerprint identification," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 696–701.
[50] R. P. Krish, J. Fierrez, D. Ramos, J. Ortega-Garcia, and J. Bigun, "Pre-registration of latent fingerprints based on orientation field," *IET Biometrics*, vol. 4, no. 2, pp. 42–52, Jun. 2015.
[51] D. Black, "Partial justification of the borda count," *Public Choice*, vol. 28, no. 1, pp. 1–15, Dec. 1976, doi: 10.1007/BF01718454.
[52] (2019). National Institute of Standards and Technology. *Main Page*. Accessed: Nov. 17, 2019. https://www.nist.gov/
[53] C. I. Watson and C. L. Wilson, "NIST special database 4," *Fingerprint Database, U.S. Nat. Inst. Standards Technol.*, vol. 17, no. 77, p. 5, 1992.
[54] C. I. Watson, "NIST special database 14: Mated fingerprint card pairs 2 version 2," U.S. Nat. Inst. Standards Technol., Gaithersburg, MD, USA, 2001.
[55] M. D. Garris and R. M. McCabe, *NIST Special Database 27: Fingerprint Minutiae From Latent and matching Tenprint Images*. Gaithersburg, MD, USA: U.S. Department of Commerce, National Institute of Standards and Technology, 2000.
[56] R. Cappelli, D. Maio, D. Maltoni, J. L. Wayman, and A. K. Jain, "Performance evaluation of fingerprint verification systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 1, pp. 3–18, Jan. 2006.
[57] Biometric System Laboratory. (2019). *Benchmark Area: Fingerprint Indexing*. Accessed: Sep. 17, 2019. [Online]. Available: https://biolab.csr.unibo.it/fvcongoing/UI/Form/BenchmarkAreas/BenchmarkAreaFIDX.aspx
[58] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explor. Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.
[59] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.



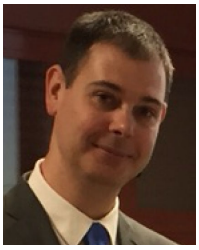
ISMAY PÉREZ-SÁNCHEZ received the bachelor's degree in computer science from Central University "Martha Abreu" of Las Villas, Cuba, in 2006, and the master's degree (Hons.) in computational science from the Tecnologico de Monterrey, Mexico, in 2019, where he is currently pursuing the Ph.D. degree in computational science. He has been working on indexing algorithms for latent fingerprints, and previously he has participated in projects related to numerical models for engineering simulations, specifically, high-performance computing topics. His research interests include high-performance computing, machine learning, and biometrics. He received the National Award from the Cuban Science Academy for the work developed in discrete element method modeling, in 2017.

BÁRBARA CERVANTES received the bachelor's degree from the Tecnologico de Monterrey, in 2010, the M.Sc. degree in artificial intelligence from The University of Edinburgh, in 2012, with a focus on learning from data, and the Ph.D. degree in computer science from the Tecnologico de Monterrey, in 2017. She is currently associated with the GIEE-ML (Machine Learning) Research Group, Tecnologico de Monterrey. She has experience of working in the higher education industry and in research laboratories. She has experience and interest in projects in different areas of artificial intelligence, including machine translation, machine learning, data analysis, and visualization.

MIGUEL ANGEL MEDINA-PÉREZ received the Ph.D. degree in computer science from the National Institute of Astrophysics, Optics and Electronics, Mexico, in 2014. He is currently a Research Professor with the Tecnologico de Monterrey, where he is a member of the GIEE-ML (Machine Learning) Research Group. He has ranked one in the Mexican Research System. He has published tens of articles in referenced journals, such as *Information Fusion*, IEEE TRANSACTIONS ON AFFECTIVE COMPUTING, *Pattern Recognition*, *Information Sciences*, and *Expert Systems with Applications*. He has extensive experience on developing software to solve pattern recognition problems. A successful example is a fingerprint and palmprint recognition framework which has more than 1.3 million visits and 135 thousand downloads. His research interests include pattern recognition, data visualization, explainable artificial intelligence, fingerprint recognition, and palmprint recognition.

RAÚL MONROY received the Ph.D. degree in artificial intelligence from The University of Edinburgh, under the supervision of Prof. Alan Bundy, in 1998. He is currently a Full Professor in computing with the Tecnologico de Monterrey. He is also the Leader of the GIEE-ML (Machine Learning) Research Group, Tecnologico de Monterrey. His current research interests include discovery and application of novel models of machine learning especially in the context of cybersecurity. Since 1998, he has been a member of CONACyT's National Research System, ranked three, and a fellow of the National Academy of Sciences.

OCTAVIO LOYOLA-GONZÁLEZ received the Ph.D. degree in computer science from the National Institute for Astrophysics, Optics and Electronics, Mexico, in 2017. He worked as a Distinguished Professor and a Researcher with the Tecnologico de Monterrey at Puebla, for undergraduate and graduate programs of computer sciences. He is currently responsible for running machine learning and artificial intelligence practice inside Altair Management Consultants Corporation, where he is involved in the development and implementation using analytics and data mining with the Altair Compass Department. He has outstanding experience in the fields of big data and pattern recognition, cloud computing, the IoT, and analytical tools to apply them in sectors where he has worked for as banking and insurance, retail, oil and gas, agriculture, cybersecurity, biotechnology, and dactyloscopy. From these applied projects, he has published several books and articles in well-known journals, and he has several ongoing patents as the manager and a researcher in Altair Compass. He is a member of the National System of Researchers in Mexico (ranked one). He received several awards from different institutions due to his research work on applied projects.



SALVADOR GARCÍA received the B.S. and Ph.D. degrees in computer science from the University of Granada, Granada, Spain, in 2004 and 2008, respectively. He is currently an Associate Professor with the Department of Computer Science and Artificial Intelligence, University of Granada, Granada. He has organized several special sessions and workshops related to data preprocessing and evolutionary learning in conferences, such as Hybrid Intelligent Systems, Intelligent Systems Design and Applications, and International Joint-Conference of Neural Networks. He has been associated with the international program committees and organizing committees of several regular international conferences, including IEEE CEC, ICPR, ICDM, and IJCAI. He has published more than 90 articles in international journals (more than 60 in Q1), H-index 44, and over 60 papers in international conference proceedings (data from the Web of Science). He is the coauthor of the books entitled *Data Preprocessing in Data Mining* (Springer) and *Learning from Imbalanced Data Sets* (Springer). His research interests include data science, data preprocessing, big data, evolutionary learning, deep learning, metaheuristics, and biometrics. He has been given some awards and honors for his personal work or for his publications in and conferences, such as the IFSA-EUSFLAT 2015 Best Application Paper Award and the IDEAL 2015 Best Paper Award. He belongs to the list of the Highly Cited Researchers in the area of computer sciences (Clarivate Analytics), from 2014 to 2018. He is also an Associate Editor of *Information Fusion* (Elsevier), *Swarm and Evolutionary Computation* (Elsevier), and *AI Communications* (IOS Press) journals.

FRANCISCO HERRERA (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in mathematics from the University of Granada, Spain, in 1988 and in 1991, respectively. He is currently a Professor with the Department of Computer Science and Artificial Intelligence, University of Granada, and the Director of the Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI). He is also an Academician with the Spanish Royal Academy of Engineering. He has been the Supervisor of 45 Ph.D. students. He has published more than 400 journal articles and receiving more than 73 000 citations (Scholar Google, H-index 135). His current research interests include among others, computational intelligence, including fuzzy modeling, computing with words, evolutionary algorithms, and deep learning, information fusion and decision making, and data science, including data preprocessing, prediction, and big data. He was a fellow of ECCAI and IFSA, in 2009 and 2013, respectively. He received several honors and awards, among others the IEEE TRANSACTIONS ON FUZZY SYSTEMS Outstanding Paper, in 2008 and 2012; the 2010 Spanish National Award on Computer Science ARITMEL to the Spanish Engineer on Computer Science; the International Cajastur "Mamdani" Prize for Soft Computing (Fourth Edition, in 2010); the 2011 Lotfi A. Zadeh Prize Best Paper Award (IFSA Association); the 2013 AEPIA Award to a Scientific Career in Artificial Intelligence; the 2014 XV Andaluca Research Prize Maimnides; the 2017 Andaluca Medal from the Regional Government of Andaluca; and the 2018 Granada Science and Innovation City Award. He has been selected as a Highly Cited Researcher in the fields of computer science and engineering (Clarivate Analytics), since 2014. He also acts as the Editor-in-Chief of the international journal *Information Fusion* (Elsevier). He also acts as an editorial member of a dozen of journals.