Michigan State University
Department of Computer Science and Engineering

Project Report:

On-line Signature Verification

By

Friederike Dorothea Griess

May 2000

Professor Anil K. Jain

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The handwritten signature is commonly used to manifest the contents of a document or authenticate a financial transaction. Signature Verification is usually done by visual inspection. A person compares the appearance of two signatures and accepts the given signature if it is sufficiently similar to the stored signatures. In the majority of situations where a signature is required, no verification takes place at all. This includes the signatures for credit card use and almost all signatures on documents that do not immediately implicate a money transaction. Automating the signature verification process will improve the current situation and eliminate fraud.

Automatic signature verification can be divided into two main areas depending on the data acquisition method: on-line and off-line signature verification. In off-line signature verification, the signature is available on a document which is scanned to obtain a digital image representation. On-line signature verification uses special hardware, such as a digitizing tablet or a pressure sensitive pen, to record the movements over the paper during writing. Both methods have advantages and disadvantages.

Off-line data is easy to acquire, a scanner is the only special hardware needed. It can also be applied to signatures that have been acquired in the past and documents that are not acquired in electronic format. On the other hand, the preprocessing steps needed to extract important information from the signature for verification are more difficult. The signature has to be separated from the background and segmented. No temporal information is available to indicate the process in which the strokes were formed so all the features must be derived from the shape of the signature. It is also easier to forge an off-line signature. To copy a written example of a signature, the forger only needs to copy the shape of the signature. On-line signature verification uses the dynamics of the signature in addition to its shape, which is not apparent from the 2-D representation and is difficult to forge. For on-line data the individual has to be present at the time of signing and actively participate in writing the signature. On the other hand, for on-line signature verification, special hardware has to be installed. This can be an obstacle for potential customers. However, we are able to record additional characteristics such as time dependencies and possibly pressure and pen tilt which are useful for verification.

Application areas for signature verification include all applications where handwritten signatures are already used such as cashing a check, signing a credit card transaction or authentication of a document with important instructions or information. The possibility to capture the signature and have it immediately available in digital form for verification introduces a range of new application areas. Basically every program that uses a password or pin can be replaced. This includes file and device access or signature verification based entry systems. The advantages are evident,

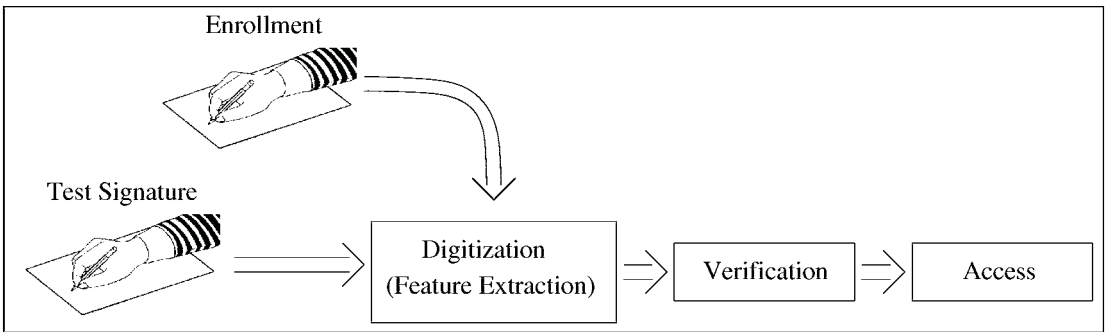

Figure 1.1: A typical signature verification system.

a signature is more difficult to steal than a password and also easier to remember for the user.

Figure 1.1 shows the diagram of a typical signature verification system. To enroll into the system, the user has to provide a set of enrollment signatures. Typically, a feature vector is extracted from the data which describe certain characteristics of the signature. For verification, the same features are extracted from the test signature and compared to the reference.

The system implemented in this project uses a digitizing tablet from the A.T. Cross company [1] as data capturing device. The CrossPad has a sampling rate of 100 to 150 samples per second and records the x- and y-coordinates of the handwriting. The pen has a touch sensitive switch in its tip such that only pen-down samples (i.e., when the pen touches the paper) are recorded. A string of consecutive pen down samples is also called a stroke. Figure 1.2 shows the data recorded for a signature. Each sample point is marked by a dot and each point is connected to its preceding and succeeding point. The same signature is shown as a function of time in Figure 1.3.

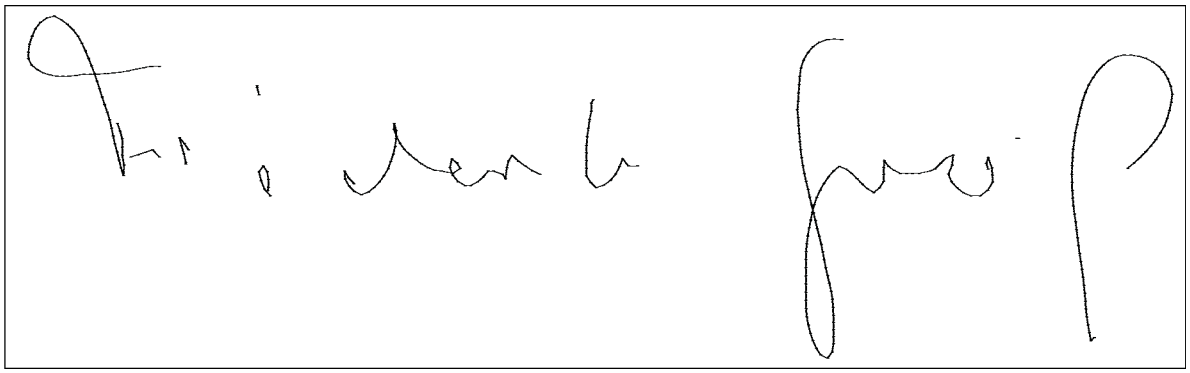

Figure 1.2: Signature data captured with a digitizing tablet. The sampled points are connected in their written order.

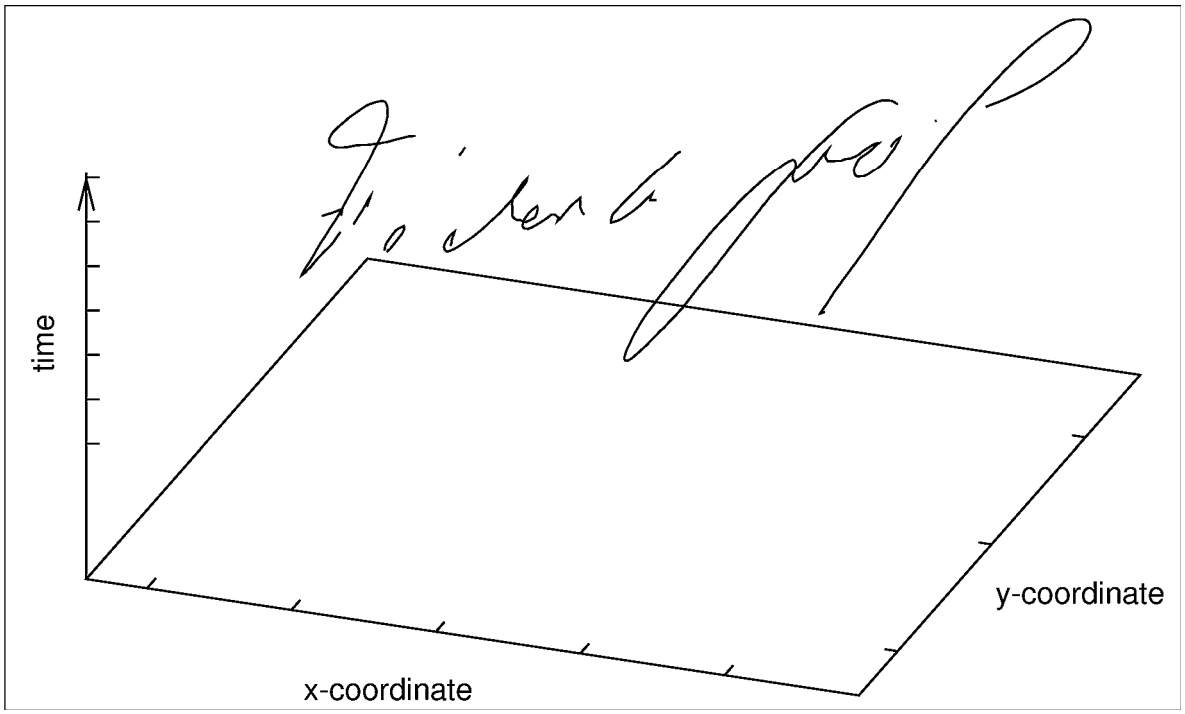

Figure 1.3: 3D-plot of the signature: the signature from Figure 1.2 is shown as a function of time.

Evaluating a verification system requires the analysis of two types of error. The percentage of genuine signatures that are wrongly rejected by the system is called the false reject rate or type I error. The percentage of wrongly accepted forgeries is called the false accept rate or type II error. The two types of error usually have different costs associated with them depending on the security requirements of the application. If high security is the main goal, then the false accept rate should be very low leading to a large false reject rate and potential user annoyance. If the user comfort with only mild security is the main goal, then a higher false accept rate must be tolerated. The performance of a system is often measured by its equal error rate, this is the point where the false accept rate and the false reject rate are approximately the same. A more meaningful performance measure is the error tradeoff curve, which shows how one error changes with respect to the other. Figure 1.4 shows an example of the error tradeoff curve. The x-axis represents the false reject rate and the y-axis shows the false accept rate. For a desired error rate of one type, the corresponding error rate of the other type can be found. Forgeries are classified into random or zero-effort forgeries and skilled forgeries. For a random forgery, the forger has either no knowledge about the original signature or does not try to imitate the shape of the signature. Genuine signatures of other writers are commonly used to simulate random forgeries. A skilled forgery tries to imitate the shape and sometimes even the dynamics of the original signature. Ideally, skilled forgeries are provided by professionals. In practice untrained individuals often provide imitations to evaluate new verification algorithms.

Compared to other biometric systems, signature verification has a long tradition.

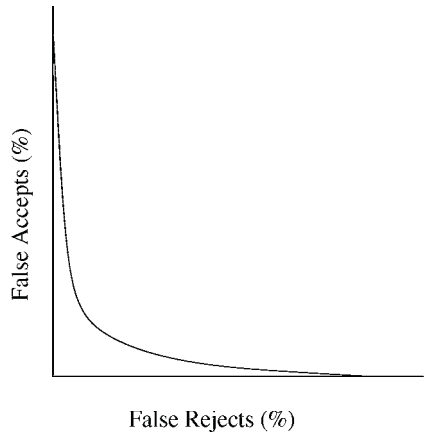

Figure 1.4: Example of an error tradeoff curve.

For this reason, it is more likely to be accepted by the general public. Fingerprints still have the stigma of police investigations and to new methods like face and iris recognition the general public has to get accustomed to. Signature verification also uses relatively simple hardware and does not depend on environmental changes such as illumination. But signature verification is still difficult since noise in the data is not only introduced through the hardware but two signatures of a single writer are never exactly the same. The signature is also the biometric feature that is easiest to forge. However, signature verification can be combined with other biometric methods, if higher security is needed.

Chapter 2

Literature Overview

A wide range of methods for online handwritten signature verification have been reported. Depending on the signature capture device used, features such as velocity, pen pressure and pen tilt are used in addition to spatial (x,y coordinates) features. Different approaches can be categorized based on the model used for verification. Since there does not exist a signature database in the public domain, every research group has collected its own data set. This makes a comparison of different signature verification systems a difficult task.

2.1 Matching Based Methods

Nalwa [2] provides a very detailed description of his signature verification system. He claims that the dynamic information of the signature is not as consistent as the spatial information. He uses both, global and local features which are primarily based on the shape of the signature. First a polygon is fitted through the sample points

and the jitter is computed. The signature is then normalized with respect to rotation and the aspect ratio. The jitter, the aspect ratio and number of strokes are kept as global features. The local properties of the signature are described by a set of characteristic functions which are derived from the notions of center of mass, torque and moments of inertia. To extract these features, the signature is parameterized over the normalized arc length to arrive at a representation which is generally independent of the signing velocity. Two sliding windows are used, one holds the coordinate frame while the other, which has a fixed distance to the coordinate frame, is called the computation window. The features are extracted from the part of the signature which falls into the computation window. The size of the computation window is usually a few hundredths of the complete length of the signature. Not all points are weighted equally, they are weighted with a Gaussian which effectively smoothes the signature. Five characteristic functions are derived, the x- and y-coordinates of the center of mass, the torque and two curvature-ellipses measures derived from the moments of inertia, all of which are functions of the normalized arc-length. The characteristic functions are simultaneously warped against their prototypes. Since the domain size of the functions and its prototypes can differ, an alignment between the argument values must be found. The warping algorithm minimizes the weighted harmonic mean of the single alignments, thus consistent parts are weighted more heavily by incorporating the standard deviation into the error measure. The amount of warping is retained as a global feature. The final dissimilarity measure is defined as the weighted harmonic mean of the global features. Results were obtained using three different databases with 904, 982 and 790 genuine signatures from 59, 102 and

43 writers, respectively. Additionally, 325, 401 and 424 forgeries were collected. For each writer 4, 5 and 6 signatures for building the prototype were evaluated. The equal error rates for 6 reference signatures varied between 2% and 5%. From the error tradeoff curves it was observed, that a small decrease in the false reject rate results in a much higher false accept rate.

The method proposed by Huang and Yan [3] is based on the motor command theory. The signature is considered as a concatenation of basic strokes that are generated by a motor controlled hand movement. During preprocessing small pen-up portions of the signature are merged into pen-down strokes and concatenated. Larger breaks are kept. The signature is then smoothed and normalized in size. The velocity profile of the signature is used to segment the strokes, where the number of sample points and the spatial length must not be below a system specific threshold. Several global features, like signing time duration, pen-up and pen-down times and aspect ratio, are extracted. For each global feature mean and standard deviation are recorded. Local features used are position, speed and acceleration. The velocity profile is used to segment the signature at points of near zero speed, if the length of the resulting segment is above a threshold. The verification procedure consists of two stages. In the first stage only the global features are used to compare a test signature to the stored templates and the signature is either rejected or passed to the local verification procedure. The global features are represented as a vector and the Euclidean distance is computed. In the second stage the profiles of the local features of each segment are matched using dynamic time warping. The Euclidean distance between the position, speed and acceleration profiles is weighted by the corresponding

segment size, which is the number of sample points. The system uses writer-dependent thresholds for each registered user in the verification process. The system was tested with data from 20 subjects who signed between 15 and 20 times each. The database also include skilled forgeries. The authors reported 5% error for false rejects and 2% error for false accepts. Experiments were also conducted with practiced names, where individuals were asked to practice a new name like a signature. The resulting error rates were higher, 7% and 12%, respectively.

Lee et al. [4] proposed a set of 49 features where only a subset of these features is selected for each writer according to his/her signing characteristics. Only global features were used, many of them are composed of minimum, maximum and averages for velocity and x- and y-coordinates. The feature selection method is based on the mean and variance of the features. If the minimum distance for the ith feature between writer a and all the other writers is bigger than the minimum distance for the jth feature, feature i is regarded as more important than feature j for writer a. This measure not only depends on the writer whose features are to be selected, but also on the rest of the database. Experiments incorporating skilled forgeries to select the best feature set for each writer were also conducted. The authors also proposed a common feature set which should be included in the individualized feature set of every writer. The decision process is implemented using a majority voting schema. Each feature is compared to the mean value for this feature from the reference set signatures and divided by the standard deviation. This normalized difference is compared to a threshold and the feature is either rejected or accepted. If at least half of the features are accepted, the signature is regarded as genuine. To augment the database, the

authors developed a statistical model to generate new signature data. The best error rate reported is 2.5% equal error rate and 7% false accept rate with the false reject rate approaching 0%. Both types of forgeries, zero-effort and skilled, were used.

Kim et al. [5] explore a similar idea. A total of 75 features are used, where many features are combinations of basic features such as average speed, aspect ratio, width or height of the signature. To find the best combination of features for each individual, $2^m - 1$ possible subsets need to be considered, where m is the number of features. Such a search would be computationally very expensive. To find the features that are most difficult to forge, the authors define an evaluation function, the Degree of Difficulty to Forge (DDF). Three different variants are presented. Each feature is assumed to have a normal distribution whose mean and standard deviation are estimated from all the signatures from all the writers. The general idea behind the DDF is as follows: if the mean of a feature differs significantly from the common mean, the feature is assumed to be hard to forge. Using 120 genuine signatures and 120 skilled forgeries from each of the 9 individuals, the equal error decreases from 5.5% to 4.3% by using personalized weights.

Wirtz [6] uses pressure information in combination with the x- and y-coordinates. The whole signature is represented as a 5-dimensional vector consisting of the x- and y-coordinates, pressure and the pen-up/pen-down information for each sampling point. Pen-up strokes are captured as well; the device used captures the pen position information even when it is not in touch with the paper. Dynamic programming is used to match the input signature to a reference. Matching is performed on a stroke basis. These are segmented with respect to the pen-up and pen-down samples. Since

the number of strokes between two signatures can vary, additional strokes in the test signature are compared to a "pseudo-stroke" and missing strokes are penalized. From a set of reference signatures, the mean and standard deviation for each stroke are computed and used to put more weight on the consistent strokes in the matching process. The similarity measure consists of two parts. The x- and y-coordinates are compared using dynamic time warping resulting in a spatial similarity. A motion similarity is obtained by comparing the optimal time axis transformation to the main diagonal of the reference with respect to the sampling points. These two measures are combined using a pseudo Mahalanobis distance. Results for 644 original signatures and 669 forgeries are reported. Different weighting schemas are evaluated and results for certain genuine writers and forgers are given. Since results are reported only for subsets of the data, it is difficult to generalize these results. For different individuals the equal error rate varies between 0% to 35% using the best weighting schema.

Gupta et al. [7] take a totally different approach. The x- and y-profiles of the signature and the speed and acceleration profiles in both the directions are used as features. The profiles are divided into valleys and peaks. Peaks and valleys are associated with a symbol, using different symbols for different properties, for example A stands for a peak and B for a valley in the x-profile. The signature is then represented as a string of symbols representing the peaks and valleys in all 6 profiles. Two strings are compared by computing the number of modifications needed to transform one string into the other. The signature is accepted if the smallest distance between the test signature and the reference signatures is smaller than the mean difference between the references. Experiments were conducted with 1229 signatures from 59

writers including 325 skilled forgeries. The lowest error rates reported for this method were 9% false rejects and 15% false accepts for skilled forgeries and 1.4% false accepts for random forgeries. Combining this method with seven global features (total time, number of sign changes in the x- and y-velocities and accelerations, pen-up time and total path length) and using a two-stage verification process, the error rates dropped to 4.5% false rejects, 10.8% false accepts for skilled forgeries and 3.8% false accepts for random forgeries. Texas Instruments has adopted the proposed method and implemented it on a microprocessor [8].

Gupta and McCabe [9] give an overview of several methods used for on-line signature verification. The authors distinguish between two approaches. In the first approach all position coordinates are assumed important and the signatures are compared on a point by point basis. In the second approach features are extracted from the x- and y-coordinates. Possible global features include total writing time and pen-up time whereas local features include signature path lengths, path tangent angles, local velocities and accelerations. Five methods to compute the distance between two signatures are presented in more detail: linear discriminant function, Euclidean distance classifier, dynamic programming matching technique, synthetic discriminant function and the majority classifier, which implements a voting schema. The reported equal error rates are between 3.8% for the majority classifier and 28% for the Euclidean distance classifier. All the results reported were obtained by Lee [10], but the size of the database and the type of forgeries (zero-effort or skilled) are not mentioned. The remainder of the work is dedicated to a comprehensive summary of earlier work which is divided into point by point comparison, feature value com-

parison and capturing the shape dynamically. The latter refers to studies where a dynamic model is used to reproduce the test signature. Error rates reported for all the methods range from false reject rates of 0.5% and false accept rates of 0% to false reject rates of 28% and false accept rates of over 21%. It is clear that no unified method or database to evaluate the performance of a signature verification system exists. The size of the database, the type of forgeries and the evaluation method used by different investigators vary substantially.

2.2 Hidden Markov Model Based Methods

Hidden Markov models are well known for their success in speech recognition. They have also been successfully applied to other recognition and verification tasks including handwriting recognition.

Yang et al. [11] use the absolute angular direction along the trajectory, which is encoded as a sequence of angles, to represent the signature. To obtain sequences of the same length, each signature is uniformly divided into a fixed number of segments. The normalized angle is then quantized into sixteen levels. Another sixteen levels are introduced for pen-up samples. Several Hidden Markov model structures were investigated including left-to-right models and parallel models. The model is trained with the forward-backward algorithm and the probabilities estimated with the Baum-Welch algorithm. In preliminary experiments the left-to-right model with arbitrary state skips performed the best. Sixteen signatures obtained from 31 writers each were used for evaluation; eight signatures were used for training and the remaining

eight for testing. No skilled forgeries were available. The experiments also showed that increasing the number of states and decreasing the observation length lead to a decrease in the false rejects and an increase in false accepts. The best results reported are a false accept rate of 4.4% and false reject rate of 1.75%.

Dolfing et al. [12] use a pressure sensitive pen to collect the signature data. Thus in addition to the x- and y-coordinates, the pressure and the tilt in the x- and y-direction information is also available. Thirty-two features were extracted from the data including both, global and local features. The model used is a left-to-right model, the observation probabilities are continuous mixtures of Gaussians, where up to four Gaussians are allowed per state to model different signature variations. The model is trained by a combination of algorithms and the estimation of the observation probability is a variant of the Viterbi algorithm. The threshold for accepting a test signature is a combination of a common offset and a writer dependent term. Data from 51 individuals was collected where each contributed 30 signatures. In addition, 3000 amateur forgeries and 240 professional forgeries were available. Equal error rates of 2.3% and 2.9% were reported depending on the type of forgery. The error rates could be halved by excluding the three shortest signatures from the experiments.

Rigoll and Kosmala [13] provide a comparison between on-line and off-line signature verification using hidden Markov models. Seven different features were investigated for their discrimination capabilities. Features used were pressure, angle, difference of the angle, velocity and acceleration and a bitmap feature that consists of nine grey values computed from a 30x30 sliding window. A different window of size 10 is used to calculate the Fourier transform of the signal. No global features

are used. For off-line data, the image is divided into a predefined number of squares containing approximately 10x10 pixels and the image is represented as a sequence of grey values, where each column is represented as a vector. The structure of the hidden Markov model used is not mentioned. The authors use discrete hidden Markov models. The Viterbi algorithm is used to compute the probability that the feature observation sequence has been generated by the respective HMM. Fourteen Writers provided 20 signatures each and in addition 60 forgeries were available. First each feature is evaluated separately before combinations of features are tried. Combining bitmap, velocity, Fourier transform and pressure features yield the best on-line error rates of 1% equal error. In the off-line case an equal error rate of 1.9% was obtained.

2.3 Other Methods

Wu et al. [14] use Linear Prediction Coding (LPC)-cepstrum and neural networks for Chinese character signature verification. A signature consists of several Chinese character symbols which are called words in the remainder. The signature is first normalized with respect to the symbol size and resampled such that each word in the signature contains the same number of sampling points. The resampled words are subdivided into frames which can overlap. For each frame, the x-coordinate LPC-cepstrum and y-coordinate LPC-cepstrum are computed. The signatures are compared using a multilayer perceptron network with backpropagation learning. For each word in a signature, a separate network is trained. The output of all the networks are added and compared to a threshold. The input layer consists of 2 times the

number of frames times the order of the LPC cepstrum units. The output layer is one unit which can take values between 0 and 1. The number of hidden units is not stated. Training data consists of approximately 10 genuine signatures and 10 skilled forgeries for each writer. The database contained 810 genuine signatures from 27 writers and 640 skilled forgeries. The best error rate, the sum of the false accept and false reject rates divided by two, is 4%. One of the stated advantages of neural networks for verification is the small storage cost for the model, since only the weights for the multilayer perceptron have to be stored. The same authors propose another method based on Fourier coefficients [15]. The signature is resampled and divided into frames like in the previous approach. Each frame is then windowed and the logarithmic spectrum is calculated. The intra-frame and inter-frame scatter matrices are combined into a feature vector for each frame. The similarity between two signatures is the sum of the differences between the feature vectors of each frame divided by the number of words in the signature. For each writer, a template is built from a set of reference signatures. The template mean and standard deviation are used to dynamically define a threshold. The same database as in the previous approach is used yielding a false accept error of 1.4% and false reject error of 2.8%.

Hastie and Kishon [16] present a statistical model for signature verification. A template signature is built from 5-10 references. To verify a new signature, the signature is first smoothed using a cubic spline and the speed is obtained as a function of time. The speed is then matched against the template using dynamic time warping. If the signature passes this test, it is segmented into letters and each letter undergoes an affine transformation to fit it to the model. The parameters for the template are

obtained by the same procedure. The authors experimented with a small database of 10 people but did not report any results.

2.4 Commercial Products

A number of commercial software companies have implemented signature verification algorithms and integrated them into commercial products. Application domains for these products are primarily computer access systems, file access systems and document authentication in combination with electronic signatures. Generally the application domain is not limited by these systems. Most of the manufacturers provide additional software and hardware to facilitate the installation. All systems allow the user to set the verification threshold according to the application and environment specifications. Thus the false accept rate and the false reject rate can be manipulated. No claims are made by these systems regarding their error rates.

Cyber-SIGN Cyber-SIGN Inc. [17] distributes its system with a software development kit, that allows the user to develop its own applications. For distributed applications, a server is provided and document authorization software can be acquired with it. A stand alone implementation is also distributed. The data capturing device used is a graphic tablet with a pressure sensitive pen from WACOM [18]. The number of signatures that are needed to enroll a user into the system is not mentioned. To perform verification, the system makes use of pressure, shape, direction, speed and velocity. Verification is done from a secure server.

DATAVISION SigRecognition The system distributed by DATAVISION [19] uses a signature pad from the same company. The software is integrated with a signature display program. The software is used for account management. Five signatures are used to enroll into the system, from which a template is generated. The template can be updated. The electronic representation of the signature has a size of 108 bytes in addition to an image of the signature that is stored. The software uses both representations for verification. The capabilities of the signature pad used to capture the data are not mentioned.

PenOp Signature The PenOp Signature software from PenOp [20] allows signing and authenticating documents online. A digitizing tablet is used to capture the signature. The signature is then converted into a so called signature stamp which is based on the captured signature. With a different user verification method (password, etc.) the signature stamp can be affixed to a document with the additional information of when and where the document was signed. The receiver can extract and verify the signature. Three signatures are used to build a signature template and the template can be updated.

LCI-SMARTpen BIAS The SMARTpen from the LCI Technology Group [21] captures forces in three directions, the speed of the writing and the angles at which the pen is held. The pen does not require a special tablet to write on, it can be used with any surface. The information is encrypted within the pen and transmitted through radio frequency transmission. The verification software is designed to be

integrated easily into existing applications. Again the parameters of the system allow a user to adjust the error rates to be tailored to the specific problem. The enrollment process is not described.

Quintet SignCrypt The software system developed by Quintet [22] is exclusively for handheld PCs and runs under WindowsCE. A digitizing tablet is used to capture the signature. Features that are used for verification are speed, slanting angles, tilting angles, numbers of strokes, space usage, aspect ratios, stylus friction, positions, pressure shapes, grouping and legibility which implies that a pressure sensitive and angle capturing pen must be used. These features are analyzed and stored as profiles that can be stored in any computer memory or embedded in the magnetic strip of a credit card for future comparisons.

2.5 Summary

Table 2.1 summarizes the methods used in the literature. Most of the approaches do a fair amount of preprocessing before extracting features from the signature. Using only global features (for example in [4]) has the advantage of being very fast, but the error rates for algorithms that also incorporate local features are generally lower. The most common method to compare the extracted local features is a variant of the Euclidean distance. Since the number of points differs between two signatures, some form of string matching (see for example [6]) is used. The number of signatures to enroll varies between 6 and 20. The equal error rates generally lie between 1% and

Method	Comparison/ Decision	Threshold and No. References	Databases and Results
String Matching	- global - strokes	- common - writer-dep. 6 - 20	20 - 103 Writers, +skilled, 10 - 30 Signatures each, 3% - 5% Equal Error
Hidden Markov Models	- Baum-Welch - Viterbi	- writer-dep. 8 - 16	14 - 15 Writers, 20 - 30 Signatures each, 1% - 4% Equal Error
Neural Networks	- Multi Layer Perceptron	- automatic	27 Writers, +skilled, ca. 30 Signatures each, 4% Equal Error
Only Global Features	Euclidean Distance	- writer-dep. 10 - 20	9 - 105 Writers, +skilled, 20 - 30 Signatures each, 2.5% - 5.5% Equal Error

Table 2.1: Comparison of signature verification methods used in the literature.

6%. Many factors influence the verification results making them difficult to compare. The number of signatures and their quality are a major influence. Certainly a large number of writers must be included in the database before the system performance can be generalized. This is not always feasible either in a single session or over a period of time. In most approaches forgeries were provided by colleagues which were sometimes given instructions and time to practice. The equal error rate per se is not a good measure to compare the results; the error tradeoff curves provide more information. All of the commercial products allow the user to adjust the system parameters to tune the system to operate at different security levels.

Table 2.2 summarizes the main properties of the commercial systems. The number of signatures used to enroll a user is generally smaller in these systems than the methods described in the literature. It is also interesting to mention, that most systems allow the reference signatures to be updated. This topic is generally not

Name	Data Capturing Device	No. of Signatures required to enroll	Updatable Templates	Error Rates
Cyber-SIGN	Digitizing tablet with pressure sensitive pen	n/a	n/a	n/a
SigRecognition	n/a	5	yes	n/a
PenOp Signature	Digitizing tablet	3	yes	n/a
LCI-SMARTpen BIAS	Pressure and angle sensitive pen	6	yes	n/a
Quintet SignCrypt	Pressure and angle capturing device	n/a	n/a	n/a

Table 2.2: Summary of commercial signature verification products.

mentioned in the literature. No software manufacturer describes the algorithms used or reveals error rates of their system.

Chapter 3

System Structure

In this chapter the methodology used by our signature verification system and the algorithms implemented are described. Section 3.1 describes briefly all the components of the system and depicts their connections. Section 3.2 illustrates the preprocessing before features are extracted from the signature as presented in Section 3.3. A pair of signatures is compared using a string matching measure, which is described in Section 3.4 and the combination of the similarity values to arrive at a final decision is given in Section 3.5.

3.1 General Methodology

For signature verification, the following topics have to be addressed:

- Preprocessing
- Feature Extraction

- Enrollment
- Matching
- Threshold Selection

A brief description of each topic is followed by an illustration of how the components work together.

Preprocessing

The input signal from a digitizing tablet or digitizing pen can be very jagged. The physical space provided to sign may vary between different applications and the pen used can affect the smoothness and the size of the signature. A common method to smooth the signature is to use a Gaussian filter. To avoid the influence of the size difference on the matching result, the signature is normalized in size. In order to compare the spatial features of the signature, time dependencies have to be eliminated from the representation. This can be achieved by resampling the signature uniformly with equi-distant spacing. Temporal features must be extracted before resampling. Preprocessing is described in detail in Section 3.2.

Feature Extraction

The features extracted from a signature can be divided into two categories: global and local features. Global features are derived from the whole signature and result in one or more values representing a certain characteristic. Local features are extracted in the local neighborhood of a sample point. Additionally, spatial and temporal features can be distinguished. Spatial features are derived from the shape of the

signature whereas temporal features capture the dynamics of the signing process. Combinations of spatial and temporal features are also possible. The features used in our system are described in Section 3.3.

Enrollment

To use a signature verification system, a user has to provide several example signatures to enroll into the system. The number of signatures usually varies between 3 and 10. These signatures are then stored as a reference set or a template is generated. The enrollment process is described in Section 3.5.

Matching

To perform verification of a test signature, it has to be compared to the reference set. The method implemented to compare the local features is based on string matching. Each point in the signature is described by a feature vector. String matching finds an alignment between the points such that the sum of the differences between the feature vectors is minimized. This technique is described in Section 3.4. Global features are easier to compare since the number of feature vectors extracted from different signatures do not differ.

Threshold Selection

The output of the string matching is a dissimilarity value which is compared to a threshold to accept the signature as genuine or reject it as a forgery. Either a common threshold is used, which is derived from the training data, or a writer-dependent threshold can be selected for each writer. In this system a hybrid approach

is used: a global threshold is chosen and a user dependent offset is calculated from the enrollment data. Section 3.5 describes the advantages and disadvantages of the different approaches for threshold selection.

Verification

The difference values between the test signature and the signatures in the reference set must be combined and this value is compared to the threshold to verify whether the test signature is genuine or not.

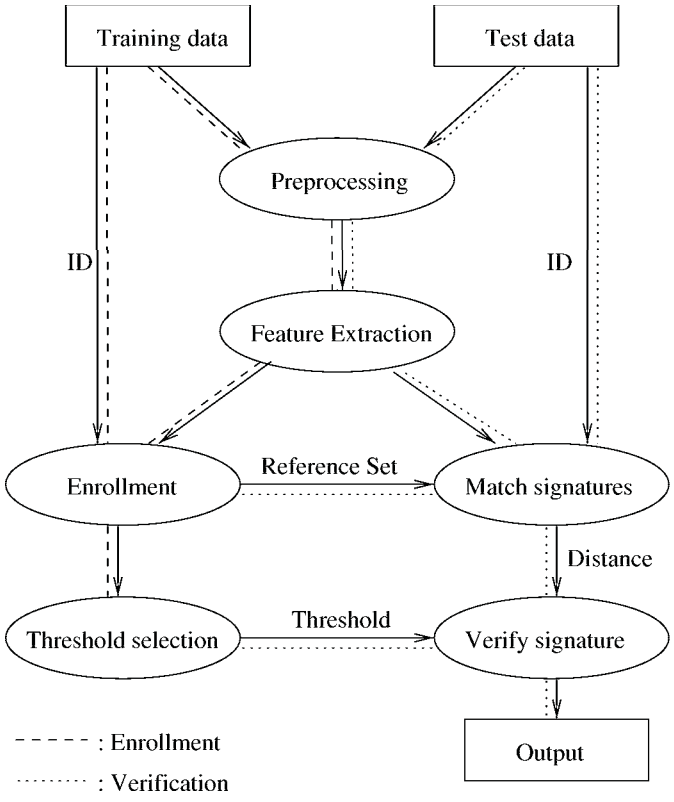

Figure 3.1: Modules of a signature verification system.

Figure 3.1 shows how the modules relate to each other. During enrollment of a new user, input to the system is a set of training signatures produced by that user. The

training data is preprocessed and the features are extracted as described in Sections 3.2 and 3.3. This data is then saved in a database together with a unique identifier that is used to retrieve the signatures during matching. In addition, a threshold is derived from the data. For verification, a test signature which has to be verified is input to the system, along with the claimed writer identity of this test signature. The same preprocessing and feature extraction methods are applied. The signature is then compared to each of the reference signatures which are retrieved from the database based on the writer identifier. The resulting difference values are combined and, based on the threshold for the writer, the signature is accepted as genuine or rejected as a forgery.

3.2 Preprocessing

The input from a signature tablet or other coordinate capturing device can be rather noisy. The amount of noise depends on the capturing device used, on the speed of the signer's writing and on the writing itself. To what extent the signature is preprocessed/smoothed must be carefully decided. With every change in the original signature, a potential writer dependency may be eliminated. The roughness of a signature may be typical for a writer, so smoothing will eliminate this characteristic. A writer may have a "small" signature, so size normalization will eliminate the differences between large and small signatures. On the other hand, size differences among signatures may be due to different amounts of space that are provided for the signer, in which case size normalization is necessary. Resampling of the signature

eliminates the speed information but the shape information can be extracted more reliably. All these effects must be considered. In this section the preprocessing steps used in the current system are described in more detail. The same preprocessing has been successfully used for handwritten character recognition [23], [24].

3.2.1 Resampling

Online data allows both spatial and temporal features to be incorporated into the matching process. To compare two signatures with respect to their shape, the signatures have to be resampled. Temporal features must be extracted before uniform resampling since all local speed information is lost during this process. The shape features can be extracted more reliably when the signature is resampled. Additionally, resampling is necessary before smoothing to ensure that the signature is uniformly smoothed. Otherwise, segments of low writing velocity would be smoothed more than segments that are written very fast. Figure 3.4 shows the effect of resampling a signature. First the signature is resampled with a small spacing to get many sample points for the subsequent smoothing. After smoothing, the signature is resampled again with wider spaces to retain only as much information as necessary for verification. More resampling points do not only mean more local information but also increase the complexity of feature extraction and matching. Redundant or useless information may be contained in many sample points which not only increases the processing time, but also decreases the accuracy of the system. Different smoothing variants need to be investigated to find a suitable size. Certain points in the signature carry

important information such as start and endpoints of a stroke and points where the direction of the writing changes. These points, referred to as critical points, are extracted before preprocessing and retained throughout the resampling and smoothing process. Critical points are described in Section 3.2.3.

To resample the signature, the distance between two critical points is measured and the effective spacing calculated as the distance divided by the number of sample points for that segment. The distance between each two consecutive points is calculated and whenever the cumulated distance exceeds the effective resampling length, a new point between these two original points is calculated using the gradient between them.

3.2.2 Smoothing

To smooth the signature, a Gaussian filter is used. One property of the Gauss filter is that its weight decreases from the center of the filter; pixels close to the center have more weight than those further away. Small changes in the signal are smoothed out, while the overall structure of the signal is kept. The x- and the y-direction of the signature are smoothed separately. The Gauss filter in one dimension is defined as

$$f_i = \frac{e^{-\frac{i^2}{2\sigma^2}}}{\sum_{j=-2\sigma}^{2\sigma} e^{-\frac{j^2}{2\sigma^2}}}$$

where f_i is the value of the filter at position i, i ranges from -2σ to 2σ, σ is the variance of the filter (measured in number of consecutive sample points) and set to 2. The total size of the filter is chosen to be 4σ since the values of the Gaussian

approach zero with increasing distance from the center. The x- and y-coordinates are correlated with the Gauss filter f_i:

$$x_t^{filtered} = \sum_{i=-2\sigma}^{2\sigma} f_i * x_{t+i}^{orig}$$

Critical points are kept without smoothing; the segments between two critical points are smoothed separately. The effect of smoothing is shown in Figure 3.5.

3.2.3 Critical Points

Certain points in the signature carry more important information than the others. Resampling or smoothing these points can affect the structure of the signature. It is important that these points are not effected by resampling or smoothing. Critical points are:

- endpoints of strokes
- points of trajectory change

Figure 3.2 shows possible trajectory changes with the corresponding critical points extracted. If several points with a trajectory change are close together, the point with the smallest curvature is taken to be the critical point.

Figure 3.3 shows how the curvature is computed. The angle between the actual point, p_i and the second to next point, p_{i+2}, along with the angle between the second to previous point, p_{i-2} are used to compute the curvature. The range for these angle is 0 to 360 degrees.

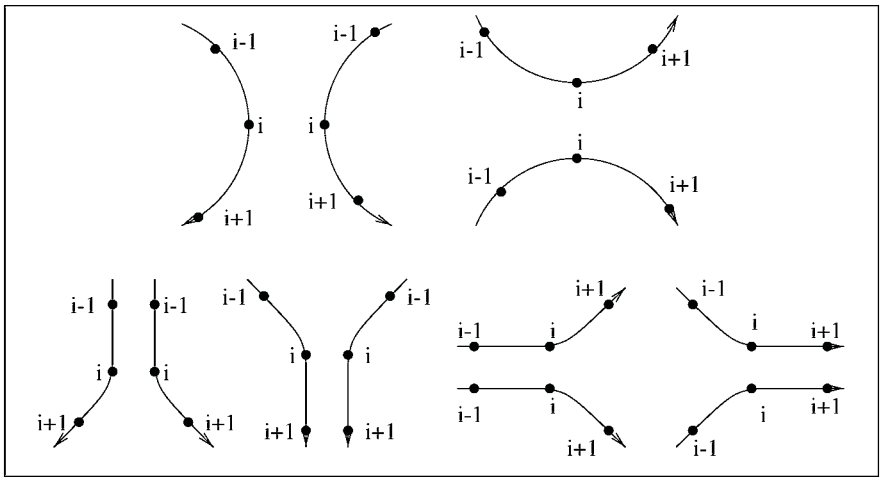

Figure 3.2: Critical Points: i denotes the critical point, $i-1$ and $i+1$ are the preceding and succeeding points, respectively. The examples at the top show situations, where the x- or y-direction of the stroke changes. The lower part of the figure shows examples, where a transition from a vertical or horizontal stroke into a curve exists. The arcs show the writing direction.

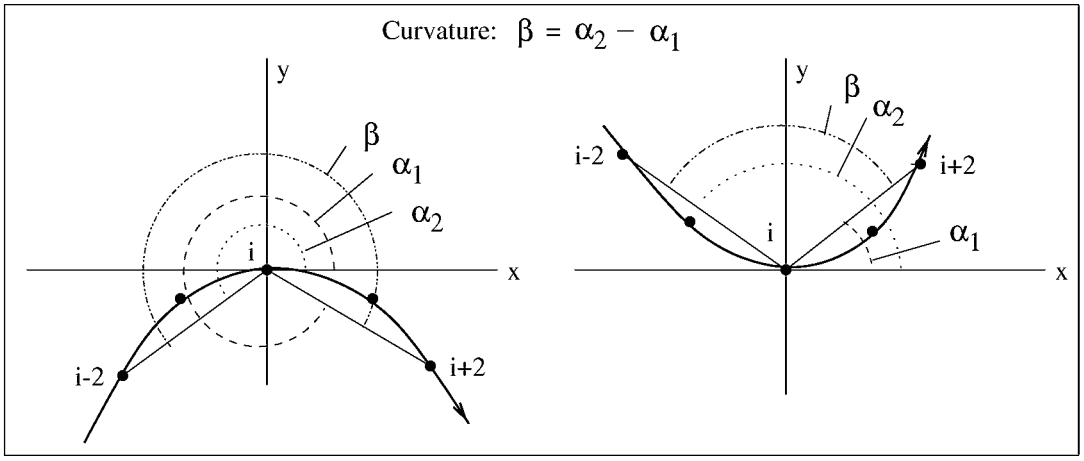

Figure 3.3: Curvature: The curvature β is calculated as the difference of the angles that the line through the second preceding point, $i-2$, resulting in angle α_2, and the line through the second succeeding point, $i+2$, resulting in angle α_1 form with the x-axis of a coordinate system centered at point i. The left example results in a curvature value greater 180 degrees, in the right example the curvature value is smaller than 180. The arcs show the drawing direction.

3.2.4 Size Normalization

Comparing two signatures and changing their size generally changes the similarity value depending on the matching method. In applications where different signing forms are used or a varying amount of space is provided for the signer, it is necessary to normalize the signature in size. On the other hand, for applications that do not change the input method, it may be useful to keep the original signature size, since this is an important characteristic of a writer.

During size normalization, the original aspect ratio can either be kept or the signature is normalized with respect to both, height and width. The latter approach eliminates writer dependent information, so in this system only the height is normalized; the original aspect ratio is kept which leads to different widths for different signatures. Often the signature is also normalized with respect to skew. We assume that the skew is a characteristic of the signing style and expect it to be consistent.

After preprocessing, all the strokes are connected into one long string. This annihilates the direct information of the stroke endpoints. This is to facilitate the use of the string matching procedure described in Section 3.4. In this context the motivation for the connection of the strings will be further explained.

The effects of all the preprocessing steps except for size normalization are shown in Figures 3.4 and 3.5.

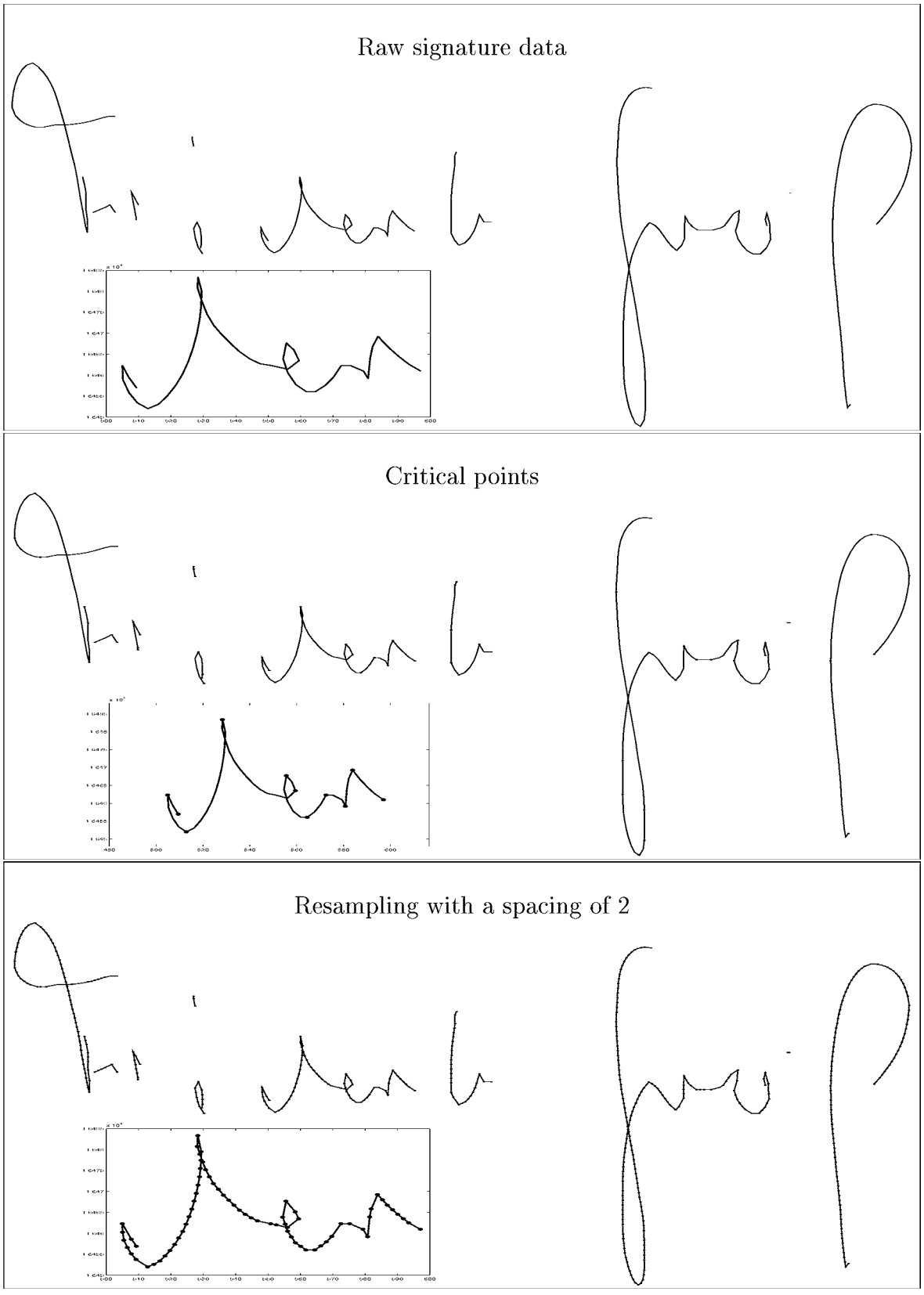

Figure 3.4: First three stages of preprocessing.

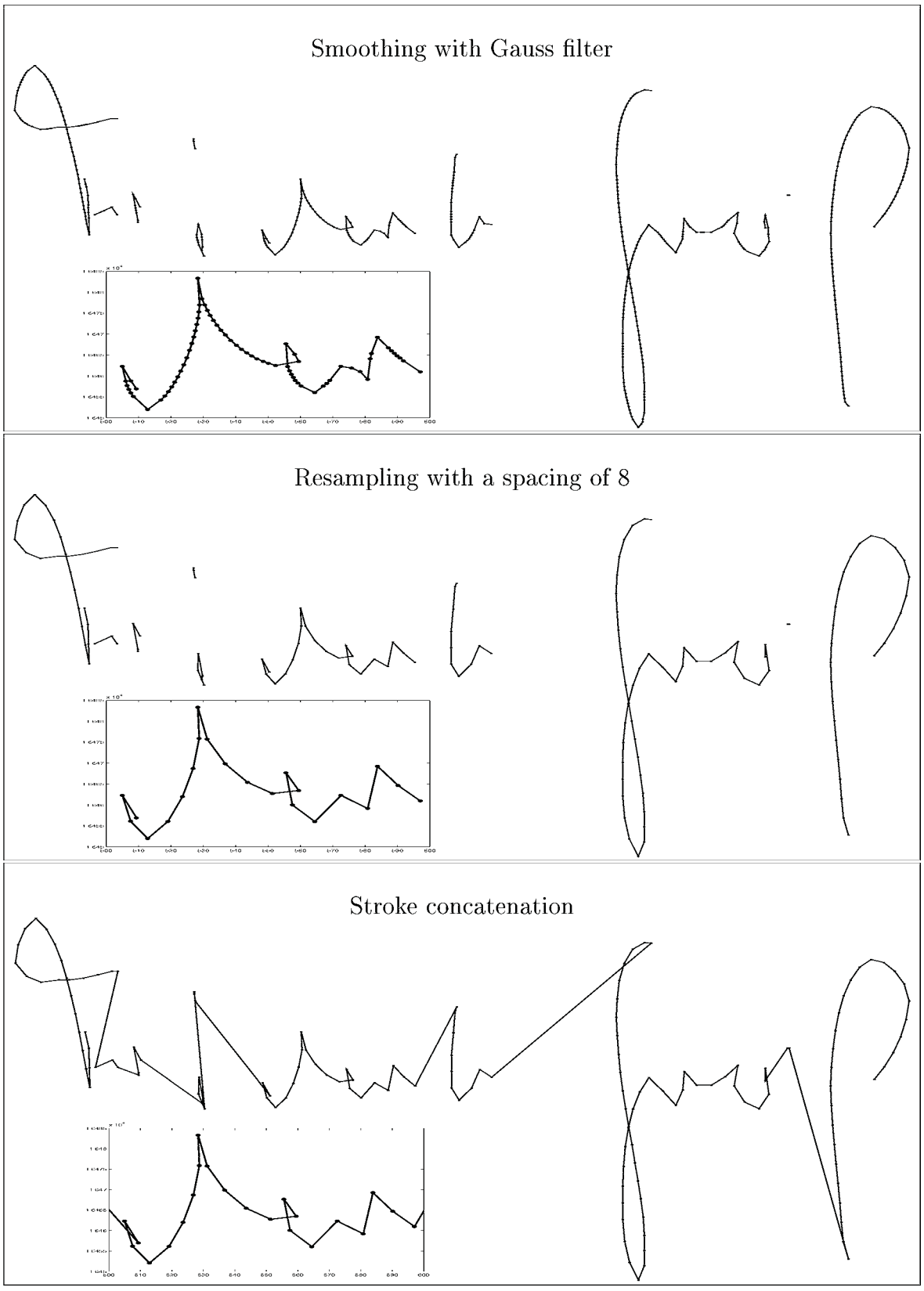

Figure 3.5: Remaining three stages of preprocessing.

3.3 Feature Extraction

Two approaches are possible after preprocessing. In the functional approach the x- and y-coordinates of the signature are input to the verification procedure. Alternatively, features can be extracted and the signatures are compared based on the feature values. Global features refer to the whole signature. Local features are local to a sample point within the signature and are derived only from the point and its local neighborhood. This section will introduce the selected features in more detail.

3.3.1 Global Features

The only global feature in this system is the number of strokes. As described above, all strokes are combined into one long stroke during preprocessing. The original number of strokes is recorded and used as a feature. Appendix A.1 shows the distribution of the mean and deviation of the number of strokes for each writer. It can be seen that the number of strokes is relatively stable for a signer.

3.3.2 Local Features

From the x- and y-coordinates of the preprocessed image, a number of features are extracted which are divided into two categories, spatial and temporal features. Spatial features are static features that are extracted from the shape of the signature. To a certain extent, these features can be derived from an image of the signature. This does not apply to temporal features such as velocity. For both types of local features, temporal information is needed, since is not possible for image data (off-line) to infer

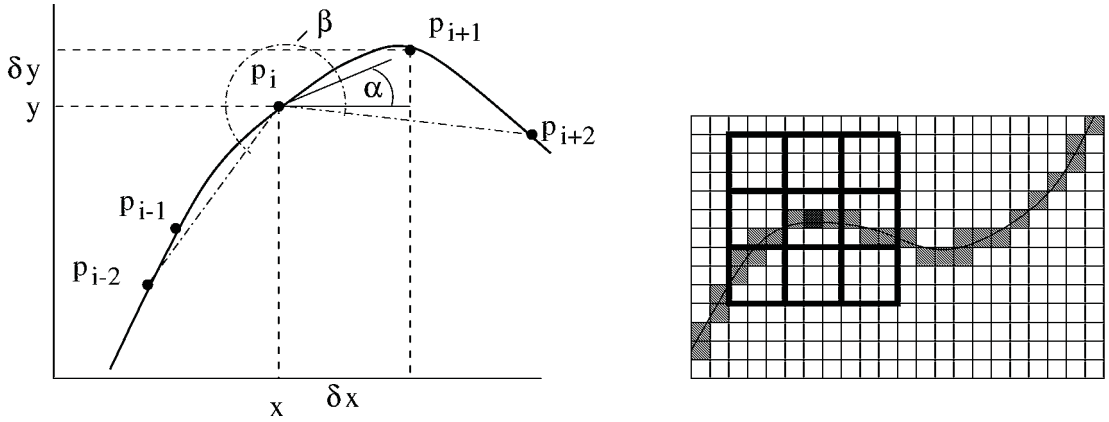

Figure 3.6: Local spatial features.

the order or the direction of the single strokes.

Spatial Features

Local spatial features that are extracted and studied for their potential for signature verification are the following:

- the change of the distance between two consecutive points, δx and δy
- the absolute y-coordinate, y
- the sine and cosine of the angle with the x-axis, $\sin\alpha$ and $\cos\alpha$
- the curvature, β
- the grey values in a 9x9 pixel neighborhood

Figure 3.6 shows all the spatial features. In this figure the features are computed at point p_i; the two preceding points are p_{i-1} and p_{i-2} and the two succeeding points are p_{i+1} and p_{i+2}, respectively. The change of the x- and y-coordinates for point p_i are

the changes with respect to the subsequent point p_{i+1}. Since the last point of a stroke does not have a successor, no features are extracted for it. It should be noted that all strokes are connected into one stroke after preprocessing, therefore only the last point of the signature is omitted. The absolute y-coordinate is the y-coordinate of each resampled point after preprocessing. The angle α is the angle between the x-axis with the straight line through points p_i and p_{i+1}. Although the angle already carries all the information, its representation as the sine and cosine has certain advantages. Similar angles have similar sine and cosine values. Consider the angular values 1 and 359. The sine values for these angles are close to zero, and the cosine values are close to one. Using the original angular values as features, a special similarity measure must be used. With the sine and the cosine the similarity can be measured simply by the Euclidean distance between the values. Both sine and cosine are used, since the directional information is encoded in the sign of the value. Furthermore, sine and cosine are nonlinear functions which have small value changes for a specific range of input values so they supplement each other. The curvature feature is the angle between the straight lines $\overline{p_i p_{i-2}}$ and $\overline{p_i p_{i+2}}$. For the points at the beginning and the end of the stroke, p_{i+2} or p_{i-2} may not exist. In this case the closest existent point is taken. The image feature calculates nine grey values in the neighborhood of the sampling point. A 9x9 pixel neighborhood is divided into nine 3x3 squares as shown in Figure 3.6. For each square, a grey value is computed as the sum of the pixel values falling in that window. The range of the grey values is theoretically between zero and nine, but in practice values usually range between zero and four. A total of nine grey values are computed for each sampling point. The image feature is costly

compared to the other features, since the signature must be transformed into pixels and many image operations are necessary to extract the features. It also adds many values to the feature vector which increases the time needed for signature matching.

Using all the above features, the feature vector has a dimensionality of 15 (2+1+2+1+9). In Section 4 the feature combinations tested will be introduced in more detail.

Temporal Features

In addition to the temporal order of the points, the speed of the writing at local points is a valuable feature. Two distinct opinions are present in the literature. Some argue that the shape of a stroke determines the speed with which it is written. Even if different writers scribble the same thing, the speed at certain points in the writing will be similar. Others argue that the speed is typical to a particular writer.

Two different variants to extract the speed from the signature are explored:

- absolute and relative speed at each resampled point
- absolute and relative average speed between two critical points

The velocity is defined as distance per time unit. The digitizing tablet captures the position of the pen 100 times per second. This already provides a uniform time unit. The distance between two consecutive points is a measure of the speed of the writing between those points. Resampling destroys the original speed information. Thus the speed has to be extracted before or during resampling. In Figure 3.7 the calculation of the speed during resampling is shown.

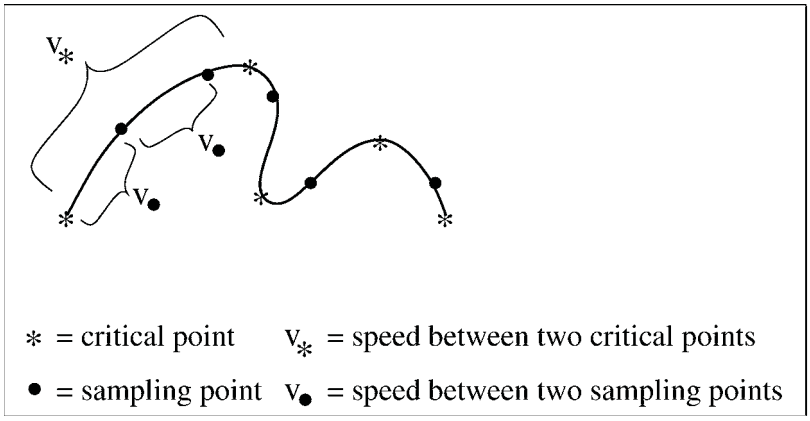

Figure 3.7: The speed between two consecutive critical points, V_*, and the speed between any two points, $V_\bullet$, is calculated as the distance between those points.

Usually the resampled point is between two points in the source signal. The distance between those two points is taken as the speed value. This value can differ from the correct speed especially when the resampled point is very close to an original point. Since the sampling rate of the digital tablet is very high, the change in the speed between two consecutive sampling points can be assumed to be smooth except for endpoints of strokes. Since each stroke is resampled individually, this does not affect the procedure. In the case where the resampled point is equivalent to an original point, the difference between this and the succeeding point is taken as the speed. For the last point in the stroke, the difference between the preceding point and the last point is taken.

Normalizing the local speed at every sample point by the average writing speed over the whole signature yields a speed feature that is independent of the overall speed. This takes into account the fact that due to writing circumstances and due to the size of the signature, the average writing speed of a writer can vary. The relative

signing speed at each point within the signature nevertheless should be stable.

Another approach is to extract the speed only at critical points and measure the average speed from one critical point to the next. This value can also be normalized by the average speed of the signature. Although critical points are extracted and kept before any preprocessing, the average velocity is computed before preprocessing, since the arc lengths between two consecutive critical points can be changed during resampling and smoothing.

3.4 String Matching

Once the local features are extracted from each point in the signature, a method must be chosen to compare two signatures. The representation of the signature so far is a collection of points, where each point is represented by a n-ary feature vector. Two matching methods are needed, one to compare two feature vectors and one to compare the entire string, if necessary.

Two points that are similar with respect to a property represented by a feature have similar values. Thus the Euclidean distance is a straightforward choice to compare the feature vectors.

String matching is a well known method to compare strings of different lengths. It finds an alignment between the points in the two strings such that the sum of the differences between each pair of aligned points is minimal. The method used here was implemented for handwritten character recognition [23], [24]. The two stings are referred to as the template string and the input string. The alignment is represented

as a set of pairings between the points:

$$\{(e^T_{t^T(1)}, e^I_{t^I(1)}), (e^T_{t^T(2)}, e^I_{t^I(2)}), \ldots, (e^T_{t^T(L)}, e^I_{t^I(L)})\},$$

where $e^T_{t^T(n)}$ is the $t^T(n)$th point in the template T, n is the position of the point in the alignment and the function $t^T(n)$ returns the position, which corresponds to the time it was drawn, of the point in the original string. A similar definition applies for $e^I_{t^I(n)}$ which corresponds to the input string. Since the strings can have different lengths, the matching algorithm must be allowed to skip points in the alignment in either signature. Skipped points in the template are called spurious points and skipped points in the input are called missing points. It is forbidden to match one point in one string to more than one point in the second string:

$$t_T(i) = t_I(j) \text{ if and only if } i = j$$

It must also be ensured that the matching obeys the temporal order of the points. If a correspondence between two points is found, no point further in time can be mapped to a point in the second string in the past:

$$t^T(1) < t^T(2) < \ldots < t^T(N_T)$$

$$t^I(1) < t^I(2) < \ldots < t^I(N_I)$$

where N_T and N_I are the total number of points in the template and input, respec-

tively. A simple alignment that always yields the minimum distance is to align no points at all. In order to avoid this, for each spurious or missing point in either signature, a penalty is added. Different penalties can be used to penalize omitted points in one of the strings more than the other. The values for the penalties must be carefully chosen to avoid aligning points that are too dissimilar and to avoid skipping too many points.

Figure 3.8 illustrates the string matching between two signatures. To keep the representation simple, only the matching of the critical points is shown. Whenever a line between two points in different signatures is drawn, these two points are mapped to each other in the alignment. An interpretation of the first constraint is that no two lines representing a mapping can share the same point. Points without any mapping lines are allowed. The second constraint means that no two mapping lines can cross each other.

To find the minimal difference, all possible alignments must be investigated. To solve this problem efficiently, dynamic programming can be used. The value for the optimal alignment of two partial strings up to the points i and j in the template and the input string, respectively, can be computed from the optimal alignments of the partial strings up to the points $i-1$ and $j-1$, respectively:

$$D(i,j) \;=\; Min \begin{cases} D(i-1,j-1) + d_E(i,j) & \\ D(i-1,j) + \text{Missing Penalty} & 1 \le i \le N_T \\ D(i,j-1) + \text{Spurious Penalty} & 1 \le j \le N_I \\ D(i,j) + \text{Missing Penalty} + \text{Spurious Penalty} & \end{cases}$$

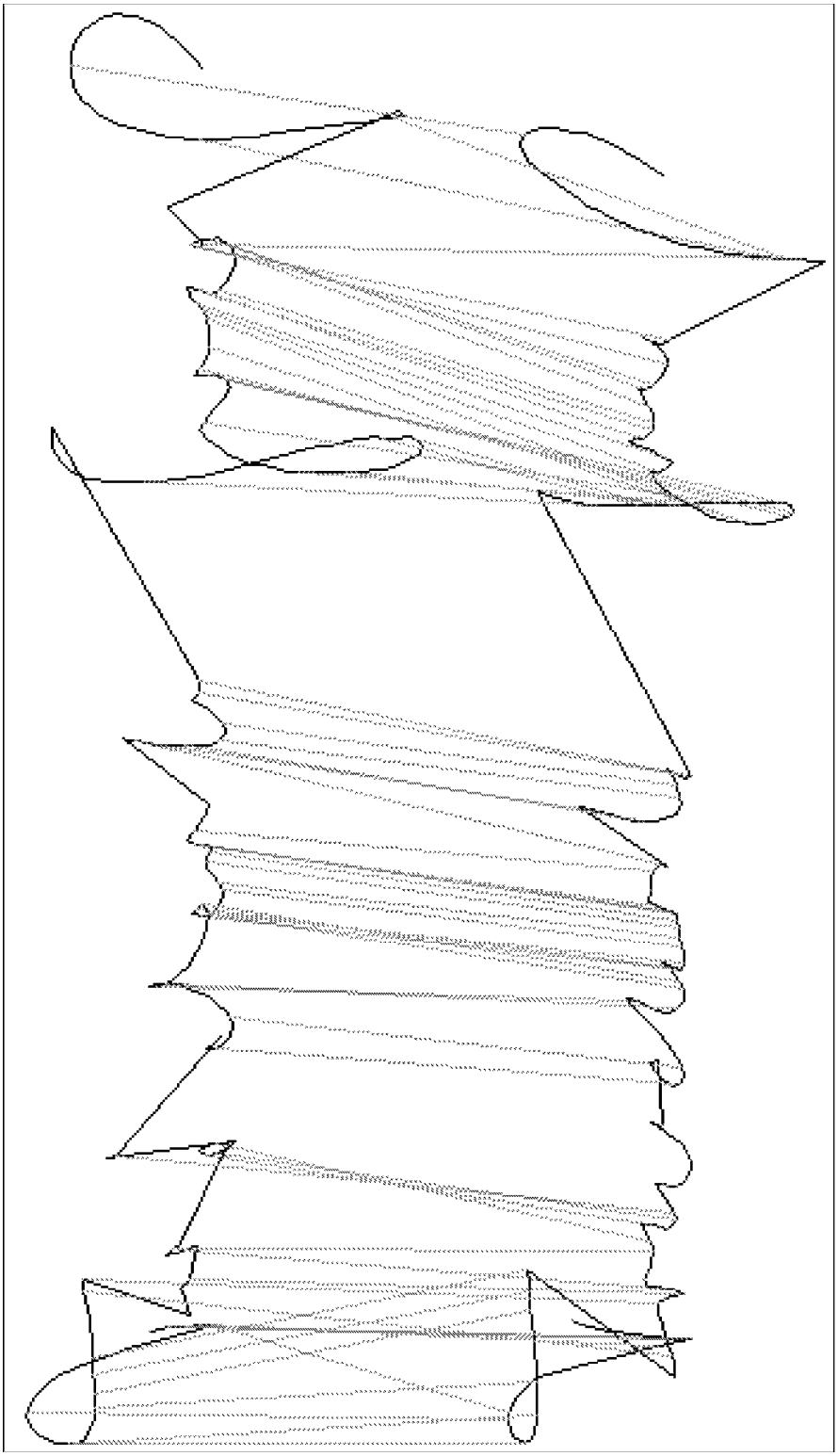

Figure 3.8: Alignment between points in two signatures of the same individual.

$$Dist(T, I) = D(N_T, N_I),$$

where $d_E(i, j)$ is the Euclidean distance between feature vector for point i and the feature vector for point j.

After the alignment between two signatures is found and the difference is calculated, the difference between the number of strokes in the two signatures must be incorporated into the overall dissimilarity measure. The formula for the overall dissimilarity is

$$Dist(T, I) = \frac{Dist(T, I)^2}{Norm\_Factor(N_T, N_I)} + (SP)|S_T - S_I|,$$

where SP is the stroke penalty, $|S_T - S_I|$ is the difference of the number of strokes in the template and the input strings and $Norm\_Factor(N_T, N_I)$ is maximum possible distance between two strings of length N_T and N_I scaled by a constant factor. Figure 3.8 shows the alignment between two signatures of the same writer and Figure 3.9 shows the alignment between two different writers.

3.5 Verification

To enroll in the system, the user has to provide a set of reference signatures. These signatures can either be used to create a template of the signature or they are stored directly. Using a template has several advantages; a test signature must be compared only to the template to calculate the similarity value and the storage requirements are low. Updating the template is simple; a weighted average between the template and

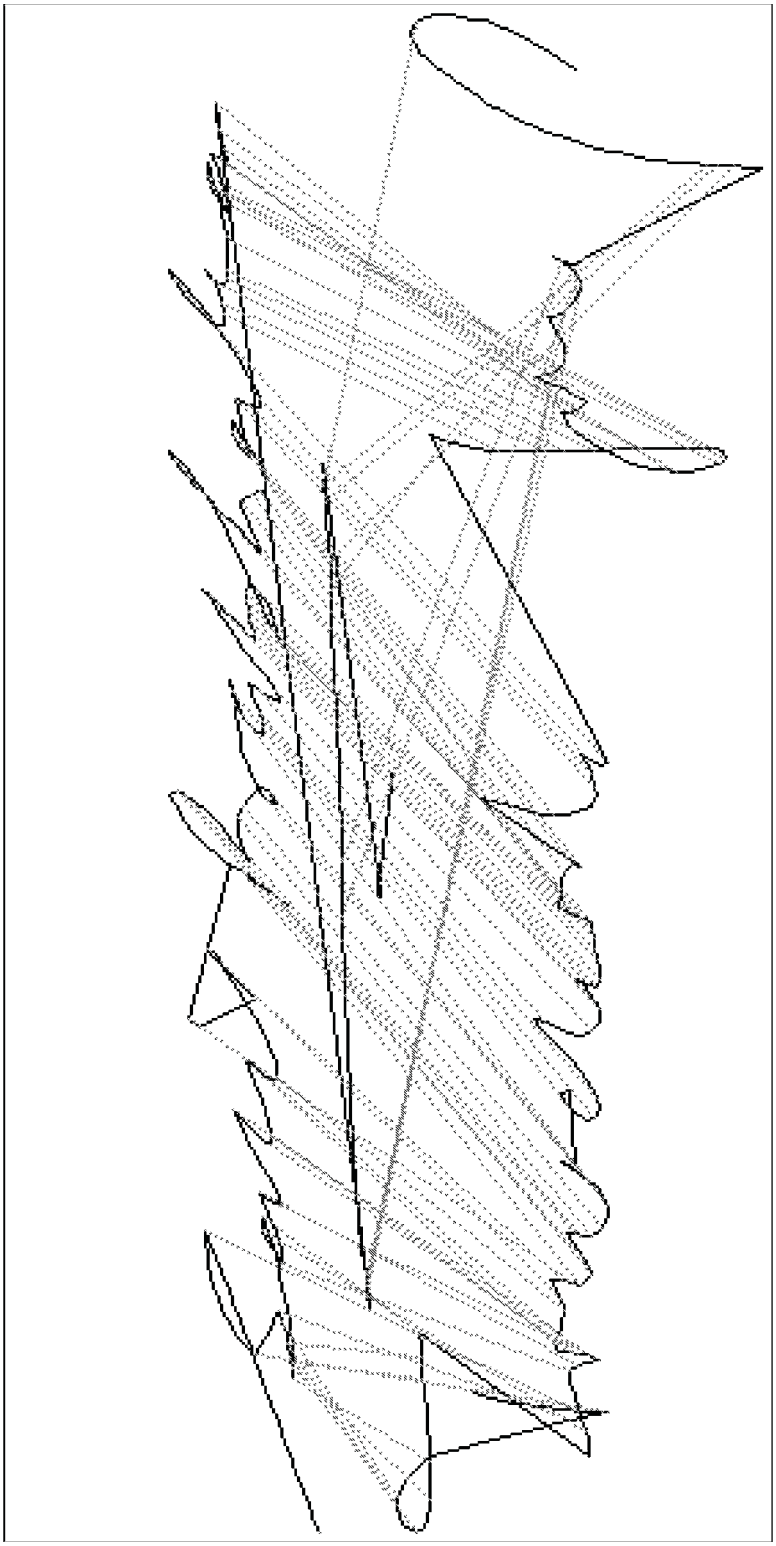

Figure 3.9: Alignment between points in signatures of two different individuals.

the new reference signature is calculated. If all the references are stored, a method to calculate a combined dissimilarity value must be defined. Updating the reference set is also simple; a signature that is outdated or redundant is selected and replaced by the new reference. A major advantage of keeping all the references is the ability to model different writing styles. This is the approach implemented in our system.

3.5.1 Combined Dissimilarity Value

In the verification process a test signature must be compared to all the signatures in the reference set. Three basic methods to combine the individual dissimilarity values into one value are investigated:

- minimum of all the dissimilarity values
- average of all the dissimilarity values
- maximum of all the dissimilarity values

The minimum value (see Section 4) has been shown to be the best choice. This is reasonable since taking the minimum difference value between the test signature and all the reference signatures will only take the reference signature most similar to the test signature into consideration. If a writer has different signing styles, the signature would only be compared to the reference modeling this particular writing style. Using the average reduces this distinction and taking the maximum value punishes writers that have multiple signing styles. If many references signatures are available, a hybrid method where a set of templates for the different writing styles

is built should be considered. It merges the advantages of both the approaches by modeling different writing styles and by incorporating a large amount of training data without having many signatures to compare. The string matching is the most time consuming step (see Section 4.3).

After the dissimilarity value is found, a decision regarding whether the signature is authentic or a forgery must be made. For this, the result of the matching will be compared to a threshold. If the dissimilarity value is above that threshold, the signature is rejected, otherwise it is accepted. The threshold can be chosen to be identical for all the writers or set individually for each writer.

3.5.2 Common Threshold

A common threshold has the advantage that all data from all writers can be used to find an optimal value. The differences between all signatures of all writers who are enrolled into the system are computed and a threshold value is selected according to an optimization criterion. Since lowering the false acceptance rate always increases the false reject rate and vice versa and different applications have different requirements, the optimal threshold will be different from application to application.

3.5.3 Writer-Dependent Threshold

Using a common threshold has the disadvantage that the characteristics of different writers are not taken into account. Some writers may have a very consistent writing style which usually results in small dissimilarity values, while another writer may be

very inconsistent or may have a very complicated signature for which higher dissimilarity values are common. These differences are not taken into consideration with a common threshold. To adapt the verification process to the properties of a single writer, writer-dependent thresholds should be used. In principle, a writer-dependent threshold can be derived only from the enrollment data. However, to get a reliable estimation of the optimal threshold, more data than usually available is necessary. To circumvent this, one starts with a common threshold and then modifies it for each writer by adding a writer specific component. This component is derived from the enrollment data. Three choices to calculate the writer specific component from the reference set are investigated:

- the minimum distance between all references
- the average distance between all references
- the maximum distance between all references

Using the minimum difference between the reference signatures as a value to be added to the common threshold proved to be the best choice. This is validated by experiments presented in Section 4.

Chapter 4

Experimental Results

The proposed method has been implemented and evaluated with 1,232 signatures from 102 different writers. The datasets used are described in Section 4.1. Section 4.2 describes the feature selection process. In Section 4.3 the effect of different parameters used during preprocessing is shown. Finally, threshold selection is addressed in Section 4.4.

4.1 Dataset

Two datasets, called DB1 and DB2, are used for evaluation. The first dataset, DB1, contains 520 signatures from 52 writers. Each writer was asked to contribute ten signatures. Additionally, for 20 writers three forgeries each were collected from individuals. These writers were shown the original signature before producing the forgeries. No signatures from professional imitators are available. These 60 forgeries will be called skilled forgeries in the remainder of the chapter. The second database,

Dataset	Number of writers	Number of signatures per writer	Total number of signatures	Total Number of forgeries
DB1	52	10	520	60
DB2	102	10 - 42	1,232	60

DB1 $\subset$ DB2

Table 4.1: Datasets for signature verification.

DB2, contains a total of 1,232 signatures collected from 102 writers.

Ten signatures per writer were collected in each session. Seventeen writers contributed more than ten signatures, which were collected over a period of up to one year. The second dataset DB2 is a superset of the first dataset DB1. Table 4.1 summarizes the data used.

To evaluate the system, the error rates for genuine signatures and forgeries must be analyzed. Since only a limited number of forgeries exist in which an attempt is made to imitate an original signature from the database, signatures from other writers serve as random or zero-effort forgeries. The error rates for the different kinds of forgeries are evaluated separately. Signatures from different writers are called *random* forgeries and the 60 imitations of the original signatures are called *skilled* forgeries, although it should be noted, that this term is commonly restricted to professional forgeries.

4.2 Feature Selection

For on-line handwriting recognition, the potential of several feature sets has been evaluated in [25]. To see if these features are also applicable to signature verification,

each feature combination is tested with database DB1. Table 4.2 shows the results.

Feature Subsets	Type of Forgery	Threshold (TH)	Percentage of genuine values above TH	Percentage of forgery values below TH
δx, δy, $\sin\alpha$, $\cos\alpha$	genuine skilled	3.9 3.3	9.4% 11.4%	9.5% 11.6%
δx, y, $\sin\alpha$, $\cos\alpha$	genuine skilled	122 143	15.7% 13.4%	15.5% 13.6%
δx, δy, y, $\sin\alpha$, $\cos\alpha$	genuine skilled	142 171	15.3% 12.4%	15.4% 12.5%
δx, y, Curvature	genuine skilled	158 150	8.9% 10.4%	9.1% 10.5%
Image features	genuine skilled	$14.4 * 10^4$ $15.3 * 10^4$	12.1% 10.6%	12% 10.5%
δx, y, $\sin\alpha$, $\cos\alpha$, Image features	genuine skilled	$13 * 10^4$ $11.1 * 10^4$	9.5% 14%	9.6% 13.9%

Table 4.2: Performance of for different feature subsets using a common threshold.

To evaluate the discriminative potential of the feature sets, every signature of every writer is compared to all other signatures. The resulting dissimilarity value should be low for two signatures from the same writer and high for two signatures of different writers. The first column of Table 4.2 lists the feature subsets used (see Section 3.3). The fourth column gives the percentage of time the signatures from the same writers differ by more than the threshold shown in column three. Column five reports the percentage of time the difference between signatures of two different writers is below the threshold. The threshold is selected such that these two percentages are approximately equal. In Table 4.3 the same dataset is evaluated with thresholds individually selected for each writer. The threshold is again selected such that approximately equal percentages in column four and five are obtained. It must be

Feature Subsets	Type of Forgery	Threshold (TH)	Percentage of genuine values above TH	Percentage of forgery values below TH
δx, δy, $\sin\alpha$, $\cos\alpha$	genuine skilled	0.5 - 45.3 0.5 - 8.5	3.6% 4%	3.5% 4.2%
δx, y, $\sin\alpha$, $\cos\alpha$	genuine skilled	14 - 500 47 - 494	10.1% 8.5%	10% 8.7%
δx, δy, y, $\sin\alpha$, $\cos\alpha$	genuine skilled	15 - 600 52 - 589	9.7% 7.6%	9.6% 7.6%
δx, y, Curvature	genuine skilled	53 - 300 53 - 283	4.4% 3.1%	4.4% 3%
Image features	genuine skilled	$5.2*10^4$ - $23.6*10^4$ $5.2*10^4$ - $29.1*10^4$	8% 4.5%	8.1% 4.7%
δx, y, $\sin\alpha$, $\cos\alpha$, Image features	genuine skilled	$5*10^4$ - $19.9*10^4$ $5*10^4$ - $26.6*10^4$	6.5% 4%	6.6% 4%

Table 4.3: Performance of different feature subsets using a writer-dependent threshold.

noticed however, that the results for the common threshold are based on much more data; 2,295 intraclass dissimilarity values and 26,520 interclass dissimilarity values are available for the common threshold case versus only 45 intraclass dissimilarity and 510 interclass dissimilarity values for writer-dependent thresholds. The writer-dependent results yield lower percentages for all feature subsets, but yield approximately the same ordering of feature subsets in terms of their discrimination ability. Based on these results, only the feature set consisting of δx, δy, $\sin\alpha$ and $\cos\alpha$ is further investigated. The feature set consisting of δx, y and the curvature also shows good results, but the discrimination values for the writer-dependent threshold is better for the first feature set. The results for the forgeries are not considered to be of much statistical relevance, since only 60 forgeries are available altogether. Figure 4.1 shows the distribution of the dissimilarity values for the feature set consisting of δx, δy, $\sin\alpha$ and $\cos\alpha$. The range of the dissimilarity values is divided into equally sized bins. A point on the curve represents the percentage of dissimilarity values that fall into that bin whose midpoint is the value given on the x-axis. Additional graphs can be found in Appendix A.2.

The best feature subset consisting of spatial features, as determined above, is combined with different temporal features. Speed features can be defined as absolute or normalized speed at every resampled point and absolute or normalized speed between two critical points (see Section 3.3). Each speed feature has been added to the "optimal" spatial feature set resulting in a local feature vector of dimensionality 5. The results of the various combinations can be found in Table 4.4. The first column in Table 4.4 lists the additional speed feature as introduced in Section 3.3. The results

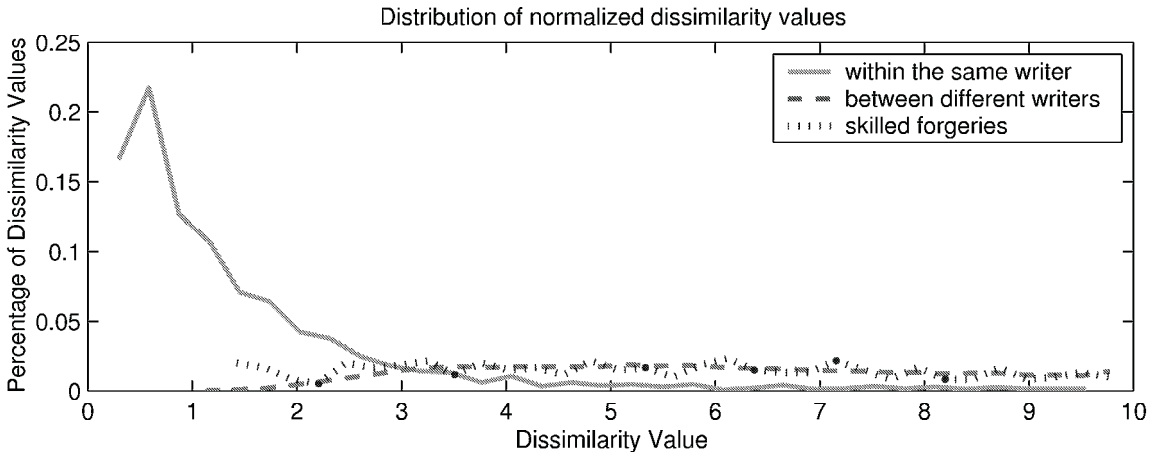

Figure 4.1: Distribution of dissimilarity values between signatures of the same writer, signatures of different writers and skilled forgeries.

Speed feature	Type of Forgery	Threshold (TH)	Percentage of genuine values above TH	Percentage of forgery values below TH
absolute speed	random skilled	8.5 5.5	6.0 % 13.6%	5.9% 13.6%
normalized speed	random skilled	6.6 3.9	4.5% 12.3 %	4.5% 12.2%
absolute speed between critical points	random skilled	12.5 8	3.9% 10.1%	3.9% 10%
normalized speed between critical points	random skilled	8 5	4.9% 10%	4.8% 10.1%

Table 4.4: Influence of temporal features using a common threshold.

Speed feature	Type of Forgery	Threshold (TH)	Percentage of genuine values above TH	Percentage of forgery values below TH
absolute speed	random skilled	2 - 20 2 - 19	3.4% 3.7%	3.5% 3.8%
normalized speed	random skilled	1.5 - 20 1.8 - 15	2.6% 2.5%	2.7% 2.4%
absolute speed between critical points	random skilled	4 - 19 4 - 20	2.5% 2.3%	2.6% 2.3%
normalized speed between critical points	random skilled	1.5 - 13 2.5 - 17.5	2.7% 2.5%	2.6% 2.6%

Table 4.5: Influence of temporal features using writer-dependent thresholds.

imply that the absolute speed between critical points is the best feature. The results for writer-dependent thresholds can be found in Table 4.5.

4.3 Preprocessing

Preprocessing transforms the input data into a representation more suitable for the subsequent stages of verification. The parameter settings during preprocessing must therefore be chosen carefully. Two different aspects are investigated in more detail, the spacing of resampling and size normalization.

As described in Section 3.2, size normalization of the signatures may be necessary depending on the application. During data collection, no individual was given any size restriction on the signature. It turns out that generally the signature size is very consistent for every writer. Therefore size normalization actually increases the error rates. Nevertheless, size normalization should be incorporated if we wish to ensure

Spacing	Processing Time in milliseconds			Type of forgeries	FRR	FAR
	Pre-processing	Feature Extraction	Matching			
4	21 - 42	474 - 1919	1800 - 2900	random skilled	2.1% 7.9%	1.5% 7.3%
8	12 - 64	358 - 1938	229 - 700	random skilled	1.6% 7.2%	2.0% 6.2%
12	21 - 32	450 - 1882	110 - 300	random skilled	1.7% 7.2%	1.9% 4.6%

Table 4.6: Results for different resampling spacings for the feature set δx, δy, $\sin\alpha$, $\cos\alpha$ using a common threshold for all users. The results are obtained with dataset DB1. In column two to column four the processing times needed for preprocessing, feature extraction and string matching are shown.

the compatibility of the matching algorithms in multiple applications.

Another important aspect is the spacing that is used to resample the signature. A small spacing leads to a more accurate sampling and retains the shape of the signature with greater detail, but also leads to an increase in the time for extracting the features and comparing the signatures. In Table 4.6 results for different resampling rates are reported. The error rates using three randomly chosen signatures as references out of the ten available signatures per writer are reported in Table 4.6. Details about the error calculation can be found in Section 4.4. In columns two, three and four the processing time needed for preprocessing, feature extraction and string matching in milliseconds is reported. As can be seen, string matching dominates the processing time, and is much faster when the signature contains fewer points. The error rates using writer-dependent thresholds are reported in Table A.2 in Appendix A.3. The error rates stay approximately the same for the three spacings that were examined. The results were obtained before the speed features were implemented, but since most

of the calculation of the speed is already done during preprocessing and the speed feature adds only one dimension to the feature vector, the results are expected to stay in the same ranges.

Figures 4.2 and 4.3 show the same signature resampled with spacings of four, eight and twelve pixels. Up to 300 pixels correspond to one inch on the writing surface. It can be seen that with smaller sampling sizes (four and eight pixels), almost every detail of the signature is kept. For a resampling size of twelve pixels, small details start to disappear.

In the remainder only a resampling with a spacing of 8 pixels is used, since it retains a lot of the detail and matching takes a reasonable amount of time.

4.4 Threshold Selection

In the previous sections the discriminatory potential of different feature sets was investigated. To derive the error rates for the system, a set of sample signatures, called the reference set, must be chosen and compared to the remaining signatures. Reference sets of sizes three and five are investigated. Two types of error are encountered in verification applications. The false reject error or type I error is the event when a genuine signature is rejected as a forgery by the system. The false accept error or type II error is the event when a forgery is accepted as a genuine signature. The two errors are not equivalent in their impact. To be able to compare the results of different systems, the equal error rates are commonly used. Two methods of fixing a threshold are investigated, the use of a common threshold for all the writers and

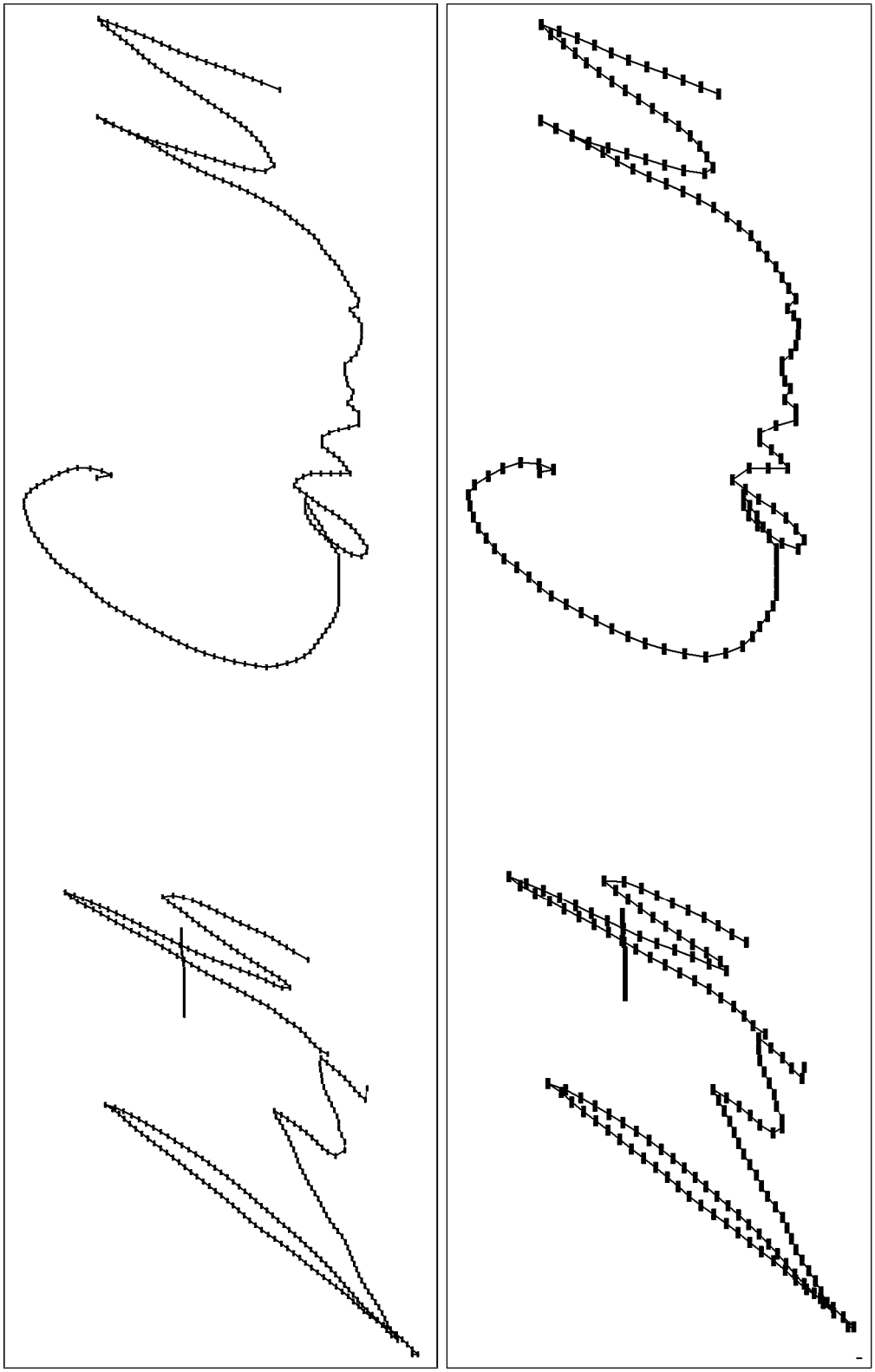

Figure 4.2: Resampling with a spacing of 4 and 8 pixels.

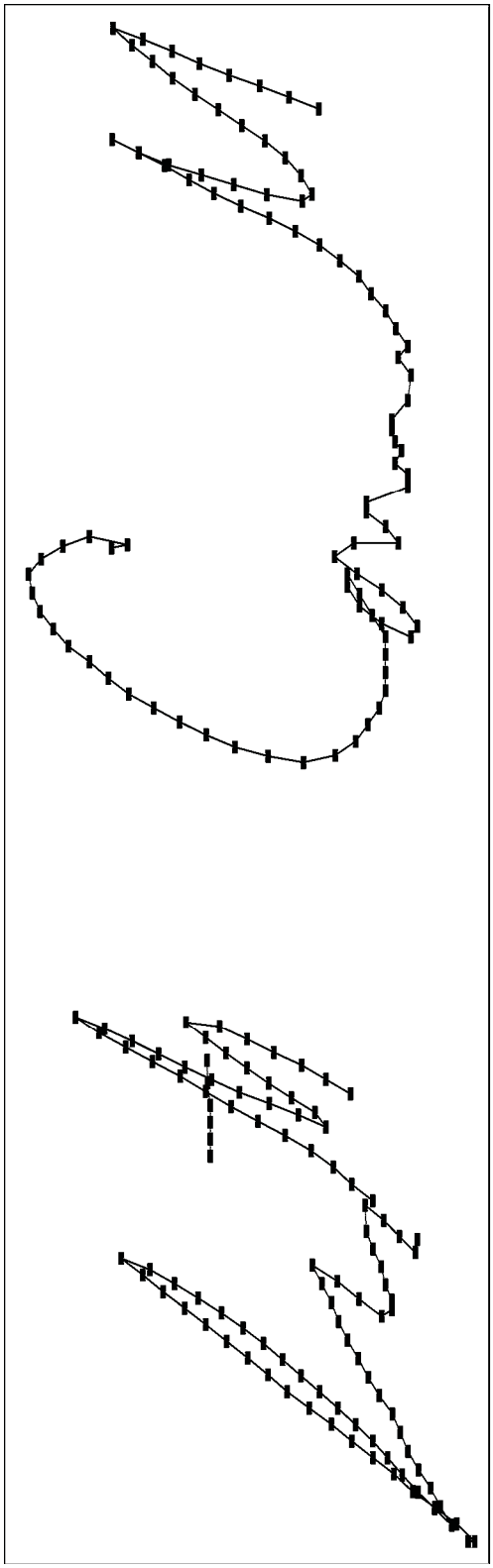

Figure 4.3: Resampling with a spacing of 12 pixels.

Preprocessing	Features	No. of reference signatures	Type of forgeries	Equal Error Rate Min.	Avg.	Max.
smoothing resampling (8) stroke concatenation	δx, δy $\sin\alpha$, $\cos\alpha$	3 5	random skilled random skilled	3% 11% 2.7 % 12%	1.9% 11.2% 1.1% 6.6%	5.5% 16% 5.8% 16%
smoothing resampling (8) stroke concatenation size normalization	δx, δy $\sin\alpha$, $\cos\alpha$	3 5	random skilled random skilled	5.2% 11% 6% 13%	2.9% 12% 3.9% 12%	6.2% 17% 9.5% 20%

Table 4.7: Equal Error rates for different preprocessing and different number of reference signatures.

the use of writer-dependent thresholds. The algorithms to derive the thresholds were explained in Section 3.5. Evaluating all possible combinations of selecting reference signatures is not feasible. Therefore, the reference signatures are drawn randomly from the available data and evaluated and this process is repeated 20 times.

4.4.1 Common Threshold

Three different methods to combine the dissimilarity values from the comparison of the test signature with the reference set are investigated, the minimum, the average and the maximum dissimilarity value as described in Section 3.5. In Table 4.7 the results using a common threshold are shown. These results are obtained using DB1. It can be seen that the minimum value yields the best error rates. Using the minimum value, only the most similar reference signature is involved in the decision making process.

The common threshold is found by testing multiple thresholds in a suitable range.

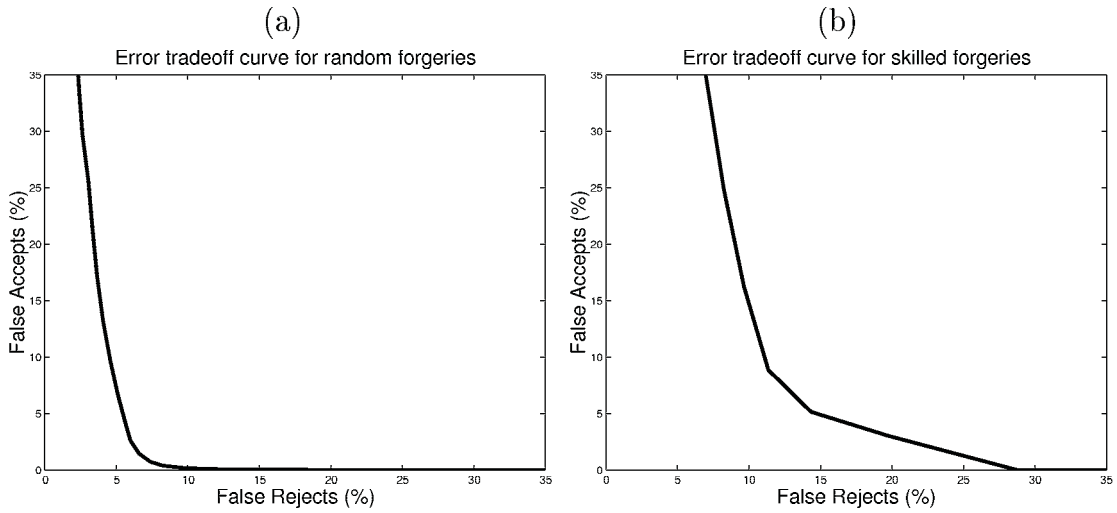

Figure 4.4: Error tradeoff curves for minimum distance without size normalization.

The range is deduced from the analysis of the range of all scores as described in Section 3.5.2. In Table 4.7 the equal error rates are reported. More information can be extracted from the error tradeoff curve and the graph of the error rates with respect to the threshold. The error tradeoff curve depicts the false accept rate with respect to the false reject rate. As expected, lowering the false accept rate results in a higher false reject rate and vice versa. The requirements of an application must choose its operating point which fixes the resulting error rates. Figure 4.4 (a) shows the error tradeoff curve for random forgeries whereas Figure 4.4 (b) shows the error tradeoff curve for skilled forgeries.

Figure 4.5 shows the error rates with respect to the threshold value. It can be seen that the false accept rate increases at a faster rate than the decrease in the false reject rate. This effectively means that a lower false accept rate only slightly increases the false reject rate. This information can also be inferred from the slopes of the error

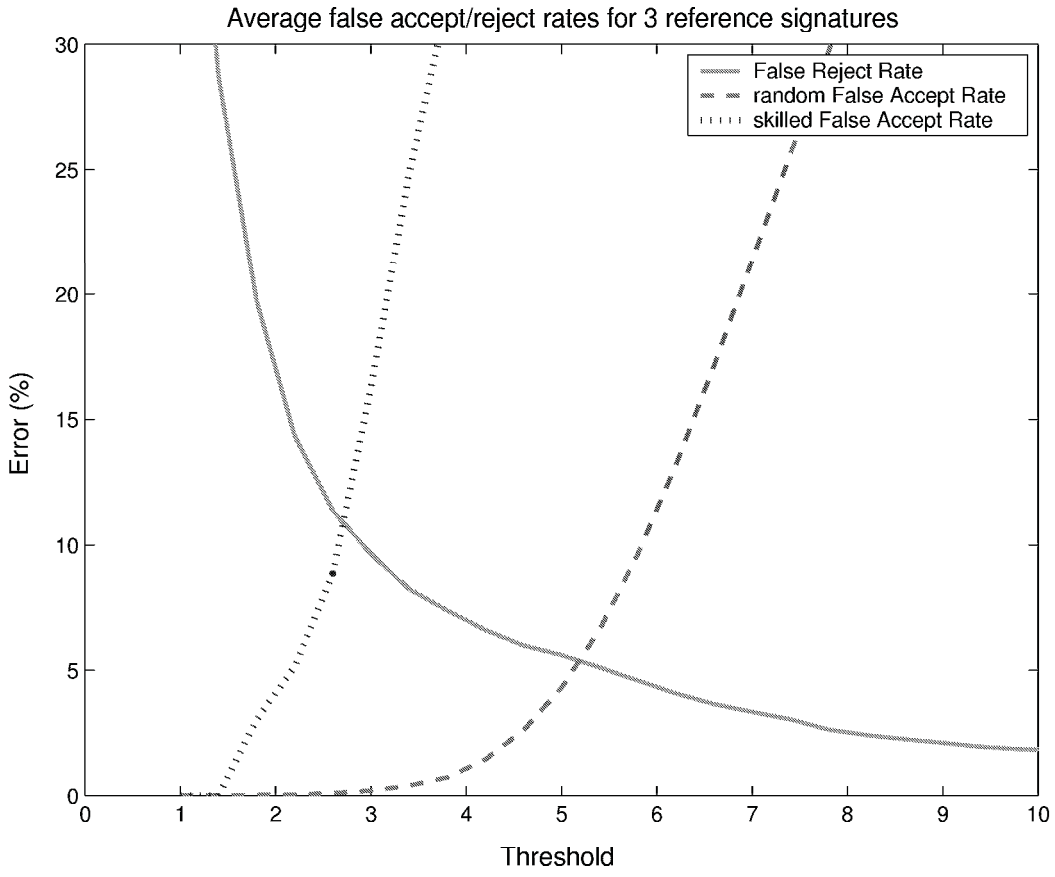

Figure 4.5: Error rates as a function of the threshold value.

Preprocessing	Features	No. of reference signatures	Type of forgeries	FRR	FAR
smooting resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$	3	random skilled	5.6% 11.3%	4.3% 8.9%
smooting resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed	3	random skilled	5.4% 11%	3.8% 10.6%
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed	3	random skilled	3.3% 7.9%	2.7% 10.3%
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed at critical points	3	random skilled	3.2% 3.3%	3.5% 4.7%
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed at critical points	3	random skilled	3.5 % 9.2%	3.1% 5.3%

Table 4.8: Equal error rates using a common threshold. The results are obtained with dataset DB2.

tradeoff curves. Additional error tradeoff curves for other feature sets can be found in appendix A.4.

For the evaluation of the various speed features, only the minimum distance is considered. Table 4.8 shows the results. The results reported are obtained using database DB2, therefore the results for the best spatial feature subset are repeated. Normalizing the speed with the overall average speed of the signature has a similar effect as size normalization for spatial features. Since the speed is extracted before size normalization, the speed feature depends on the size of the signature. If a person signs in different sizes, it is likely that the overall signing time is longer for a bigger signature, so the relative speed at a given point should be the same. If the application

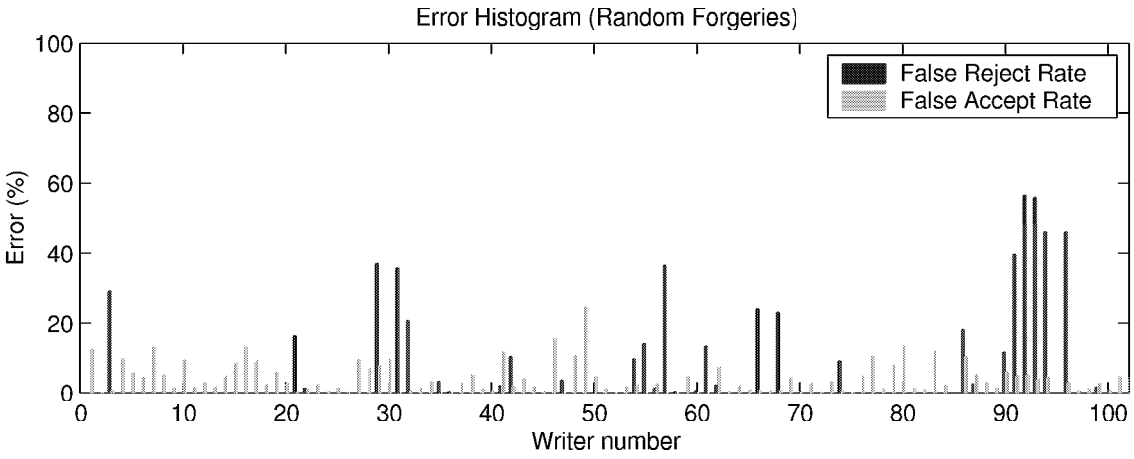

Figure 4.6: Individual error rates for each writer.

requires size normalization, a normalized speed feature should be used.

Figure 4.6 shows the error rates for each writer for the common threshold using the feature subset consisting of δx, δy, $\sin\alpha$ and $\cos\alpha$. It is interesting to see that a few writers have very high error rates. False reject rate and false accept rate are displayed. The next section describes the use of writer specific thresholds which addresses this problem.

4.4.2 Writer-Dependent Threshold Selection

Table 4.9 shows the results for the same datasets as in the previous section using writer-dependent thresholds. The values of the individual thresholds are found empirically. The same range of potential thresholds is evaluated and the best (i.e., the one yielding equal values for the false accept rate and false reject rate) threshold is chosen. This result is "ideal" it reflects the best rates that can be achieved. The goal of every automatic threshold selection method must be to come as close as possible

Preprocessing	Features	No. of reference signatures	Type of forgeries	FRR	FAR
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$	3	random skilled random skilled	1.7% 0.5%	1.6% 0.4%
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed	3	random skilled	1.5 % 0.5%	1.6% 0.4%
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed	3	random skilled	1.3% 1.2%	1.2% 0.6%
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed at critical points	3	random skilled	0.6% n/a	0.6% n/a
smoothing resampling (8) stroke concatenation	δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed at critical points	3	random skilled	1.5% 1.15%	1.4% 0%

Table 4.9: Optimal results for writer-dependent thresholds.

to these values. The low error rates for the skilled forgeries can be explained by the small amount of data. With only three skilled forgeries per writer, the probability of finding a good separating reference set is quite high.

The three methods chosen to calculate the thresholds are the minimum, maximum and average dissimilarity values between the signatures of the reference plus a constant. The constant depends on the feature set and must be determined empirically. Table 4.10 shows the results using the different approaches with the spatial feature set. It can be noted that the minimum value again yields the best error rate. The database used to find the best method is DB1.

Features	Threshold selection method	FRR	random FAR	skilled FAR
δx, δy, $\sin\alpha$, $\cos\alpha$	Min + offset	3.5%	0.35%	9.8%
	Avg + offset	5.1%	4.4%	11.9%
	Max + offset	3.2%	6.7%	8.2%

Table 4.10: Error rates for automatic writer-dependent threshold selection. The results were obtained with dataset DB1.

Features	Threshold selection method	FRR	random FAR	skilled FAR
δx, δy, $\sin\alpha$, $\cos\alpha$	Min + offset	5.3%	2.7%	2.9%
δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed	Min + offset	4.0%	1.3%	10.1%
δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed	Min + offset	3.5%	1.3%	16.9%
δx, δy $\sin\alpha$, $\cos\alpha$ absolute speed at critical points	Min + offset	2.8%	1.6%	n/a
δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed at critical points	Min + offset	3.9%	1.1%	7.5%

Table 4.11: Error rates for automatic writer-dependent threshold selection incorporating speed features. The results are obtained with dataset DB2.

Table 4.11 shows the results for the feature sets incorporating the speed. These results are obtained with dataset DB2. The global component of the threshold, which is combined with the offset calculated for each writer, can be altered to change the operating point of the system accordingly to the needs of the application. A lower global component results in a lower false accept rate but a higher false reject rate. A large number of thresholds have to be fully evaluated to be able to compare the performance of the system using different speed features. The results shown in Table 4.11 are difficult to compare, since the changes in the error rates as a function of the global component cannot be predicted. The results however show, that the error rates are generally lower than those when using a common threshold. Results using five reference signatures can be found in Appendix A.3. Figure 4.7 shows the error using the minimum dissimilarity value as the global threshold. When compared to Figure 4.6 it is obvious that the same writers still have high error rates. The writer-dependent threshold seems to work best for people whose error rates are already low using the common threshold. This may be due to the impact of the global threshold, which is derived from the data of all the writers. Using the minimum dissimilarity scaled by a constant factor may work better for unconventional writers.

4.5 Summary

The best individual feature set without temporal information contains the local features δx, δy, $\cos x$ and $\cos y$. When comparing the test signature to the reference set, the minimum dissimilarity value results in the lowest error rates. The equal er-

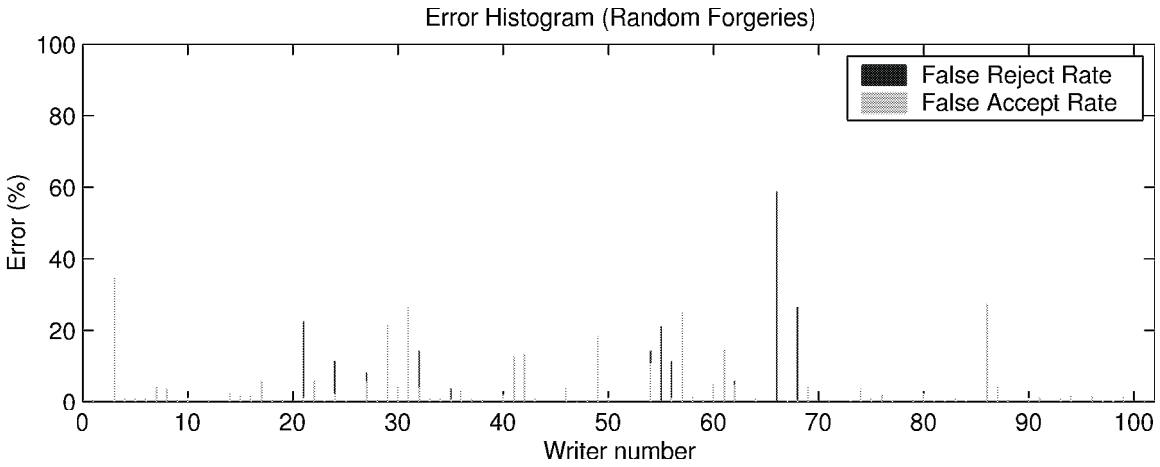

Figure 4.7: Individual error rates for each writer.

ror rates using a common threshold for all writers is around 5%, the best achievable error rates with writer-dependent thresholds are 5.2% false reject rate and 2.3% false accept rate. It is not feasible to derive the individual threshold exclusively from the enrollment data, therefore a method that combines the information from the database with the individual reference set has been used.

Incorporating the speed into the feature set improves the error rates to around 3% for a common threshold and 2.8% false reject rate and 1.6% false accept rate using writer-dependent thresholds.

Chapter 5

Conclusions

A system for on-line signature verification has been implemented. For verification, the user must provide a set of reference signatures to enroll into the system. The signatures are then preprocessed and a set of features are extracted. To verify a test signature, the same preprocessing and feature extraction processes are applied. The test signature is then matched to all the reference signatures. The method used to match each pair of signatures is based on string matching. The dissimilarity values obtained by the matching is then compared to a threshold to decide whether the signature is genuine or a forgery. Different methods to combine the dissimilarity values have been investigated and the minimum value shows the best results. For threshold selection two choices are investigated: a common threshold for all the writers and a writer-dependent threshold for each writer.

The best results for a common threshold are obtained with the feature set consisting of the local features δx, δy, $\sin\alpha$, $\cos\alpha$, the normalized speed between all sampling points and the number of strokes as a global feature. The error rates for

a common threshold are 3.3% false accepts and 2.7% false rejects. Writer-dependent thresholds are computed from the reference signatures. All the reference signatures are matched with each other. The best feature set for writer-dependent thresholds consists of the absolute speed between critical points as the speed feature. Using the minimum difference value plus a feature set dependent offset results in 2.8% false accepts and 1.6% false rejects.

The verification results obtained by us are comparable with prior results reported in the literature (see Section 2). Furthermore, the threshold which is chosen in this system (such that the false reject and false accept rates are approximately equal) can be adjusted to match the demands of the application.

5.1 Related Issues

5.1.1 Security

An application using signature verification must save the signatures and/or its extracted features obtained in the enrollment process. Often the test signature must be sent to a verification server over a network. Whenever biometric data is saved or transmitted, high security standards have to be met. Stolen passwords or pin numbers can be invalidated and replaced, but this is not possible for biometric information. One of the advantages of on-line methods is that the user must be present and actively participate in signing. This provides some protection against forgeries. It is unlikely that on-line signature verification will be used for e-commerce applications,

where manipulation of the hardware cannot be controlled. The enrollment data and the test signature should always be encrypted before transmitting and storage.

5.2 Future Work

It is still an open question as to how the reference set should be updated. The signature of an individual usually changes over time, so a deterioration of the verification rates can be expected if the reference set remains fixed. One possibility would be to ask the user to periodically provide new reference signatures. This would also insure that no forged signatures are used for updates. An automatic update system would be most comfortable for the user. Here the choice of which reference signature should be updated must be made. A simple choice would be to always replace the oldest signature. More sophisticated methods that incorporate the use of dissimilarity values to the other references should be taken into consideration.

Our signature database does not contain any real forgery data. It is still unsure how skilled forgeries can be collected. The results would be more valuable if true forgeries that imitate the shape of the original signature would be available. Additionally, more signatures must be collected. The current test database, consisting of signatures from 102 writers, is at most representative of an application that is running within a small company. Larger companies or applications that use signature verification for their clients have a much larger signature database and the scalability of our system needs to be investigated.

Appendices

Appendix A

Additional Tables and Figures

A.1 Distribution of strokes per signature

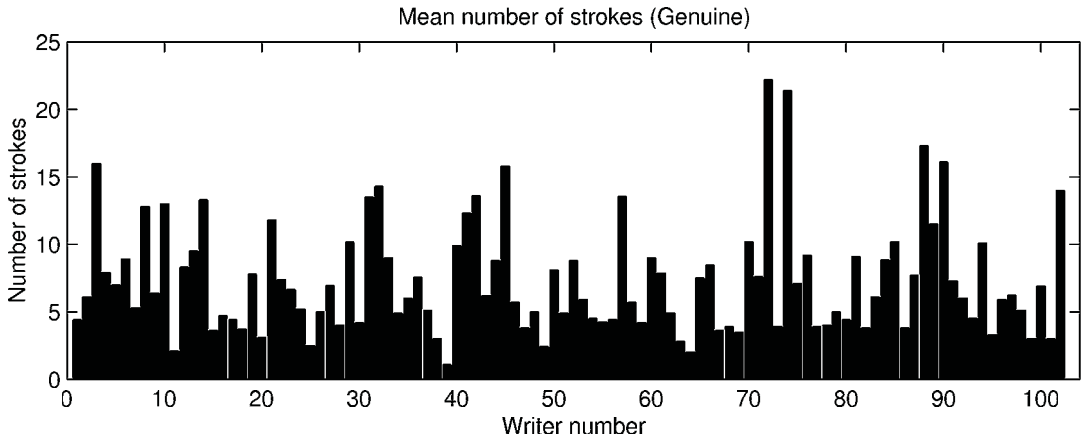

Figure A.1: Number of strokes per signature.

Minimum number of strokes	Maximum number of strokes	Minimum deviation	Maximum deviation	Average deviation
1	31	0	6.5	0.9

Table A.1: Minimum and maximum number of strokes and deviations.

Figure A.1 shows the average number of strokes in the signature of every individual in our database. The minimum number of strokes in a signature is 1, the maximum is 31. The deviation of the number of strokes is different for every individual. The minimum deviation for an individual is 0, the maximum deviation in the number of strokes for an individual is 6.5. This information is summarized in Table A.1.

A.2 Distribution of dissimilarity values

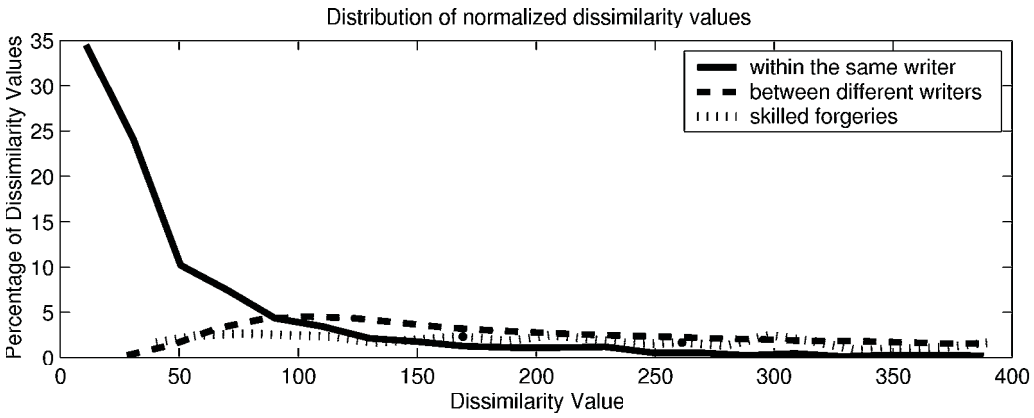

Figure A.2: Distribution of dissimilarity values for the feature subset consisting of δx, y, $\sin\alpha$ and $\cos\alpha$.

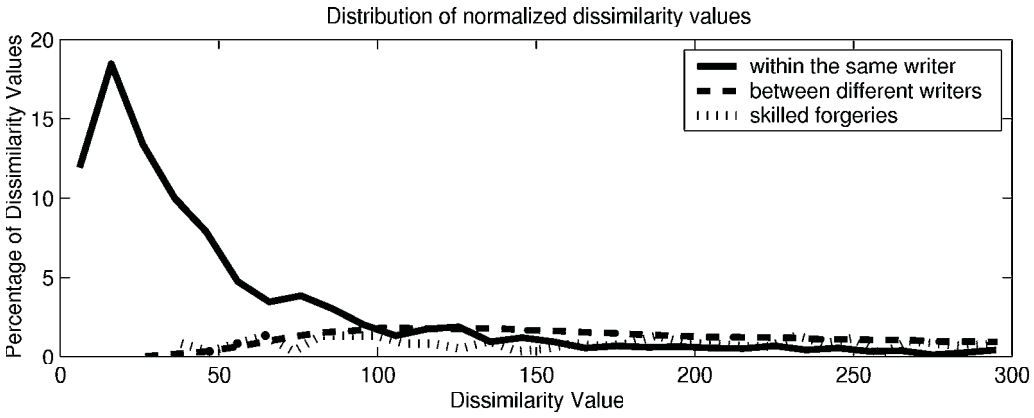

Figure A.3: Distribution of dissimilarity values for the feature subset consisting of δx, δy, y, $\sin\alpha$ and $\cos\alpha$.

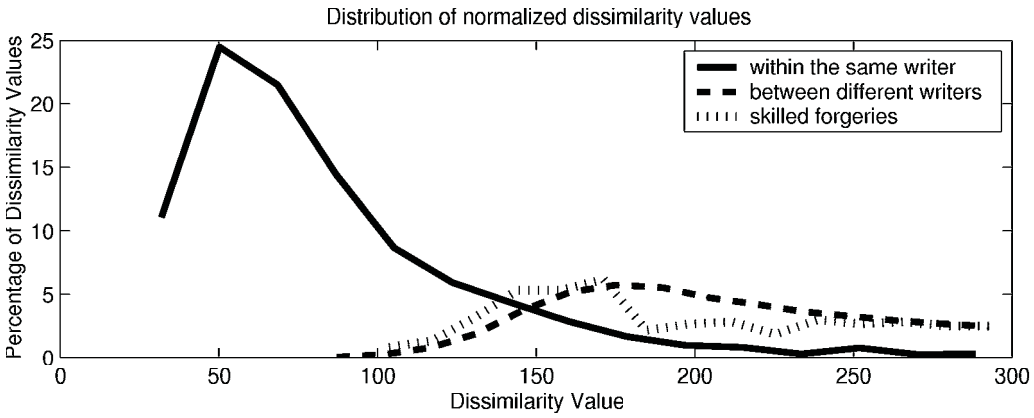

Figure A.4: Distribution of dissimilarity values for the feature subset consisting of δx, y and curvature.

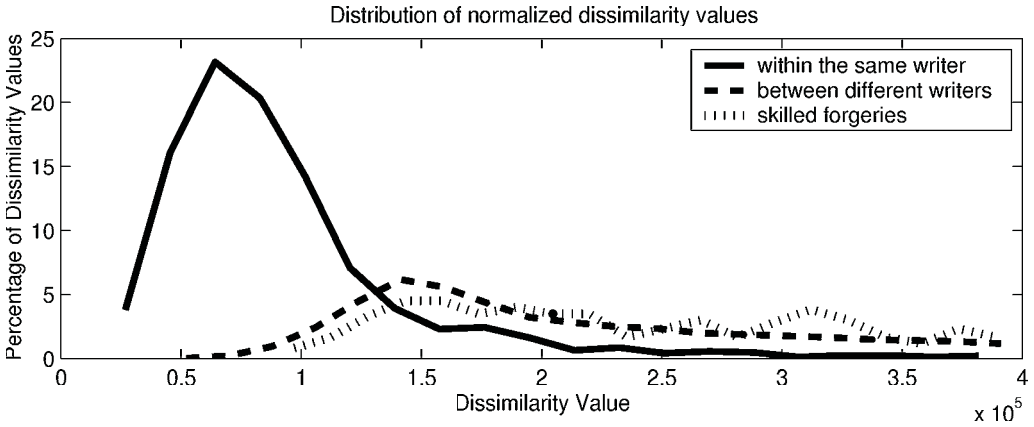

Figure A.5: Distribution of dissimilarity values for the feature subset consisting of the image features.

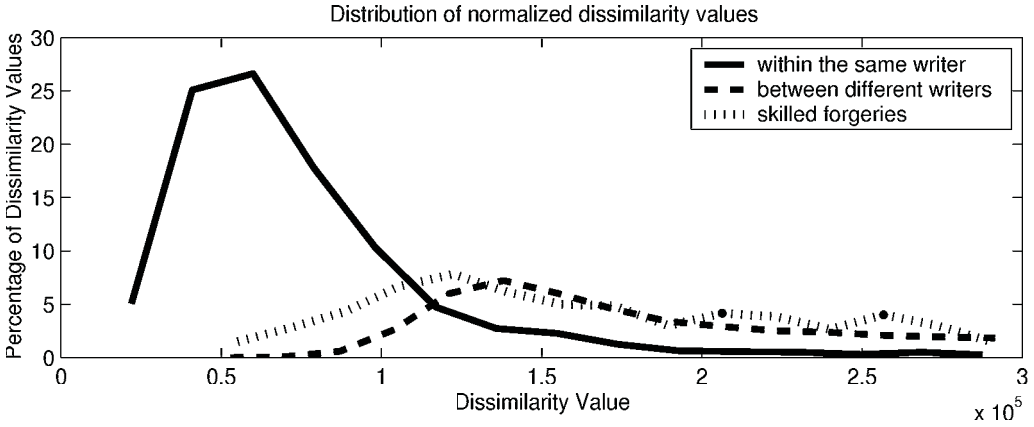

Figure A.6: Distribution of dissimilarity values for the feature subset consisting of the δx, y, $\sin\alpha$, $\cos\alpha$ and image features.

A.3 Additional Results

Spacing	Type of forgeries	FRR	FAR
4	random skilled	0.7% 1.2%	0.8% 1%
8	random skilled	0.22% 0.25%	0.25% 0.16%
12	random skilled	0.2% 0.25%	0.2% 0.6%

Table A.2: Results for different resampling spacings for the feature set δx, δy, $\sin\alpha$, $\cos\alpha$ using writer-dependent threshold for all users. The results are obtained with DB1.

Features	No. of references	Type of forgeries	common TH		writer-dep. TH	
			FRR	FAR	FRR	FAR
δx, δy, $\sin\alpha$, $\cos\alpha$	5	random skilled	3.5% 7.9%	3.5% 10.3%	0.8% 0.2%	0.8% 0%
δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed	5	random skilled	3.6% 7.1%	2.5% 12.2%	0.9% 0.37%	0.8% 0.33%
δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed	5	random skilled	2.1% 9.9%	3.5% 2.7%	0.12% 0.1 %	0.12% 0.05%
δx, δy, $\sin\alpha$, $\cos\alpha$, absolute speed at critical points	5	random skilled	2.5% 7.2%	1.1% 5.9%	0.7% 0.6%	0.7% 0.5%
δx, δy, $\sin\alpha$, $\cos\alpha$, normalized speed at critical points	5	random skilled	2.3% 7%	4.0% 6.7%	0.14% 0.13%	0.15% 0%

Table A.3: Equal error rates for common and writer-dependent thresholds using five reference signatures. The results are obtained with DB2.

A.4 Error Tradeoff Curves

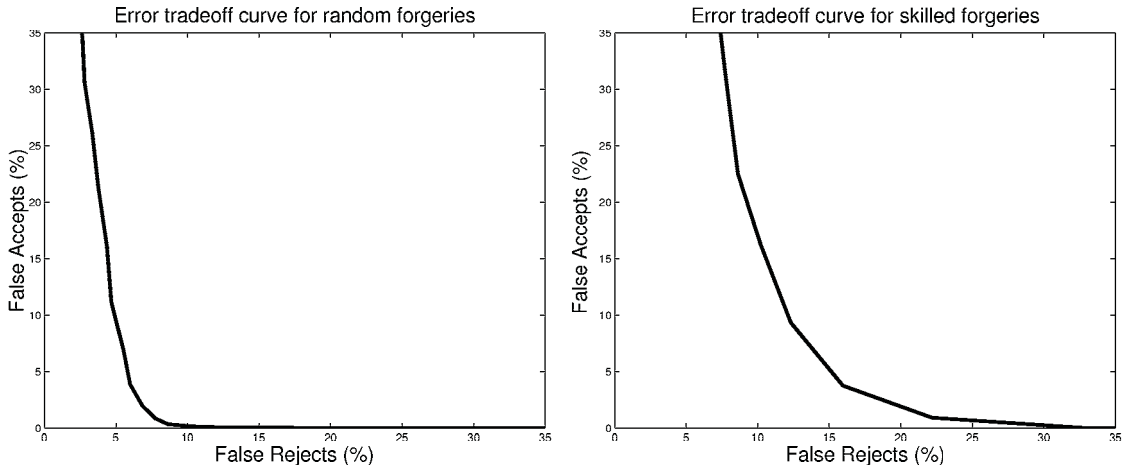

Figure A.7: Error tradeoff curves for the feature subset consisting of δx, δy, $\sin\alpha$, $\cos\alpha$ and absolute speed between sampling points.

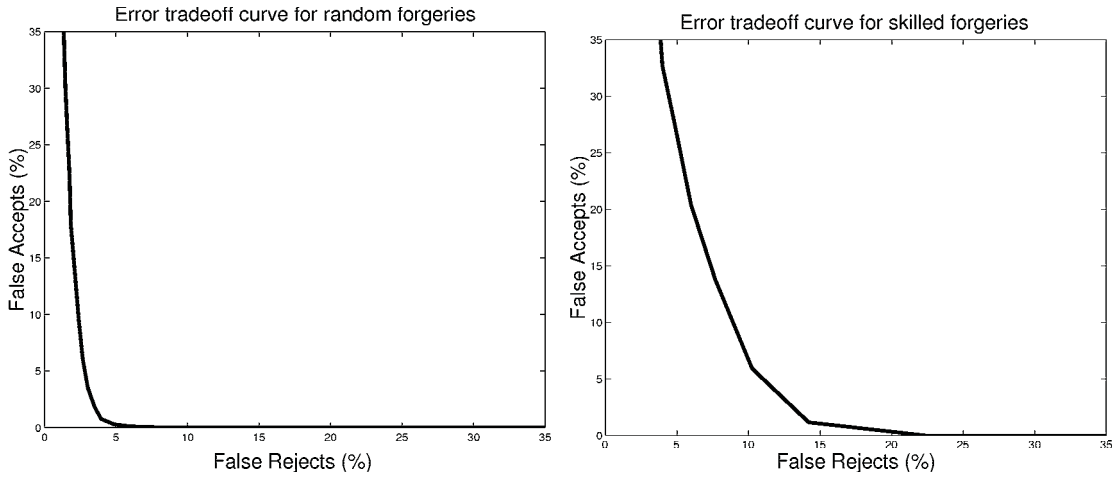

Figure A.8: Error tradeoff curves for the feature subset consisting of δx, δy, $\sin\alpha$, $\cos\alpha$ and normalized speed between sampling points.

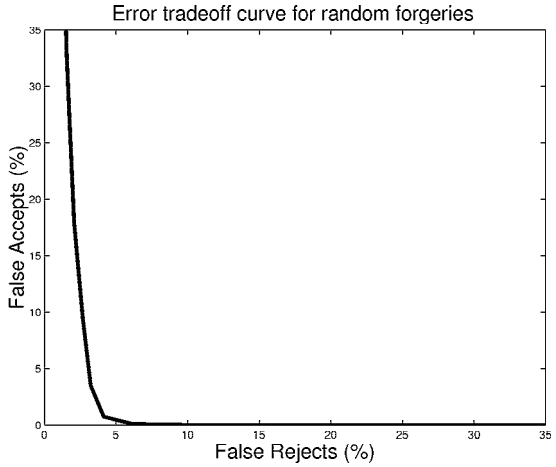

Figure A.9: Error tradeoff curve for the feature subset consisting of δx, δy, $\sin\alpha$, $\cos\alpha$ and absolute speed between critical points.

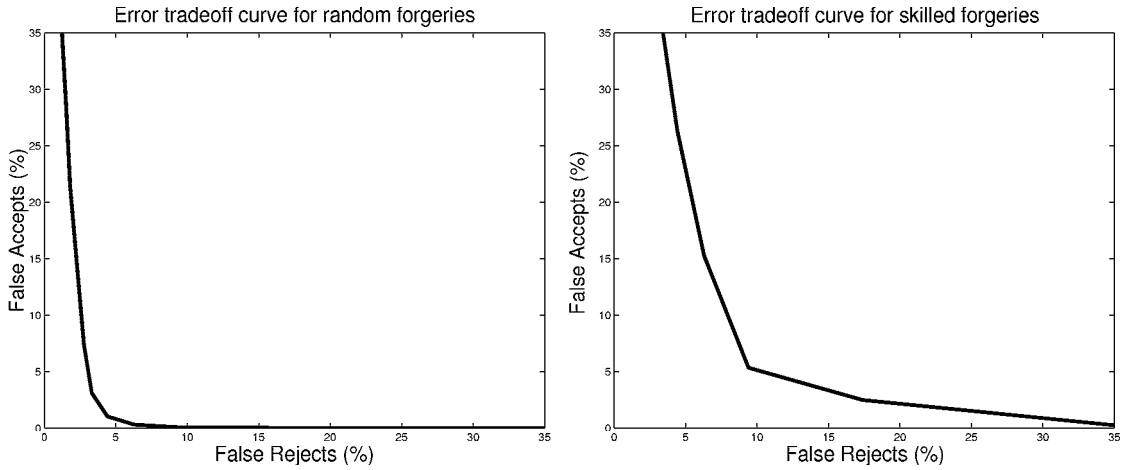

Figure A.10: Error tradeoff curve for the feature subset consisting of δx, δy, $\sin\alpha$, $\cos\alpha$ and normalized speed between critical points.

Bibliography

Bibliography

[1] A. T. Cross Company. [Online] Available *http://www.cross.com/cross/home.asp?*, May 25 2000.

[2] V. S. Nalwa, "Automatic on-line signature verification," *Proceedings of the IEEE*, vol. 85, pp. 215–239, February 1997.

[3] K. Huang and H. Yan, "On-line signature verification based on dynamic segmentation and global and local matching," *Optical Engineering*, vol. 34, pp. 3480–3487, December 1995.

[4] L. L. Lee *et al.*, "Reliable on-line human signature verification systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 643–647, June 1996.

[5] S. H. Kim *et al.*, "Applying personalized weights to a feature set for on-line signature verification," in *Proceedings of the Intl. Conference on Document Analysis and Recognition*, vol. 2, pp. 882–885, 1995.

[6] B. Wirtz, "Stroke-based time warping for signature verification," in *Proceedings of the Intl. Conference on Document Analysis and Recognition*, vol. 1, pp. 179–182, 1995.

[7] G. Gupta and R. Joyce, "A study of shape in dynamic handwritten signature verification," tech. rep., James Cook University of North Queensland, Computer Science Dept., 1997.

[8] H. Dullink *et al.*, "Implementing a dsp kernel for online dynamic handwritten signature verification using the tms320 dsp family." [Online] Available *http://www.ti.com/sc/docs/psheets/abstract/apps/spra304.htm*, December 1995.

[9] G. Gupta and A. McCabe, "A review of dynamic handwritten signature verification." http://cay.cs.ju.edu.com/ãlan/Work/HSV-Lit_rev.html, September 1997.

[10] L. L. Lee, *On-line systems for Human Signature Verification.* PhD thesis, Cornell University, 1992.

[11] L. Yang *et al.*, "Application of hidden markov models for signature verification," *Pattern Recognition*, vol. 28, no. 2, pp. 161–170, 1995.

[12] J. Dolfing *et al.*, "On-line signature verification with hidden markov models," in *Intl. Conference on Pattern Recognition*, vol. 2, pp. 1309–1312, IEEE, 1998.

[13] G. Rigoll and A. Kosmala, "A systematic comparison between on-line and off-line methods for signature verification with hidden markov models," in *Proceedings of the Intl. Conference on Pattern Recognition*, vol. 2, pp. 1755–1757, 1998.

[14] Q.-Z. Wu *et al.*, "On-line signature verification using lpc cepstrum and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 27, pp. 148–153, February 1997.

[15] Q.-Z. Wu *et al.*, "On-line signature verification based on logarithmic spectrum," *Pattern Recognition*, vol. 31, no. 12, pp. 1865–1871, 1998.

[16] T. Hastie *et al.*, "A model for signature verification," tech. rep., AT&T Bell Laboratories, February 1992.

[17] Cyber SIGN, Inc. [Online] Available *http://www.cybersign.com/techoverview.htm*, June 29 1999.

[18] Wacom Technology Co. [Online] Available *http://www.wacom.com/productinfo/pl300.html*, May 25 2000.

[19] DATAVISION Corporation. [Online] Available *http://www.datavisionimage.com/sigrec.htm*, June 29 1999.

[20] PenOp, Inc. [Online] Available *http://www.penop.com/penop/penop.nsf/htmlmedia/index.html*, July 19 1999.

[21] LCI SMARTpen, Inc. [Online] Available *http://www.smartpen.net/site/index.htm*, July 12 1999.

[22] Quintet. [Online] Available *http://www.quintetusa.com/pro.htm*, September 10 1999.

[23] S. D. Connell and A. K. Jain, "Template-based online character recognition." To appear in Pattern Recognition, 2000.

[24] S. D. Connell and A. K. Jain, "Learning prototypes for on-line handwritten digits," in *Proceedings of the 14th International Conference on Pattern Recognition*, Brisbane 1998.

[25] S. D. Connell, "A comparison of hidden markov model features for the recognition of cursive handwriting," Masters Thesis MSU-CPS-96-31, Michigan State University, Computer Sciences Dept., September 1996.