Interactive Systems Lab

Carnegie Mellon University
Pittsburgh, PA, USA

Universität Karlsruhe
Germany

On-line Signature Verification

REPORT

Stefan Kreckwitz

Supervisors:
Prof. Dr. rer. nat. Alexander Waibel
Dr. ing. Hermann Hild

Acknowledgements

I would like to thank Alex Waibel for giving me the possibility to do this research project at the Interactive Systems Labs in Pittsburgh. I would also like to thank the company SMI for giving me permission to use the Handwriting Recognition Toolkit, Hermann Hild for his advise and Roald Wolff for helping me with ideas.

Contents

Chapter 1

Introduction

1.1 Biometrics

Along with the growing automation of our modern life, there is an increasing need for reliable identity verification. Currently there are two popular ways for solving this security problem. One is related to something "that you have", such as credit cards, physical keys, etc., and the other depends on "something that you know", such as passwords, Personal Identification Number (PIN), etc. [6]. As a Result of the extensive usage of these methods, people have to keep various cards and remember tens of passwords. Losing a card or forgetting a password may bring users into great trouble. In the meanwhile, banks, telecommunication companies and governments are suffering from losing hundreds of millions of dollars per year due to the breaches of current card or password based security systems [1]. Biometrics based verification systems solve these problems as they rely on "something that you are". They use features such as signature, face profile, fingerprint, voice print, eye retina and hand geometry to identify a person by something that cannot be lost, forgotten or stolen.

1.2 On-line Signature Verification

This work deals with the automatic signature verification which belongs to the biometric based methods. In contrast to a writer identification system where the system must establish a writer's identity by comparing attributes of his handwriting with all the writers enrolled in a reference data base, a verification system decides on the claimed identity of a writer by a one-to-one comparison process.
There are two types of signature verification systems, on-line and off-line, that are differentiated by the data acquisition method. In an off-line system, the signature image is digitized with a scanner or camera after the complete signature has been written on paper. An on-line system acquires the signature trace in real time with a digitizing tablet or an instrumented pen. Since an

on-line system can utilize not only the shape information of the signature but also the dynamic information, it is also called a dynamic verification system, whereas an off-line system is called a static verification system. With special hardware, dynamic verification systems can acquire additional features such as the air movement of the pen, the pen pressure on the paper or the pen inclination angles. All these dynamic features reflect the unique habits of the signer and are extremely hard to observe and imitate. A static verification systems cannot recover these features with good accuracy. Therefore on-line signature verification systems usually show a better performance than off-line systems. Here we focus on identity verification by on-line signature verification (OSV). Such a system has all the advantages of a biometrics based system, is extremely user-friendly and the dynamic features give reason for a possible high security level. Furthermore there are, as we will see in the next section, more and more computers equipped with the necessary hardware. Besides the great opportunities of an on-line signature verification system (OSVS) there are also difficulties we have to solve. The biggest problem is the fact that signature patterns vary very much even those patterns of a same individual. Thus, signature verification is a challenging task in the biometric-based authentication.

1.3 Possible Applications

Already today many package delivery companies use special devices in order to record signatures. Likewise some shops are already equipped with devices which record the signature necessary for the purchase by credit card by means of a graphics tablet. In both cases is the hardware for an on-line signature verification system already available and by the application of such a system large saving potentials possible.
Another large field of possible applications are mobile computers, which use a pressure sensitive screen instead of a keyboard and a mouse. Also in combination with these ever more widespread devices, the OSV is the only biometric procedure, which can be applied without any additional hardware.

1.4 Objective of this Work

In this report the development of an on-line signature verification system is described. This development covers the steps of data acquisition, preprocessing, feature extraction and classification experiments.
The report is organized as follows. Chapter 2 describes the data acquisition, Chapter 3 and 4 deal with the preprocessing respectively the feature extraction. Since the dynamic time warping algorithm is a core technology for almost each of the experiments, Chapter 5 describes the algorithm and its different applications for signature verification. Chapter 6 introduces the evaluation of a signature verification system (SVS) and Chapter 7 exhibits the experiments done. Finally, Chapter 8 summarizes the results and outlines future work.

Chapter 2

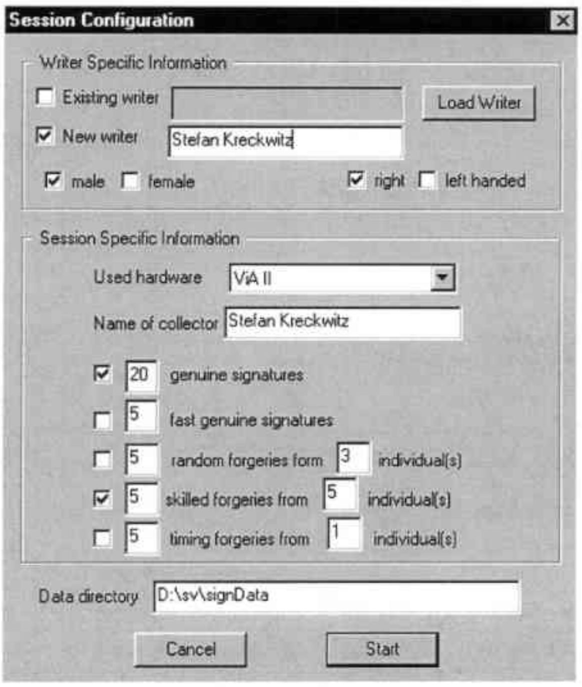
Data Acquisition

2.1 Types of Genuine Signatures and Forgeries

In on-line signature verification one can distinguish between two kinds of genuine signature and three kinds of forgeries [4]. The first type of genuine signatures corresponds to signatures written by the subjects in the way they do it most of the time without any restrictions. The second type are the so-called "fast" signatures where the subjects were told to write their signatures as fast as possible. The major idea behind that is, that customers will occasionally accelerate their writing. Furthermore these fast written signatures are useful to evaluate the robustness of a system with respect to variations in writing speed.
As mentioned, the forgeries can also be divided into three groups: random, skilled and timing forgeries. A forgeries is considered random, when the forger only knows how to spell the name of the subject whose signature he tries to forge. A random forgery is done without any prior knowledge about the appearance of the genuine signature. An imitation attempt is considered skilled, when the forger posses information about the appearance of a genuine signature and goes through a training process which consists of exercising imitations. The last category, the timing forgery, is a skilled forgery where the forgers are provided with additional information about dynamic properties of the genuine signature. In the simplest case this is the average duration of a genuine signature.

2.2 The Data Collection

For the data collection a C++ program was developed which enables the collection of all types of genuine signatures and forgeries. The dynamic information for the timing forgeries are implemented as a repeatable animation of a genuine signature. Figure 2.1 shows the start page of a collection session. If the writer has used the program before, his settings can be loaded or a previous session displayed. Besides some personal information like the sex and hand, the date is saved. Furthermore, every session can be individually configured by the ses-



Session Configuration

Writer Specific Information

Existing writer | Load Writer

[✓ New writer] Stefan Kreckwitz

[✓ male] female | [✓ right] left handed

Session Specific Information

Used hardware: ViA II

Name of collector: Stefan Kreckwitz

[✓] 20 genuine signatures

5 fast genuine signatures

5 random forgeries form 3 individual(s)

[✓] 5 skilled forgeries from 5 individual(s)

5 timing forgeries from 1 individual(s)

Data directory: D:\sv\signData

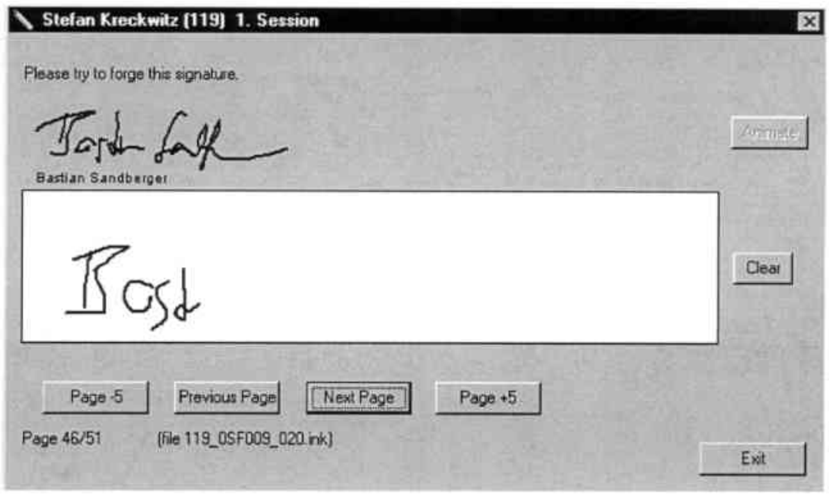
Cancel | Start

Figure 2.1: The start page of the signature collection program.

sion settings. During the collection there is a short instruction and in case of a forgery the necessary information is displayed above the input canvas. The pen trace is displayed on the computer monitor as the signing takes place. Each signature is saved as a sequence of points

$$p_i = (p_{i,x}, p_{i,y}, p_{i,p}, p_{i,t})$$

where $p_{i,x}$ is the horizontal and $p_{i,y}$ the vertical position on the screen in pixel, $p_{i,p}$ the binary pressure value which indicates the pen status ($p_{i,p} = 0$ for a pen up and $p_{i,p} = 1$ for a pen down) and $p_{i,t}$ the system-intern time value in ms.
Using this program, a data base consisting of 795 genuine signatures from a population of 31 subjects and 1295 forgeries written by 32 individuals was built up (see appendix A.1 for details). Some genuine signature were collected with a time delay of several months, which enables experiments that deal with the changing of signature during a long period of time. Figure 2.3 exhibits some examples of genuine signatures and corresponding forgeries.


Stefan Kreckwitz (119) 1. Session

Please try to forge this signature.

Bastian Sandberger

Page 46/51 (file 119_0SF009_020.ink)

Figure 2.2: Within a session: The user is asked to forge a signatures.

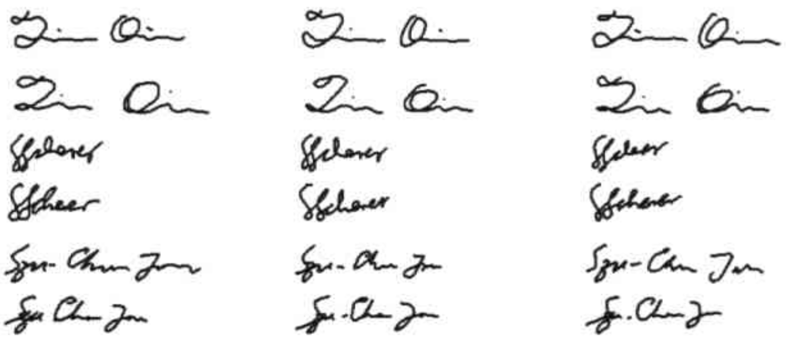
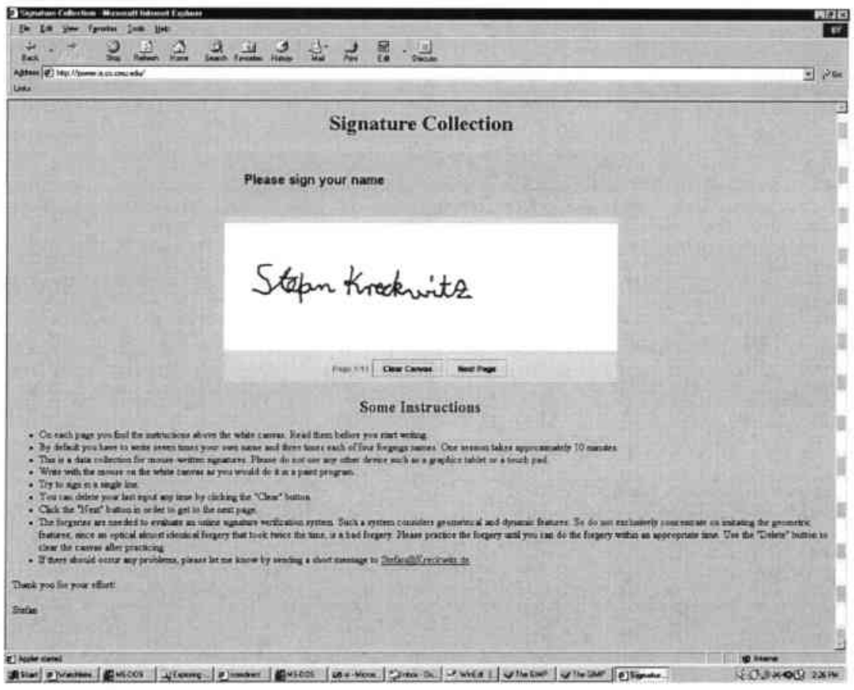


Figure 2.3: An excerpt of the data base. The upper three signatures are originals, the row below exhibits corresponding forgeries.


Signature Collection

Please sign your name

[illegible]

Some Instructions

- On each page you find the instructions above the white canvas. Read them before you start writing.
- By default you have to write seven times your own name and three times each of four forgery names. One session takes approximately 10 minutes.
- This is a data collection for mouse written signatures. Please do not use any other device such as a graphics tablet or a touch pad.
- Write with the mouse on the white canvas as you would do it in a paint program.
- Try to sign in a single line.
- You can delete your last input any time by clicking the "Clear" button.
- Click the "Next" button in order to get to the next page.
- The forgeries are needed to evaluate an online signature verification system. Such a system considers geometrical and dynamic features. So do not exclusively concentrate on imitating the geometric features, since an optical almost identical forgery that took twice the time, is a bad forgery. Please practice the forgery until you can do the forgery within an appropriate time. Use the "Delete" button to clear the canvas after practicing.
- If there should occur any problems, please let me know by sending a short message to Stefan@Krekwitz.de

Thank you for your effort!

Stefan

Figure 2.4: The collection applet in an internet browser. Below the input canvas are several instructions.

2.3 Mouse-written Signatures

The input of an OSVS is a sequence of points which could also be sampled with an ordinary computer mouse instead of special pen-input hardware. Of course, mouse-input looses some natural properties of a human signature. But nevertheless it is interesting to see, whether the developed methods are capable of doing a reliable signature verification on mouse-written signatures. As almost every computer has a mouse, it was obvious to develop a program which allows the users to take part in the data collection through the internet. The resulting program is written in Java and consists of a so-called servlet which organizes the data base on the server and an applet which is the user interface on the client side (see Figure 2.4). This solution makes the program for many people conveniently accessible.
Since it is more time consuming to sign with a mouse, the data collection concentrated merely on genuine signatures and skilled forgeries. The program was kept simple, so loading or individual configuration of sessions is not yet possible. The resulting data base consists of 210 genuine signatures written by 21 individuals and 270 forgeries (see appendix A.2 for details).

Chapter 3

Preprocessing

Although all the procedures used for preprocessing originate from on-line handwriting recognition, there is a substantial difference concerning the target of preprocessing. That is the preprocessing in an OSVS may not eliminate under any circumstances writer specific peculiarities. Thus only such processing steps are executed, which do not eliminate the peculiarities of the subject, but suppress coincidental noise and intra-personal variations. Figure 3.1 exhibits the preprocessing steps described below.

3.1 Normalization

The size of a signature is not a writer dependent habit. For this reason a linear normalization is performed that normalized the height of each signature to 1. After the normalization process the lowest p_y-value is 0 and the highest 1. The p_x-values are normalized accordingly with the same factor. Thus, the most left point has the p_x-value of 0, the highest p_x-value is dependent on the signature. With *miny* as the index of the point with the smallest p_y-value, *maxy* the index of the biggest p_y-value and *minx*, *mint* accordingly the index of the smallest p_x- respectively p_t value, the transformation of a single point p_i^{Orig} of the original point sequence to a point p_i^{Norm} of the normalized point sequence can be formalized as follows:

$$p_{i,x}^{Norm} = \frac{p_{i,x}^{Orig} - p_{minx,x}^{Orig}}{p_{maxy,y}^{Orig} - p_{miny,y}^{Orig}} \tag{3.1}$$

$$p_{i,y}^{Norm} = \frac{p_{i,y}^{Orig} - p_{miny,y}^{Orig}}{p_{maxy,y}^{Orig} - p_{miny,y}^{Orig}} \tag{3.2}$$

$$p_{i,t}^{Norm} = \frac{p_{i,t}^{Orig} - p_{mint,t}^{Orig}}{p_{maxy,y}^{Orig} - p_{miny,y}^{Orig}} \tag{3.3}$$

$$p_{i,p}^{Norm} = p_{i,p}^{Orig} \tag{3.4}$$

3.2 Smoothing

In order to get rid of noise originated form erratic hand motions or inaccuracies of the input device, a smoothing is performed on the normalized data point sequence. The smoothing method averages the position and time of every data point within a smoothing window. The smoothing process is formalized by the following equations:

$$p_{i,x}^{Smooth} = \sum_{j=i-W_{start}}^{i+W_{end}} w_{(i-j)} p_{j,x}^{Norm} \tag{3.5}$$

$$p_{i,p}^{Smooth} = p_{i,p}^{Norm} \tag{3.6}$$

$p_{i,y}^{Smooth}$ and $p_{i,t}^{Smooth}$ are defined accordingly to $p_{i,x}^{Smooth}$. Here W_{start} respectively W_{end} stand for the first and last element of the window function w which are taken into account to get the new values.

3.3 Resampling

The pen-input device samples a sequence of data points which can be either equidistant in time or in space, where the equidistant intervals vary dependent on the used hardware. In order to eliminate this variability, the point sequence is resampled by a linear interpolation algorithm to be equidistant in space. This means that after resampling the Euclidean Distance d between two consecutive points has the same value:

$$\|p_i^{Resamp} - p_{i-1}^{Resamp}\| = \sqrt{(p_{i,x}^{Resamp} - p_{(i-1),x}^{Resamp})^2 + (p_{i,y}^{Resamp} - p_{(i-1),y}^{Resamp})^2} = d \tag{3.7}$$

Hereby is d a fixed value dependent on the normalized height of each signature.

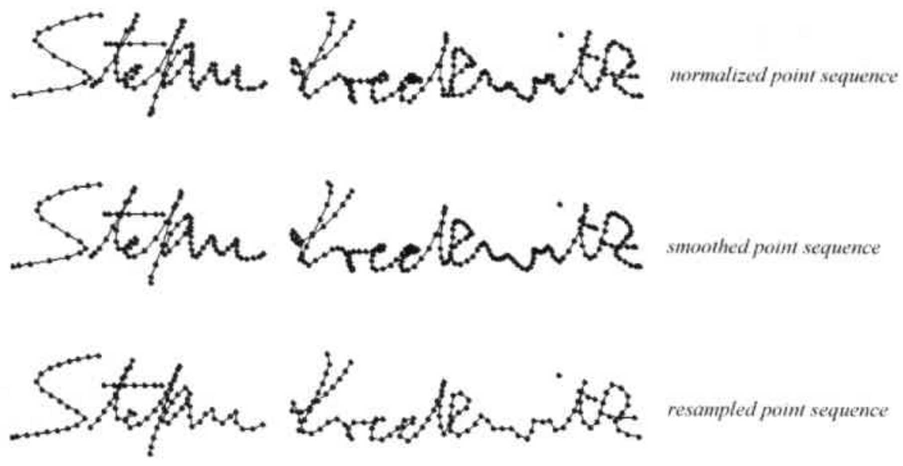


Figure 3.1: Preprocessing steps.

Chapter 4

Feature Extraction

The input of the feature extraction step is the preprocessed data point sequences p_i^{Resamp}. As we will only use this point sequence in further processing steps, we will shorten the notation and write p_i instead of p_i^{Resamp}.
The extracted features should help the system to distinguish between genuine signatures and forgeries. For an on-line signature verification system, these features can be divided into two classes: parameter and function features.
Parameter features are also known as static features which describe the process of writing for a segment within a signature or for the whole signature. Extracted parameters can be the number of strokes, the maximal writing speed, the height-length-ratio of the signature, etc.
Function features regard the signature as a function of time and are for this reason sometimes called dynamic features. Features of this class such as the absolute y-position $F^{abs_y}(i)$ or the horizontal velocity $F^{vel_x}(i)$ are represented as functions where the time is implicitly given by the point-index i.

4.1 Function Features

4.1.1 Relative x-Position

After the preprocessing, the possible x-values depend on the length of a signature. The absolute x values are not useful as features because tight or wide written beginnings of a signature can lead to different x values at the end of the signature. For this reason, the relative x-position to the predecessor point was chosen as a more robust feature for x-positions. We define this feature formally by

$$F^{rel_x}(i) = \begin{cases} 0 & : \quad i = 0 \\ p_{i,x} - p_{(i-1),x} & : \quad i > 0 \end{cases} \tag{4.1}$$

4.1.2 Absolute y-Position

The vertical position of each point is an extremely important feature. This feature needs no extra computation, we can simply use the normalized y-position:

$$F^{abs_y}(i) = p_{i,y} \tag{4.2}$$

4.1.3 Pen-down Feature

The pen-down feature $F^{penDown}(i)$ indicates the pressure of the pen at point i. Likewise as the y-position, we can copy the corresponding value of the data point sequence:

$$F^{penDown}(i) = p_{i,p} \tag{4.3}$$

4.1.4 Direction and Curvature

The following features provide information about the direction and the curvature of the trajectory for each point.
The direction is a translation invariant feature which is determined by a discrete approximation of the first derivatives with respect to the arc length, $\frac{dx}{ds}$ and $\frac{dy}{ds}$, where $ds = \sqrt{dx_2 + dy_2}$. The direction feature consists of two components:

$$F^{\cos\theta}(i) = \frac{\Delta x(i)}{\Delta s(i)} \tag{4.4}$$

$$F^{\sin\theta}(i) = \frac{\Delta y(i)}{\Delta s(i)} \tag{4.5}$$

where

$$\Delta x(i) = p_{i+1,x} - p_{i-1,x}$$
$$\Delta y(i) = p_{i+1,y} - p_{i-1,y}$$
$$\Delta s(i) = \sqrt{\Delta x(i)^2 + \Delta y(i)^2}$$

Figure 4.1 illustrates the definition of the direction feature.

The curvature feature (Figure 4.2) is not only translation, but also rotation invariant. It is defined as the second derivatives $\frac{d^2x}{ds^2}$ and $\frac{d^2y}{ds^2}$ which are approximated by the angle between two elementary segments: $\phi(i) = \theta(i+1) - \theta(i-1)$. This angle is encoded by its cosine and sine. Using the subtraction formulas for sine and cosine these values can be computed as:

$$F^{\cos\phi}(i) = \cos\theta(i+1)\cos\theta(i-1) + \sin\theta(i+1)\sin\theta(i-1) \tag{4.6}$$

$$F^{\sin\phi}(i) = \sin\theta(i+1)\cos\theta(i-1) + \cos\theta(i+1)\sin\theta(i-1) \tag{4.7}$$

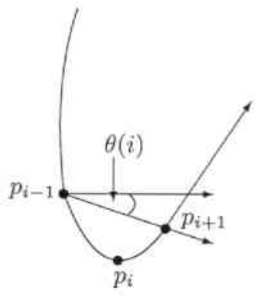


Figure 4.1: Estimation of writing direction

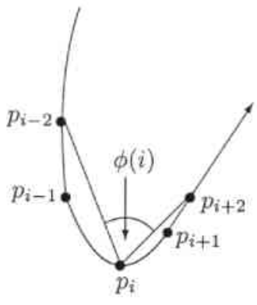


Figure 4.2: Estimation of curvature

4.1.5 Velocity and Acceleration

These two features represent local dynamic properties of a signature. In the literature, two ways for computation of these features can be found. The first way leads to one dimensional features that approximate tangential velocity or acceleration. As there are several good reasons to believe that the dynamic properties measured by this feature can be crucial, we use two dimensional features that measure the horizontal and vertical velocity and speed components separately instead of the one dimensional ones. The implementation of these features is straight forward with the definitions from physics. The horizontal velocity for example is computed as

$$F^{vel_x}(i) = \frac{|p_{(i-1),x} - p_{i,x}| + |p_{i,x} - p_{(i+1),x}|}{p_{(i+1),t} - p_{(i-1),t}} = \frac{\Delta s}{\Delta t} \tag{4.8}$$

$F^{vel_y}(i)$ can be computed accordingly. Acceleration is the derivation of velocity and thus the corresponding horizontal acceleration feature is defined as

$$F^{acc_x}(i) = \frac{F^{vel_x}_{(j-1)} - F^{vel_x}_{(j+1)}}{p_{(i+1),t} - p_{(i-1),t}} = \frac{\Delta v}{\Delta t} \tag{4.9}$$

Here $F^{acc_y}(i)$ is also defined accordingly. In order to get smoother features for velocity and acceleration, a time-window can be used instead of taking merely the neighboring values.

4.2 Parameter Features

We define parameter features for segments which are sequences of consecutive points within a signature. This work makes use of two different segmentations. In 7.1 we will use the whole signature as one single segment and in 7.4 we will use local vertical position extremas as segment boundaries. The following description is independent of the implemented segmentation-technique.
A segment S_i is defined by a sequence of points:

$$S_i = (p_{S_{i,start}}, ..., p_{S_{i,end}}) \tag{4.10}$$

where $S_{i,start}$ is the index of the first point of the sequence inside the preprocessed point sequence and $S_{i,end}$ is the index of the last point. This definition allows every unit from a single point to a whole signature to be one segment. Furthermore the segments can be overlapping and space between two neighboring sequences is also possible. Additionally, we define $S_{i,minx}$ to be the index of the point inside the segment with the smallest p_x value. Accordingly, we define $S_{i,maxx}$, $S_{i,miny}$,$S_{i,maxy}$,$S_{i,mint}$ and $S_{i,maxt}$. To express that a point j is inside the Segment S_i we simply denote $j \in S_i$ which is equal to $S_{i,start} \leq j \leq S_{i,end}$

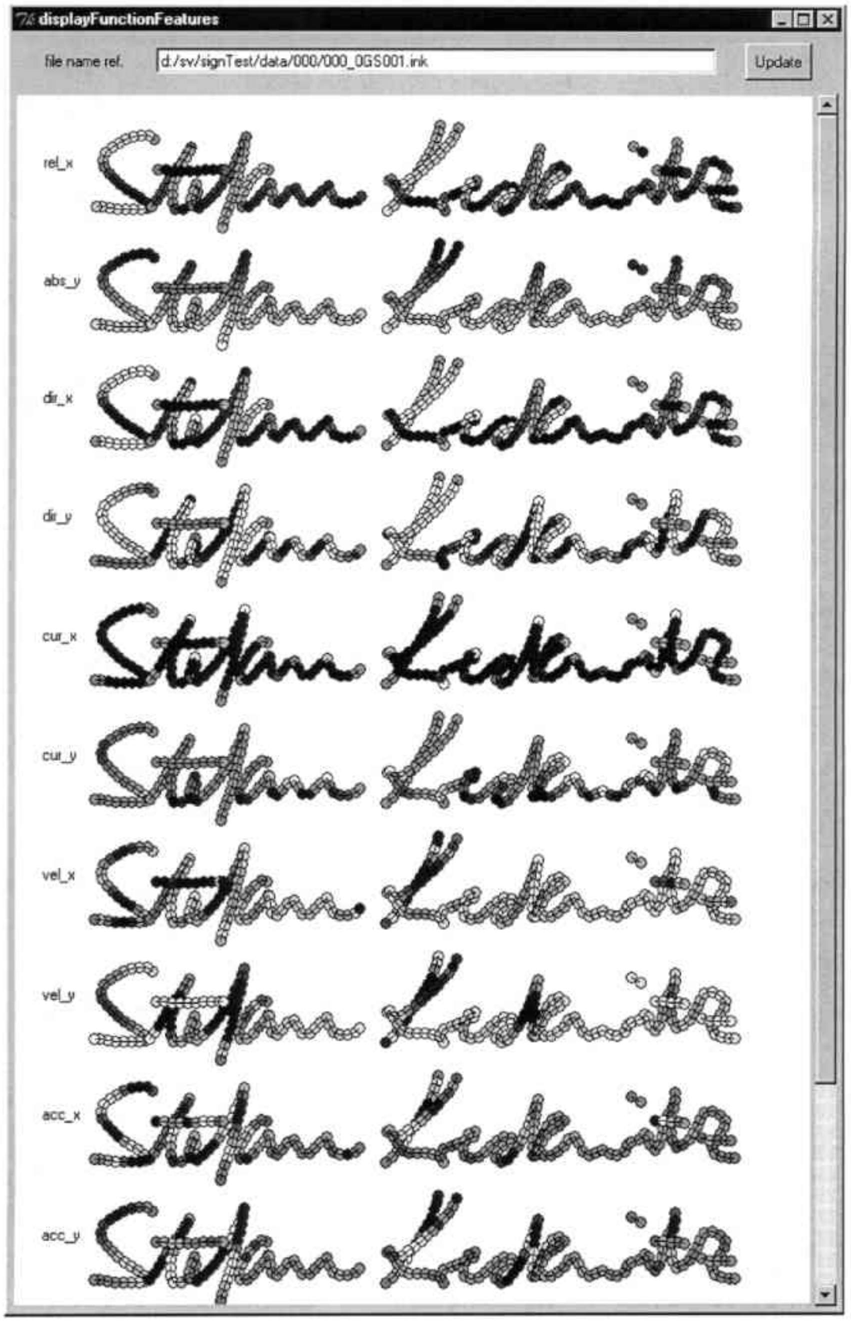


Figure 4.3: Illustration of the computed function features. Here a black point corresponds to a hight features value, a white one to a low value.

4.2.1 Number of Strokes

This feature counts the number of pen-lifts within the segment and can be formalized by

$$P^{StrokeN}(S_i) = |\{j | p_{j,p} = 1, p_{(j+1),p} = 0, j \in S_i\}| \tag{4.11}$$

Hereby $|\{.\}|$ stands for the number of elements in a set.

4.2.2 Width

The width of a segment is simply the maximal horizontal distance between two points of the segment:

$$P^{Width}(S_i) = p_{S_{i,maxx},x} - p_{S_{i,minx},x} \tag{4.12}$$

4.2.3 Aspect

We define the aspect of a segment as the length-width ratio. This feature is undefined if there is no vertical expansion within the segment. Otherwise the feature is given by

$$P^{Aspect}(S_i) = \frac{p_{S_{i,maxx},x} - p_{S_{i,minx},x}}{p_{S_{i,maxy},y} - p_{S_{i,miny},y}} \tag{4.13}$$

4.2.4 Duration Features

There are three durations of interest: The total duration $P^{T_{tot}}(S_i)$ is the time that was needed to write the whole segment. The other durations measure the time the pen was on the paper respectively in the air during the writing process.

$$P^{T_{tot}}(S_i) = p_{S_{i,end},t} - p_{S_{i,start},t} \tag{4.14}$$

$$P^{T_{down}}(S_i) = \sum_{\substack{S_{i,start} \leq j < S_{i,end} \\ p_{j,p} * p_{(j+1),p} = 1}} p_{(j+1),t} - p_{j,t} \tag{4.15}$$

$$P^{T_{up}}(S_i) = \sum_{\substack{S_{i,start} \leq j < S_{i,end} \\ (1-p_{j,p}) * (1-p_{(j+1),p}) = 1}} p_{(j+1),t} - p_{j,t} \tag{4.16}$$

4.2.5 Trace Length Features

Likewise the duration, the trace length can also be specified for the total segment or the part the pen was on the paper respectively in the air. These three trace lengths can be denotated as:

$$P^{L_{tot}}(S_i) = \sum_{S_{i,start} \leq j < S_{i,end}} \|p_{(j+1)} - p_j\| \tag{4.17}$$

$$P^{L_{down}}(S_i) = \sum_{\substack{S_{i,start} \leq j < S_{i,end} \\ p_{j,p} * p_{(j+1),p} = 1}} \|p_{(j+1)} - p_j\| \tag{4.18}$$

$$P^{L_{up}}(S_i) = \sum_{\substack{S_{i,start} \leq j < S_{i,end} \\ (1-p_{j,p}) * (1-p_{(j+1),p}) = 1}} \|p_{(j+1)} - p_j\| \tag{4.19}$$

Where $\|.\|$ is the Euclidian distance between two points.

4.2.6 Pen-up Direction

In most cases only the picture of a signature will be available for a forger. Of course this is not sufficient to find out the correct order of the delayed strokes i.e. i-dots or t-lines. In particular in context of names with several letters which permit delayed strokes, there is a multiplicity of different combinations in which the delayed lines can be made. The pen-up direction feature codes the sequence of horizontal movements between two strokes. This permits testing whether the delayed strokes of two signatures were written in the same sequence. We formally represent the pen-up direction to feature as a sequence of directions. A threshold value controls thereby with which significance one of the two directions B (for backward movement) or F (for forward movement) is present. If this significance is not achieved, then it is a horizontal neutral movement, which is coded with an N.

$$P^{penUpDir}(S_i) = (dir_1, dir_2, ...), dir \in \{B, N, F\} \tag{4.20}$$

4.2.7 Function Feature Extremals and Average

Based on the function feature there are several possibilities to define parameter features. The following features pick out one of the extremals or determine an average value of one feature.

$$P^{min_{F^{feat}}}(S_i) = min\{F^{feat}(j) | j \in S_i\} \tag{4.21}$$

$$P^{max_{F^{feat}}}(S_i) = max\{F^{feat}(j) | j \in S_i\} \tag{4.22}$$

$$P^{avg_{F^{feat}}}(S_i) = \frac{\sum_{j \in S_i} F^{feat}(j)}{S_{i,end} - S_{i,start}} \tag{4.23}$$

Hereby $feat$ is a placeholder for any of the function features described above.

Chapter 5

The DTW-Algorithm

The Dynamic time warping (DTW) algorithm [7] is a well known technique, which had its first application in speech recognition. Since this algorithm was used as one of the core technologies for almost every experiment of this work, this section describes the algorithm, the modifications made for an efficient OSV and the different possibilities the DTW can be used within the verification process.
A major problem of signature verification is the fact that each signature is written with different speed. Unfortunately, a linear time adjustment cannot eliminate this effect, since there are parts in each signature, which are temporally more or less strongly shortened by a faster signature. The DTW-algorithm solves this problem since the algorithm enables a nonlinear time adjustment. This is accomplished by evaluating various permitted pairings between the n-dimensional points of two sequences and selecting the best alignment path through these points based on some optimally criteria and search constraints.

5.1 DTW-Distance

The experiments use different combinations of function features as input. A combination of function features builds a feature vector

$$F(i) = \begin{pmatrix} F^{feat_1}(i) \\ F^{feat_2}(i) \\ \dots \\ F^{feat_n}(i) \end{pmatrix}$$

All the feature vectors of one signature are put together in one pattern

$$F = (F(1), F(2), ..., F(N))$$

where N is the length of the preprocessed data point sequence. Since the DTW compares two patterns, we write

$$F_{Ref} = (F_{Ref}(1), F_{Ref}(2), ..., F_{Ref}(I))$$

for the reference pattern and

$$F_{Test} = (F_{Test}(1), F_{Test}(2), ..., F_{Test}(J))$$

for the test pattern.
As a measure of the difference between two feature vectors $F_{Ref}(i)$ and $F_{Test}(j)$, a distance $d(i,j)$ is employed between them. The definition of this distance is crucial for the system-performance. Performed experiments showed that a quadratic distance with a maximal component distance leads to the best results. Such a maximal component distance prevents that a single component dominates the total distance by extremely large values. Thus, the following distance measurement was implemented:

$$d(i,j) = \sum_{k=1}^{n} \min\{(F_{Ref}^{feat_k}(i) - F_{Test}^{feat_k}(i))^2, maxDist\} \tag{5.1}$$

Hereby $maxDist$ is the upper boundary for the distance a single component can contribute to the total vector-distance.
The cumulative distance $c(i,j)$ between two sequences from the beginning of the signature to point (i,j) is calculated as

$$c(i,j) = min\{c(k,l)|(k,l) \in Pred_{(i,j)}\} + d(i,j) \tag{5.2}$$

Hereby is $Pred_{(i,j)}$ the set of permitted predecessors of point (i,j). The initialization of the above recursion is done by

$$c(1,1) = d(1,1) \tag{5.3}$$

There are a lot of different possibilities for defining the set of predecessors. We will discuss this topic in a later section.
With the cumulative distance the overall distance between two sequences is

$$C(F_{Ref}, F_{Test}) = c(I,J) \tag{5.4}$$

The next section will introduce the alignment path which enables a normalization by the number of accumulated distances.

5.2 DTW-Alignment Path

Sometimes we are not interested in the distance between two patterns, but in the alignment path. The optimal warping path can then be found recursively by starting at point (I,J) and backtracking to the beginning of the signature accordingly to the decision made in each point. To enable such a backtracking, it is necessary to save the selected predecessor of each point.
The path is formally not a function, but a relation, which represents the sequential allocation of feature vectors. If we assume that the path has the length L_P, then the last allocation of the path is

$$Path(L_P) = (I,J) \tag{5.5}$$

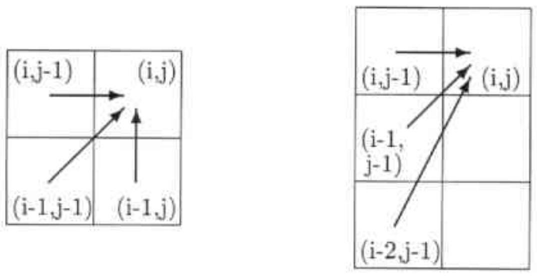


Figure 5.1: DTW transitions. On the left the symmetric DTW, on the right the Bakis model.

Starting at (I, J), the $(i-1)^{th}$ point of the relation is given by the selected predecessor in $Path(i)$:

$$Path(i-1) = argmin_{(k,l)}\{c(k,l)|(k,l) \in Pred_{Path(i)}\} \tag{5.6}$$

The described algorithm guaranties that the first relation $Path(1)$ will be $(1, 1)$. If we would like to refer to one of the two components of $Path(i)$, we write $Path_0(i)$ for the first component and $Path_1(i)$ for the second component:

$$Path(i) = (Path_0(i), Path_1(i))$$

By the length of the alignment path it is now possible to normalize the DTW-distance:

$$D(F_{Ref}, F_{Test}) = \frac{1}{L_P(F_{Ref}, F_{Test})} C(F_{Ref}, F_{Test})$$

Here $L_P(F_{Ref}, F_{Test})$ is the length of the path which aligns the reference and the test pattern.

5.3 Possible Transitions and Restrictions

As mentioned before, there are several possibilities for permitted transitions. In this work the following two approaches were used:
The simple or symmetric DTW

$$Pred^{symm}_{(i,j)} = \{(i-1, j), (i-1, j-1), (i, j-1)\} \tag{5.7}$$

and the Bakis model.

$$Pred^{Bakis}_{(i,j)} = \{(i-2, j-1), (i-1, j-1), (i, j-1) \tag{5.8}$$

Figure 5.1 illustrates both models.

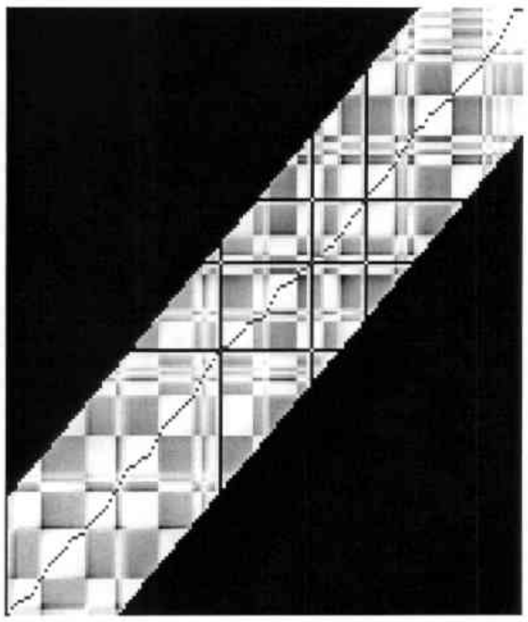


Figure 5.2: A so called DTW matrix. Each point within the rectangle presents a single vector-distance by its gray-value. White corresponds to small and black to big distances. The DTW-path starts in the lower left and ends in the upper right corner. The search window causes the black triangles which constrain the valid area for the alignment path. The three vertical and horizontal lines within the valid area are due to the distances caused by the pen-down feature.

The Bakis model does not allow vertical transitions which prevents degenerated paths with many consecutive horizontal or vertical transitions. However, if one sequence is much shorter than the other, then the Bakis model cannot find an alignment path which ends in (I, J). If the input is a complete signature, this is a good indication of a forgery, but if we use shorter segments as input, relatively big differences in the length of the point sequences are possible. In this cases we will use the symmetric version which leads in every case to a DTW-distance and a corresponding alignment path.
Furthermore it is possible to constrain the permitted transitions by a search window, which allows only those points to be considered which are inside a restricted area. A typical search window is a narrow band along the diagonal of the DTW-matrix. Such a search window saves a lot of computations since the local distances $d(i, j)$ for points outside of the search window do not need to be calculated. It also prevents degenerated alignments. Figure 5.2 and figure 5.3 illustrate the alignment by the DTW-algorithm.

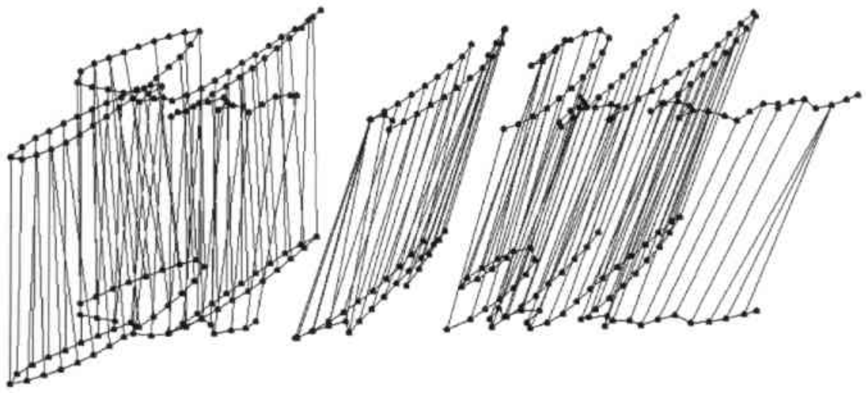


Figure 5.3: The alignment of two signatures corresponding to 5.2.

5.4 Weighting of Feature Components

In order to obtain good verification results with the DTW, it is not sufficient to copy the features as they have been calculated into the feature vectors. Such a procedure would ignore the different quality of the features as a classification criteria. For this reason a weight ω^{feat} is assign to each feature component. This weight controls the contribution to the total distance between two feature vectors. Here optimal weights were determined experimentally by systematical repetition of the test verifications with different weights.

5.5 Main References

In some of the following experiments a main reference signatures for each person was needed. The simplest way to find such a main reference is to chose one by random. This could be sufficient in some cases, but it will sometimes lead to an untypical main reference. The DTW-distance measurement gives us the possibility to find a more reasonable main reference. For this we compute the cumulated distance

$$Cum_j^i = \sum_{k \neq j} D(F_{Ref_j^i}, F_{Ref_k^i}) \tag{5.9}$$

of each reference pattern Ref_j^i from subject i to all the other reference patterns and chose the Reference j^* with the smallest cumulated distance as the main reference:

$$j^* = argmin_j\{Cum_j^i\} \tag{5.10}$$

Chapter 6

Evaluation of a SVS

The performance of a signature verification system is generally evaluated according to the error representation of a two-class pattern recognition problem, i. e. with the type I (FRR: false rejection rate) and type II (FAR: false acceptation rate) error rates. These error rates vary with the acceptance/rejection threshold. The usual representation of the performance of a system is therefore a diagram like figure 6.1, which illustrates the tradeoff between the two error rates. Each point in such a diagram corresponds to one threshold value.
Dependent on the field of application quite different results can be desirable. A credit card company for example might accept a worse FRR, if the FAR is near zero, in order not to bring the customers into unpleasant situations. For an access control system however a very low FAR would be necessary.
Since different systems have a different FAR and FRR curves, the equal error rate (EER) was defined as a possibility for comparing systems. The EER is defined such that it gives the lowest theoretical common error rate possible for a given data set and is determined as the intersection point of the FAR-FRR curve and the function $f(x) = x$. The EER was used in many experiments as the optimization criteria. But this does not necessarily mean that the system with the lowest EER is the best choice for a special application.

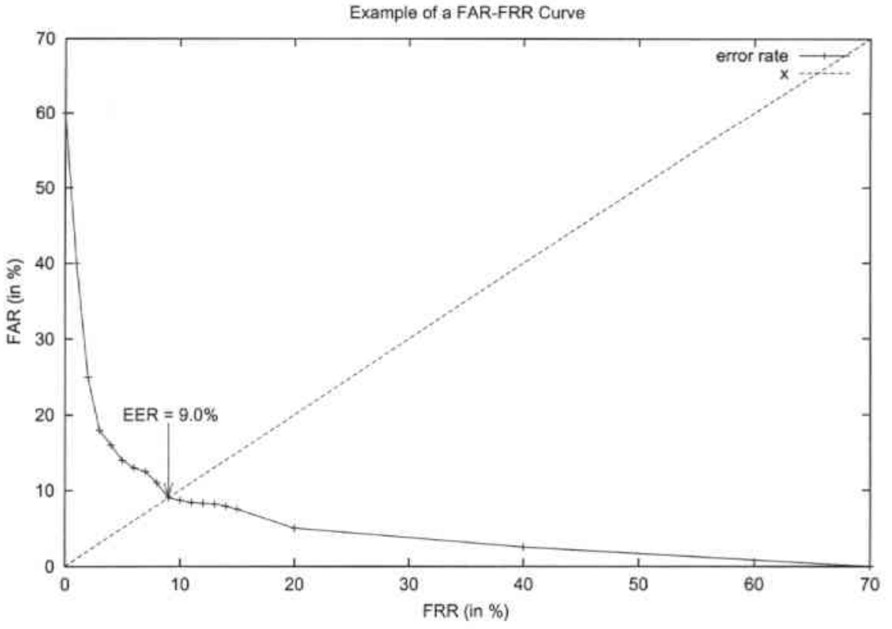


Figure 6.1: An example of a false acceptance - false rejection curve. Here the the intersection point of the FAR-FRR curve and the function $f(x) = x$ determines an equal error rate of 9.0%

Chapter 7

Experiments

The experiments were evaluated on the data-base built up during the project (see A.2 for details). Seven original signatures of each person were used as references. This is half the signatures collected in one session and it corresponds to an effort which every user will readily make. The remaining genuine signatures and the skilled forgeries were exclusively used for test evaluations.
For the time being we assume that N_R is the general number of reference patterns for each signature and that Ref_j^i is the j-th reference pattern of the signature from the individual i. We denote in the following experiment descriptions P^{feat} respectively F^{feat} for any parameter or function feature. $P_{R_j^i}^{feat}$ respectively $F_{R_j^i}^{feat}$ is the j-th feature or parameter pattern of the subject i. $Test^i$ is a test pattern for subject i which can be either a genuine signature or a forgery.

7.1 Verification by Global Parameters

Description of the Verification Process

One of the simplest techniques for a signature verification is to consider a signature as one single segment and to perform the verification based on the parameter features of this large segment. One might think that one cannot place very high expectations against such a procedure, but it will lead, even on a slow computer, fast and resource-saving to results, which might make more complex processing steps unnecessary. Thus, in the following experiment it was tried to detect obvious forgeries by means of global parameters. In other words: It was tried to achieve an FAR as small as possible at a FRR near 0.
Each of the parameter features described above is a candidate for this verification task. However, experiments showed, that it is possible to reduce the number of features without a considerable loss of classification accuracy. Eventually the following feature set was used for this experiment:

$$P^{StrokeN}, P^{Aspect}, P^{T_{tot}}, P^{L_{tot}}, P^{penUpDir}$$

For each of those features, except the pen-up direction feature, the mean and standard deviation can be determined by the following two formulas:

$$\mu_i^{P^{feat}} = \frac{1}{N_R} \sum_{j=1}^{N_R} P_{Ref_j^i}^{feat} \tag{7.1}$$

$$\sigma_i^{P^{feat}} = \sqrt{\frac{\sum_{j=1}^{N_R} (P_{Ref_j^i}^{feat} - \mu_i^{P^{feat}})^2}{N_R - 1}} \tag{7.2}$$

Since every feature has its own characteristic, the threshold for the classification was set for each feature separately to Θ^{feat}. With this definitions we can describe the verification decision of a test pattern $Test^i$ by the feature $feat$ with the following rule:

$$|P_{Test^i}^{feat} - \mu_i^{P^{feat}}| < \Theta^{feat} \sigma_i^{P^{feat}} \tag{7.3}$$

The pen-up feature is a sequence of coded directions, so it is not possible to compute something like a mean or standard deviation. In order to detect crude forgeries by this feature, we determine the well-known edition distance between the feature of the test signature and those of all the reference signatures. The edition distance is a special case of the DTW-algorithm described above and determines here the minimal number of necessary deletions, substitutions and insertions for a mapping of two pen-up direction features. A signature is accepted by this feature, if the smallest distance between the pen-up directions of the test and reference signatures is below a previously defined threshold:

$$\min_j editdist(P_{Test^i}^{penUpDir}, P_{Ref_j^i}^{penUpDir}) < \Theta^{penUpDir} \tag{7.4}$$

A signature is accepted, if each of the five parameter feature indicates a genuine signature.

Experimental Results

The described procedure is extremely easy to implement, resource-saving and it achieves surprisingly good results as one can see in figure 7.1. More than 60% of the forgeries can be detected at a false rejection rate of 0% and with a false rejection rate of 1% even 76.6% of the forgeries can be found. Also the equal error rate of 10.2% is remarkable good for such a simple procedure. A second experiment (see figure B.1 for details) where only the total time was taken into account for the verification revealed the reason for the performance: Most writers tried to imitate the appearance of the signature they had to forge as good as possible and neglected the dynamical aspects. So it is possible to achieve an ERR of 11.2% just by considering the total duration of the signatures. The third experiment with global parameters (see figure B.2) applied the described procedure on random forgeries, which are usually fast written, but with few similarity to the original signature. An equal error rate of 6.5% is good, but only 60% of the obvious forgeries can be detected at an false rejection rate of

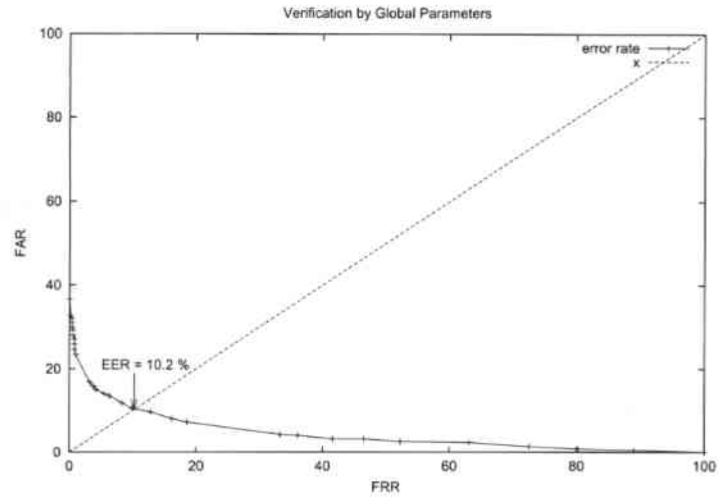


Figure 7.1: Verification by global parameters.

0%.
One can see, the procedure is on the one hand able to discover obvious forgeries, on the other hand it is by means of the global scope able to determine deviations that might not be noticed by a procedure with an exclusively local scope. It supposed itself to use the described technique as a pre-selection for a locally operating procedure in order to profit from both scopes. Corresponding experiments are described below. In each of those experiments the thresholds were set to values that correspond to false rejection rates of 0%. The verification by global parameters was then used as a pre-selection which tried to detect obvious forgeries. Subsequently only those signatures that passed the pre-selection were tested by the more sophisticated procedures.

7.2 Verification by DTW-Distance

Description of the Verification Process

A very widespread way for verifying signatures is the classification by the DTW-distance. After choosing a set of function features, the patterns $F_{Ref_j^i}$ for each reference signature j from subject i can be computed accordingly to the described feature extraction (see chapter 4). Subsequently, an average distance μ_{Ref^i} between two references of each subject is calculated:

$$\mu_{Ref^i} = \frac{\sum_{\substack{j<k \\ k<=N_R}} D(F_{Ref_j^i}, F_{Ref_k^i})}{\sum_{j=1}^{N_R-1} j} \tag{7.5}$$

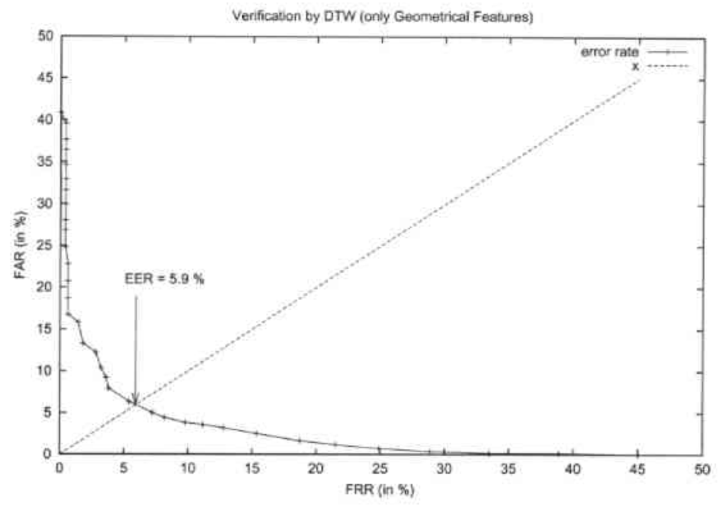


Figure 7.2: DTW applied to geometrical features.

This average distance tells us, in which order of magnitude the distance between genuine signatures can be expected. Subsequently, the resulting values are used to determine the standard deviations as a measurement for the expected deviation of the distance to the average value:

$$\sigma_{Ref^i} = \sqrt{\frac{\sum_{\substack{j<k \\ k<=N_R}} (D(F_{Ref^i_j}, F_{Ref^i_k}) - \mu_{Ref^i})^2}{\sum_{j=2}^{N_R-1} j}} \tag{7.6}$$

By comparing a test pattern F_{Test^i} with a reference pattern $F_{Ref^i_j}$, the classification should assign the test pattern to the genuine signatures, if the following rule is fulfilled:

$$D(F_{Ref^i_j}, F_{Test^i}) - \mu_{Ref^i} < \Theta\sigma_{Ref^i} \tag{7.7}$$

Hereby Θ is a previously defined threshold which controls the trade-off between false acceptance and the false rejection.

There are now different possibilities of combining the single results from the comparisons of the test sample with the references into a total result. It showed up that the best performance can be obtained if the rule has to be fulfilled for one or more reference patterns. This in combination with a very low threshold Θ is superior to the majority decision rule and other consents-finding methods tested.

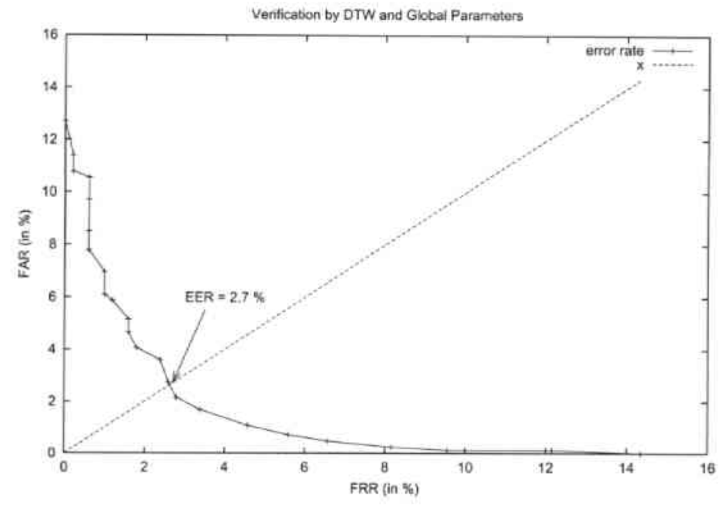


Figure 7.3: Verification by DTW and Global Parameters.

Experimental Results

In the preceding experiment with global parameters it became clear that the dynamic features supply a very substantial contribution to the verification process. For this reason the DTW war first applied exclusively to geometrical features. Here it showed up that satisfying results (see figure 7.2) can be obtained without dynamic features.
If one adds the velocity and acceleration feature, then the procedure comes to an EER of amazingly 2.7% (figure B.3). Together with a pre-selection based on global parameters the EER cannot be further reduced, but the FAR at a FRR of 0% drops form 27, 5% to less than 13%. Figure 7.3 exhibits the performance in detail.

7.3 Verification by Average Patterns

Description of the Verification Process

Instead of comparing a test sample with several reference samples and performing a consent finding afterwards, in this experiment we try to determine an averaged reference pattern. This averaged reference pattern will consist of two vectors per frame: mean vectors and standard deviation vectors. Thus the averaged reference sample shows the expected values and deviations of the feature components for each time frame.
We select the main representative $j^*$of each individual as described in section 5.5. Afterwards we compute an alignment Path $Path^{(j^*,j)}$ for all reference sam-

ples j. Since each path assigns several feature vectors to each feature vector of the main reference sample, a set $\mathcal{F}_i$ of feature vectors result for each point i in the main reference sample:

$$\mathcal{F}_i = \{F_{Ref_j}(Path_1^{(j^*,j)}(k)) | Path_0^{(j^*,j)}(k) = i, j = 1, ..., N_R\} \qquad (7.8)$$

The motivation is, as already mentioned, to determine a "statistical feature vector" in the form of average values and standard deviations. The average pattern will have the same length as the main reference. Now we can compute the mean vectors μ_i by

$$\mu_i = \frac{1}{|\mathcal{F}_i|} \sum_{f \in \mathcal{F}_i} f \qquad (7.9)$$

Subsequently each component k of the standard deviation vector σ_i is calculated by

$$\sigma_{ik} = \sqrt{\frac{\sum_{f \in \mathcal{F}_i} (\mu_{ik} - f_k)^2}{|\mathcal{F}_i| - 1}} \qquad (7.10)$$

The verification of an unknown sample can now done by means of a single DTW distance between the test and the averaged reference sample. The only thing which has to be modified in the DTW distance is the determination of the distance d. We replace d by d' which is calculated as

$$d'(i,j) = \sum_k \left(\frac{1}{\sigma_{ik}} |F_{Test}(j)_k - \mu_{ik}|\right) \qquad (7.11)$$

We can extend this approach by taking an neighborhood into account when we compute the means and variances. This can easily be done by means of a neighborhood-window.

Experimental Results

With an ERR of 5, 2% (see figure 7.4), the described procedure achieves satisfying results. Also here an improvement can be obtained by a combination with the verification by global parameters (figure B.4). Even though the results cannot keep up with those obtained with the DTW approach, it is worthwhile pursuing the approach further. In particular with more than the seven reference patterns used here, the performance should increase clearly. Furthermore no improvement could be obtained so far by different weighting of the feature as well as the introduction of a neighborhood between the sets of feature vectors. Both ideas are still promising.

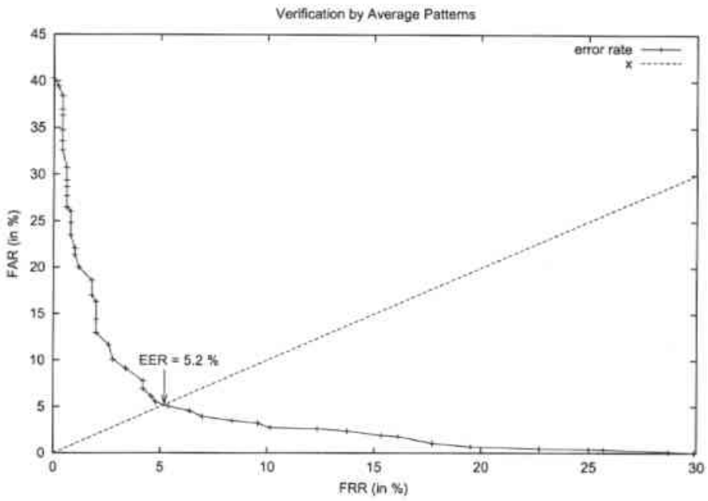


Figure 7.4: Verification by average patterns.

7.4 Verification by Neural Networks on Segment Level

Description of the Verification Process

Segmentation

So far we have compared signatures on two different levels. Either we have compared the features of single points of a signature, or the parameters features of an entire signature. Thus it is obvious to regard a unit between these two extremes. In order to be able to do this, the signatures must be segmented. That means divided into different parts. In the literature, different possibilities for such a segmentation are described. The most common methods take local extremals, local minima or minima of the horizontal or tangential velocity as boundaries between segments. Unfortunately, many procedures described by other scientists have the problem that many errors are caused by different segmentations of two signatures from the same writer. Here another way was chosen which prevents this. We start by selecting the main reference (see 5.5) and segment only this signature by local vertical position extremas. Thus a sequence of N_S^i segments for the main reference j^* of the individual i results:

$$S_{Ref_{j^*}^i}(0), ..., S_{Ref_{j^*}^i}(N_S^i)$$

Now we make use of the good results of the DTW and compute the alignment paths for every reference to the main reference. By the means of these alignment paths we can afterwards transform the segment boundaries of the main reference feature to the other references. As the result we get several examples for each segment.

A Data-driven approach

We will see that there are a lot of features which can be computed for the determined segments. So the question arises how to use all these features for a good verification. There are several attempts to select only those features that are very typical for each individual. Unfortunately, these procedures are not very convincing and neither are the results. Another popular approach is based on rules how to weight the features. However, these rule based approaches have well-known difficulties and usually show a worse performance than data-driven methods. For this reason, a neural network was implemented in order to decide for a segment whether it belongs to a genuine or a forged one.

Organization of test and training set

The neural network makes it necessary to rearrange the allocation of the training and test data. This is necessary because we will train the neural network with genuine signatures and forgeries. Since the reference samples were used for the calculation of input-features (see next section), we need further genuine signatures beside the reference samples and forgeries for the training-process. For this reason the set of individuals for the training and test was divided. In order to get more representative results, the experiments were done by different organizations of test and training sets. One single individual was used in each case for the test and the remaining individuals for training. The organization of the references is preserved as described in the beginning of this chapter.

Input feature vectors

We can use every parameter feature as input feature. The observed parameter features are not sufficient as an input for the neural network, since the net needs some information about the references. For this reason we use the references from each individual i to compute the mean $\mu^{feat}_{S_k^{Ref^i}}$ and standard deviation $\sigma^{feat}_{S_k^{Ref^i}}$ of each feature $feat$ and each segment k. Then we get a normalized distance-feature value for a test pattern by

$$v^{feat}_{S_k^{Test^i}} = \frac{P^{feat}(S_k^{Test^i}) - \mu^{feat}_{S_k^{Ref^i}}}{\sigma^{feat}_{S_k^{Ref^i}}} \tag{7.12}$$

So we get for a test pattern three values for each feature and segment:

$$v^{feat}_{S_k^{Test^i}}, \mu^{feat}_{S_k^{Ref^i}}, \sigma^{feat}_{S_k^{Ref^i}}$$

The DTW-distance gives us a second possibility to describe the deviations between a reference and a test pattern. One can simply use the the DTW-distances to the N_R references to get further components. We define

$$D_{S_k}^{featSet}(F_{Test^i}, F_{Ref_j^i}) \tag{7.13}$$

as the segment DTW-distance for segment k between the test pattern and reference pattern j from individual i based on the feature set $featSet$. By means of the reference patterns we compute the average distance $\mu_{S_k^{Ref^i}}^{DTWdist_{featSet}}$ between two reference patterns and the standard deviation $\sigma_{S_k^{Ref^i}}^{DTWdist_{featSet}}$. Now the DTW-distance can be normalized:

$$\upsilon_{S_k^{Ref_j^i}}^{DTWdist_{featSet}} = \frac{D_{S_k}^{featSet}(F_{Test^i}, F_{Ref_j^i}) - \mu_{S_k^{Ref^i}}^{DTWdist_{featSet}}}{\sigma_{S_k^{Ref^i}}^{DTWdist}} \tag{7.14}$$

All the measured deviations are put together into one large vector δ:

$$\delta(S_k^{Test_i}) = \begin{pmatrix} \upsilon_{S_k^{Test^i}}^{feat_1} \\ \mu_{S_k^{Ref^i}}^{feat_1} \\ \sigma_{S_k^{Ref^i}}^{feat_1} \\ \upsilon_{S_k^{Test^i}}^{feat_2} \\ \mu_{S_k^{Ref^i}}^{feat_2} \\ \sigma_{S_k^{Ref^i}}^{feat_2} \\ \ldots \\ \ldots \\ \upsilon_{S_k^{Ref_1^i}}^{DTWdist_{featSet_1}} \\ \ldots \\ \upsilon_{S_k^{Ref_{N_R}^i}}^{DTWdist_{featSet_1}} \\ \upsilon_{S_k^{Ref_1^i}}^{DTWdist_{featSet_2}} \\ \ldots \\ \upsilon_{S_k^{Ref_{N_R}^i}}^{DTWdist_{featSet_2}} \\ \ldots \\ \ldots \end{pmatrix} \tag{7.15}$$

Dimension reduction

With all the available features, a 100-dimensional feature space is easily possible. This means the neural net will have 100 input units and a several times higher number of transitions into the hidden layer, which is equivalent to a high number of free parameters. After a rule of thumb, one needs at least ten training patterns for each free parameter in a neural net. If we use only ten units in one hidden layer, we would need ten thousand segments to fulfill the minimal requirements. After these thoughts it is obvious to employ one of the well-known dimension reduction techniques. For this experiment, the linear discriminant analysis (LDA) [9] was chosen. This transformation is suitable for dimension reduction and it increases additionally the separability of the two classes. After computing the LDA-matrix LDA with all the available training data, a linear transformation can be done by multiplying the feature-vector δ with the LDA-matrix:

$$\delta_{LDA}(S_k^{Test_i}) = LDA * \delta(S_k^{Test_i}) \tag{7.16}$$

Now a dimension reduction can be done by taking the first N_{inp} vector elements.

The Net Configuration

The neural network architecture used is a time delay neural network (TDNN) [8]. Figure 7.5 exhibits the selected configuration. The input layer has no time delay and a width of N_{inp} input units according to the previous dimension reduction. The single hidden layer has N_{hid} units and also no time delay. The segment-output layer delivers the output in the first training stage. It has two neurons in each frame, one indicated a genuine signature and one a forgery. During the training iterations of the first stage the network is trained to decide whether a segment is part of a genuine signature or a forgery. After several iterations the second training stage starts. In this stage the final output layer decides whether a sequence of consecutive segments belongs to a genuine signature or forgery. This layer has also two neurons with the same meaning as in the segment output layer.

Training and Testing

Training takes place in two stages. In the first stage the segment output layer is trained in such a way that a genuine signature causes the upper neurons of this layer to output a 1 and the lower neurons neurons to output a 0. For a forgery this target-output is inverted.
The second stage of the training process makes use of the time delay. Here the output layer is trained to combine successive results of the segment output layer into a single result. In the test phase the outputs of the upper neurons are summed-up with the negatively weighted outputs of the lower neurons. The resulting value is then divided by the number of summed values and compared afterwards with a threshold value. If the summed value is larger than the threshold, then the neural network detects a genuine signature, otherwise a forgery.

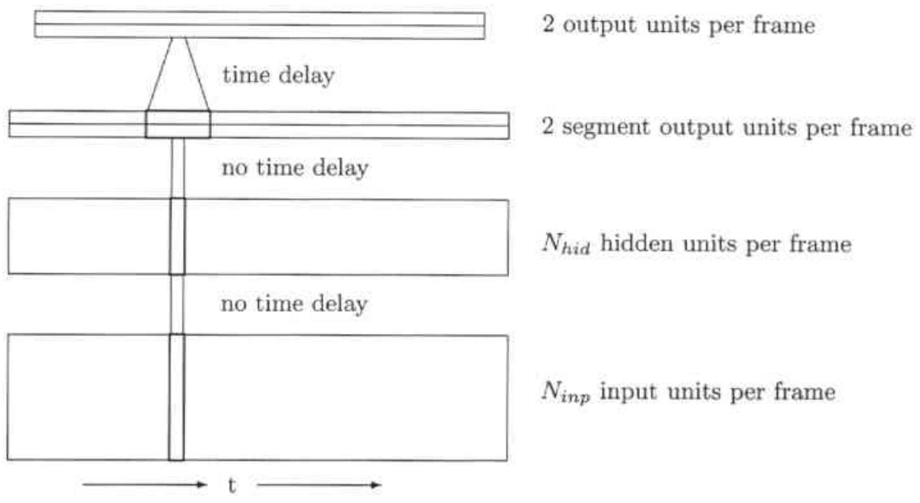

2 output units per frame
time delay
2 segment output units per frame
no time delay
N_{hid} hidden units per frame
no time delay
N_{inp} input units per frame
t

Figure 7.5: The architecture of the implemented TDNN.

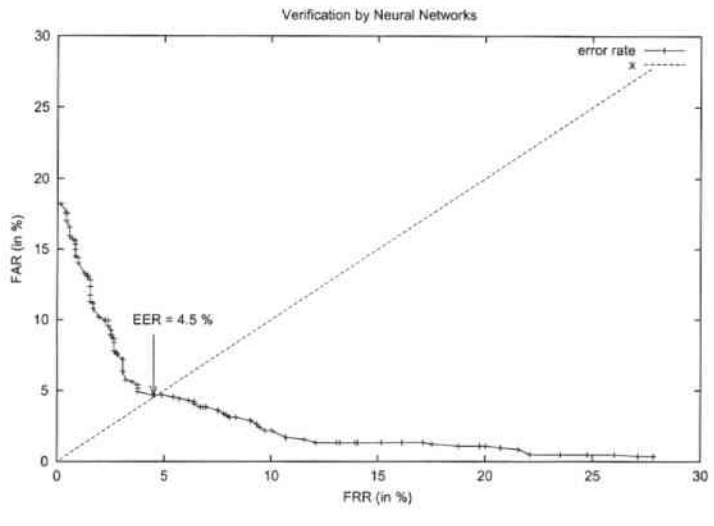


Figure 7.6: Verification by Neural Networks.

Experimental Results

The following features were chosen for this experiment:

$$P^{Width}, P^{T_{tot}}$$

$$P^{min_{Ffeat}}, P^{max_{Ffeat}}, P^{avg_{Ffeat}}, feat \in \{abs_y, cos\theta \sin\theta, cos\phi, sin\phi, vel_x, vel_y\}$$

The DTW-distance was computed only for the feature set consisting of all available function features (see 4.1).
This experiment was executed with the described network architecture with $N_{inp} = 20$ and $N_{hid} = 15$.
With the presented procedure an EER of 4,5% was obtained. Thus, the performance is worse than those of the DTW approach. However, if one regards the false acceptance rate with a false rejection close zero, then the neural network is clearly superior to the DTW.
Also here, improving this procedure seems to be worthwhile. Since all networks were trained with the same number of iterations and the same parameters, some percents of accuracy were lost due to the missing fine tuning. Furthermore a neural network is dependent on the available amount of data. While the DTW approach will not get better just by adding more data, one can assume that the neural network will exceed its past results when more training data is available.

7.5 Verification of Mouse-written Signatures

Experimental Results

Some of the experiments presented above were repeated with the mouse-written signatures. Since there were fewer mouse-written original signatures of each person available, only five instead of seven reference patterns were used. Due to the small number of reference patterns, none of the average pattern experiments were executed.
The first experiment was again the verification by means of global parameters. Compared with the experiment in 7.1. only the thresholds were slightly changed. Figure B.5 shows the results of this experiment. Since it is even difficult to write the own signature fast with a mouse, the total time loses as reliable classification criterion, which impairs the total performance of the verification negatively.
By means of the DTW of procedure already better results are obtained (see Figure B.6), which are again exceeded by inclusion of the global parameters (Figure 7.7).
As good results as the verification of the pen-written signatures are here hardly conceivable. That is above all because the mouse-written signatures of an individual deviate very much from each other. Nevertheless, with a special preprocessing it seems to be possible to improve the results obtained so far. In particular a sophisticated baseline normalization [10] which rotates the written input to a nearly horizontal orientation could bring clear improvements.

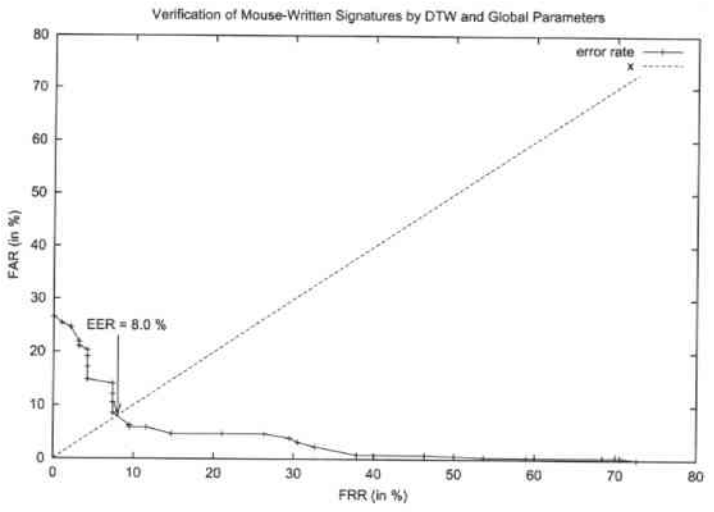


Figure 7.7: Verification of Mouse-Written Signatures by DTW and Global Parameters.

Chapter 8

Summary

8.1 Summary

A data base consisting of almost 800 genuine signature and 1300 forgeries was created. This data base enables almost representative results and an extension of the amount of data is easily possible with the developed programs.
Four basically different approaches to the problem of OSV were described. The approach based on average patterns and the application of neural networks on the segment level are based on new ideas. Each of the four experiments led to encouraging results on the available data base (table 8.1). Thus a further investigation of these approaches seems to be promising. The best results were obtained by the DTW approach in combination with a pre-selection based on global parameters. Here an equal error rate of 2, 7% could be obtained.
The best result (8% EER) on mouse-written signatures was also obtained by the DTW together with a pre-selection based on global parameters. Table 8.2 summarizes the results of the mouse-experiments.

8.2 Related Works

The comparison with results of other scientists is very difficult, since there is no uniform database. Furthermore there is only little information about the data-acquisition process of other researchers available. Besides this some scientists improve their results by mixing random and skilled forgeries or by collecting an extremely high number of signatures from one individual.
The most extensive outline with many literature references can be found in [2] and [3]. Additionally I would like to refer to the following works:
Mario E. Munich and Pietro Perona ([11]) developed a continuous version of the DTW for a translation-invariant curve alignment. The database consisted of 56 persons with 25 signatures each. Ten signatures from each writer were used for the test, 15 as references. As result it was stated that the EER was 0.3% worse by application of the new technique than by the original DTW. The EER of the

Experiment	Figure	EER	FRR 0%	FRR 2%	FRR 5%
total time	B.1	11.2%	45.9%	24.5%	16.0%
global parameters	7.1	10.2%	36.7%	20.0%	11.5%
DTW (geom.)	7.2	5.9%	40.9%	13.0%	6.5%
DTW	B.3	2.7%	27.3%	6.0%	0.9%
DTW & glob. par.	7.3	2.7%	12.7%	3.7%	0.9%
average patterns	7.4	5.4%	40.3%	13.0%	5.3%
avg. pat. & glob. par.	B.4	4.5%	18.7%	7.6%	3.9%
neural network	7.6	4.5%	18.2%	10.2%	4.6%

Table 8.1: A summary of the experimental results. The first column describes the experiment, the second refers to the figure with the detailed results, the third column contains the equal error rate and the remaining columns show the error rates at different false rejection rates.

Experiment	Figure	EER	FRR 0%	FRR 2%	FRR 5%
global parameters	B.5	21.4%	52.3%	50.0%	36.0%
DTW	B.6	13.4%	55.5%	50.8 %	31.0%
DTW & glob. par.	7.7	8.0%	26.6%	24.6%	14.5%

Table 8.2: Summary of the results obtained on mouse-written signatures.

unmodified DTW was 2.6%.
Luan L. Lee and Toby Berger [4] worked with personalized feature sets and a majority-decision-rule that classified by global parameters. The feature-selection took place under consideration of forgeries. This technique yield 2.5% EER and an asymptotic performance of 7% FAR at a zero FRR. The data base constisted of 105 subjects and 5600 genuine signatures (13-1000 per subject).
Seong Hoon Kim, Myoung Soo Park and Jaihie Kim ([14]) also applied personalized feature sets. With 9 subjects with in each case 120 genuine signatures and 120 forgeries they obtained an EER of 4.28%.
Kai Huang and Hong Yan ([12])have chosen the local speed minima as segmentation points. They applied a pre-selection by global parameters and a DTW on segment-features. The scientists indicate a FRR of 5% at 2% FAR as their best result. For the test and training were 394 genuine signatures from 20 people and 466 forgeries available. Nine of the original signatures were used as references.
In [15] Kai Huang and Hong Yan describe a verification by means of a fractal transformation-technique. They obtained a FRR of 2.3% at 2.7% FAR for random forgeries.
Ma Mingming and Sharda Wijesoma ([13]) examined three models: a frequency function model, a shape-related parameter model and a dynamics-related parameter model. They obtained results between 4.62% and 8.96% EER on their data-base consisting of 1230 genuine signatures from 41 persons and 410 skilled

forgery samples generated by two different forgers.
Brigitte Wirtz ([16] and [17]) developed a time- and position-based averaging of represantive input signatures for a stroke-based verification approach. Based on 6000 origingal signatures from 20 writers and 6000 forgeries from 100 forgers an EER of 9.89% was obtained.

8.3 Future Work

This was the first research project at the Interactive Systems Labs which dealt with signature verification. Thus, there are still many questions open and a lot of research to be done. The most important topics for future work can be summarized as follows:

- Data base
  The created data base is still much too small in order to be able to indicate representative results. In particular because it concerns a security issue, a database consisting of signatures from far more than 100 different individuals would be desirable. Additionally high-quality forgeries are missing. Especially the dynamic features were quite neglected. How the results with the global parameters show, many forgeries can be detected merely by the total duration with the help of very simple procedures. Thus, future data collections should increase the acquisition of timing forgeries. Future data collections should also try to collect a larger number signatures from some persons. This would enable investigations with the goal to observe how an increasing number of reference patterns can reduce the error rates.

- Additional features
  Due to the missing hardware not all possible features could be recorded during the data collection. Thus the pressure of the pen-tip on the paper was only binary encoded. Meanwhile there are already instrumented pens which measure this pressure in 256 or more levels. In particular this feature seems to be very promising, what is acknowledged by several papers. Furthermore features like air movement and the inclination angles of the pen have not been recorded yet.

- Adaptation
  A commercial system should be able to adapt to over the time slightly changing signatures. On the one hand additional data is needed which represent modifications of signatures over several months or years. On the other hand the procedures represented in this work have to be adjusted.

- Intelligent Reference Acquisition
  In some cases very untypical signatures result from a lack of hardware or a mistake of the user. These patterns are particularly harmful for the system if they are added to the reference samples. In this case they will

negatively influence the verification process as long as the system operates. For this reason a good system should test whether the digitized samples are consistent. If the system discovers an inconsistency, it should inquire with the user and replace the possible incorrect signature by a new one.
It is evident that some signatures are more easily to forge than others. For this reason an intelligent reference acquisition should approximate the complexity of the signature by a measure which has to be defined and set the number of necessary reference signatures accordingly. By a higher number of reference samples for an easily to imitate signature, there is some good reason to believe that a more precise model of the signature can be determined which compensates the lacking complexity. Whether such a procedure really improves the verification accuracy, has to be examined.

- Verification on a PDA
  As already mentioned in the introduction, the PDAs represent a potential market for OSVS. However, to build an OSVS for PDAs is a challenging task, since the screen is often very small, the digitization is quite crude and PDAs have usually slower processors and less memory than desktop computers. Also here there are special adjustments of the described procedures necessary to build a system which is executable on a PDA.

Appendix A

Data Bases

ID	#genuine signatures	#skilled forgeries
000	10	15
001	10	15
002	10	15
003	10	15
004	10	15
005	10	15
006	10	15
007	10	15
008	10	15
009	10	15
010	10	15
011	10	15
012	10	15
013	10	15
014	10	15
015	10	15
016	10	12
017	10	9
018	10	6
019	10	3
020	10	0
21	210	270

Table A.1: The data base of the mouse-written signatures. One line contains the writer identity number, the number of signatures made by this writer and the number of available forgeries for the signature from the corresponding individual.

ID	#gen. sign.	#rand. forg.	#skilled forg.	#sessions
000	55	25	25	4
001	35	25	30	3
002	75	25	30	2
003	35	15	25	2
004	15	15	30	1
006	35	10	39	1
007	35	10	25	1
008	35	10	30	1
009	15	10	30	1
010	15	10	30	1
100	15	15	25	1
101	0	0	0	2
102	15	15	25	1
103	15	15	25	1
104	15	15	25	1
105	15	15	25	1
106	55	15	25	3
107	35	15	25	2
108	15	15	25	1
109	15	15	25	1
110	15	10	25	1
111	15	10	25	1
112	35	10	25	2
113	15	10	25	1
114	15	10	30	1
115	15	10	30	1
116	20	10	25	1
117	40	10	25	2
118	20	10	25	1
119	20	10	30	1
120	20	10	30	1
121	20	10	30	1
32	795	430	865	50

Table A.2: The data base of the pen-written signatures. Each line consists of an identity-number, the number of signatures from the corresponding individual, the number of random and skilled forgeries of the signature from this individual and the number of sessions done by the corresponding individual.

Appendix B

Further Results

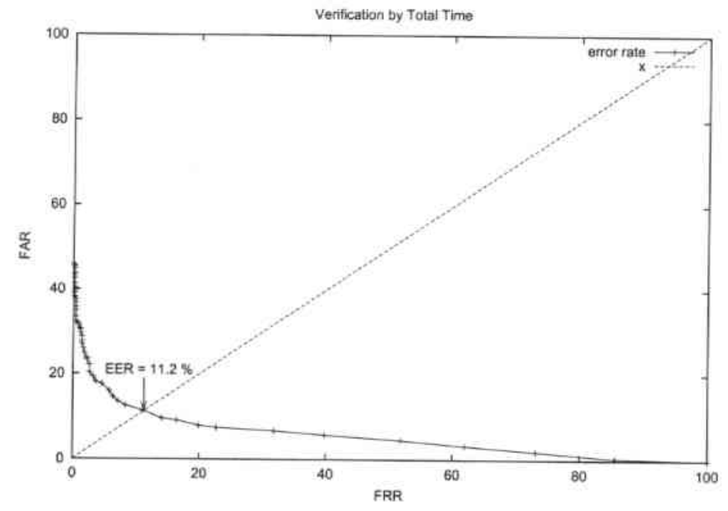


Figure B.1: Verification by total time as the only global parameter.

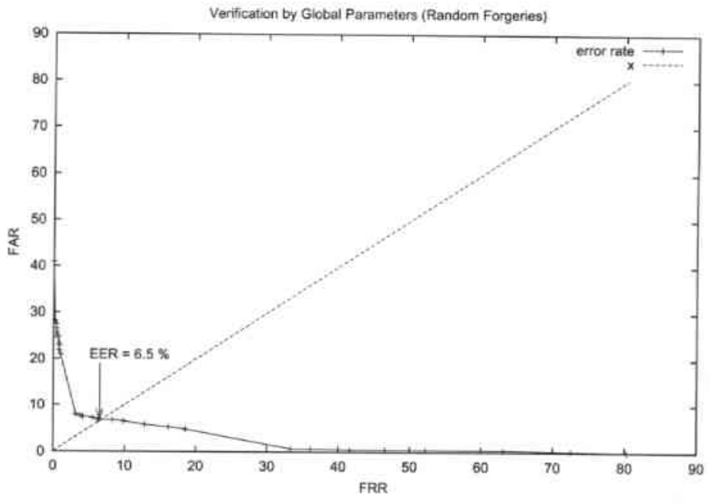


Figure B.2: Verification of random forgeries by global parameters.

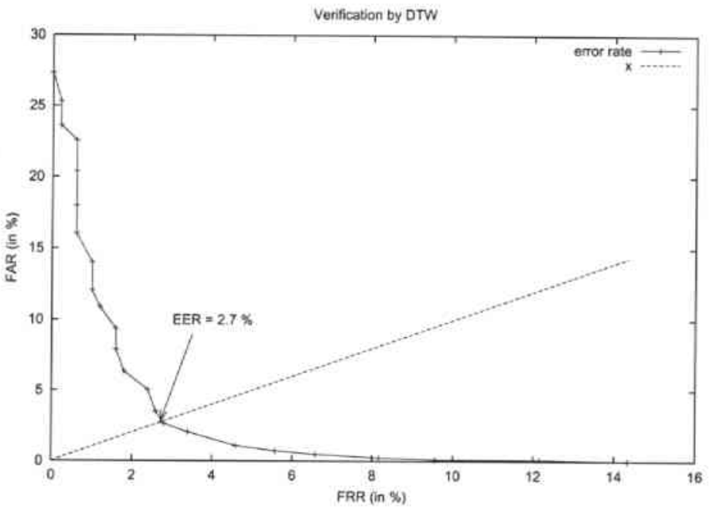


Figure B.3: Verification by DTW.

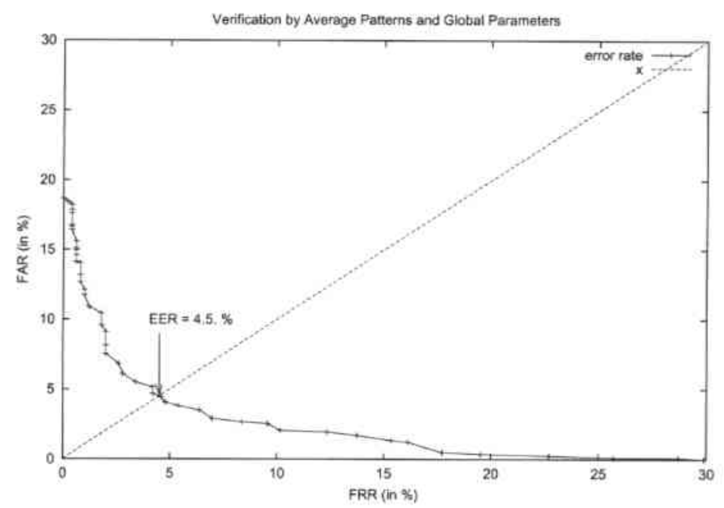


Figure B.4: Verification by average patterns and global parameters.

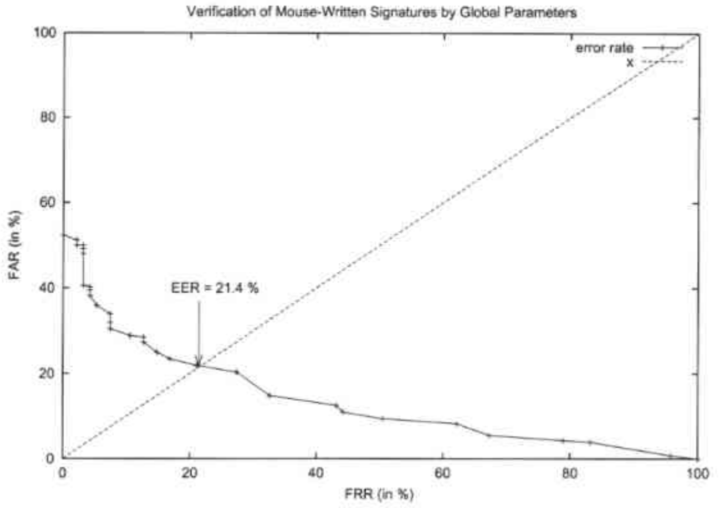


Figure B.5: Verification of Mouse-Written Signatures by global parameters.

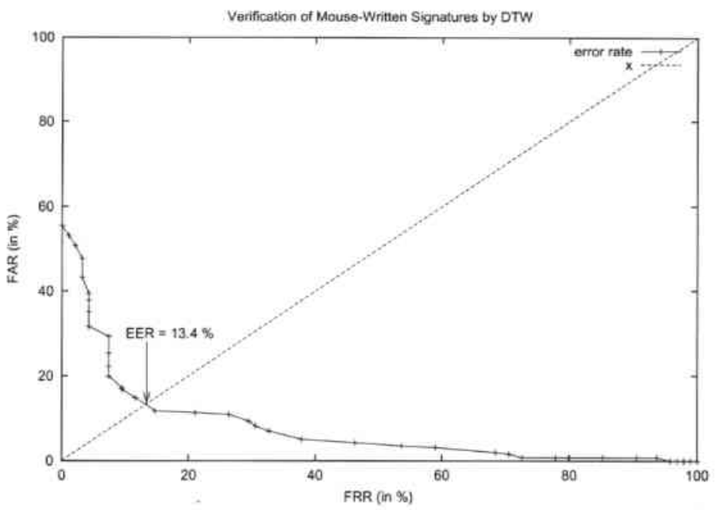


Figure B.6: Verification of Mouse-Written Signatures by DTW.

Bibliography

[1] G. Roethenbaugh, "Biometrics Explained," Available: http://www.icsa.net/services/consortia/cdb-c/sec4.html

[2] R. Plamondon and G. Lorette, "Automatic signature verification and writer identification - the state of the art", Pattern Recognition, vol 22, pp. 107-131, February 1989

[3] F. Leclerc and R. Plamondon, "Automatic signature verification: the state of the art 1989-1993", International Journal of Pattern recognition and Artificial Intelligence, vol 8, pp. 643-660, March 1994

[4] L. L. Lee and T. Berger, "Reliable on-line human signature verification system for point-of-sales applications"

[5] L. L. Lee and T. Berger, "Reliable on-line human signature verification systems", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 18, No. 6, 643-647, 1996

[6] D. D. Zhang, "Automated biometrics - technologies and systems", Kluwer Academic Publishing, 2000

[7] H. Sakoe and S. Chiba "Dynamic programming algorithm optimization for spoken word recognition", IEEE Trans. Acoust. Speech Signal Process. ASSP-26(1), 43-49, 1978

[8] A. Waibel, T. Hanazawa, G. Hinton, K. Shinao and K. Lang, "Phonem recognition using time-delay neural networks", IEEE Trans. Acoust. Speech and Signal Process, March 1989

[9] K. Fukunaga, "Introduction to statistical pattern recognition", Academic Press, 1990

[10] S. Manke, "On-line Erkennung kursiver Handschriften bei grossen Vokabularen", Shaker Verlag, 1998

[11] M. E. Munich, P. Perona, "Continuous dynamic time warping for translation invariant curve alignment with application to signature verification", Proc. of the Int. Conf. on Computer Vision, 108-115, 1999

[12] K. Huang, H. Yan, "On-line signature verification based on dynamic segmentation and global and local matching", Optical Engineering, 3480-3487, December 1995

[13] M. Mingming, W. S. Wijesoma, "Automatic on-line signature verification based on multiple models", CIFEr '00 Conf., 30-33, 2000

[14] S. H. Kim, M. S. Park, J. Kim, "Applying personalized weights to a feature set for on-line signature verification", Proc. ICDAR '95, 882-885, 1995

[15] K. Huang, H. Yan, "Signature verifiaction using fractal transformation", Proc. ICDAR '99, 851-854, 1999

[16] B. Wirtz, "Stroke-based time warping for signature verification", Proc. ICDAR '95 Vol. 1, 179-182, 1995

[17] B. Wirtz, "Average prototypes for stroke-based signature verification", Proc. ICDAR '97, 268-272, 1997