Robust core-point-ROI based fingerprint identification using a sparse classifier

Alexandru Paul Condurache and Alfred Mertins
Institute for Signal Processing,
University of Lübeck,
D-23538 Lübeck, Germany
Email: condurache,mertins@isip.uni-luebeck.de

***Abstract*—We address the problem of automated fingerprints-based person identification from poor-quality fingerprints. Our solution includes the definition of a region of interest centered over the fingerprint at a reference point. From this region of interest a feature vector is computed that is invariant to some geometrical transforms but also to point transforms of the gray levels in the region of interest. This feature vector is then classified by means of a sparse classifier. We successfully test our algorithms on a publicly available fingerprints database and show that they are robust to a set of issues afflicting current fingerprint-identification systems in the case of poor-quality fingerprints.**

I. Introduction

The pattern of ridges and valleys of the skin covering the interior side of distal phalanges (at the tip of a finger) represents a fingerprint. The fingerprints represent unique characteristics of each human being and as such have been the biometric feature of choice for a long time. Currently they receive attention from the machine vision community in the quest for automatic person-identification systems to be used for various purposes ranging from access control to electronic banking.

A typical fingerprint identification algorithm consists of a feature extraction step followed by a classification step. During classification, the query fingerprint is assigned to one of the available classes – one for each person in the target group—of enrolled fingerprints. The fingerprints are considered to be available in the form of digital images coming either from a sensor or from digitized latent fingerprints. The latent fingerprints are collected by forensic modalities from various items. Person identification with respect to a target group/database is achieved only if the classification can be conducted with sufficient confidence.

There are several types of features that can be extracted from a fingerprint [11]. Level-1 features are related to general characteristics of the fingerprint, like the location of singular points (e.g., points characterized by large ridge curvature). Level-2 features include more particular characteristics, these so called minutiae features are, for example, the locations of ridge bifurcations and endings. Level-3 features are related to fingerprint details like ridge width, edge contour or pores.

The classification step is usually conducted with the help of either image correlation, phase matching, skeleton matching or minutiae matching [11]. Minutiae-based approaches are most commonly used [6], [5]. They mimic to a certain extent the way a human expert usually goes to work on classifying a fingerprint.

It is considered that for fingerprint identification Level-2 and Level-3 features are needed [11], and under such circumstances, there are many effective solutions to the problem of fingerprint identification [18]. However, the problem space is not yet fully covered, as there are still some unanswered questions that lead to new research opportunities [11]. Major difficulties are encountered when working with poor quality fingerprints, yet another difficulty is represented by the small overlapping area between a query fingerprint and the enrolled fingerprints. Fingerprints are additionally afflicted by nonlinear distortions and, in the case of latent fingerprints, a complex background. Under such circumstances, the extraction of fingerprint features of Level-2 and Level-3, and hence fingerprint identification by methods based on these types of features, becomes challenging.

In this contribution we propose a fingerprint identification method that works despite the above mentioned difficulties. However, we assume that for each enrolled user, several fingerprints of (at least) one finger are available. From the above-mentioned feature categories, our solution uses only one Level-1 feature, i.e., the location of the core point. The core point is the most central point of the fingerprint, around which the ridge orientation changes rapidly [12], [16]. We use the core point as reference and select around it a region of interest (ROI) of the fingerprint image. The ROI serves the purpose to concentrate our analysis on the same area of the fingerprint, irrespective of how it was positioned over the imaging sensor. From the ROI, a feature vector is extracted that includes a certain selection of Discrete Cosine Transform (DCT) coefficients. The feature vector concentrates information that is less-likely to be afflicted by image noise, difficult backgrounds or scars and scratches. Such information is related to the global pattern of a fingerprint. A query fingerprint is then assigned to a specific finger from the target database, with the help of a sparse classifier. This classifier exhibits a set of characteristics that make it well suited for our problem of robust fingerprint recognition. The sparse classifier naturally offers the possibility to compute the confidence in the computed result. If this confidence is not high enough, the identification of the respective fingerprint should be conducted—in the case of poor-quality fingerprints—by a human expert. To correctly identify a finger despite small fingerprint overlap area, or the availability of a partial or partially corrupted fingerprint, we harness the abilities of both the sparse classifier and the feature vector to handle occlusions. Also, the sparse classifier works well despite the availability of a very small number of training samples per class, which is usually the case for fingerprint identification. Furthermore, with a sparse classifier the precise choice of features is less important than the number of features [21]. The sparse classifier has been previously used in the context of fingerprint analysis, but for the purpose of pore-matching in high-quality fingerprints [15]. To handle distortions, we assume that, in general, the training set includes fingerprints with the usual linear and nonlinear geometric transformation caused by the acquisition procedure. At the same time, the feature extraction is designed to offer a feature vector with additional invariance properties to some geometric transformations but also to point-transformations of the fingerprint-image's gray levels. Furthermore, the memory footprint (measured in Bits) of the feature vector is far smaller than that of the ROI or the original image.

We summarize our main contributions as follows: first, we introduce a new definition of the core point and describe methods to detect it accordingly, second, we define a transform-based feature vector that captures basic fingerprint-information that is available even in the most difficult fingerprint images, and third, we introduce the sparse classifier for fingerprint identification.

In Section II we describe the feature extraction process and the sparse classifier. In Section III we demonstrate our solution on a public database. Finally Section IV contains the conclusions.

II. Methods

For each fingerprint image, from a ROI placed at the core point, a feature vector is computed. The sparse classifier then assigns a feature vector to the class whose training vectors span the subspace closest to it. For sparse classification, the dimension of the feature space is also important. It needs to be chosen in relation to the number of training samples and the mean number of representatives per class (see Section II-B2).

A. Feature extraction

To detect the core point, we use the orientation of the ridges. The features we've used are some DCT coefficients of the core-point-centered ROI. The particular choice of DCT coefficients offers mild rotation and translation invariance and at the same time robustness to changes in the mean of the fingerprint image. To find out which DCT features to include in the feature vector we have applied a feature selection procedure [10].

1) Core-point detection: The core point is currently defined as the point of maximal curvature of the concave ridges of a fingerprint [12]. Initial approaches to detecting the core point were based on the Poincar index [13]. They work well only for good-quality fingerprints. To detect the core point in poor-quality fingerprints, robust methods are needed. Such methods

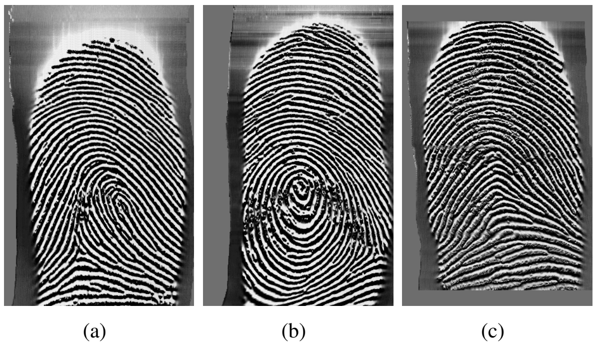
Fig. 1. Various types of fingerprints: loop (a), whorl (b) and plain arch (c).

are proposed in [12], [3], [16] but they have difficulties with arch-type fingerprints (see Figure 1), due to the definition used for the core point. In this contribution, we use another definition of the core point. First we assume that the available fingerprints are approximately vertically oriented, then we use the observation that starting from the top of the fingerprint and going down, the ridges are less and less flat—whereby with flat we mean similar to a horizontal line (see Figure 1). We define the core point as the point where such ridge flatness becomes minimal.

Next we measure ridge-flatness by the sine of the angle between the orientation vector and the x axis in a fingerprint image (see Figure 2 (a)). The orientation vector is evaluated from a local neighborhood Ω at each pixel as the eigenvector of the orientation tensor corresponding to the minimal eigenvalue. The orientation tensor is computed as [1]:

$$\mathbf{J} = \left[\begin{array}{cc} B(R_y \cdot R_y) & B(R_x \cdot R_y) \\ B(R_x \cdot R_y) & B(R_x \cdot R_x) \end{array} \right]$$

with B being a smoothing kernel whose impulse response is related to Ω, and R_i is the directional derivative in the direction i. Therefore, the grey-level variation along the orientation vector is minimal.

The sine image (with the values of the sine function of the orientation angle at each pixel) is considered to exhibit two classes, one for orientation angles close to $\frac{\pi}{2}$ and one with orientation angles close to zero. Using Otsu's method for unsupervised binary classification [19]—that optimizes a separability measure—we binarize the sine image, defining labels such that the class of pixels with orientation close to zero represent the object and the rest the background (see Figure 2 (b)).

The binarized sine image is then morphologically processed [9] to detect the core point. First, we break it into two parts at the region of the core point by eroding with an appropriate disc-like structuring element of diameter d_1 and then dilating with a similar structuring element but with a diameter $d_2 < d_1$ (see Figure 2 (c)). Then we select the object region in the upper central part of the image. For this purpose we use as seed points the object pixels in the upper quarter (along the y axis) of the image. As the pixels may be

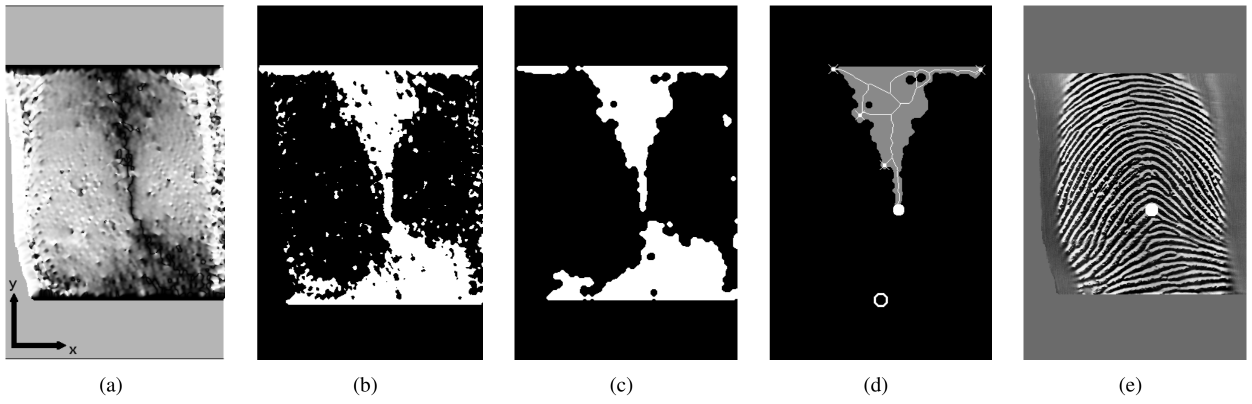

Fig. 2. Processing chain for core-point detection: sine image (a), binarization result (b), breaking at the core-point region (c), upper central object-part with skeleton and detected terminations – the core point is marked by a disc and the reference position by a circle (d), final result (e).

divided among several object structures in the upper quarter, we take only those pixels belonging to the structure closest to the image center. All object points linked to these over an eight-neighborhood (i.e., all neighboring pixels situated at an Euclidian distance $d \leq \sqrt{2}$) are selected. We then compute the morphological skeleton of the result and detect the termination points [14] (see Figure 2 (d)). The (usually) lowest most central termination point is the sought core point. We find this point as the termination point closest to the empirically established reference position given by $\left[\frac{5m}{6}, \frac{n}{2}\right]$, with m the number of lines and n the number of columns in the image.

2) The feature vector: As our purpose is to design a system robust to partial occlusions of the fingerprint as well as to partial and total corruptions (like e.g., overlays), the feature vector must also exhibit such properties. At the same time, we would like to reduce the number of dimensions with respect to the size of the training sample to avoid problems related to the curse of dimensionality. Therefore, in contrast to other appearance-based methods [12] we use here transform features. We did not use LDA and other related methods, because we want a method unrelated to the number of classes—i.e., the number of fingers in the query database. The PCA is of limited use, as there are hardly any common high-eigenvalue variation modes, considering how dissimilar the various types of fingerprints are (see Figure 1). Furthermore, with PCA we would theoretically need to recompute the transformation matrix with each new finger enrolled in the database. Here we make use of the DCT. To find out which DCT coefficients to use, we have used feature selection.

The feature vector is computed from an ROI centered at the core point (see Figure 3 (a)). The size of the ROI should be chosen in relation to the resolution of the fingerprint imaging device. For some fingerprint images the ROI may go over the boundaries of the image, the corresponding region is filled with zeros—being thus treated as total occlusions. Our feature vector contains DCT coefficients of the ROI from a certain region of the DCT space (see Figure 3 (b)).

The selected DCT coefficients are largely invariant to translation. By eliminating the coefficients corresponding to DC and very low frequencies, the feature vector is invariant to changes in the mean of the gray-levels of the fingerprint image. The specific choice of DCT coefficients represents a subsampled, alias-pledged representation (see Figure 3), that in comparison to the original ROI has mild rotation invariance properties. We concentrate thus on the global pattern of a fingerprint, the most general and often available fingerprint information.

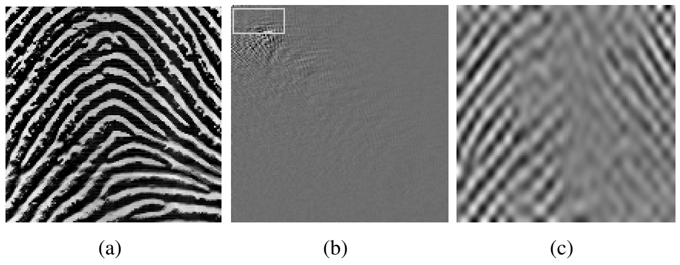

Fig. 3. ROI (a), DCT coefficients – the marked region gives our feature vector (b) and ROI reconstructed with the selected coefficients (c).

a) Feature selection: Assuming we have a small labeled data set at our disposal, we can use it to conduct fast feature selection similar to [10]. The objective function is the squared training error of a set of linear discriminants. We start from a vector with k DCT coefficients randomly selected from the DCT space with a total of $N \gg k$ coefficients. For each selected coefficient, we compare the objective function for a feature vector including it with that obtained for a feature vector excluding it. At the same time we rank the coefficients by the value of the difference between the two objective functions. We then eliminate the least relevant coefficient and replace it with another one randomly chosen from the remaining $N-k$ coefficients. We then repeat the procedure q times. After q repetitions, the better coefficients will be selected more often (see Figure 4). We compute our feature vector from that region of the DCT space which has the coefficients most often selected during feature selection. We select this region of feature concentration, instead of using precisely the selected features, because we can't assume that the data sample used for feature selection is optimally representative for the random variable feature vector.

b) Mild rotation invariance: By mild rotation invariance we mean that for relatively small rotations, the selected DCT coefficients of the original and the rotated image differ insignificantly. To clarify further what we mean by mild rotation invariance here, we use the analogy of a fingerprint to a planar wave. We observe that a planar wave of frequency ω_1 is less variant to rotations than a planar wave of frequency $\omega_2 > \omega_1$, when both are considered within the same finite area. For example, a rotation of five degrees (with the center of the coordinate system at the center of the image) the squared error between the original and the rotated image increases with the frequency.

A similar phenomenon takes place in the case of our feature vector as well. It contains a low-frequency representation of the ROI, similar to the low-frequency planar wave form above, while the original ROI behaves similar to the higher-frequency planar wave.

B. Sparse Classification

Sparse classification is a type of nearest subspace method. The distance function used is the quality of the reconstruction of a test sample by a linear combination of basis vectors from a class subspace. The coefficients of this reconstruction are computed based on the principle of parsimony. It offers thus a principle-based approach in comparison to the rather heuristic methods employed in other nearest-subspace methods. In the case of sparse classification it is assumed that any sample in a class can be expressed as a linear combination of other samples in the same class. Thus, a new sample will lie in the linear span of the training samples. The training samples represent the basis vectors for the class subspace. Such considerations have previously been used for classification as early as [20] and have ignited research in the field of appearance-based classification. Sparse classification is usually fast. It has a complexity order $O(t^3 + ndt)$ [8], with t being the number of nonzero elements in the sparse vector, n denoting the size and d the dimension of the training set. In comparison, fast nearest-subspace methods have a complexity order of $O(nd^2)$ [2]. For sparse classification to work properly, the assumption that the basis vectors of each class subspace are present in the training set must hold.

Building on such premises, sparse-representation based classification looks for the sparsest representation of a test sample in terms of a matrix of training samples. This representation is sparse because it should contain only samples from the class to which the test sample belongs [21].

1) Sparse representations: Let the training matrix be denoted by $\mathbf{T} = [\mathbf{T}_1, \ldots, \mathbf{T}_k]$, containing the class-submatrices $\mathbf{T}_i = [\boldsymbol{v}_{i,1}, \ldots, \boldsymbol{v}_{i,N_i}]$ with $i = 1, \ldots, k$, where N_i is the number of samples in class i and k the number of classes. The total number of vectors in $\mathbf{T}$ is $n = \sum_{i=1}^{k} N_i$, and each vector $[\boldsymbol{v}_j]_1^n$ has m entries. Then, for each new sample $\boldsymbol{y}$ we ideally have:

$$\begin{aligned} \boldsymbol{y} = &\quad \mathbf{T}\boldsymbol{x} \\ = &\quad x_{j,1}\boldsymbol{v}_{i,1} + \cdots + x_{j,N_i}\boldsymbol{v}_{i,N_i} \end{aligned} \tag{1}$$

where the coefficient vector $\boldsymbol{x} = [x_j]_1^n$ has entries x_j different from zero only for the training-space samples from the class i to which $\boldsymbol{y}$ belongs. This system of equations is usually overcomplete with the number n of vectors in the training space being well larger than the dimension m of the vectors. Thus, there are infinitely many solutions to (1), and the ones that interest us are the ones with a sparse representation $\boldsymbol{x}$.

Assuming an equal number of training samples per class, the more classes the more sparse $\boldsymbol{x}$. Thus we search not for some $\hat{\boldsymbol{x}} \in \mathbb{R}^n$, but for the *sparsest* vector that solves equation (1). We find it by optimizing over the ℓ^0 pseudo norm, solving:

$$\hat{\boldsymbol{x}} = \arg\min \parallel \boldsymbol{x} \parallel_0 \quad \text{subject to} \quad \mathbf{T}\boldsymbol{x} = \boldsymbol{y} \tag{2}$$

Ideally, assuming that a vector $\boldsymbol{y}$ is represented solely with the training vectors from the correct class (that is, the coefficients in $\boldsymbol{x}$ for training vectors from other classes are all zero), the vector $\boldsymbol{y}$ can be classified by looking up to which class the nonzero entries in $\boldsymbol{x}$ belong. In practice, of course, questions about the required amount of sparsity and the uniqueness of the sparsest solution arise. In [7] it was shown that if some $\boldsymbol{x}$ with less than $\frac{m}{2}$ nonzero entries verifies $\boldsymbol{y} = \mathbf{T}\boldsymbol{x}$, then this is the unique sparsest solution. This means that we have a good chance of finding the correct and unique sparsest solution to (1) even for two-class problems or for configurations where the number of training samples per class is not the same over all classes, provided we have enough samples in the training set.

2) Classification: In practice, because the solution to equation (2) is computationally difficult to find, we solve instead:

$$\hat{\boldsymbol{x}} = \arg\min \parallel \boldsymbol{x} \parallel_1 \quad \text{subject to} \quad \mathbf{T}\boldsymbol{x} = \boldsymbol{y} \tag{3}$$

Minimizing over the ℓ^1 norm instead of the ℓ^0 pseudo norm yields the same solution if $\boldsymbol{x}$ is sparse enough [4].

The vector $\boldsymbol{x}$ found after we solve (3) will usually have the largest entries for one class only, and small non-zero entries for other classes as well. Next, to introduce the decision rule, we use the following notation: $1_i(x)$ is the indicator function for class i and $\boldsymbol{v}_1 \odot \boldsymbol{v}_2$ is the component-wise product of two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, i.e., the Hadamard product. Thus, $1_i(x_j) \odot \boldsymbol{x}$ selects the entries of $\boldsymbol{x}$ that belong to class i. $C(\boldsymbol{y})$, with $C : \mathbb{R}^m \to \{1, \ldots, k\}$ is the function that assigns a class label to the vector $\boldsymbol{y}$. A classification rule that harnesses the class-subspace structure has been proposed in [21]. It assigns the test sample $\boldsymbol{y}$ to the class whose training-set vectors best reproduce $\boldsymbol{y}$. The decision rule reads:

$$C(\boldsymbol{y}) = \arg\min_i \parallel \boldsymbol{y} - \mathbf{T}(1_i(x_j) \odot \hat{\boldsymbol{x}}) \parallel_2 \tag{4}$$

a) Confidence index: The sparse classification framework provides room for the introduction of a degree of confidence in the classification result. In order to express such a confidence, for the decision rule in (4), a *sparsity* *concentration index* is defined as [21]:

$$SCI(\boldsymbol{x}) = \frac{l \max_i \left(\frac{\| 1_i(x_j) \odot \boldsymbol{x} \|_1}{\| \boldsymbol{x} \|_1} \right) - 1}{l - 1} \qquad (5)$$

A decision is accepted only if $SCI(\hat{\boldsymbol{x}}) \geq \tau$, otherwise the corresponding $\boldsymbol{y}$ is labeled as "unsure". The parameters l and τ need to be set empirically. Here we propose heuristically to choose the parameter l of the SCI as $l = 1.5 \cdot C$, where C is the number of classes. The precise value for τ should be established by cross-validation.

b) Number of features: It can be shown that, if $\boldsymbol{x}_0$ has $t \ll n$ nonzero elements, then $d \geq 2t \log(\frac{n}{d})$ features are sufficient to obtain the correct sparse solution by means of the optimization (2).

III. Experiments and discussion

In the last decade fingerprint-based biometric applications have received a lot of attention from the machine vision community. Fingerprint Verification Contest (FVCs) have been organized in 2000 [17], 2002, 2004 [6] and 2006 [5]. These are mainly geared towards fingerprint verification and not fingerprint identification/recognition as is the case with this contribution. Therefore, a direct comparison is not possible.

Furthermore, the sparse classifier offers the possibility to default a decision in favor of a higher-accuracy classifier/human observer, if the confidence is not high enough. This has a major influence on the way our algorithm is used. The test setup is designed to take into account such particularities of our approach.

We have conducted experiments on the DB3 database. This database has been used within the FVC 2004 [6] and is currently available online. The DB3A database contains 800 fingerprints, eight fingerprints for each of 100 different fingers and is meant for test purposes. The DB3B database contains 80 fingerprints, eight fingerprints for ten different fingers. They have been acquired with a thermal sweeping sensor (Atmel FingerChip), at a resolution of 512 *dpi* and an image size of 300×480 pixels. To test our algorithm we have divided the data from DB3A into eight equal parts, each part containing 100 different fingerprints and conducted eight-fold cross validation. The establishment of various parameters for our method has been done with the help of DB3B. The size of the ROI was 141×141 pixels.

For our algorithm, the enrollment time for a fingerprint is approximately 0.4 seconds. This includes 0.39 seconds for detecting the core point and 0.016 seconds for computing the feature vector. The match time for a fingerprint with respect to a database containing 700 fingerprints, i.e., seven fingerprints per finger and 100 different fingerprints is 1.29 seconds, including about 0.88 seconds for classification. The experiments were conducted under MATLAB R2010b on a Core 2 Duo E6600 (2.66 GHz) machine with 4 GB of RAM.

A. Core-point detection

The parameters of the core-point detection method are: $\Omega = 11$, $d_1 = 5$ and $d_2 = 4$. The filters implementing the directional derivative had a length $l_d = 9$. For testing we have used a manual ground-truth of core points, established by the first author. To account for human imprecisions, we have declared a core point to be found successfully if it falls into a region of 21×21 pixels centered at the manual core point. Within this setup, we have successfully found the core point in 720 cases from 800.

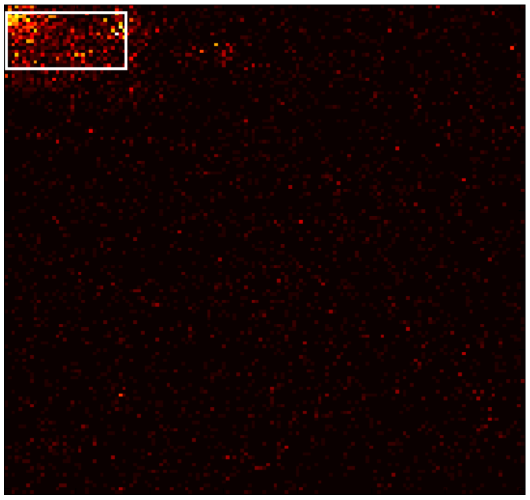

Fig. 4. Results of feature selection. The chosen coefficients are those in the white rectangle. The brighter a pixel is the more often the DCT at the corresponding position has been selected.

B. Feature vector

To find out which DCT coefficients are optimally suited for our purposes, we have conducted a feature selection using DB3B. The number of potentially available DCT coefficients is $N = 141^2$. We've conducted feature selection with $k = 500$ coefficients, and iterated $q = 1500$ times. The results are shown in Figure 4. The brighter the pixels, the more often the respective DCT coefficient was selected. Our DCT region corresponds with the concentration region of the most selected DCT coefficients. The region R_b has the size 17×35. Thus, we get a feature vector with $d_{R_b} = 595$ dimensions. We have also conducted experiments with the DCT coefficients from a smaller region R_s of the size 17×17, such that it covers approximately the left half of R_b. With the resulting vector of $d_{R_b} = 289$ dimensions, we obtained very similar identification results. For our experiments we had $n = 700$ and $t = 7$, therefore in both cases d verifies the bound on the minimal number of features.

C. Finger identification

We have computed a type of ROC by varying the confidence threshold τ between zero and one in steps of 0.01 and computing for each threshold the mean rate of correct decisions and the mean rate of "uncertain" decisions, where the means were taken over the cross-validation results

To find out how sensitive is our method to an accurate detection of the core-points, we have computed two ROCs.

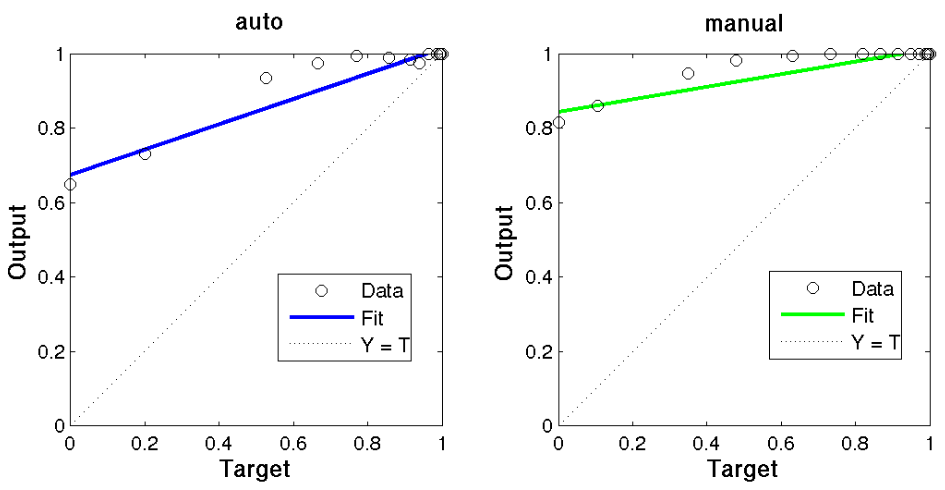

Fig. 5. Left: the ROC for automatically detected core-points and right: the ROC for manually detected core-points. Target gives the *"uncertain"* rate and output the *"correct decision"* rate.

For the first we used features computed based on the set of automatically detected core points and for the second based on the manually selected core points. The results are shown in Figure 5. As it can be seen, the accuracy of the detection of the core point influences the results up to an "uncertain" rate of 0.5.

We consider 80% correct decisions to be the minimal rate for successful fingerprint identification from poor-quality fingerprints. Using the automatic core points, we obtain 80% correct decisions for a 30% "uncertain" rate. This means that from the 70% of the data for which a decision is obtained, 80% are correct decisions. For manual core-point detection, we reach an 80% correct decision rate for 0% "uncertain" rate.

To simulate bad fingerprints, we have low-pass filtered the images using a Gaussian kernel. The performance of our method remained unchanged when using filters up to a minimal 3dB bandwidth of $\frac{\pi}{6}$. Our method can handle occlusions up to 30% of the ROI. We have encountered such occlusions in our database for those fingerprints whose core points were placed near the borders of the analyzed image, such that only about 70% of the ROI was filled with fingerprint information, the rest being filled with zeros. As long as the core point was correctly detected, all these images were correctly classified.

Besides the DCT, we have tested several other feature extraction methods on DB3B, these are the Linear Discriminant Analysis (LDA), the Principal Components Analysis (PCA), and we have also tested a feature vector that contains a downsampled (by a factor of three) ROI. As expected, none of them worked well, with correct-decision rates of under 15% for $\tau = 0$.

IV. Summary, Conclusions and outlook

We have described a fingerprint-identification framework designed to work with poor-quality fingerprints. Under these circumstances we make use only of the most basic fingerprint features, the core point. At the same time, we build a feature vector that captures only part of the information found in the ridge-pattern of a finger, namely that part that is imprinted on a grabbed item under most difficult conditions for the subsequent fingerprint acquisition. Therefore, the feature-extraction process yields a feature vector that is not particularly rich in information, and to obtain satisfactory results, we need to compensate for this in the classification phase. By its properties the sparse classifier is optimally suited to work under such conditions. It works well despite occlusions or corruptions of large parts of the analyzed fingerprint image as well as with a less-informative feature vector (as long as the size of the feature space is well chosen and the training set covers most of the variability to be expected in the test sample), and with a small number of training examples per class.

We believe that the novel definition of the core point proposed here is able to deal with all types of fingerprints, i.e., including arch-type fingerprints, however, the particular method we use for the detection of the core point leads to slightly imprecise results. This makes our core-point detection method not optimally suited to detect core points for fingerprint alignment (with respect to rotation and translation). The algorithms we describe here profit from the ability to work with arch-type fingerprints and are designed to be robust to such imprecision.

While our method is not heavily influenced by small errors in positioning the core point, a complete failure definitely leads to a wrong decision for the analyzed fingerprint. We are currently investigating both improved core-point detection methods and enrollment-failure detection methods that evaluate if the core point can be detected at all in a given image.

The feature vector contains actually a downsampled bandpass representation of the fingerprint ROI. The low frequencies related to average gray level and large, low-frequent structures, like, e.g., a cut or bruise, are ignored as are high-frequency noise structures. At the same time the ridge pattern is blurred (and undersampled). The autocorrelation of the fingerprint ROI remains high over larger rotations and/or shifts after the blurring as opposed to before the blurring. Therefore our algorithms are insensitive to both small translations of the ROI due to imprecise core-point detection and small rotations due to differences in placing the fingerprint over the sensor. The precise choice of DCT coefficients has been established by feature selection.

In contrast to many previous works, we use the implicit assumption that several fingerprints of the same finger are available—even though theoretically the sparse classifier could return a decision with even one training sample per class. This assumption becomes increasingly valid the more ubiquitous fingerprint-based systems become. Even if in the beginning a finger gets just one impression stored, the more often the respective person uses the system, the more impressions of the same finger—under various transformations/influences—become available. Clearly this information should be harvested. In our simulations we worked under the assumption that the query fingerprint is bad, but the enrolled fingerprints are relatively good. Should bad fingerprints also be available we expect an improvement of the results of our method.

The "uncertain" decision, which comes naturally for the sparse classifier, should be interpreted as there is not enough information to make a decision with enough confidence, thus, in this case, a human observer should analyze the respective fingerprint. This is the setup for which our algorithm has been designed. Clearly, under such circumstance the combined correct decision rate of our framework will be greatly improved.

Acknowledgment

The authors would like to thank Florian Müller for the fruitful discussions pertaining to feature selection.

References

[1] T. Aach, I. Stuke, C. Mota, and E. Barth. Estimation of multiple local orientations in image signals. In *Proceedings of ICASSP-2004*, pages III 553–556, Montreal, Canada, May 17–21 2004. IEEE.

[2] Ronen Basri, Tal Hassner, and Lihi Zelnik-Manor. Approximate nearest subspace search with applications to pattern recognition. *CVPR*, pages 1–8, 2007.

[3] A. M. Bazen and S. H. Gerez. Systematic methods for the computation of the directional fields and singular points of fingerprints. *IEEE TPAMI*, 24:905–919, July 2002.

[4] E.J. Candès and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE TIT*, 52(12):5406–5425, 2006.

[5] R. Cappelli, M. Ferrara, and D. Maltoni. Minutia cylinder-code: A new representation and matching technique for fingerprint recognition. *IEEE TPAMI*, 32(12):2128–2141, 2010.

[6] R. Cappelli, D. Maio, D. Maltoni, J.L. Wayman, and A.K. Jain. Performance evaluation of fingerprint verification systems. *IEEE TPAMI*, 28:3–18, 2006.

[7] D. Donoho and M. Elad. Optimal sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization. *Proc. Nat'l Academy of Sciences*, pages 2197–2202, March 2003.

[8] David Donoho and Yaakov Tsaig. Fast solution of ell-1-norm minimization problems when the solution may be sparse. Technical Report 18, Stanford University Department of Statistics, 2006.

[9] E. R. Dougherty. *An introduction to morphological image processing.* SPIE Optical Engineering Press, 1992.

[10] T. Gramss. Word recognition with the feature finding neural network (FFNN). In *Proc. IEEE Workshop Neural Networks for Signal Processing*, pages 289–298, Princeton, NJ, USA, Oct. 1991.

[11] A.K. Jain, J. Feng, and K. Nandakumar. Fingerprint matching. *IEEE Computer*, pages 36–44, February 2010.

[12] A.K. Jain, S. Prabhakar, L. Hong, and S. Pankanti. Filterbank-based fingerprint matching. *IEEE Transactions on Image Processing*, 9(5), May 2000.

[13] K Karu and A. K. Jain. Fingerprint classification. *Pattern Recognition*, 29(3):389–404, 1996.

[14] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[15] F. Liu, Q. Zhao, L. Zhang, and D. Zhang. Fingerprint pore matching based on sparse representations. In *Proc. of ICPR*, pages 1630–1633, Istanbul, Turkey, August. 2010.

[16] M. Liu, X. Jiang, and A.C. Kot. Fingerprint reference-point detection. *Eurasip JASP*, 4:498–509.

[17] D. Maio, D. Maltoni, J. L. Wayman, and A. K. Jain. Fvc2000: Fingerprint verification competition. *IEEE Transactions on PAMI*, 24(3):402–412, 2002.

[18] D. Maltoni, D. Maio, A.K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2009.

[19] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-9(1):62–66, 1979.

[20] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE TPAMI*, 13(10):992–1006, 1991.

[21] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE TPAMI.*, 31(2):210–227, 2009.