Int. J. Appl. Math. Comput. Sci., 2015, Vol. 25, No. 3, 659–674
DOI: 10.1515/amcs-2015-0048

amcs

SIGNATURE VERIFICATION: A COMPREHENSIVE STUDY OF THE HIDDEN SIGNATURE METHOD

JOANNA PUTZ-LESZCZYŃSKA [a]

[a]Faculty of Electronics and Information Technology
Warsaw University of Technology, ul. Nowowiejska 15/19, 00-663 Warsaw, Poland
e-mail: jputz@elka.pw.edu.pl

Many handwritten signature verification algorithms have been developed in order to distinguish between genuine signatures and forgeries. An important group of these methods is based on dynamic time warping (DTW). Traditional use of DTW for signature verification consists in forming a misalignment score between the verified signature and a set of *template signatures*. The right selection of template signatures has a big impact on that verification. In this article, we describe our proposition for replacing the template signatures with the *hidden signature*—an artificial signature which is created by minimizing the mean misalignment between itself and the signatures from the enrollment set. We present a few hidden signature estimation methods together with their comprehensive comparison. The hidden signature opens a number of new possibilities for signature analysis. We apply statistical properties of the hidden signature to normalize the error signal of the verified signature and to use the misalignment on the normalized errors as a verification basis. A result, we achieve satisfying error rates that allow creating an on-line system, ready for operating in a real-world environment.

Keywords: verification, on-line recognition, time warping, hidden signature.

1. Introduction

A handwriting signature understood as a graphical sign that represents a person seems to be the most powerful behavioral biometric, because document signing has been present for decades in everyday live. During the last few years, the traditional signing on a paper has slowly been replaced by digitizer tablets that are used to capture a signature. However, most of them are only put on documents as images, and no automatic verification is performed. This has a number of reasons.

The main issue is that verification algorithms' quality is lower than for other biometrics. This is caused by the fact that, in contrast to most of other biometric characteristics, signature verification has to deal not only with the signatures belonging to other people (*random forgeries*) but also with trained forgeries (*skilled forgeries*). Another is that the genuine realizations of a person' signature may differ from instance to instance. Signature instances can be understood as representations of the original signature, which is (at some point of life) not subjected to variations due to fatigue, emotional states, etc. Signature instances can differ not only in amplitudes or values at certain points, but also in dynamics. The issues mentioned have been under research for the last 35 years. The first attempt at handwriting verification was undertaken by Herbst and Liu (1977). Since then, this subject has been widely explored by scientists, both at universities as well as at commercial companies.

In this paper we are dealing with the so-called on-line signature verification, meaning that the signatures under analysis are registered with the use of a digitizer tablet (Fig. 1), and each point of the signature is labeled with its time moment. Using tablets to collect the signatures has an additional important effect. Not only can the pen position be registered, but also additional characteristics, such as the pen pressures and pen position angles as they change in time. Consequently, each signature instance is registered in the form of several sequences (the sequence of x-pen positions, the sequence of y-pen positions, pen pressure, etc.). It can be mathematically treated as a multidimensional time series in (finite) time $\mathbf{T}$. The time series dimension is determined by the number of signature characteristics which the device is able to capture, e.g., position coordinates $x \in X, y \in Y$, pen pressure $p \in P$ and angles: azimuth $z \in Z$ and altitude $l \in L$ (Fig. 1). An on-line signature, denoted by g, can be represented as

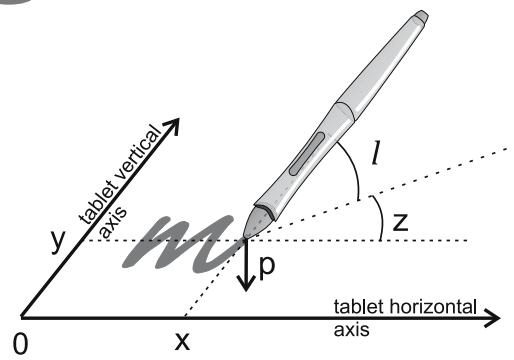

Fig. 1. Signature on-line acquisition by Wacom Intuos (WACOM, 2015).

a parametric curve in $X \times Y \times P \times Z \times L$, i.e., a mapping

$$g : \mathbf{T} \to X \times Y \times P \times Z \times L, \tag{1}$$

where

$$g = \begin{bmatrix} g^x & g^y & g^p & g^z & g^l \end{bmatrix}^T. \tag{2}$$

The set of all possible signatures will be denoted by $\mathbf{G}$.

If we use only a sub-vector of functions, with coordinates in a list *list*, we denote it by g^{list}. For example, if $list = x, y$, we have

$$g^{x,y} = \begin{bmatrix} g^x & g^y \end{bmatrix}^T. \tag{3}$$

In our tests, not all the signature data are used. Each signature is represented by a time sequence $g = \{g(t), \quad t = 1, 2, \ldots, M_g\}$ of the length M_g whose elements $\begin{bmatrix} g^{\Delta x} & g^{\Delta y} & g^p \end{bmatrix}^T$ consist of the pen tip coordinates $\begin{bmatrix} g^{\Delta x} & g^{\Delta y} \end{bmatrix}^T$ and the pen tip pressure g^p. The nonparametric features Δx, Δy are the x and y coordinate differences between two consecutive points, and p stands for pen pressure. We use $\Delta x, \Delta y$ instead of x, y, as they are invariant with respect to translation. Additionally, in conjunction with the DTW algorithm, they are quite successful (Kholmatov and Yanikoglu, 2005; Putz-Leszczynska and Pacut, 2009) in capturing similarities and ignoring irrelevant differences. The literature also suggests that the angle signals are not useful, so they were removed, too.

An essential issue concerning signature biometrics is the way the handwritten signature is captured, together with the comparison methods that depend on the signature representation. The wide review of both on-line and off-line verification methods by Impedovo and Pirlo (2008) shows possible implementations of signature-based biometric systems together with their results. All the algorithms can be split into two different comparison techniques, thus obligating one to choose from two distinct types of templates represented by the following:

- **Template signature(s):** The template is represented by one template signature (Schmidt and Kraiss, 1997), or a number of template signatures (Sakamoto *et al.*, 2001; Yoshimura and Yoshimura, 1992) selected from an enrollment set (signatures captured during registration).

  Two issues are related to this approach: (i) selecting the number of template signatures and (ii) choosing the decision making method.

  Usually the number of signatures in a template is defined without any explanation (Sakamoto *et al.*, 2001; Schmidt and Kraiss, 1997; Yoshimura and Yoshimura, 1992). The decision making methods are also arbitrary. One way of choosing a template signature(s) is to split a set of enrollment signatures into as many clusters as the number of signatures in a template and then, for each cluster, to select the "best" signature with the use of the min-max method (Yoshimura and Yoshimura, 1992). Another possibility is to choose the template signatures, as the ones with the lowest mean distance in comparison with the rest of enrollment signatures (Sakamoto *et al.*, 2001). The distance between two signatures can be computed for example with DTW.

- **Model:** The template is represented by a statistical model. During the estimation phase, a model is estimated with the use of enrollment signatures. This can be used for both nonparametric and parametric features. A popular approach when dealing with nonparametric features is to use a hidden Markov model (HMM) (Muramatsu and Matsumoto, 2003; Van *et al.*, 2007), a statistical model in which the system being modeled is assumed to be a Markov process with an unobserved state. Neural networks (NNs) are commonly used for parametric features (Xiao and Leedham, 1999; Marinai *et al.*, 2005). Neural networks and support vector machines (SVMs) can be used to model nonlinear complex relationships between inputs and outputs of enrollment signatures (Hong-Wei and Zhong-Hua, 2005; Fauziyah *et al.*, 2009).

In this paper we propose a method that combines both the approaches. We still use a signature instance in the template, but it is not one of the enrollment signatures. Instead, it is an artificial signature which is an estimate of the abstract representation of the signature that is kept (stored) in the signer's brain—the ideal signature. Therefore, for every signature there exists an abstract representation, called the hidden signature, from which every other instance can be derived. This abstract representation can be estimated from a collection of available signature instances (i.e., enrollment signatures) and this methodology can be regarded as a model creation

as well.

In the present paper we outline our hidden time methodology, and also show how this solution works with DTW methods, including time influence. In the next section, we present the hidden signature idea, methodology and its comparison together with the testing approach. In Section 3, we discuss the usage of statistical properties of the hidden signature to normalize the error signal of the verified signature and to use the misalignment on the normalized errors as a verification base. Section 4 summarizes the results.

2. Hidden signature

It is obvious that a good template assures successful verification. Therefore, the template creation stage is so important. A large number of algorithms like these based on DTW do not create a model but use original signatures collected during the enrollment process *(enrollment signatures*) at the verification stage (*template signatures*). Here an important question arises—which of the N enrollment signatures $\{g_1, g_2, \ldots, g_N\}$ should be selected for the template (*template signatures*). There are three possible solutions:

- all the enrollment signatures are used in the template,
- a subset of enrollment signatures forms the template,
- one signature.

The first solution generates two problems. One is the resulting size of the template which could be too big to be kept on a smartcard. Secondly, the more signatures are in the template, the more comparisons are needed and the more comparison time increases. The second solution partially addresses the problems mentioned, but generates others, namely, how many template signatures should be used in the template and which statistic should be used to select template signatures from among the enrollment ones.

The last option can be implemented in two ways. The simpler way is to select the "best signature" from the enrollment signatures, the one that minimizes the average misalignment within the enrollment set (DTW can be used for that). However, there is a chance that the selected signature would not represent the whole enrollment set. An alternative option is to create a 'model' signature. A common solution to sample variability is averaging. Namely, several samples are combined by an arithmetic average to produce a new specimen. This solution results in a small template size (because we have one signature), and the possibility that it would represent the whole enrollment set is much bigger than for the 'best signature'.

Various statistical theorems assume that, under a wide class of assumptions, the variability of the resulting sample representations is reduced and, if measured by the variance, the reduction is proportional to the number of averaged samples, if they are independent. Unfortunately, this popular approach cannot be applied to signatures. Since the signature consists of a series of points, averaging must be performed on signature points, by passing from point to point, rather than on the entire series: the points of two or more signature instances to be compared must be adequately selected. Indeed, the corresponding points of the signatures were not necessarily written at the same time moment, or they do not belong to the same area of the paper image of the signature. Readily, the duration of the signing varies from signature to signature, showing again that the signature points to be averaged cannot be selected according to identical time labels. The problem here to average such sequences remains unsolved.

That is why we introduced a more complex solution that is a *hidden signature.* By that we understand an "ideal" signature realization. This ideal signature exists only in its owner's mind. During that signing process this signature is distorted by the writing conditions, and that is why realizations of one person's signature differs. Hence, during the template creation we want to inverse this process and estimate the ideal signature with the use of a person's signatures captured during the enrollment process—we calculate the hidden signature. Thus, we introduce the definition that the *hidden signature* is an artificial signature that minimizes the average misalignment between itself and the signatures from the enrollment set:

$$\hat{g} = \arg\min_{g \in \mathbf{G}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{D}\big(g, g_i\big), \tag{4}$$

where $\mathbf{G}$ denotes the set of all possible signatures and $\mathcal{D}\big(g_1, g_2\big)$ stands for the misalignment between g_1, g_2 (the definition (13)). An example of a hidden signature is visualized in Fig. 2.

From the mathematical point of view, the hidden signature is a solution of a nonlinear minimization problem. Nonlinear problems are usually solved by iterative refinement. In each iteration, a single hidden signature $\hat{g}$ estimation is computed. The aim of every successive iteration is to minimize the average misalignment V between the hidden signature approximation and the signatures from the enrollment set $\{g_1, \ldots, g_N\}$. We call it *the average misalignment*:

$$V = \min_{j=1,\ldots,H} \frac{1}{N} \sum_{i=1,\ldots,N} \hat{\mathcal{D}}(\hat{g}_j, g_i), \tag{5}$$

where $\hat{\mathcal{D}}(\hat{g}_j, g_i)$ denotes the misalignment score between two signatures $\hat{g}_j$ and g_i (the definition (13)).

This general approach can have many versions and refinements. We considered two methods of hidden signature estimation: genetic calculations, and iterative

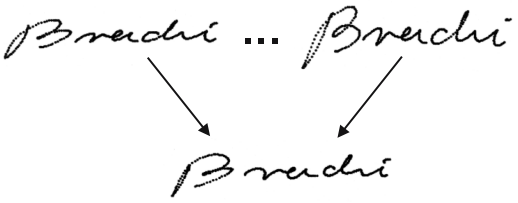

Fig. 2. Enrollment signatures (top), the hidden signature estimated from the enrollment signatures (bottom). The estimated hidden signature is visually similar to the signatures from the enrollment set, having a comparable global slope and size proportions.

point-by-point averaging. The latter is still preferred because of its good verification parameters and short calculation time. This algorithm is constructed as an iterative procedure that in each step transforms all the enrollment signatures to a common time scale by a nonlinear time transformation (warping) and then finds their average in this scale. In this new (warped) time scale, the enrollment signatures are averaged, and this average is called the hidden signature. By probability theory, the hidden signature is close to the mean value of the real signature after warping, thus assuring the proper level of invariance with the training set.

With some additional assumptions, iterative point-by-point averaging extends the least squares approach. The method of least squares assumes that the best fit is realized by minimization of the sum of the squared deviations (least-squares error). Since our problem is a conjunction of least-squares modeling and optimal warping, the hidden signature estimates the expected values of points of optimally warped enrollment signatures. Iterative point-by-point averaging can be regarded as an extension of the least-squares approach since in each iteration we find least-squares averaging among warped enrollment signatures.

In this article, we will present the details of implementations of both methods and their comprehensive comparison.

2.1. Hidden signature estimation: DTW bases. Each signature is represented by a time sequence $g = \{g(t), \quad t = 1, 2, \ldots, M_g\}$ of the length M_g whose elements $\left[g^{\Delta x}\ g^{\Delta y}\ g^{p}\right]^T$ consist of the pen tip coordinates $\left[g^{\Delta x}\ g^{\Delta y}\right]^T$ and the pen tip pressure g^p. Direct (point-by-point) comparison of time sequences(signatures) is not adequate, because two signatures g_1 and g_2 may differ in the time dimension.

This problem can be solved with the use of dynamic time warping (DTW), which also measures the dissimilarity between the signatures. The sequences are "warped" nonlinearly in time according to the *warping path* $\boldsymbol{w}$, which is a parametric (discrete) curve that aligns g_1 and g_2 (Fig. 3) and "connects" each point of the aligned sequences g_1 and g_2:

$$\boldsymbol{w}(\ell) = [w^{\mathrm{t}}(\ell)\ \ w^{\tau}(\ell)]^T, \quad \ell = 1, \ldots, L_w, \tag{6}$$

where

$$w^{\mathrm{t}}(\ell) \in \{1, \ldots, M_1\}, \quad w^{\tau}(\ell) \in \{1, \ldots, M_2\},$$

L_w denotes the warping path length. Functions $w^t(\ell)$ and $w^{\tau}(\ell)$ are nondecreasing in $\ell = 1, \ldots, L_w$. The warping path defines a dynamic alignment of elements of g_1 and g_2, aligning $g_1(w^t(\ell))$ with $g_2(w^{\tau}(\ell))$. A warping path between r and g will be denoted by $\boldsymbol{w}(r, g)$. By $\boldsymbol{w}(\ell; r, g)$ we will denote the coordinates of the ℓ-th point of this warping path. For the visualization purposes, successive points of the warping path are connected with straight lines.

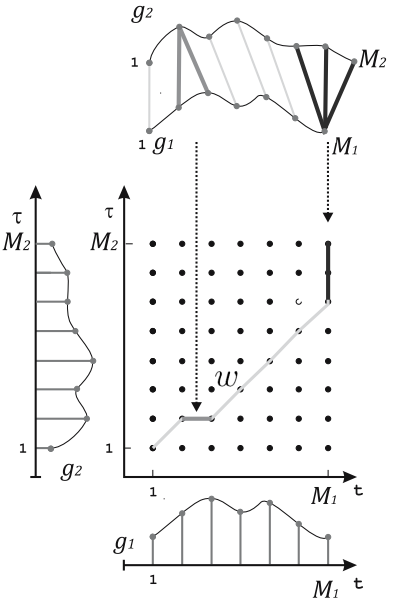

Fig. 3. Warping path of two sequences g_1 and g_2 on a two-dimensional grid: discrete warped time $\tau \in [0, M_2]$ vs. discrete reference time $t \in [0, M_1]$.

Assuming the sequence values belong to some feature space Φ, a comparison between two sequences $g_1,\ g_2$ requires a measure of the *local cost function*:

$$d : \Phi \times \Phi \to \mathbb{R} \geq 0. \tag{7}$$

The local cost function can be measured using typical distance metrics, for example, with the L_2 distance

$$d\big(g_1(t), g_2(\tau)\big) = \|g_1(t) - g_2(\tau)\| \tag{8}$$

or the squared L_2 distance

$$d\big(g_1(t), g_2(\tau)\big) = \|g_1(t) - g_2(\tau)\|^2, \tag{9}$$

where $t \in 1, 2, \ldots, M_1,\ \tau \in 1, 2, \ldots, M_2$.

The local cost function (9) has been provided for DTW by Putz-Leszczynska and Pacut (2005). It has several useful qualities, in particular the advantage of being differentiable. This local cost function was used in the research presented in this study.

The first step of the dynamic time warping algorithm is to build the *local cost matrix* which is defined as

$$\Delta(g_1, g_2) = \Big\{ d\big(g_1(t), g_2(\tau)\big)\ ,\ t = 1, 2, \ldots, M_1, \\ \tau = 1, 2, \ldots, M_2 \Big\}. \tag{10}$$

The algorithm finds the *alignment path* which "runs" through the local cost matrix (Fig.4).

The *misalignment* score

$$\mathcal{D}(g_1, g_2, \boldsymbol{w}) = \sum_{\ell=1}^{L_w} d\big(g_1(w^t(\ell)), g_2(w^\tau(\ell))\big) \tag{11}$$

is based on the sum of local costs $d\big(g_1 w^t(\ell)), g_2(w^\tau(\ell))\big)$ between elements of sequences r and g at the points belonging to the warping path. In the set of all possible warping paths W_{g_1,g_2} associated with given time sequences, we can define an *optimum* warping path $\hat{\boldsymbol{w}}(g_1, g_2)$ between g_1 and g_2, which minimizes its associated misalignment, namely,

$$\hat{\boldsymbol{w}}(g_1, g_2) = \arg \min_{\boldsymbol{w} \in \mathrm{W}_{g_1,g_2}} \mathcal{D}(g_1, g_2, \boldsymbol{w}). \tag{12}$$

According to (12), the minimum misalignment between sequences r and g is thus equal to

$$\hat{\mathcal{D}}(g_1, g_2) = \min_{\boldsymbol{w} \in \mathrm{W}_{g_1,g_2}} \mathcal{D}(g_1, g_2, \boldsymbol{w}) \\ = D(g_1, g_2, \hat{\boldsymbol{w}}(g_1, g_2)). \tag{13}$$

The DTW algorithm finds the optimum alignment path, which "runs" through the low–cost areas—"valleys" in the cost matrix (Fig. 4).

2.2. Hidden signature estimation: Iterative point-by-point averaging (IPPA). The iterative point-by-point averaging procedure consists of the initialization phase and the estimation phase. We assume that during each iteration the hidden signature is estimated only once.

2.2.1. Initialization. The first important decision is to define the hidden signature length. We considered several options and the results are discussed Section 3.3. The simplest way is to calculate the hidden signature length as the simple average of signatures from the enrollment set, independently for each user, namely ,

$$M = \frac{1}{N} \sum_{n=1}^{N} M_n. \tag{14}$$

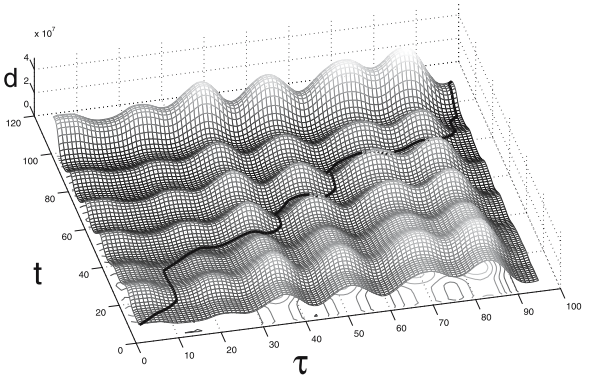

Fig. 4. Local cost matrix mesh and the optimal warping path (black solid line).

Finally, we assume that during the initialization all the enrollment signatures $\{g_1, \ldots, g_N\}$ are linearly resampled into a time length M (Fig. 5)$\{g_1^0, \ldots, g_N^0\}$. Therefore, it is possible to average the warped enrollment signature, thus obtaining the initial approximation of the hidden signature $\hat{g}^{[0]}$, namely,

$$\hat{g}^{[0]}(t) = \frac{1}{N} \sum_{n=1}^{N} g_n^0(t) \\ = \frac{1}{N} \sum_{n=1}^{N} \Big(g_n(\lfloor \tau \rfloor) + (\tau - \lfloor \tau \rfloor) \\ \times \big(g_n(\lfloor \tau + 1 \rfloor) - g_n(\lfloor \tau \rfloor) \big) \Big), \tag{15}$$

where $t = 1, \ldots, M$ and τ is a grid value spaced by $1/(1 - M)$:

$$\tau = 1 + (t - 1) \frac{M_g - 1}{M - 1}, \tag{16}$$

$\lfloor \tau \rfloor$ being a floor-type discrete value of τ.

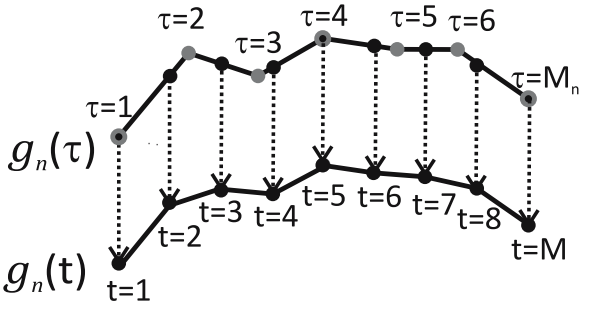

Fig. 5. Linearly resampling into a time length M.

2.2.2. Iteration stage. Next, in each successive iteration $k = 1, 2, \ldots$, the newly computed hidden signature approximation $\hat{g}^{[k-1]}$ is used to calculate

N optimal warping paths $\boldsymbol{w}^{[k]}$ between $\hat{g}^{[k-1]}$and the enrollment signatures (Fig. 6):

$$\hat{\boldsymbol{w}}^{[k]}(\hat{g}^{[k-1]}, g_n) = \arg\min_{\boldsymbol{w}} \mathcal{D}(\hat{g}^{[k-1]}, g_n, \boldsymbol{w}), \quad (17)$$

$n = 1, \ldots, N$. The above warping paths are used to transform the enrollment signatures into a new hidden signature approximation $\hat{g}^{[k]}$ space using DTW:

$$g'_n(t; \hat{\boldsymbol{w}}(\hat{g}^{[k-1]}, g_n)) = \sum_{\ell: w^{\mathrm{t}}(\ell)=t} g_n(\hat{w}^{\tau}(\ell)), \quad (18)$$

$t = 1, \ldots, M$. As a result of this operation, N warped enrollment signatures are obtained, whose lengths are equal to the assumed hidden signature length M. Consequently, a new hidden signature approximation can be calculated as an average of the warped enrollment signatures:

$$\hat{g}^{[k]}(t) = \frac{1}{N} \sum_{n=1}^{N} g'_n(t; \hat{\boldsymbol{w}}^{[k]}(\hat{g}^{[k-1]}, g_n)), \quad (19)$$

$t = 1, \ldots, M$.

This process is repeated iteratively, because a change in the warping paths implies changes in the warped enrollment signatures and then a change in the hidden signature approximation. In each successive iteration, the hidden signatures approximations $\hat{g}^{[k]}$ are created with the use of the optimal warping paths (17), resulting from the optimal misalignment between the hidden signature and the enrollment signatures. As a result, in each successive iteration, the new hidden signature approximation minimizes the average misalignment between itself and the enrollment signatures. The iteration process is repeated until the difference between successive hidden signatures is lower than a predefined threshold ε:

$$\mid \hat{g}^{[k]} - \hat{g}^{[k-1]} \mid < \varepsilon. \quad (20)$$

A visualization of this modified iterative point-by-point averaging is presented in Fig. 6. According to the presented description of the method, our algorithm works as follows.

2.3. Genetic algorithm (GA). Genetic algorithms are routinely used to generate useful solutions to nonlinear optimization problems, thus they fit the nonlinear minimization problem of finding the hidden signature (Davis, 1991).

In a genetic algorithm, a population of strings evolves toward better solutions. Traditionally, solutions are represented as binary strings (arrays of 0s and 1s). However, different encodings are also possible; here, the signatures are represented by arrays of floating point values.

Algorithm 1. Iterative point-by-point averaging (IPPA).

Step 1. Choose initial signatures $\mathbf{O_E} = \{g_1, \ldots, g_N\}$

Step 2. Calculate the initial hidden signature approximation $\hat{g}^{[0]}$ according to (15)

Step 3. Repeat:

(a) Calculate the k-th set of warping paths $\hat{\boldsymbol{w}}^{[k]}$ according to (17) with the use of the k-th hidden signature approximations

(b) Calculate the k-th hidden signature approximation $\hat{g}^{[k]}$ according to (19) with use of set of warping paths $\hat{\boldsymbol{w}}^{[k]}$

until a stopping condition is reached.

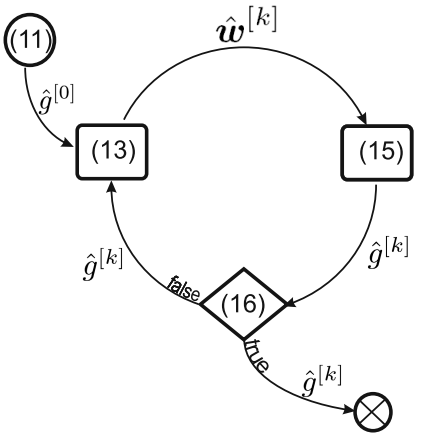

Fig. 6. Calculation process starts the initial calculation (15), the iteration process starts for $k = 1$. Each successive iteration (13) and (15) ends with the stopping condition (20). If it is not satisfied, the iteration is repeated.

2.3.1. Initialization and selection. We assume that the evolution starts from an initial signatures population $\mathbf{O}_E = \{g_1, \ldots, g_N\}$, i.e., enrollment signatures. In each iteration, the fitness of every signature in the current population is evaluated according to the *fitness function*:

$$f(g_j) = \Big(\sum_{k=1,\ldots,N} \hat{\mathcal{D}}(g_k, g_j) \Big)^{-1}. \quad (21)$$

We decide that parent signatures are randomly selected through a *roulette wheel selection*. This is a way of choosing members from the population of signatures in a way that is proportional to their fitness.

2.3.2. Reproduction. Usually, the next step is to generate a new population (*offspring*) of solutions from those selected through genetic operators: crossover and/or mutation. Here, we decide to use only the crossover operation to calculate the new offspring population of

signatures $O = \{o_1, \ldots, o_N\}$. This decision was motivated by the assumption that the initial population already contains signatures that are close to the sought optimum. Therefore, expanding the search space of the algorithms is not necessary.

We propose a crossover operation that is specially designed by us for on-line signatures. This crossover operation deals with the different lengths of signature instances, which makes direct alignment of points impossible. Our idea is to "morph" between two "parent"

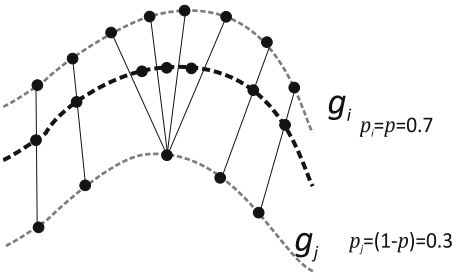

Fig. 7. Crossover operation depending on a weight p.

signatures, with the use of a random weight p (see Fig. 7). For signatures of equal lengths, the process is straightforward. For signatures of unequal lengths, we use DTW to assign every point in signature g_i to one or more points of signature g_j and vice versa. In the case of a one-to-one assignment, the resulting k-th point of an offspring signature o is simply a weighted average of the two input points:

$$o(k) = p \cdot g_i\big(\hat{w}^{\mathrm{t}}(\ell)\big) + (1-p) \cdot g_j\big(\hat{w}^{\tau}(\ell)\big). \tag{22}$$

In other cases, i.e., A points of a signature g_j are assigned to one point of a signature g_i, we propose that the number of points C in the "offspring" signature be calculated in proportion to the weight p:

$$C = \lceil p \cdot A \rceil. \tag{23}$$

Then, we calculate each of the resulting C points of the offspring signature o as a weighted average of one point of a signature g_i and the assigned points of signature g_j, with an additional weighting by a weighting vector $\mathbf{b}_{c,\ell}$:

$$\begin{aligned} &o(k+c) \\ &= p\, g_i(t) + (1-p) \sum_{\ell:\boldsymbol{w}^{\mathrm{t}}(\ell)=t} \Big(\mathbf{b}_{c,\ell}\, g_j\big(\hat{w}^{\tau}(\ell)\big)\Big), \\ &\quad c = 0, \ldots, C-1. \end{aligned} \tag{24}$$

The weighting vector $\mathbf{b}_{c,\ell}$ depends on the number of the resulting points in the offspring signature and it is different for each resulting point. The weight p can be random *(without coefficient control)*, or dependent on the fitness of the signatures in the population $\rho(g)$ (see (21)) *(with coefficient control)*. We decided that the weights p should be individually determined for each pair of aligned points of two morphed parent signatures.

Using the crossover operation, we obtain N offspring signatures. From the population of N parent and N offspring signatures, we select N best signatures according to the fitness function. Therefore, the fitness function values in the new most recent population (new hidden signatures approximations) $\hat{\mathbf{G}} = \{\hat{g}_1, \ldots, \hat{g}_N\}$ can only be equal to or better than the values in the previous population. Due to the fitness function being proportional to the average misalignment between the signature and the signatures from the enrollment set, the selection of the best offspring signatures is in fact minimizing the average misalignment (5) in each successive iteration.

Commonly, the algorithm terminates when either a maximum number of iterations or a satisfactory minimization solution has been reached.

According to the presented description of the method, our genetic algorithm works as follows.

Algorithm 2. Genetic algorithm (GA).

Step 1. Choose initial signatures $\mathbf{O}_E = \{g_1, \ldots, g_N\}$

Step 2. Evaluate the fitness of each individual in the population $f(g_j), \quad j = 1, \ldots, N$

Step 3. Repeat:

(a) Select the individuals to reproduce from using the roulette wheel selection $\mathbf{P}^i$

(b) Breed new offspring signatures through crossover $O^{[i]} = \left\{o_1^{[i]}, \ldots, o_N^{[i]}\right\}$

(c) Evaluate the fitness values of the offspring $f(o_j^{[i]}), \ j = 1, \ldots, N$

(d) Select the new most recent population $\hat{\mathbf{G}}^{[i]}$ containing the best N signatures from the current offspring $O^{[i]}$ and the previous population $\hat{\mathbf{G}}^{[i-1]}$, according to their fitness function f

until stop condition is reached.

In this scheme, two different cases are considered for the reproducing group:

1. the most recent population: $\mathbf{P}^i = \left\{\hat{\mathbf{G}}^{[i-1]}\right\}$,
2. the most recent and the initial population: $\mathbf{P}^i = \left\{\hat{\mathbf{G}}^{[i-1]} \cup \mathbf{O}_E\right\}$.

The first one consists exclusively of the offspring; the

second case additionally includes the enrollment set.

Because the algorithm allows modifications, we may select four versions *(v1–v4)* of the presented algorithm (Table 1).

Table 1. Versions of the genetic algorithm.

	Most recent and initial population	Most recent population
Without coefficient control	GAv1	GAv2
With coefficient control	GAv3	GAv4

2.4. Testing methodology. Verification errors are parameters that describe the capabilities of algorithms. In biometrics, a typical null hypothesis is tested: *"the sample and the template originate from the same biometric specimen"*. If the true null hypothesis is rejected, a false rejection is registered. The proportion of verification transactions with false rejections is the *false rejection rate* (FRR). In opposite situation, when the false null hypothesis is accepted, a false acceptance is registered, giving in a finally the *false acceptance rate* (FAR).

The presented errors describe an access control system. Additionally, it is possible to describe an *equal error rate* (EER) that indicates that the proportion of false acceptances is equal to that of false rejections.

We proposed testing the quality of a given biometric verification system which involves a two-dimensional division of the data set. For a given user, the biometric samples used for creating the biometric template are called the *enrollment set*. The remaining biometric samples used for comparison are the *comparison set*.

Biometric security system parameters are set during the *estimation phase*, processed on one part of a biometric data set (*estimation set*), containing data for a group of users. Typically, one of the important method parameters which is set during the estimation phase is a verification threshold level. Then, during the *testing phase*, the whole system is tested on the remaining part of the data set (*testing set*). As a result of this phase, the FAR and FRR errors are obtained. The proposed division of a database enables us to determine the testing procedure with the EER induced threshold.

In this method, the threshold is adjusted in the estimation phase and it is set to the value that corresponds to the EER (Fig. 8):

$$\theta : \mathrm{FRR}(\theta) = \mathrm{FAR}(\theta) \stackrel{\mathrm{def}}{=} \mathrm{EER} \tag{25}$$

In situations where the verification method depends on a parameter vector $\boldsymbol{v}$, the EER is a function of vector $\boldsymbol{v}$,

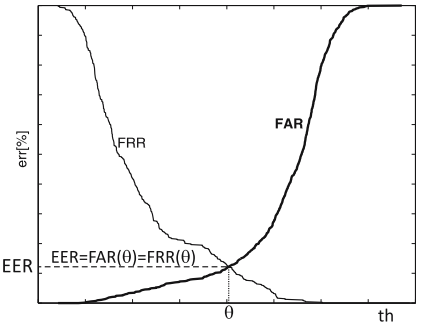

Fig. 8. Estimation phase: FAR and FRR curves calculated for the estimation sets.

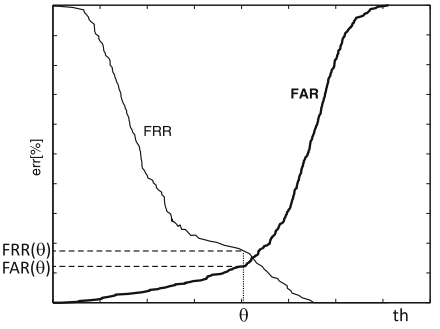

Fig. 9. Testing phase: FAR and FRR curves calculated for the testing sets.

namely,

$$\theta_{\boldsymbol{v}} : \mathrm{FRR}(\theta_{\boldsymbol{v}}, \boldsymbol{v}) = \mathrm{FAR}(\theta_{\boldsymbol{v}}, \boldsymbol{v}) \stackrel{\mathrm{def}}{=} \mathrm{EER}(\boldsymbol{v}) \tag{26}$$

Note that, formally, $\theta_{\boldsymbol{v}}$ becomes a random variable but its value is determined on a data set independent of the set used in testing. In this case the parameter vector is set to the value corresponding to the minimum value of the EER and θ is set to the respective value of $\theta_{\boldsymbol{v}}$, namely,

$$\begin{aligned} \hat{\boldsymbol{v}} &= \arg\min_{\boldsymbol{v}} EER(\boldsymbol{v}), \\ \theta &= \theta_{\hat{v}}. \end{aligned} \tag{27}$$

To check the generalization ability of the verification method, the threshold calculated in the estimation phase is used in the testing phase. Thus, as a result of the testing phase, the FAR and FRR errors are obtained (Fig. 9).

All the on-line verification algorithms presented in this article were tested on the MCYT on-line database, which is a part of the MCYT multi-modal database (Ortega-Garcia *et al*., 2003). It contains data belonging to 100 persons, with 25 genuine signatures and 25 forgeries per user. For tests, the original database was divided into estimation and tasting data. The estimation data used for the estimation phase included data for 40 persons, whereas the remaining data for 60 persons were used as the testing data. Additionally, 1000 of different divisions

of the database was used. These 1000 randomly chosen divisions were the same with every experiment. For each experiment, the ERR, FAR and FRR errors are stored. The algorithm errors are calculated as average values and standard deviations of the stored ERR, FAR and FRR errors. For each user, his/her enrollment set $\mathbf{O}_E$ contains the first 10 genuine signatures from the database. It is a simulation of a real-world situation. As the database signatures are ordered chronologically, these signatures were collected before the remaining ones and, in consequence, this set can be treated as a set collected during the enrollment process.

2.5. Comparison between the methods for hidden signature estimation. In the previous sections, we presented three methods for hidden signature estimation (Fig. 2): one based on a genetic algorithm and two employing iterative point-by-point averaging.

These methods differ in their approach to find the hidden signature approximation. The genetic algorithm increases the number of genuine signatures, by creating artificial signatures only with the use of enrollment signatures. Then, from the large group of genuine signatures, the hidden one is selected as that which minimizes the average misalignment score between itself and the enrollment signatures.

In contrast to the genetic algorithm approach, iterative point-by-point averaging narrows the search to the average signatures obtained from the enrollment set. The algorithm is designed to find the hidden signature time space, in which the average signature minimizes the average misalignment score between itself and the enrollment signatures.

The comparison between the three methods of hidden signature estimation touches upon several aspects, like numerical comparison with the average misalignment with the use of the verification system or creation time.

2.5.1. Numerical comparison. We used the average misalignment (5) to compare numerically the four versions of the genetic algorithm, GAv1, GAv2, GAv3, GAv4, and the modified iterative point-by-point method, MIPPA.

First we evaluated four versions of the genetic algorithm in order to select their best versions. The differences in the obtained average misalignment were not significant, yet the GAv4 version achieves the minimum score in the lowest number of iterations, compared with the other versions of the genetic algorithm, and for those reasons we selected GAv4 for further comparison between the methods.

As seen in Fig. 10, for a user from the MCYT database the best average misalignment was obtained with MIPPA. It can be noticed that the biggest difference

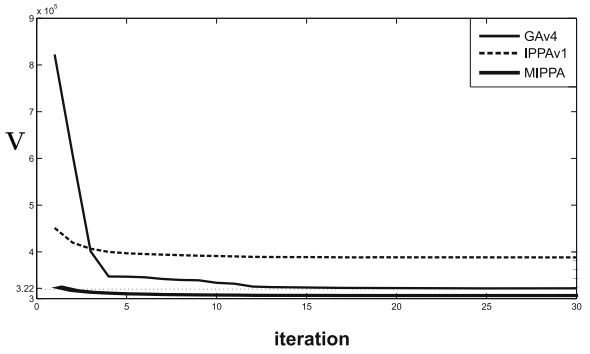

Fig. 10. Average misalignment obtained for three methods: the GA, IPPA and MIPPA, for a user from the MCYT database.

between the average misalignment in the first and last iteration was observed for the genetic algorithm. The differences in the resulting levels of the average misalignment between the methods depend on a user.

Note that, for different users, the average misalignment scales are different, because they are not normalized between the users. What is common for each of the presented methods is that the average misalignment converges asymptotically. A visualization on how the hidden signature approximation changes along the iteration is presented in Fig. 11 (MIPPA method).

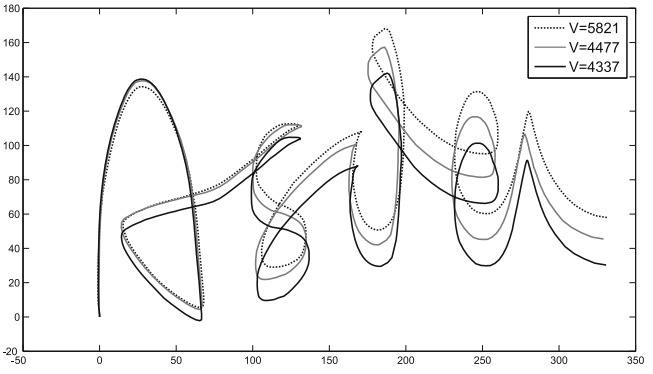

Fig. 11. Hidden signature approximation along the iterations.

The MIPPA method, according to its deterministic character, always approaches the same value of the minimum average misalignment. The genetic algorithm, due to its probabilistic nature, can approach different values of $\mathbf{V}$ for the same enrollment signatures (Fig. 12). The resulting hidden signatures are visually similar and have similar average misalignment values (Fig. 13).

It should be noted that the hidden signature, as an abstract representation of a signature, is unique for a given person. As each of the methods estimates the hidden

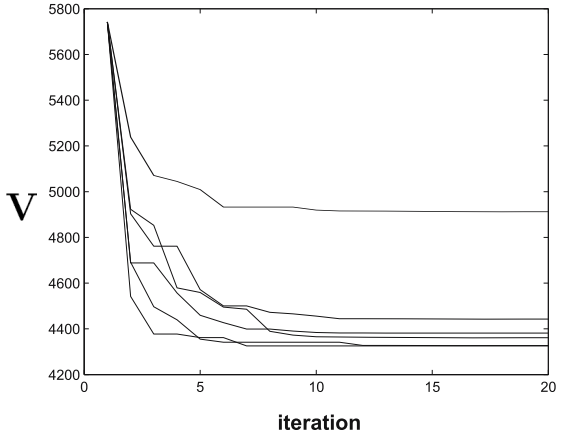

Fig. 12. Average misalignment obtained for GA4 for the same enrollment set.

signature according to the hidden signature definition (2), the resulting hidden signature approximations can be regarded as the hidden signature in the sense of each method: the GA and MIPPA.

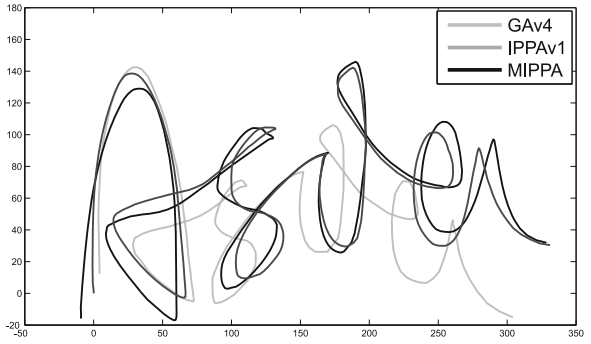

Fig. 13. Hidden signature approximation obtained with GAv4, IPPA and MIPPA.

2.5.2. Verification comparison. The *basic DTW* verification procedure presented in Section 2.1 was used to distinguish between the selected methods (GAv4, MIPPA) for hidden signature estimation. Additionally, we wanted to compare the verification results obtained with the template formed from a hidden signature and the "best" enrollment signature. The *basic DTW* verification procedure was used together with the universal forgery idea which forms the final version of the algorithm. The universal forgery idea was presented by Putz-Leszczynska and Pacut (2013).

During the experiment, for each user, three different templates were created from the same set of the enrollment signatures:

1. "best" enrollment signature,
2. GAv4 hidden signature,
3. MIPPA hidden signature.

The algorithm was tested on the MCYT on-line database and only the estimation EER errors were calculated for the set of 40 users from the MCYT database.

Table 2. ERR for different signatures used as the template for same algorithm. The results were obtained for the 40 users.

Template	EER[%]
1."best" enrollment signature	3.4
2. hidden signature, GAv4	2.33
3. hidden signature, MIPPA	2.33

The comparison between the calculated errors suggests that the verification algorithm works better when the hidden signature is used as a template (cf. Table 2). It proves that this artificial signature could be very helpful. The results obtained with the use of iterative point-by-point averaging are comparable with those of the genetic algorithm.

2.5.3. Conclusions. It is important to note that the average misalignment obtained for the best genetic algorithm GA and the MIPPA methods is similar and generally depend on the user. However, MIPPA converges to its minima in a number of iterations which is lower than for the genetic algorithm. What is more, for the same enrollment signatures, MIPPA always estimates the same hidden signature, while the genetic algorithm, due to its heuristic nature, does not always converge to the same solution (Fig. 12).

The suggestion that the differences in average misalignment (for some users) between the methods are insignificant is confirmed by the fact that EER values are at the same level (see Table 2) for all the hidden signature estimation methods.

The modified iterative point-by-point averaging method, MIPPA, is the fastest, with the iteration time an order of magnitude shorter than in the case of the other methods. For a signature with the enrollment signatures of a length of around $M_n = 100$, the hidden signature estimation with MIPPA method took 6 seconds, while the genetic algorithm GA needed 251 seconds. For commercial purposes we optimized the MIPPA algorithm and the acceptable 6 seconds were reduced to 1 second. Taking into account the computation time, ERRs, and average misalignment, the modified version of iterative point-by-point averaging seems to be the best solution for obtaining the hidden signature because of its good equal error rate and short calculation time.

3. Error time series used in DTW

In Section 2.5, we showed that the use of the hidden signature improves on-line signature verification based on DTW. Here, we extend our approach to exploit the properties of the hidden signature by using its sample standard deviation. In this way the hidden signature properties were use to standardize the verified signature, thus obtaining an error time series which can be used for verification.

3.1. Error time series formation. We proposed division of the MCYT on-line database (Ortega-Garcia *et al.*, 2003) into three sets. Namely, for each of the 100 users, we divided the data into the enrollment set $\mathbf{O}_T$, and two testing sets $\mathbf{O}_G$ $\mathbf{O}_F$:

- $\mathbf{O}_T$: the enrollment set (1000 genuine signatures)—contains the first 10 genuine signatures from the database for each user,
- $\mathbf{O}_G$: 1500 genuine signatures—contains the remaining 15 genuine signatures from the database,
- $\mathbf{O}_F$: 2500 skilled forgeries—contains 25 skilled forgeries.

Additionally, for each user, we store the hidden signature $\hat{g}$, estimated with the use of the enrollment set $\mathbf{O}_T$.

For the purpose of error signal formation, we will regard the signature instances as realizations of a stochastic process, and will estimate certain parameters of this process. In particular, we will calculate the covariance matrix of the signature time series. This can be typically done either by sample averaging or by time averaging. Since we have multiple realizations of each signature, we do not have to make any stationarity assumptions and can use sample averaging.

DTW nonlinear transformation of the enrollment signatures into the hidden signature space can be treated as multiple realization of the signature process in the hidden signature time space. Then their sample average

$$\bar{g}(t) = \frac{1}{N}\sum_{n=1}^{N} g'_n(t; \hat{\boldsymbol{w}}(\hat{g}, g_n)) = \hat{g}(t), \qquad (28)$$

$t = 1, \ldots, M$, is the hidden signature. Additionally, the sample covariance matrix for, e.g., $g^{\Delta x,\Delta y,p}$ (Definition 3), is defined as

$$\Sigma_g(t) = \begin{bmatrix} \mathbf{s}^{2\{\Delta x\}}(t) & \mathbf{s}^{\{\Delta x,\Delta y\}}(t) & \mathbf{s}^{\{\Delta x,p\}}(t) \\ \mathbf{s}^{\{\Delta y,\Delta x\}}(t) & \mathbf{s}^{2\{\Delta y\}}(t) & \mathbf{s}^{\{\Delta y,p\}}(t) \\ \mathbf{s}^{\{p,\Delta x\}}(t) & \mathbf{s}^{\{p,\Delta y\}}(t) & \mathbf{s}^{2\{p\}}(t) \end{bmatrix}, \qquad (29)$$

where

$$\mathbf{s}^{\{\Delta x,\Delta y\}}(t) = \frac{1}{N}\sum_{n=1}^{N}(g_n^{'\Delta x}(t) - \hat{g}^{\Delta x}(t))(g_n^{'\Delta y}(t) - \hat{g}^{\Delta y}(t)). \qquad (30)$$

The sample variances for each coordinate (feature) are the diagonal elements of $\Sigma_g(t)$. For instance, for Δx, we have

$$\mathbf{s}^{2\{\Delta x\}}(t) = \frac{1}{N-1}\sum_{n=1}^{N}(g_n^{'\Delta x}(t) - \hat{g}^{\Delta x}(t))^2, \qquad (31)$$

$t = 1, \ldots, M$, where $g^{\Delta x}(t)$ is defined according to (3).

By $\mathbf{O}'_T$, $\mathbf{O}'_G$, $\mathbf{O}'_F$ we denote the sets containing signatures from the MCYT database after DTW nonlinear transformation into the hidden signature time space (18). Figure 14 presents the signatures Δx time series belonging to a selected user in a time space of his/her hidden signature. The grey shade shows the boundary defined at every point t by $\hat{g}^{\Delta x}(t) \pm 2\mathbf{s}^{\Delta x}(t)$. It is easy to notice that skilled forgeries exceed the boundary "more often" than the genuine signatures. According to this observation, we standardize the warped verified signature

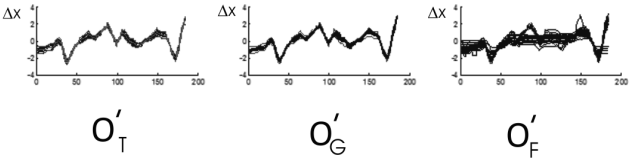

Fig. 14. Signatures in the hidden signature time space, $q^{'\Delta x} = \{\mathbf{O}'_T, \mathbf{O}'_G, \mathbf{O}'_F\}$.

$q^{'}$ (18) for every feature from *list*. This is done by point-by-point subtraction of the hidden signature $\hat{g}$ from the average values, and division by the sample standard deviations $\mathbf{s}$, e.g., for Δx,

$$q^{''\Delta x}(t) = \frac{q^{'\Delta x}(t) - \hat{g}^{\Delta x}(t)}{\mathbf{s}^{\Delta x}(t)}, \qquad (32)$$

$t = 1, \ldots, M$. The resulting standardized sequences $q^{''\Delta x}$, $q^{''\Delta y}$, $q^{''p}$, are called later the *error time series*, because they represent normalized errors between the verified signature and the hidden signature at each point. $\mathbf{O}''_T$, $\mathbf{O}''_G$, $\mathbf{O}''_F$ denote the sets of the error time series (Fig. 15).

A visual inspection reveals that the error time series for genuine signatures and skilled forgeries differ in the amplitude. Based on that, we constructed and tested the error time series approach.

3.2. Signature verification based on error time series: The error time series approach (ETSA). The

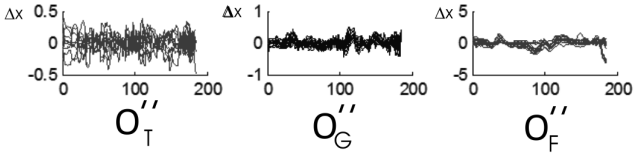

Fig. 15. Error time series, $v^{''\Delta x} = \{\mathbf{O}''_T, \mathbf{O}''_G, \mathbf{O}''_F\}$.

error time series $q^{''}$ is standardized with the use of the hidden signature. Namely, the closer the verified signature q to the hidden signature $\hat{g}$, the closer the values at each point of the error signal to zero.

The goal of the verification is to identify signatures q that are not genuine signatures, i.e., to test the following hypotheses for each verified signature:

H_v: *q is not a hidden signature instance.*

It is well known (Durrett, 2010) that for an independent and identically distributed M-element x normal sample,

$$\frac{x-\bar{x}}{\mathbf{s}_x} \sim \mathbf{t}_{M-1}, \tag{33}$$

where by $\mathbf{t}_{M-1}$ we denote the t-Student distribution. The expected value is equal to

$$E\Big(\frac{x-\bar{x}}{\mathbf{s}_x}\Big) = E\Big|\Big(\frac{x-\bar{x}}{\mathbf{s}_x}\Big)\Big| = 0 \quad \text{for } M > 2, \tag{34}$$

and variance to

$$\mathrm{Var}\Big(\frac{x-\bar{x}}{\mathbf{s}_x}\Big) = \frac{M-1}{M-3} \quad \text{for } M > 3. \tag{35}$$

Without the normality assumption the same properties hold, for a sufficiently large sample. "Sufficiently large" depends on the actual distribution of x and is roughly assumed to be more than 10.

We may then assume that (34), (35) hold approximately for $q''^{\mathbf{c}}$, where $\mathbf{c} \in$ *list* and *list* $= \{\Delta x, \Delta y, p\}$.

Based on (34), we proposed the test statistic of the form

$$T_1^{list}(q'') = \sum_{\mathbf{c}\in list} \big(\overline{q''^{\mathbf{c}}}\big)^2, \tag{36}$$

and based on (35) we proposed another test statistic:

$$T_2^{list}(q'') = \sum_{\mathbf{c}\in list} \mathbf{s}^2(q''^{\mathbf{c}}), \tag{37}$$

where $\overline{q}$ the mean of q and $\mathbf{s}^2(q)$ denotes the sample variance of q .

We use the threshold θ for verification. It is calculated during the estimation phase according to the testing procedure with the EER induced threshold (Section 2.4). We may reject H_v if the selected statistic T_1 or T_2 is greater than θ, e.g., $T_1 > \theta$.

3.2.1. Verification experiments. There are several cases to consider in algorithm construction:

- *Case 1*: Selection between the T_1 and T_2 statistics.
- *Case 2*: Selection between the raw data XY_{raw} and data after standardization XY (standardization before the DTW and error signal formation),
- *Case 3*: Taking into consideration the relative signature length difference γ condition. This is a parametric feature calculated between the corresponding signature lengths (numbers of signature points) M_{r^*}, M_q:

$$\gamma = \frac{|M_{r^*} - M_q|}{M_{r^*}}. \tag{38}$$

- *Case 4*: Including weights in the final form of the test statistic.

3.2.2. Cases 1 and 2. We evaluate both test statistics for verification quality. The results obtained in the estimation process on the whole MCYT database are presented in Table 3. The tests show that the proposed algorithm yields lower verification errors with the use of the T_2 statistic for a majority of cases. It is also clear that it is better to use normalized signatures before template creation and comparison.

The best EER value obtained in the experiment was 7.94%, which means that the results calculated only for $\{\Delta x, \Delta y\}$ produced a lower ERR than the one for a wider set $\{\Delta x, \Delta y, p\}$. To improve this result, we decided to include weights that modify the test statistic.

Table 3. Comparison of the EER (estimating phase) for different test statistics and data normalization.

list	EER[%] for T_1^{list}		EER[%] for T_2^{list}	
	XY_{raw}	XY	XY_{raw}	XY
$\{\Delta x\}$	16.07	20.55	14.27	10.4
$\{\Delta y\}$	19.0	14.93	15.8	8.94
$\{p\}$	19.07	11.47	29.72	12.87
$\{\Delta x, \Delta y\}$	16.07	13.4	13	**7.94**
$\{\Delta x, p\}$	15.92	10.87	27.67	11.8
$\{\Delta y, p\}$	15.92	10.6	28	11.19
$\{\Delta x, \Delta y, p\}$	15.07	10.32	27.05	10.32

3.2.3. Cases 3 and 4. In Cases 3 and 4, we modify the T_2 statistic by including the relative signature length difference γ (38) and the weights

$$T_3(q'') = v^{\gamma}\gamma + \sum_{\mathbf{c}\in\{\Delta x,\Delta y,p\}} v^c \mathbf{s}^2(q''^{\mathbf{c}}), \tag{39}$$

where the weights are the verification parameters $\boldsymbol{v} = [v^{\gamma} \;\; v^{\Delta x} \;\; v^{\Delta y} \;\; v^{p}]$. The test was conducted on the MCYT on-line database. The threshold θ and weights were estimated during the estimation phase according to the testing procedure with the EER induced threshold (Section 2.4). The obtained results,

- estimating phase: $\mathrm{EER} = 1.72 \pm 0.25\%$,
- testing phase: $\mathrm{FRR} = 1.82 \pm 0.86\%$ and $\mathrm{FAR} = 1.74 \pm 0.3\%$,

are very good in comparison with the *basic DTW* approach (see Table 2) and other approaches tested on this database (see Table 4), which generated errors higher than 2%.

The T_3 statistic was selected as the final statistic in the error time series approach. The results obtained in tests on the MCYT database are very promising. Moreover, the overall simplicity allows this method to be used in mobile or embedded systems.

Table 4. On-line systems comparison.

Authors	FAR [%]	FRR [%]	ERR [%]	Database
Quan *et al.* (2006)			7	MCYT
Miguel-Hurtado *et al.* (2007)			8	MCYT
Van *et al.* (2007)			3.37	MCYT
Guru and Prakash (2007)	9.16	5.42	5.3	MCYT
Galbally *et al.* (2007)			3.5	MCYT_330
Faundez-Zanuy (2007)			5.4	MCYT_330
Nanni and Lumini (2008)			5.2	MCYT
Pascual-Gaspar *et al.* (2009)			2.16	MCYT
Integrated system	1.74	1.82	1.72	MCYT

3.3. Optimal length of the hidden signature. The length of a hidden signature estimated by MIPPA method must be given at the beginning of the procedure. This decision may have an impact on verification reliability.

In the experiments described in Section 2.5, the hidden signature length M_u was set to the average length of N enrollment signatures of a given user u:

$$M_u = \mu_u = \frac{1}{N}\sum_{n=1}^{N} M_{u,n}, \tag{40}$$

where $M_{u,n}$ denotes the length of the n-th enrollment signature of a user u. This approach to hidden signature length estimation will be referred to as *user-dependent*. In the MCYT database, values of μ_u change from 50 to 900.

Apart from the user-dependent approach, another possibility is to set a single hidden signature length for all users (*user-independent* approach):

$$\bigwedge_{u \in U} M_u = const. \tag{41}$$

The user-independent length can be, for example, set to the average length of enrollment signatures over the whole database:

$$\mu = \frac{1}{U}\frac{1}{N}\sum_{u=1}^{U}\sum_{n=1}^{N} M_{u,n}. \tag{42}$$

For the MCYT database, this value equals $\mu = 350.65$.

3.3.1. Experiments. Two groups of experiments were performed in order to compare the two approaches. Each case was tested on the same data, and the error time series approach with the T_3 statistic was used for verification testing, according to the testing procedure with the EER induced threshold (Section 2.4).

In the user-dependent group, different user signature lengths were tested. The lengths depended on the average lengths of the users enrollment signatures μ_u (Table 5). The user-independent group of tests was held for the length set according to the database average length μ (Table 6).

Table 5. Results obtained for the user-dependent length of hidden signatures.

M_u	Estimation	Testing	
	EER[%]	FRR[%]	FAR[%]
$1.5\mu_u$	1.77 ± 0.26	1.87 ± 0.9	1.79 ± 0.29
$1.1\mu_u$	1.75 ± 0.25	1.85 ± 0.88	1.78 ± 0.29
μ_u	1.72 ± 0.25	1.82 ± 0.86	1.74 ± 0.30
$0.9\mu_u$	1.67 ± 0.28	1.76 ± 0.85	1.69 ± 0.36
$0.8\mu_u$	1.64 ± 0.28	1.76 ± 0.87	1.67 ± 0.36
$\mathbf{0.7\mu_u}$	$\mathbf{1.61 \pm 0.3}$	$\mathbf{1.73 \pm 0.76}$	$\mathbf{1.64 \pm 0.39}$
$0.6\mu_u$	1.75 ± 0.36	1.83 ± 0.79	1.77 ± 0.46
$0.5\mu_u$	1.75 ± 0.38	1.86 ± 0.79	1.78 ± 0.47

Table 6. Results obtained for the user-independent length of hidden signatures.

M_u	Estimation	Testing	
	EER[%]	FRR[%]	FAR[%]
0.25μ	1.88 ± 0.39	1.94 ± 0.66	1.92 ± 0.55
0.5μ	1.69 ± 0.33	1.81 ± 0.86	1.73 ± 0.45
μ	$\mathbf{1.64 \pm 0.23}$	$\mathbf{1.76 \pm 0.89}$	$\mathbf{1.66 \pm 0.26}$
1.5μ	1.69 ± 0.22	1.81 ± 0.88	1.71 ± 0.23
2μ	1.73 ± 0.23	1.82 ± 0.86	1.74 ± 0.20

Both groups of experiments showed that increasing the length of the hidden signature does not improve the verification quality. For the user-independent length, the best result was achieved for $M = \mu$, while for the user-dependent length, the lowest errors were obtained for $0.7\mu_u$. In other words, the approximation of the hidden signature is better if all the enrollment signatures are transformed into a hidden signature time space of a length lower than the enrollment signatures $M_u \leq M_{u,n}$ (15). Consequently, some information is lost during

the transformation. Conversely, for $M_u > M_{u,n}$, new points are created in the enrollment signature during the transformation. These new points are created based on the current signature points. Because there is no new information, the transformation into a wider hidden signature space ($M_u > M_{u,n}$) will always result in an approximation worse than the original data.

The errors obtained for the user-dependent approach turned out to be slightly better. This can be easily explained. In the user-dependent case, the length differences between the user signatures and the computed hidden signature will be relatively small, which is not the case in the user-independent case.

The difference in the resulting errors for both cases is still not big enough to justify selection of one approach over the other; however, as the user-dependent approach is easier to implement and database-independent, it was chosen for the estimation of hidden signature length.

3.4. Number of signature realizations required to build the hidden signature. Due to commercial requirements, we needed to know how many enrollment signatures are needed to correctly create the hidden signature. This analysis was done with the use of a POS (point of sale) low-requirement algorithm implementation of the algorithm presented in Section 2.1, employing the hidden signature.

The limitations of the target POS terminal posed problems in the implementation. The main difficulties stem from its low computing power, limited internal memory and the lack of a floating-point arithmetics support. As a result, a special implementation of the library functions was needed. Due to this limitation, the results in Table 7 are a bit worse then the one presented in the previous sections.

Again the experiments were performed according to the testing procedure with the EER induced threshold presented in Section 2.4. The errors presented in Table 7 are the obtained mean values of errors, with related standard deviations. The obtained results differ by up to 1.4%, however, the difference between 8 and 10 is not significant. Testing for a higher number of signatures was not possible with this database due to its size.

Table 7. Experiments for different number of signature used for hidden signature estimation.

Number of signatures	Estimation EER[%]	Testing FRR[%]	Testing FAR[%]
5	4.22 ± 0.53	4.43 ± 1.42	4.29 ± 1.21
6	3.56 ± 0.50	3.70 ± 1.33	3.51 ± 0.98
7	3.57 ± 0.49	3.72 ± 1.29	3.56 ± 0.95
8	2.94 ± 0.45	3.05 ± 1.21	2.90 ± 0.92
9	3.03 ± 0.47	3.14 ± 1.18	2.98 ± 0.88
10	2.92 ± 0.49	3.03 ± 1.22	2.87 ± 0.77

4. Summary and conclusions

There is a great demand for automatic signature verification systems on the commercial market. These systems have a potential to replace the traditional signature verification currently done by clerks. Although the clerks are trained to discriminate between genuine signatures and forgeries, their qualifications are much lower than the ones of forensic handwriting experts, who can identify a forged signature with almost a 100% accuracy (Kam *et al*., 2001). At the same time, they identify almost 7% of genuine signatures as forgeries, which suggests that their approximated ERR = 3.5%. Additionally, some weaknesses of human nature, like tiredness and absent-mindedness, may also have big influence on the clerks' verification ability. For these reasons, automatic signature verification systems can be applied where there is a need for fast and precise signature verification. To be commercially applicable, such systems must satisfy three main conditions:

- the system quality must be better than the one of clerks,
- the system quality must be similar to the one of forensic handwriting experts,
- enrollment and verification must be relatively quick.

Over the last 30 years, the problem of handwritten signature verification has changed a lot. The initial hardware problems disappeared. Nowadays, the market offers a plenty of digitizing tablets, which are able to capture signatures with a high frequency and resolution. Another problem, namely, the lack of available signature data, has been addressed by a number of databases, of which some are publicly available together with signature verification competitions that are being held at least biyearly. Due to the dynamic nature of signature biometrics, the methods employing dynamic information usually perform better. As a result, the majority of approaches used today employ nonparametric features. The two most commonly used solutions are systems based on DTW and HMM. HMM methods need more complex computations to be done, while DTW is much simpler, and after customization it can work really fast.

However, DTW-based methods need a selected set of template signatures. This entails the following problems:

- How many template signatures should be used in the template?
- Which kind of statistics should be used for template signature(s) selection?

The template creation in DTW is seemingly very easy. However, it is limited to selecting a subset of a person's signatures. This is the first problem—we do not know *how many* signatures should be selected for the

template. Secondly, even if we resolve the first problem, we still do not know *which* of the signatures should be selected. It has to be emphasized that this selection will have an impact on the decision if a verified signature is accepted or not.

Although signature verification systems based on DTW have been extensively studied in the past few decades, template signatures selection for DTW has remained relatively unexplored. In this work, we proposed how to solve the issues mentioned. We covered this main direction of our research in the theorem that for every signature one can construct an abstract representation, called the hidden signature, from which every other instance can be derived. This abstract representation can be in computed from a collection of available signature instances.

Our approach to DTW employs the hidden signature idea. This artificial signature can effectively replace the template signature in algorithms employing DTW. We proposed the main directions for hidden signature estimation: the genetic algorithm (Section 2.3) and iterative point-by-point averaging (Sec. 2.2). These two directions differ in their approach to finding the hidden signature. The genetic algorithm increases the number of genuine signatures, by creating artificial signatures only with the use of enrollment signatures. Then, from the large group of genuine signatures, the hidden signature is selected as the one that minimizes the average misalignment score between itself and the enrollment signatures. For this approach, we proposed a custom crossover operator, which allows the crossover operation between two sequences that differ in time.

In contrast to the genetic algorithm approach, iterative point-by-point averaging narrows the research to the average signatures obtained from the enrollment set. The algorithm is designed to find the hidden signature time space, in which the average signature minimizes the average misalignment score between itself and the enrollment signatures.

In Section 2.5, we presented a comparison between the hidden signatures estimated with the proposed methods. For that comparison, we used the average misalignment, together with verification errors (see Table 2) obtained for a DTW-based verification system (Section 2.1). During the tests, the hidden signatures were used as the templates. We showed that the average misalignment obtained for the best genetic algorithm and iterative point-by-point averaging methods is similar and generally depend on the user. Also the verification errors were at the same level. Because (as previously stated) the enrollment has to be fast, we selected the modified iterative point-by-point averaging method, whose iteration time was an order of magnitude shorter than that for the genetic algorithm.

In Section 3 we introduced the error time series approach that exploits the properties of the hidden signature by using its certain statistics. We performed experiments to determine the optimal length of a hidden signature (Section 3.3). It turned out that the best way is to use the user-dependent lengths, which are a little shorter that the average length of enrollment signatures.

The errors calculated with this final system are better than the results of other systems presented in literature and tested on the same MCYT database (Table 4). Additionally, systems and hidden signature quality was also confirmed during the ESRA 2011 competition (BioSecure Signature Evaluation Campaign). To summarize our work in this area, we gathered the four successive proposed on-line verification system variants. We succeeded in achieving a high quality system, which shows low errors with low deviations during the testing.

References

Davis, L. (1991). *Handbook of Genetic Algorithms*, Vol. 115, Van Nostrand Reinhold, New York, NY.

Durrett, R. (2010). *Probability: Theory and Examples*, Cambridge University Press, Cambridge.

Faundez-Zanuy, M. (2007). On-line signature recognition based on VQ-DTW, *Pattern Recognition* **40**(3): 981–992.

Fauziyah, S., Azlina, O., Mardiana, B., Zahariah, A.M. and Haroon, H. (2009). Signature verification system using support vector machine, *MASAUM Journal of Basic and Applied Sciences* **1**(2): 291–294.

Galbally, J., Fierrez, J., Freire, M. and Ortega-Garcia, J. (2007). Feature selection based on genetic algorithms for on-line signature verification, *IEEE Workshop on Automatic Identification Advanced Technologies, Alghero, Italy,* pp. 198–203.

Guru, D. and Prakash, H. (2007). Symbolic representation of on-line signatures, *International Conference on Computational Intelligence and Multimedia Applications 2007, Sivakasi, Tamil Nadu, India*, pp. 313–317.

Herbst, N. and Liu, C. (1977). Automatic signature verification based on accelerometry, *IBM Journal of Research and Development* **21**(3): 245–253.

Hong-Wei, J. and Zhong-Hua, Q. (2005). *Signature Verification Using Wavelet Transform and Support Vector Machine*, Springer, Berlin/Heidelberg.

Impedovo, D. and Pirlo, G. (2008). Automatic signature verification: The state of the art, *IEEE Transactions on Systems, Man, And Cybernetics, Part C: Applications and Reviews* **35**(5): 609–635.

Kam, M., Gummadidala, K., Fielding, G. and Conn, R. (2001). Signature authentication by forensic document examiners, *Journal of Forensic Sciences* **46**(4): 884–888.

Kholmatov, A. and Yanikoglu, B. (2005). Identity authentication using improved online signature verification method, *Pattern Recognition Letters* **26**(15): 2400–2408.

Marinai, S., Gori, M. and Soda, G. (2005). Artificial neural networks for document analysis and recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(1): 23–35.

Miguel-Hurtado, O., Mengibar-Pozo, L., Lorenz, M. and Liu-Jimenez, J. (2007). On-line signature verification by dynamic time warping and Gaussian mixture models, *41st Annual IEEE International Carnahan Conference on Security Technology, Ottawa, Canada,* pp. 23–29.

Muramatsu, D. and Matsumoto, T. (2003). An HMM online signature verifier incorporating signature trajectories, *7th International Conference on Document Analysis and Recognition, Washington, DC, USA*, Vol. 1, pp. 438–442.

Nanni, L. and Lumini, A. (2008). A novel local on-line signature verification system, *Pattern Recognition Letters* **29**(5): 559–568.

Ortega-Garcia, J., Fierrez-Aguilar, J., Simon, D., Gonzalez, J., Faundez-Zanuy, M., Espinosa, V., Satue, A., Hernaez, I., Igarza, J.-J., Vivaracho, C., Escudero, D. and Moro, Q.-I. (2003). MCYT baseline corpus: A bimodal biometric database, *IEE Proceedings: Vision, Image and Signal Processing* **150**(6): 395–401.

Pascual-Gaspar, J.M., Cardenoso-Payo, V. and Vivaracho-Pascual, C.E. (2009). Practical on-line signature verification, *in* M. Tistarelli and M.S. Nixon (Eds.), *Advances in Biometrics*, Lecture Notes in Computer Science, Vol. 5558, Springer, Berlin/Heidelberg, pp. 1180–1189.

Putz-Leszczynska, J. and Pacut, A. (2005). Dynamic time warping in subspaces for on-line signature verification, *Advances in Graphonomics: Proceedings of IGS 2005, Salerno, Italy*, pp. 108–112.

Putz-Leszczynska, J. and Pacut, A. (2009). Model approach to DTW signature verification using error signals, *Advances in Graphonomics: Procedings of IGS 2009, Dijon, France*, pp. 166–169.

Putz-Leszczynska, J. and Pacut, A. (2013). Universal forgery features idea: A solution for user-adjusted threshold in signature verification, *Transactions on Computational Collective Intelligence* **7770**(9): 152–172.

Quan, Z.-H., Huang, D.-S., Xia, X.-L., Lyu, M.R. and Lok, T.-M. (2006). Spectrum analysis based on windows with variable widths for online signature verification, *Proceedings of the International Conference on Pattern Recognition (ICPR '06), Hong Kong, China*, Vol. 2, pp. 1122–1125.

Sakamoto, D., Morita, H., Ohishi, T., Komiya, Y. and Matsumoto, T. (2001). On-line signature verifier incorporating pen position, pen pressure and pen inclination trajectories, *AVBPA 2001, Halmstad, Sweden,* pp. 318–323.

Schmidt, C. and Kraiss, K.-F. (1997). Establishment of personalized templates for automatic signature verification, *Proceedings of the 4th International Conference on Document Analysis and Recognition, Ulm, Germany*, Vol. 1, pp. 263–267.

Van, B., Garcia-Salicetti, S. and Dorizzi, B. (2007). On using the Viterbi path along with HMM likelihood information for online signature verification, *IEEE Transactions on Systems, Man, and Cybernetics, Part B* **37**(5): 1237–1247.

WACOM (2015). `http://www.wacom.com`.

Xiao, X. and Leedham, G. (1999). Signature verification by neural networks with selective attention, *Applied Intelligence* **11**(2): 213–223.

Yoshimura, I. and Yoshimura, M. (1992). On-line signature verification incorporating the direction of pen movement. An experimental examination of the effectiveness, *in* S. Impedovo and J.C. Simon (Eds.), *From Pixels to Features III: Frontiers in Handwriting Recognition*, Elsevier, New York, NY, pp. 353–362.

Joanna Putz-Leszczyńska holds a Ph.D. and an M.Sc. in informatics. Since 1999 she has been with the Warsaw University of Technology, presently as an assistant professor at the Institute of Control and Computation Engineering. From 2003 to 2013 she worked as a research assistant at the Biometric Laboratory of the Research and Academic Computer Network NASK. Currently, she works as a system architect at Asseco Poland. She is interested in biometrics, identification, security and global optimization heuristics.

Received: 11 March 2014
Revised: 3 September 2014
Re-revised: 15 December 2014