

***Handwritten Signature Verification:  
A Hidden Markov Model Approach***

**Pierre le Riche**

December 2000



Thesis presented in partial fulfilment  
of the requirements for the degree of  
**Master of Electronic Engineering**  
at the  
**University of Stellenbosch**

Supervisors: **J.A. du Preez**  
**B.M. Herbst**

## **Declaration**

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and has not previously in its entirety or in part been submitted at any university for a degree.

**Signature:**

**Date:** September 30, 2000

## Abstract

Handwritten signature verification (HSV) is the process through which handwritten signatures are analysed in an attempt to determine whether the person who made the signature is who he claims to be.

Banks and other financial institutions lose billions of rands annually to cheque fraud and other crimes that are preventable with the aid of good signature verification techniques. Unfortunately, the volume of cheques that are processed precludes a thorough HSV process done in the traditional manner by human operators.

It is the aim of this research to investigate new methods to compare signatures automatically, to eventually speed up the HSV process and improve on the accuracy of existing systems.

The new technology that is investigated is the use of the so-called *hidden Markov models* (HMMs). It is only quite recently that the computing power has become commonly available to make the real-time use of HMMs in pattern recognition a possibility.

Two demonstration programs, *SigGrab* and *SecuriCheque*, have been developed that make use of this technology, and show excellent improvements over other techniques and competing products. HSV accuracies in excess of 99% can be attained.

# Opsomming

Handgeskrewe handtekening verifikasie (HHV) is die proses waardeur handgeskrewe handtekeninge ondersoek word in 'n poging om te bevestig of die persoon wat die handtekening gemaak het werklik is wie hy voorgee om te wees.

Banke en ander finansiële instansies verloor jaarliks biljoene rande aan tjekbedrog en ander misdrywe wat voorkom sou kon word indien goeie metodes van handtekening verifikasie daargestel kon word. Ongelukkig is die volume van tjeks wat hanteer word so groot, dat tradisionele HHV deur menslike operateurs 'n onbegonne taak is.

Dit is die doel van hierdie navorsing om nuwe metodes te ondersoek om handtekeninge outomaties te kan vergelyk en so die HHV proses te bespoedig en ook te verbeter op die akkuraatheid van bestaande stelsels.

Die nuwe tegnologie wat ondersoek is is die gebruik van die sogenaamde *verskuilde Markov modelle* (VMMs). Dit is eers redelik onlangs dat die rekenaar verwerkingskrag algemeen beskikbaar geraak het om die intydse gebruik van VMMs in patroonherkenning prakties moontlik te maak.

Twee demonstrasieprogramme, *SigGrab* en *SecuriCheque*, is ontwikkel wat gebruik maak van hierdie tegnologie en toon uitstekende verbeterings teenoor ander tegnieke en kompeterende produkte. 'n Akkuraatheid van 99% of hoër word tipies verkry.

## Acknowledgements

I would like to thank the following people for their contributions towards this thesis:

- My supervisors, Prof. B.M. Herbst and Prof. J.A. du Preez, for their valuable input, support and encouragement. Without them, this research simply would not have happened.
- Dr. J.G.A. Dolfing, for meeting with us and making available his signature database. Without it, we would have had no common reference to compare our system with others.
- Alan McCabe, for publishing his literature study on HSV on the Internet. It provided valuable pointers to previous work and saved me days of research in the library.
- Pieter Roux, for picking up the bill for some of the hardware, and for concealing his panic when some of the other projects fell behind schedule.
- Everyone else I may have forgotten.

## Keywords

Handwritten Signature Verification

Hidden Markov Models

Dynamic Time-warping

## Acronyms

HSV	-	Handwritten Signature Verification
HMM	-	Hidden Markov Model
DTW	-	Dynamic Time-Warping
FAR	-	False Acceptance Rate
FRR	-	False Rejection Rate
EE	-	Equal Error (Rate)
RMS	-	Root Mean Square
STD	-	Standard Deviation
FFT	-	Fast Fourier Transform
DFT	-	Discrete Fourier Transform

# Contents

1	Introduction	1-1
1.1	What is handwritten signature verification?	1-1
1.1.1	The goal of HSV systems in general	1-3
1.1.2	Stumbling blocks for HSV	1-7
1.2	Previous work & related approaches	1-10
1.3	The purpose of this research	1-13
1.4	Summary	1-14
2	Review Of Current Approaches	2-1
2.1	Statistical Feature Extraction	2-1
2.2	Point-for-Point comparison	2-3
2.2.1	Segmenting Signatures	2-4
2.2.2	Dynamic Time-Warping	2-8
2.3	Summary	2-19
3	Gathering Signature Data	3-1
3.1	Data Capture	3-1
3.2	Pre-processing	3-4
3.2.1	Translation and Scaling	3-7
3.2.2	Rotation	3-10
3.2.3	Further Fitting	3-15
3.2.4	Interpolation and Calculated Features	3-20
3.2.5	Dimension Reduction	3-24
3.3	Summary	3-27

4	Hidden Markov Models	4-1
4.1	Identifying shortcomings in other methods	4-2
4.2	Introduction to HMMs	4-5
4.2.1	Obtaining the most likely state sequence	4-8
4.2.2	Obtaining a HMM model	4-10
4.3	HMMs as used in HSV	4-11
4.1	Summary	4-18
5	Experimental Evaluation	5-1
5.1	Performance with J.G.A. Dolfing's Database	5-1
5.2	Performance with In-House Data	5-10
5.2.1	Initial Results with a Static Model	5-12
5.2.2	Optimising the System	5-15
5.3	Summary	5-24
6	Conclusion and Recommendations for Future Work	6-1
Appendix A	HSV Implementation: SigGrab Program	A-1
A.1	Installing SigGrab	A-2
A.2	Capturing Signatures	A-3
A.3	Signature Graphs	A-6
A.4	Exporting Signatures	A-9
A.5	HMM Analysis	A-12
A.5.1	Setting Up and Training a HMM	A-13
A.5.2	Evaluating Signatures with the HMM Model	A-16
A.5.3	Automatic HMM Model Generation	A-18

## List of Figures

- 1.1 Ideal HSV situation: Signatures occupy separate feature spaces
- 1.2 Real HSV situation: Feature spaces overlap
- 1.3 Typical FA & FR curves
- 1.4 Natural variation of signatures over time
- 2.1 Segmentation on pressure
- 2.2 Signature curvature
- 2.3 Segmentation scale factors
- 2.4 Signature after segmentation
- 2.5 Test signature after segmentation]
- 2.6 Sample time-warping grid
- 2.7 Sample DTW cost matrix, costs at each element indicated
- 2.8 DTW cost matrix with cost to point, previous element and route indicated
- 2.9 Plot of signals A and B before DTW
- 2.10 Plot of signals A and B after DTW
- 2.11 2 genuine signatures
- 2.12 X-coordinate signals
- 2.13 X-coordinates after DTW
- 2.14 Best path
- 2.15 DTW between genuine signature and forgery
- 3.1 Wacom ArtZ II graphics tablet
- 3.2 FFT of y-coordinate of sample signature
- 3.3 Signature (x,y) plots
- 3.4 Signature pressure profiles
- 3.5 Signatures after scaling and translation
- 3.6 Pressure after scaling
- 3.7 Signatures after principal axis rotation
- 3.8 Signature with no apparent left-to-right progression
- 3.9 Signatures after average angle rotation
- 3.10 Signatures after average pen direction rotation
- 3.11 Signatures after final fitting
- 3.12 Additional fitting without speed compensation

- 3.13 DTW fitting with speed compensation
- 3.14 Pascal code to determine cubic spline
- 3.15 Correlations between signature features
- 4.1 Segments with high variation
- 4.2 Signature broken down into sequence of groups
- 4.3 Signature sample groups
- 4.4 3-state first order HMM model
- 4.5 5-state first order left-to-right HMM model
- 4.6 Gaussian PDF in one dimension
- 4.7 5-state HMM with duration modelling (3 substates)
- 5.1 Sample signatures from Dolfing's database
- 5.2 Problematic signatures
- 5.3 Typical accuracy graphs for Dolfing's database
- 5.4 SecuriCheque application
- 5.5 Sample signatures
- 5.6 Results with first 15 signatures as training set
- 5.7 Signature scores vs. duration after training
- 5.8 System accuracy vs. duration after training
- 5.9 Accuracy vs. number of states (15 training signatures)
- 5.10 Accuracy vs. state count plots for various training set sizes
- 5.11 Accuracy vs. state count plots on the same graph
- 5.12 Accuracy for various numbers of substates
- A.1 SigGrab Setup Screen
- A.2 SigGrab Main Screen
- A.3 Signer Search Screen
- A.4 Signature Capture Screen
- A.5 Sample Signature Graph
- A.6 Scaled, rotated and translated signatures
- A.7 Sample DTW grid
- A.8 Sample signature export file
- A.9 Graphical export function
- A.10 Sample graphical signature export
- A.11 HMM Model setup window
- A.12 Projection of pen angles onto (X,Z) and (Y,Z) planes

A.13 Eval All Results

A.14 Automatic HMM generation

# Chapter 1

## Introduction

### 1.1 What is handwritten signature verification?

Handwritten signature verification (HSV) is the process of verifying a person's identity by means of his or her handwritten signature. This is of particular importance for financial institutions where clients' identities have to be verified in a quick and unobtrusive way. Until the advent of the computer age, banks were confined to verifying signatures manually. In fact, most banks still verify cheques by hand. While humans may become reasonably successful at verifying signatures (after some practice, and even then it is a time-consuming process), the sheer volume of transactions make the use of humans prohibitively expensive.<sup>1</sup> Often, only cheques above a certain amount are checked for validity, while forgeries below the cut-off amount will be accepted with no verification whatsoever.

While there are more reliable ways of identifying a person, few give such a good mix between cost, reliability and acceptability to the client as a HSV system. The following table shows some common on-line biometric identification techniques, and rates their cost to implement, reliability and acceptability to the client (from 1 to 5 stars, with 5 stars being the best):

---

<sup>1</sup> It is estimated that more than 4 million cheques are processed in South Africa every day.

Biometric	Cost to Implement	Reliability	Acceptability
Iris Scan	*	*****	*
Retina Scan	*	*****	*
Fingerprint	**	*****	***
Hand Geometry	***	****	****
Voice Recognition	****	***	*****
Signature	****	****	****
Face Recognition	***	**	*****

While a retina and fingerprint scanner are arguably the most reliable, people are reluctant to stare into the infra-red beam for fear of eye damage, and the criminal connotation of fingerprints make people uneasy about their use. Some people are initially concerned about the fact that their signatures are stored in a HSV system, for fear that a criminal might gain access to it, but once reassured of confidentiality people become more receptive. The cost of implementation includes the cost of a controlling computer. The first three systems require expensive external hardware, while the bulk of the additional cost of a voice recognition system and HSV system is the software.<sup>2</sup>

Some of these methods are so-called *passive methods*, meaning that no action is required for identification. Iris scanning, for instance, only requires the presence of the individual in front of the camera. Active methods, like HSV, require a gesture from the individual. Such methods indicate not only the presence of the individual, but also his or her active participation (and co-operation). Some state laws require a conscious action for a contract to be legally binding – here a retina or iris scan would not be suitable, but a handwritten signature would be.

The advantage that all biometric verification systems have above traditional systems like PIN numbers, is that biometric systems identify the individual while other systems simply match the pattern. A static signature may be forged by simply copying from an old document, however the dynamics of the original signature are hidden to the forger. This includes the signing rhythm or speed, the pressure applied to the pen

<sup>2</sup> The most feature-rich commercial graphics tablet series currently available is the Intuos line from Wacom. The most suitable model for HSV, the Intuos A6, currently retails for \$179 (R1200).

tip, as well as the way in which the pen is held. These dynamics are unique to the individual and are hard to forge.

### 1.1.1 The goal of HSV systems in general

Verifying handwritten signatures is not as easy as it may first seem. Scientists in the 1960s predicted that voice recognition would become a reality in a couple of years. It is now 40 years later and a system capable of interpreting spoken language (and discerning identity) as well as a human is still some way off. While signatures are indeed less complex than spoken language, it is still a daunting task to get a computer to approach the accuracy of a human expert when it comes to detecting forgeries.<sup>3</sup>

In a typical signature verification system, features are collected from the test signature and are then compared to the features or model of the genuine signatures. In the ideal case the features of genuine signatures and forgeries would be mutually exclusive. If this were the case, then the problem would be simple indeed:

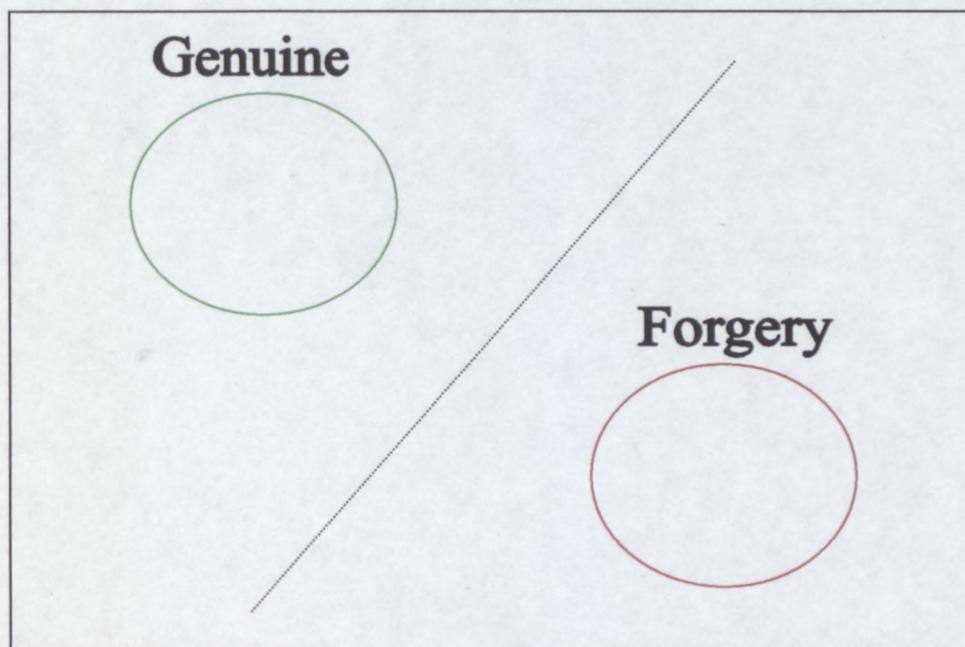


Figure 1.1: Ideal HSV situation: Genuine signatures and forgeries occupy separate feature spaces

<sup>3</sup> Given that the expert has unlimited time available, and that both the computer and human expert only has access to the 2D image of the signature. Computers are much more competitive if the dynamic information is available.

The problem would simplify to selecting an appropriate boundary between features. Depending on which side of the boundary the features lie, the signature would be either a forgery or a genuine signature. In reality the situation looks more like this:

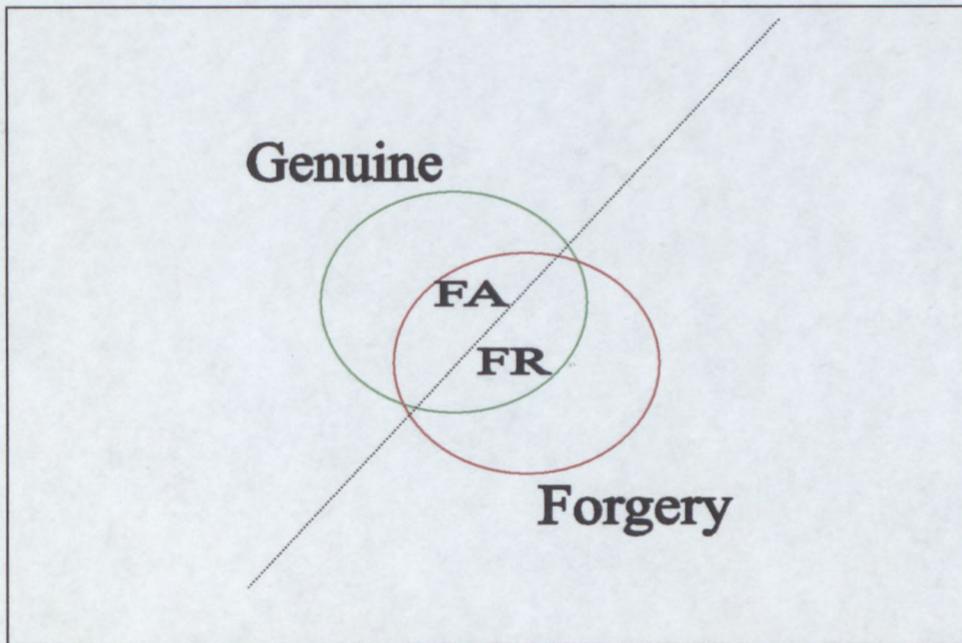


Figure 1.2: Real HSV situation: Feature spaces of genuine signatures and forgeries overlap

The features of genuine signatures and forgeries overlap in the feature-space, and no matter where the decision boundary is placed there will always be some signatures that are misidentified. Two kinds of errors may occur:

- False Acceptances (FA): These occur when forgeries go undetected. The percentage of forgeries that go undetected is called the false acceptance rate (FAR).
- False Rejections (FR): These occur when a genuine signature is mistakenly identified as a forgery. The percentage of genuine signatures that are rejected by a system is called the false rejection rate (FRR).

Apart from optimising the system to keep the overlap as small as possible, attention also has to be paid to the choice of the decision boundary. The decision boundary

determines the ratio of the FAR to FRR. It is clear from the previous figure that a shift of the boundary to decrease the FAR will increase the FRR, and vice versa. This holds true for most real world situations:

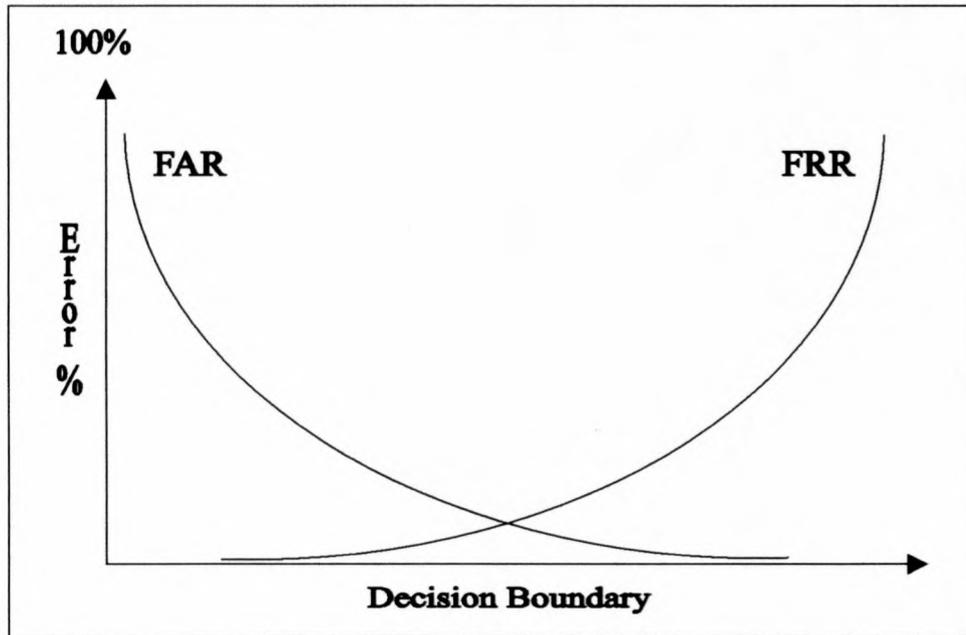


Figure 1.3: Typical FA & FR Curves

If the decision threshold is slackened so as to allow more genuine signatures through then more forgeries will also be accepted. Conversely, if the system is very strict (to reject more forgeries), then more genuine signatures will also be rejected. This is what gives the FAR-FRR curve its characteristic form. The point where the FAR and FRR curves intersect is commonly called the equal error (EE) point. This is often used as a measure of the performance of the verification system, although the shapes of the curves are usually of more practical importance.

The EE point is rarely used in a practical system as the decision boundary. The application dictates where the boundary should be situated. In some situations a large FAR may be tolerated in exchange for a small FRR, or vice versa. Banks for instance cannot risk alienating their clients by rejecting genuine signatures, and so accept a higher FAR.<sup>4</sup> In an access system to a high-security laboratory for instance, the FAR

<sup>4</sup> Banks are naturally not very forthcoming about their losses, but it is estimated that credit card fraud in the USA runs into billions every year.

should be kept as low as possible, since a user can simply sign again if the initial signature is rejected.

The aim of the research conducted was to improve on previous systems by lowering the error rates, possibly making HSV usable in situations where it was previously not a feasible option. Assume for the moment that a HSV system installed in a bank has a 1% FRR and 1% FAR, and it is given that about 1 in 10000 signatures are forgeries. A quick calculation shows that only about 1 in 100 signatures that are rejected are actual forgeries. This system would still burden the bank clerks, and puts the seemingly excellent percentages into perspective. There is always room for improvement...

In the evaluation of a HSV system, and particularly when comparing the performance of two systems, it is important to note the type and quality of the forgeries used to test the system. Forgeries are commonly divided into two categories: Skilled forgeries and zero-effort forgeries:

- Skilled forgeries: These signatures are produced by a person other than the original signer, but someone who has had access to the signer's signature. This category ranges from signatures made by a person that was actively coached by the original signer to produce a forgery to the best of his/her ability, to signatures made by a person that has merely seen the original. Forgeries that have been traced off an original can be seen as a skilled forgery.
- Zero-effort forgeries: These signatures are made by a person who has never seen the original, but may know the name of the person whose signature is being forged.

In the category of skilled forgeries, there are many gradients. Clearly a signature made by a person that was coached should be superior to one made by a person that has only seen the original. In citations of the performance of HSV systems, many advertisers claim a low FAR and FRR with skilled forgeries, but omit further details. Without a common signature database to test these different systems on, it is extremely difficult to compare them. While there are many speech databases available for use in comparing speech/speaker identification systems, freely available signature

databases are rare. This is understandable due to the risk that signers take in having their signatures published. It could be possible to compile a signature database made by people signing fictitious persons' names. Unfortunately it might not be acceptable as a serious benchmark, since signatures are generally formed over a period of many years. Fictitious signatures, not having this development phase, may not be representative of signatures in general.

Although performance evaluation as described above is essential, the evaluation is not always a true indicator of the performance of the technique since the test signatures often do not adequately represent the population at large. Plamondon and Lorette [18] note that there is a great deal of variability in signatures according to country, age, time, habits, psychological or mental state, and physical and practical situations. Building a test database that is representative of the real-world situation is a daunting task, since it is hard enough to find volunteers to willingly sign 10 or 20 (or even more) times.<sup>5</sup> People are reluctant to have their signature stored on a computer, and feel uncomfortable knowing that someone will practice forging it. For this reason, signature databases are often built up by people from the research facility where the system is developed, and contain few signatures from people that are old, disabled, suffering from a common disease (e.g. arthritis), or poor. Percentages of such people in the population are significant, and these people's signatures are likely to pose the greatest challenge to HSV systems. Not surprisingly, it has been reported that the FAR and FRR are typically higher when submitted to a more representative group.

### **1.1.2 Stumbling blocks for HSV**

No two signatures from the same person are ever the same. In fact, some signature experts note that if two signatures of the same person written on paper were identical they could be considered forgery by tracing. Successive signatures by the same person will differ, both locally and globally, by orientation and scale.

---

<sup>5</sup> Signatures made during the same session tend to be more closely matched than signatures made over a period of time. Repeated signing sessions are thus required to allow proper evaluation of the system.

The position of the signer (sitting or standing), as well as the conditions under which the signature was made may affect it. A signature written in haste may differ greatly from the signatures used to train the system. Unless signatures signed under similar circumstances were available when the system was set up, the system would have a hard time verifying the individual's identity. When a signature is being written to be used for comparison this can also produce a self-conscious, unnatural signature.

Longer signatures contain more distinguishing features than shorter signatures. Most HSV systems struggle more with shorter signatures than with long ones, since there is less to distinguish between true and forged signatures.

Signatures often undergo subtle change as time goes by. The two genuine signatures below were made exactly one year apart:

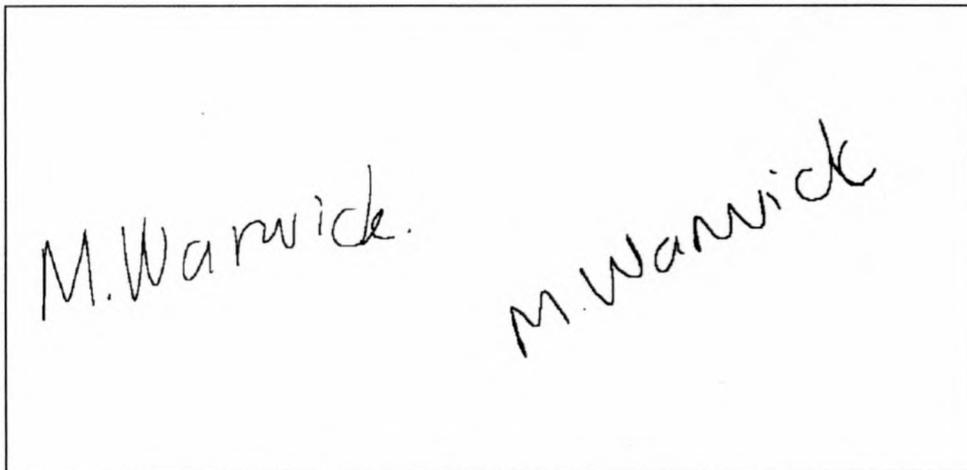


Figure 1.4: Natural variation of signatures over time

Note the more rounded appearance of the initial M in the more recent signature. This would play havoc with a HSV system that uses the curvature in the signature to perform segmentation.

Some signers continually vary between two or more signatures. This makes it very difficult for a system to obtain a model for the person's signature, unless it explicitly supports more than one signature per person.

Unlike many other biometric verification systems like retina-scanning or fingerprints, HSV requires the subject's cooperation in order to be successful. This can also be considered an advantage, depending on the application.

In spite of these problems, it has been suggested that human experts are very good at identifying forgeries, but perhaps not so good at verifying genuine signatures. For example, Herbst and Liu [20] cite references indicating that as high as 25% of genuine signatures were either rejected or classified as no-opinion by trained document examiners while no forgeries were accepted. Untrained personnel were found to accept up to 50% forgeries. Of course, this was with static signatures. If they had had access to the dynamic data, the results might have been better.<sup>6</sup>

Osborn [21] notes that handwriting shows great variation in speed and muscular dexterity. He states that the process of forging a signature, if it is to be successful, involves a double process requiring the forger to not only copy the features of the writing imitated but also to hide the writer's own personal writing characteristics. If the writing is free and rapid it will almost certainly show, when carefully analysed, many of the characteristics of the natural writing of the writer no matter what disguise may have been employed. Indeed, in a dynamic HSV system, the signer is required to imitate not only the form of the signature, but also the speed (which tends to be rapid for most signatures), the pressure exerted and (in some systems) even the way the pen is held. This should bring forward the forger's own characteristics which could be used to expose the forgery.

Osborn further notes that the variations in handwriting are themselves habitual and this is clearly shown in any collection of genuine signatures produced at different times under a great variety of conditions, which when carefully examined, show running through them a marked, unmistakable individuality *even in the manner in which the signatures vary as compared with one another*.

From the automatic HSV system it is thus required to identify those characteristics that are unique to the signature, to note the natural variations present and account for

---

<sup>6</sup> On the other hand, the dynamic data may be difficult for a human to interpret.

them in the model. When comparing a signature it should determine whether the differences between the test signature and the model it has built up occur naturally in the signer's signature, and judge whether it is a genuine signature or a forgery.

## 1.2 Previous work & related approaches

There are many approaches towards verifying identity with HSV. These include extracting discrete information from the signature, neural networks, dynamic time-warping, and recently, hidden Markov models. Some systems even combine two or more of these methods.

Computing power and storage limitations have always been a stumbling block for practical implementations of HSV systems. One of the ideals has been to be able to store the distinguishing characteristics of a person's signature on a magnetic stripe card, negating the need for the signature to be on the back of a credit card (which allows easy forgery). Without knowing what the signature looks like, it would make it very difficult for the credit card thief to produce an accurate forgery. Unfortunately, the storage capacity of standard magnetic stripe cards is severely limited (less than 60 bytes), and most of the space is already in use. This precludes their use for storing a detailed model of a signature. Smartcards may well solve this problem, but the extent of the current investment in magnetic stripe card technology probably means that it will still be a while before credit card companies will adopt a newer technology.

With the growing trend of global networking, the idea of a centralised signature verification database is not far-fetched. A person could sign on a special digitiser tablet and this signature could be passed on electronically for verification by a central computer.

Of course, speed is a great issue when the number of transactions that takes place each day is taken into account. Many of the older HSV techniques rely on very basic features of signatures, because the computing power to be able to extract more intricate information simply was not available at the time. With computing power

constantly increasing, methods that were previously deemed to be too slow are now becoming contenders. One such method is the use of hidden Markov models (HMMs), which has also (amongst many others) found application in the speech processing field.

An important distinction has to be made between static (or off-line) HSV and dynamic (or on-line) HSV. Static HSV comes into play when only the static image of the signature is available. Most of the early work on HSV (early 1970's and before) was concerned with static HSV. Dynamic HSV takes advantage of the signing dynamics of the signature, and thus requires signing on a special digitising tablet. Details like pressure applied to the pen, pen angles, etc. are also often recorded. This allows a much higher accuracy than that obtainable with static HSV. Currently the accuracy figures for static HSV linger in the 85%-90% region, while accuracies in excess of 99% are now becoming possible with dynamic HSV. This thesis will focus on dynamic HSV. For a discussion of static HSV see Plamondon and Lorette [18] and Leclerc and Plamondon [19].

Commercial HSV products are coming onto the market faster than before, and many new systems are being advertised, in particular on the Internet. McCabe [6] has done a recent survey of available products. Some are listed here:

- A product called *PenOp* is being marketed by Peripheral Vision of New York and it is claimed that the software may be used in configuring systems so that the users must login using handwritten signatures.
- Another product called *Sign-On*, it is claimed, allows HSV to be built-in to a variety of widely used software enabling the system to use a handwritten signature instead of a password. It uses, besides the signature image, acceleration, stroke angles, start and stop pressures (if available) and other factors. The signature information can be updated each time a successful verification occurs. The product uses six signatures plus a final verifying signature to build a reference signature. The test signature is now compared with the reference signature resulting in one of three judgements: true, forgery

or ambiguous. The product is claimed to have a 2.5% FRR and a 2.5% FAR, but details of performance evaluation are not available.

- Yet another product is *Cadix ID-007* which is claimed to be suitable for user authentication and which requires a pressure-sensitive pen and tablet for HSV. The Microsoft Windows based software examines the test signature according to three different criteria: the shape of the signature, the speed at which it was written and the pressure of the pen stroke. Verification of a signature with the ID-007 system typically takes less than one second. No details of how it performs are available.
- *Countermatch* is the name of a HSV product from AEA Technology in the UK. The product uses three sample signatures to build a reference signature. It is claimed that the product is suitable for signatures written in any language but no details of techniques used are provided.
- Another UK company, British Technology Group, markets a product called *Kappa*. *Kappa* uses signature shape as well as the timing and rhythm of the signature and claims to use a new high accuracy pattern matching algorithm developed at the University of Kent, but no details were available. It uses a user specific feature set designed for low FRR, but it is not clear how this set is selected. It also provides a *shape only* option that allows paper records of signatures to be computerized. The *Kappa* system has been tested in a public trial at a sub-Post Office where some 8500 signatures were collected. A FRR of 1.8% with one test signature and 0.85% with three test signatures has been reported for individuals that were able to provide a *satisfactory enrolment model*, that is, for people who have a signature that does not require *special measures* for verification. The system identifies at enrolment time those people that are believed to require special measures for verification. It is not known how many individuals were rejected at enrolment time.

### 1.3 The purpose of this research

The purpose of this research is to analyse current HSV techniques, gather information, and attempt to improve on the methods in use. The methods investigated include dynamic time-warping (DTW) and hidden Markov models (HMMs), with emphasis on the latter. The goal is to develop a fully functional and automated HSV system capable of capturing signatures, analysing them, and offering a judgement on their authenticity (given the availability of a set of genuine signatures serving as reference).

Using hidden Markov models (HMMs) is relatively new to the field of HSV. There are very few references to HMMs in the literature – the only well-documented reference that was encountered is the work of Dr. J.G.A Dolfing [5]. This is mostly due to the computational complexity and iterative nature of HMM analysis. Previous generation computers simply did not have the processing power and storage capacity to cope with it.

In HMM analysis a model is obtained for a signer's signature by analysing a set of genuine reference signatures. This model becomes a sequence of statistical distributions that predicts the sequence of samples in a genuine signature. Whenever a signature is evaluated, its actual sequence of samples is compared to the prediction obtained from the model. The better the match between the two, the higher the probability of it being a genuine signature. Of course this is an oversimplification of the process. For a detailed discussion of HMMs turn to Chapter 4.

HMMs require that the input data be *well-conditioned*<sup>7</sup>. If this is the case, then data can be fed into the HMM for training or evaluation *as is*, i.e. the raw sample data is input directly into the HMM. This makes HMM analysis very easy to use once the training and evaluation algorithms are in place.

In Dolfing's work, which is a follow-up of his handwritten text recognition work, signatures are divided into segments and these segments are individually evaluated by use of HMMs. Segmenting at points of low speed (or where the vertical speed

---

<sup>7</sup> For HSV that implies that all signatures should be of similar scale, position and rotation.

changes sign) results in the signature being divided more or less into segments corresponding with the natural strokes in the signature. While this segmentation is necessary for handwritten text recognition to be able to recognise individual letters, its use is questionable in the field of HSV. The HMM training process is usually better at modelling the sequence of samples on its own, than when it is interfered with and forced to accept certain segmentation boundaries.

The HMM analysis in this research differs from previous attempts in that no prior segmentation is performed, additional HMM features like *duration modelling* are added, and improved methods of pre-processing the signatures are also investigated.

The principal aim is to improve on the results obtained with previous approaches, but the practicality of the implementation (e.g. storage and processing power requirements) is also important.

## 1.4 Summary

The problem of being unable to quickly and accurately verify identity by means of handwritten signature has been around for a long time, and will remain a problem far into the future. It is unlikely that the tradition of using handwritten signatures as means of identity verification will be replaced by a more robust method in the near future, especially considering the cost implications for banks and other financial institutions.

With the advent of the computer age, many new avenues have opened up for the automatic analysis of signatures. With ever-increasing processing power, more and more powerful techniques can be employed that may not have been practical (or even possible) previously.

Currently, there are many commercial implementations of HSV available, but from the reported results it is clear that there is still room for improvement in the field. HSV is not as easy as it may at first appear: Genuine signatures from the same person

vary (especially if taken over a period of time) and some signers even have more than one distinct signature. The HSV system must be able to identify these variations and when evaluating a signature to decide whether it is genuine or a forgery, it has to judge whether the observed variations in the signature are normal for the signer or not.

It is the goal of this research to develop a practical implementation of an automatic HSV system with better accuracy than what is currently obtained by competing products. The main thrust of the research is using hidden Markov models as the core of the analysis technique.

## Chapter 2

### Review of Current Approaches

Most HSV techniques involve five phases: data acquisition, pre-processing, feature extraction, a comparison process, and performance evaluation. More advanced systems may add additional phases like an update of the model after a successful verification, etc.

There are two common approaches to HSV. In the first approach, discrete features like averages and totals are calculated for the test signature and these are compared with the expected values for a genuine signature. The second approach assumes that all samples taken from the signatures are important and these are typically compared to the reference signature or model point-for-point.

#### 2.1 Statistical Feature Extraction

Stemming from many older techniques, particularly from static HSV, the extraction of statistical features usually requires much less storage and processing power than current techniques. Example signatures are typically analysed to obtain likely values of these features. Once a signature is subjected to the system for evaluation, these same discrete features are extracted and compared to the average values obtained during the training phase. Some features used in systems like these are:

- Total time taken in writing the signature
- Signature path length: displacement in the x and y directions and the total displacement.
- Path tangent angles: profile of their variation and average or root mean square (RMS) values

- Signature accelerations: variations in horizontal and vertical accelerations, centripetal accelerations, tangential accelerations, total accelerations, as well as their average or RMS values.
- Pen-up time: total pen-up time or the ratio of pen-up time to total time
- Pressure: average pressure or pressure variations

These are just some of the commonly used features. In one of the more than 100 patents covering HSV systems, Parks, Carr and Fox [14] propose in excess of 90 features for comparison.

The advantage of only storing these features is that a reference signature is typically not required. Not only does this save on storage space, but is also more reassuring to the user, since the signature database is less subject to possible criminal use in the event that it falls into the wrong hands.

Without a reference signature, the features that are used are typically rotation and size invariant. When taking averages or summing values, details and variations local to certain parts of the signature are obscured. Consequently a lot of information is lost in the feature extraction phase, which limits the performance of such systems.

There are many ways of comparing these features to obtain a measure of how well signatures match. Lee [15] mentions 5 different approaches, but most systems employ an Euclidean distance classifier in some form or another.

Results reported by Lee for statistical feature extraction varies from an EE rate of 28% right down to 3.8% using 42 features. Forgeries varied from zero-effort to some skilled forgeries.

Crane and Ostrem [16] collected a database of 5220 genuine signatures from 58 people. 648 forgeries were collected from 12 forgers who were allowed to practice the signatures. 3 of the 55 signers were rejected at enrolment time due to very high variance in their signatures. In the experiments 44 features were extracted and signers

were given three attempts before a result was determined. Error rates varied from 0.5% to 3%.

Lam and Kamins [17] studied a HSV system based on a Fourier transform of the signature (after considerable pre-processing). They used the 15 harmonics with the highest frequencies for verification. The system was poorly evaluated, with only one signer and 19 forgers attempting to forge his signature. The reported FRR was 0% and FAR was 2.5%.

There are many other methods that have been reported, but details are often withheld due to the commercial value of innovations in this field. The number of patents granted in this field attests to this fact. The use of neural networks is quite conspicuous in its absence from this list.

## **2.2 Point-for-point comparison**

In point-for-point comparisons, every sample of the test signature is compared to a corresponding sample in the reference signature or model. Whether the signature is deemed as genuine or a forgery depends on how well it fares in this point-for-point comparison. Typically a score is built up by summing the Euclidean distance from each sample to its corresponding sample in the reference. If the total exceeds a certain threshold, it is deemed a forgery, otherwise it is accepted. Since there invariably are differences between genuine signatures from the same person, determining which samples to compare to which becomes a problem. There are various methods used to address this. Some of these methods will be discussed next.

## 2.2.1 Segmenting Signatures

Segmentation of signatures is an old technique that is still often employed. Signatures are divided into smaller segments by preset rules. In some studies, statistics for these segments are extracted and compared to those of the reference (which can be seen as a hybrid between point-for-point and statistical feature extraction). Segmentation reduces the problem of which samples to compare to which to some extent, but the choice of segmentation boundaries still requires some thought. It is most often seen in systems that owe their origin to handwriting recognition systems (handwritten text to ASCII text converters), since handwriting recognition systems need to know the boundaries of letters to be able to recognise them.

Most systems segment at points of high curvature<sup>1</sup>, since these most often correspond to the end points of natural strokes in the signature. The pressure is often also used to provide further segmentation. The signature below is segmented with a pressure threshold of 20% (of maximum pressure). A hysteresis of 10% is built in to improve stability:

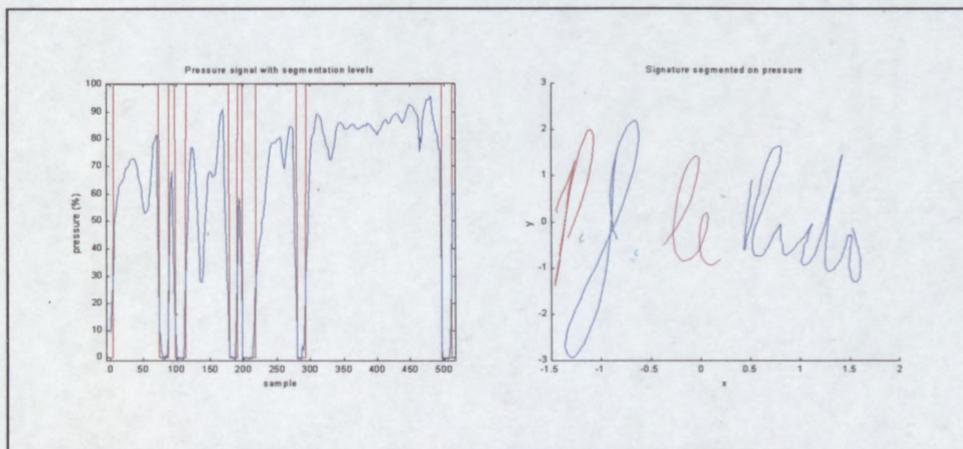


Figure 2.1: Segmentation on pressure

<sup>1</sup> There are various definitions of curvature as used in HSV. The second derivative of the signature curve is most often used in mathematical textbooks, and is also the definition used here.

Experimentation has shown that the pressure signal is of little use in segmentation except for dividing the signature into the separate words (i.e. identifying the pen-ups). The pressure profile seems too erratic to use in any further segmentation. Further segmentation is done with the curvature signal. The curvature signal exhibits peaks at the turning points in the signature, which correspond well to the edges of letters. The curvature signal for the signature above is shown below:

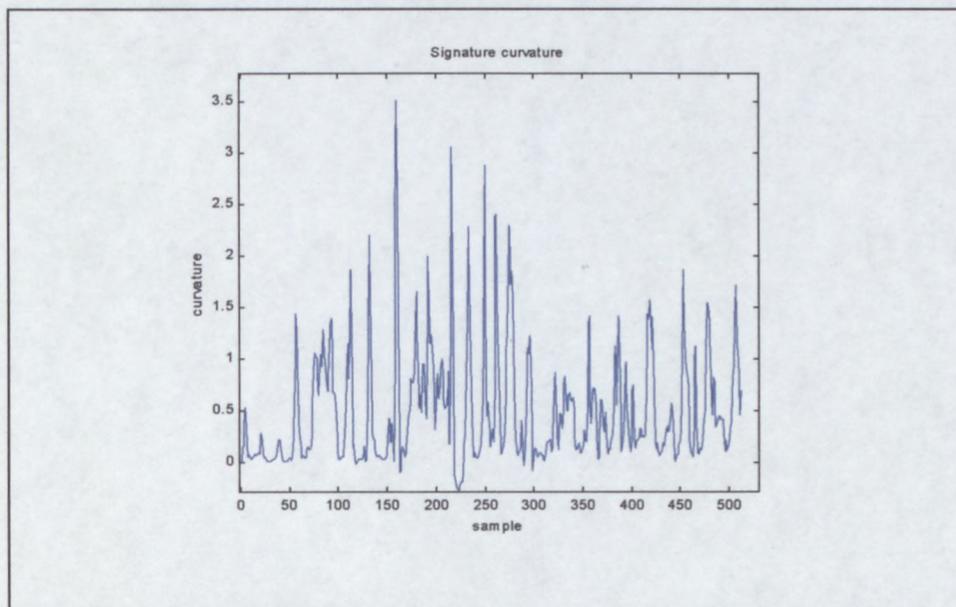


Figure 2.2: Signature curvature

From the curvature graph it is clear that there are many peaks and thus candidates for segment breaks. Deciding at which peaks to segment is the next problem. Picking the highest peaks makes sense, but many of the highest peaks are bunched together. If it is decided to segment the signature into 10 parts and the 10 highest peaks are used, the middle part of the signature will be heavily segmented and the last part not at all. To remedy this situation, the curvature is scaled by the distance towards the (currently) closest segmentation point, and the signature is segmented iteratively. Some segmentation scaling plots for this signature are shown in the next figure:

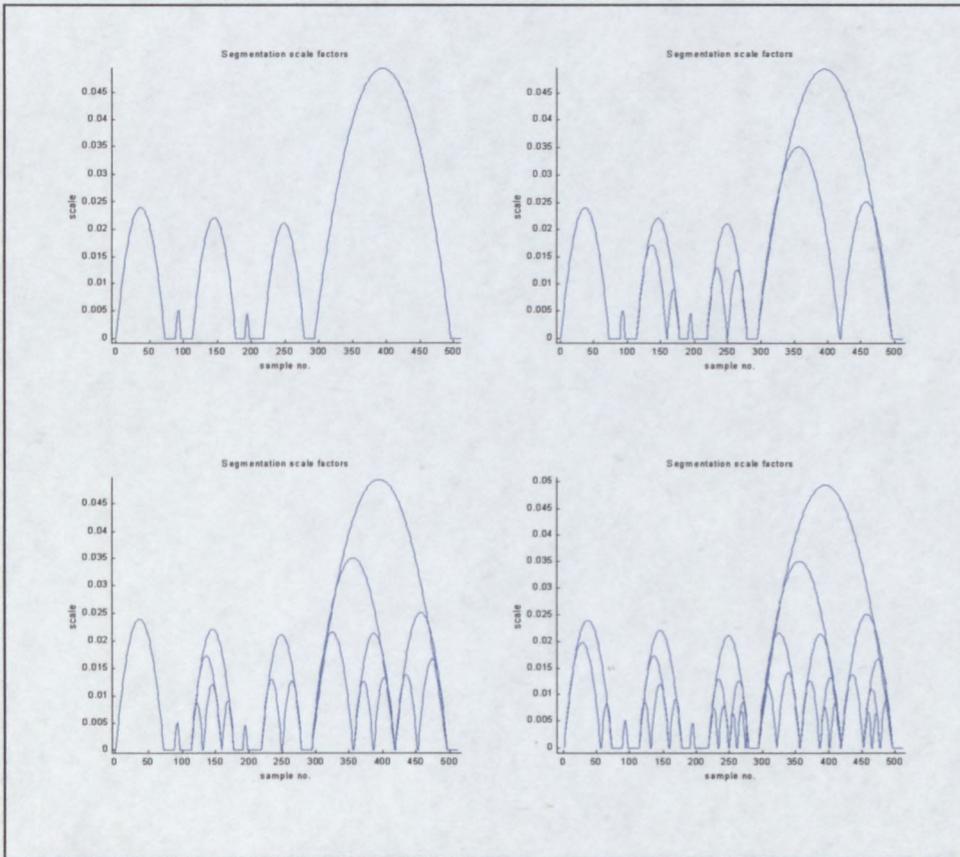


Figure 2.3: Segmentation scale factors

In this case the curvature is scaled by a quadratic function of the distance to the nearest segment break. The plots show how the scaling factors are adjusted as more and more segment breaks are chosen. The top-left plot shows the scaling function after the pressure segmentation. The top-right plot shows the scaling function after a further 3 segment breaks were made. The bottom-right plot shows the final scaling function after segmentation has been completed.

Scaling the curvature in this way distributes the segments more evenly than otherwise.

The plot below shows the final signature after segmentation has been completed:

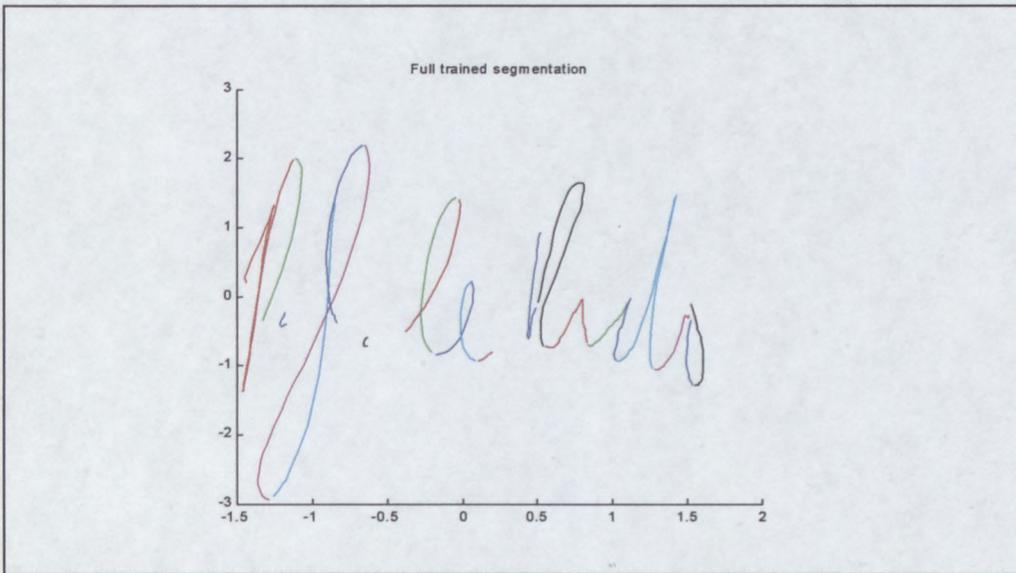


Figure 2.4: Signature after segmentation

The signature is segmented more or less into strokes, with the exception of the first part of the 'P', where the strokes were too short and were subsequently not segmented. Compare this to the segmentation of another signature by the same person:

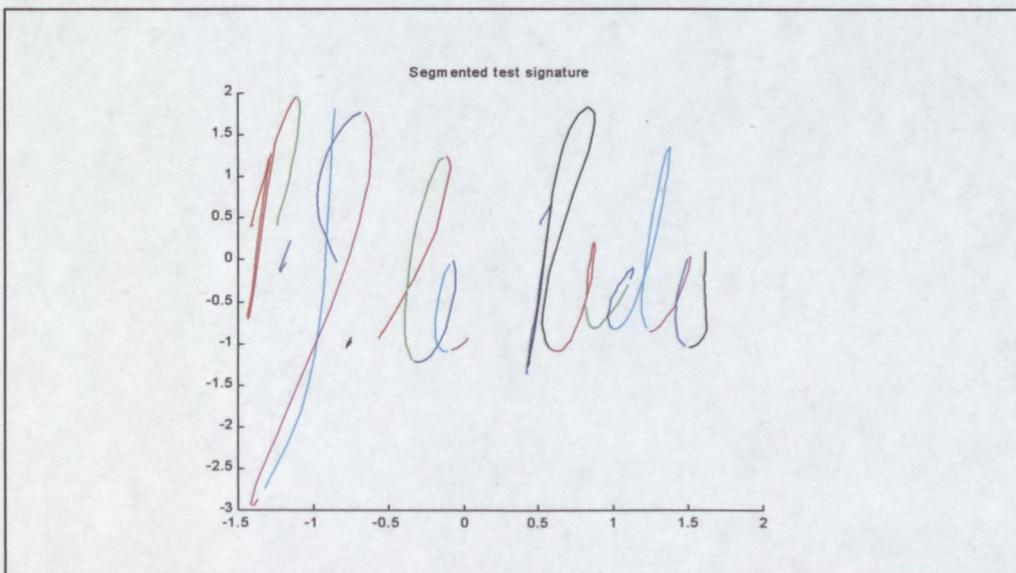


Figure 2.5: Test signature after segmentation

The signatures were segmented in the same areas, which is reassuring, but the physical break points of the segments were somewhat shifted. Look at the bottom part

of the ‘J’ for example. Since the break points are shifted, the endpoints and initial/final angles are of little use in comparing the segments. Furthermore, most segments approach a straight line, which in itself contains very little information. This conspires to make it difficult to do an accurate comparison of signatures based on their segments when segmentation is performed in this way.

While other methods of segmentation have been reported in the literature, claiming better results, “hard” segmentation still seems counter-productive. Dolfing [5] reports an EE of 1.9% with hard segmentation subsequently fed into a hidden markov model (HMM). Results obtained in a later chapter show that allowing the HMM to do its own “soft” segmentation gives superior results.

### **2.2.2 Dynamic Time-warping**

Dynamic time-warping (DTW) literally means that the time-axes are warped to obtain the best point-for-point fit between two signals. The duration of the samples in the signals are varied to allow a better match between the two signals. The following rules apply:

- Every sample in the original signal must occur in the warped signal.
- All samples must be in their original order.
- The start and end-points must remain the same.

When using DTW in HSV techniques, the test signature is typically warped to fit as best possible on the reference signature. After the test signature has been warped onto the reference, the two signatures are compared point-for-point and a score obtained. The score is then compared to a preset threshold to determine whether it should be deemed a forgery or genuine signature. The advantage of using DTW before doing a point-for-point comparison is that different length signatures can be compared, and it is now possible to compensate for the natural variations in the lengths of the segments comprising the signature.

The first step in DTW is the generation of a cost matrix. What is needed is a matrix with elements that are the *cost* between all samples from the first signature to all samples in the second signature. This cost measure can be any arbitrary function, but typically the Euclidean distance between the samples are used.<sup>2</sup> This matrix is used in determining what sequence of samples in the signatures should be matched up with each other in time to obtain a closer match.

From here the two signatures are called signature A and signature B. It is immaterial which signature is the reference and which is the test, the process works the same. For this discussion the origin of the cost matrix is placed in the bottom-left corner with signature A along the horizontal axis and signature B along the vertical. Coordinates are indicated in (horizontal,vertical) format with (1,1) being bottom-left.

Assume the length of the signature A is  $x$  samples, and signature B is  $y$  samples long. The size of the cost matrix will be  $(x,y)$  samples. The  $(i,j)$ th element will be the *cost* (typically the Euclidean distance) between sample  $i$  in signature A and sample  $j$  in signature B. (We are only interested in the sample points for now, points in-between will later be obtained by interpolation.)

If an arbitrary path is traced through the cost matrix from the bottom left (1,1) to the top-right  $(x,y)$  and a signature is constructed by taking the samples from signature A that correspond with the columns, we get a time-warped version of signature A. Similarly, taking the samples from signature B that correspond with the rows we get a time-warped version of signature B for the same path.

Obtaining the best point-for-point match between the two signatures reduces to finding the path from (1,1) to  $(x,y)$  that minimizes the sum of all the costs along its path. Note that the path traced from (1,1) to  $(x,y)$  can never go left or down, since that would violate the restriction that samples must remain in their original order. This also means that there is now a finite number of possible paths, given by:

---

<sup>2</sup> The features comprising the sample vector are usually scaled independently to add weighting factors to each feature before calculating the euclidean distance.

$$N = \sum_{s=0}^{\min(x,y)} \frac{(x+y-s)!}{(x-s)!(y-s)!s!}$$

Fortunately we don't need to investigate all of them, and the determination of this path is actually quite straight-forward.

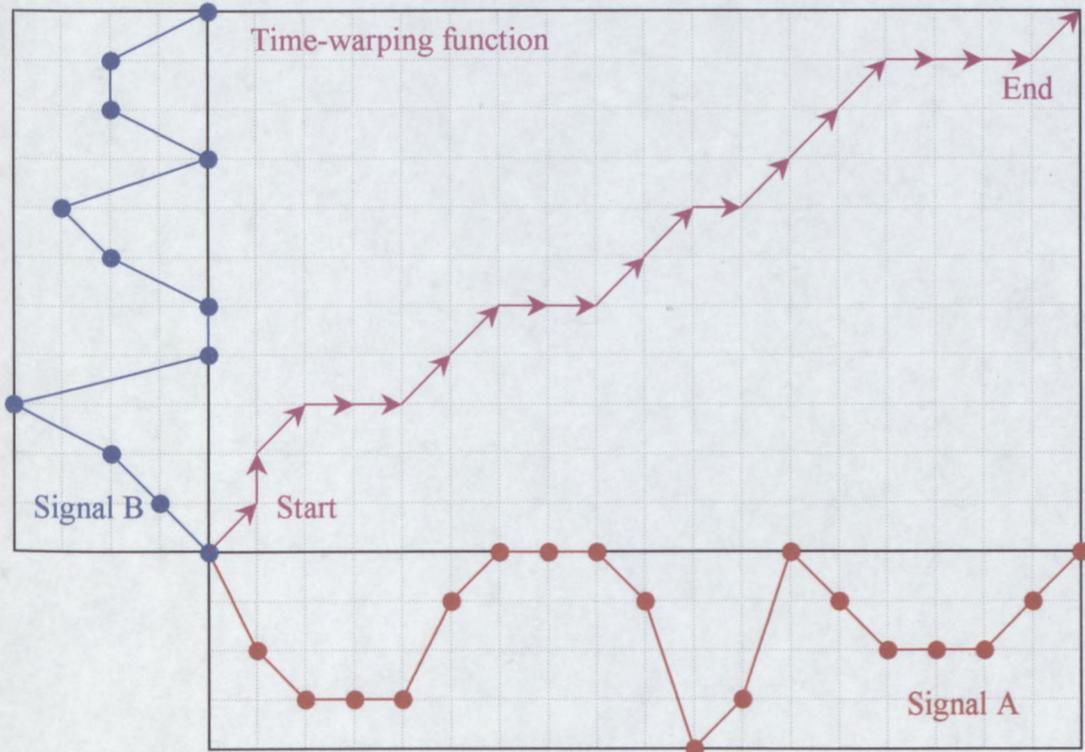


Figure 2.6: Sample time-warping grid

Starting at (1,1) and working row-by-row from left-to-right the following is calculated for each element in the cost matrix:

- The *previous* element with the lowest total cost to it from (1,1). To adhere to the rule of keeping the order of samples intact for both signatures, this previous element can only be the element to the left, below, or below-left. Calculating row-by-row and from left-to-right ensures that the values for the possible previous elements will be known.
- The lowest total cost to the current element from (1,1). This is calculated by taking the total cost to the previous element (determined above) and adding the cost at the current element.

After this process is complete, we have 3 values for each element of the cost matrix:

- the cost/distance between the corresponding points in the signatures
- the previous element
- the sum of all costs to this element

We now need to extract the best path linking (1,1) with (x,y). Since the previous element for all elements have been determined, determining the cheapest path can be done by starting at (x,y) and following the trail of previous elements back to (1,1). The total cost to element (x,y) is usually taken as a measure of how well this path matched up the two signatures. Taking samples from signature A from the column indices of the path, and samples from signature B from the row indices, we obtain the warped signatures with the best fit.

To illustrate the discussion, consider the following example: We have two signals that we wish to time-warp so that they fit on each other as best possible. Suppose they are as follows

Signal A						
Time	1	2	3	4	5	6
Value	1	1	2	5	2	1

Signal B					
Time	1	2	3	4	5
Value	3	7	5	1	1

In this case we use the Euclidean distance<sup>3</sup> as measure of how much samples in the two signatures differ, thus the cost matrix will look like this:

<sup>3</sup> For a single dimension, as in this case, it is simply the absolute value of the difference between the two sample values.

<b>Signal B samples</b>	5	0	0	1	4	1	0
	4	0	0	1	4	1	0
	3	4	4	3	0	3	4
	2	6	6	5	2	5	6
	1	2	2	1	2	1	2
	1	2	3	4	5	6	
<b>Signal A samples</b>							

Figure 2.7: Sample DTW cost matrix, costs at each element indicated

Next we determine for each element the previous element, as well as the total cost to the element. We start at coordinate (1,1) in the cost matrix and work row by row so that (6,5) is calculated last. For example: (2,1) can only be reached from (1,1), so the total cost to (2,1) is the total cost to (1,1) (which is 2) plus the cost at (2,1) which is 2. The total cost to point (2,1) is thus 4. Similarly (1,2) can only be reached from (1,1) as well, and in similar manner we determine the total cost to point (1,2) as 8. Point (2,2) can be reached from (1,1), (2,1) or (1,2). The cost from (1,2) is equal to  $6+6=12$ , compared to the costs from (1,1) and (2,1) which are both equal to  $2+6=8$ . Subsequently we pick (1,1) as the previous element for (2,2) and the total cost to (2,2) is thus 8 (although (2,1) would also be a valid choice). We continue in this manner and calculate the rest of the elements in the cost matrix, until it looks like this:

<b>Signal B samples</b>	5	↓12	↙12	↙13	↓15	↓9	↙8
	4	↓12	↙12	↓12	↓11	↙8	←8
	3	↓12	↙12	↙11	↓7	↙10	←14
	2	↓8	↙8	↙9	↙7	↙12	↙14
	1	2	←4	←5	←7	←8	←10
	1	2	3	4	5	6	
<b>Signal A samples</b>							

Figure 2.8: DTW Cost matrix with cost to point, previous element and cheapest route indicated

The cost to each cell is indicated, as well as the direction to the previous element. Obtaining the best path (which is indicated in orange) simply requires traversing from (6,5) back to (1,1) by following the directions through previous elements.

From the best path obtained from the DTW process, the two signals are warped to produce the following new signals:

Time-warped Signals							
Time	1	2	3	4	5	6	7
New A Value	1	1	2	5	5	2	1
New B Value	3	3	3	7	5	1	1

If both the old and new graphs of the two signals are drawn, it can be seen more clearly what has happened:

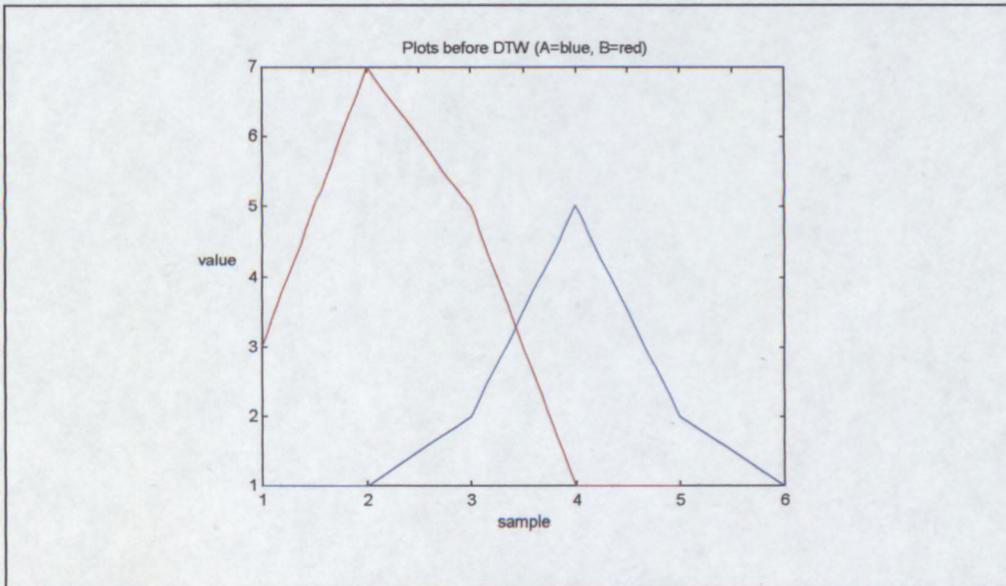


Figure 2.9: Plot of signals A and B before DTW

After the DTW is performed the signals look like this:

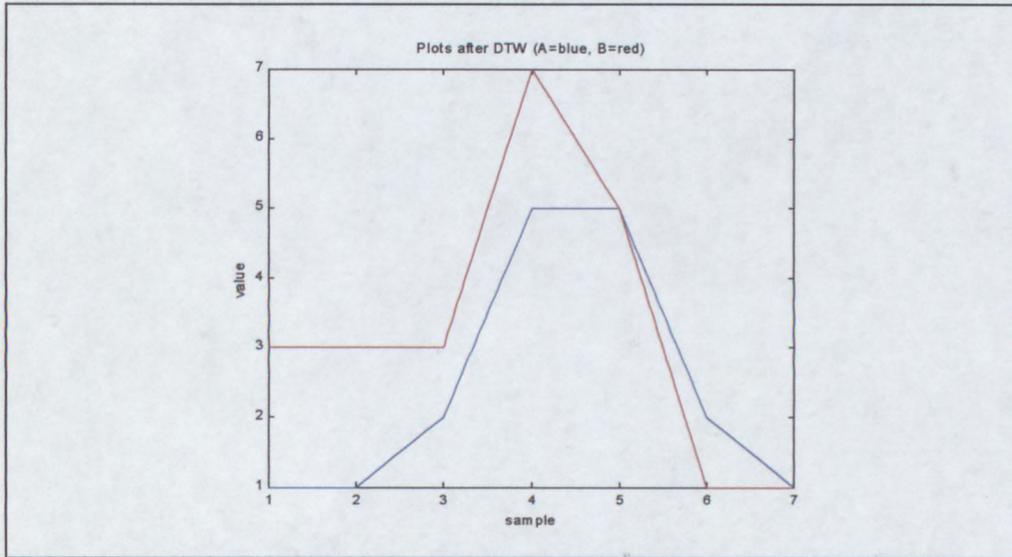


Figure 2.10: Plot of signals A and B after DTW

The two signals were warped in time so that they fit better on each other. Sample values were not changed, and their order was kept intact, but the duration of each sample has in effect been changed.

This is a very simple example. In HSV, signals will typically be multi-dimensional and also much longer (depending on the sample rate).

The figure below shows two genuine signatures from the same person after suitable scaling, translation and rotation have been performed (these processes are discussed in the next chapter):

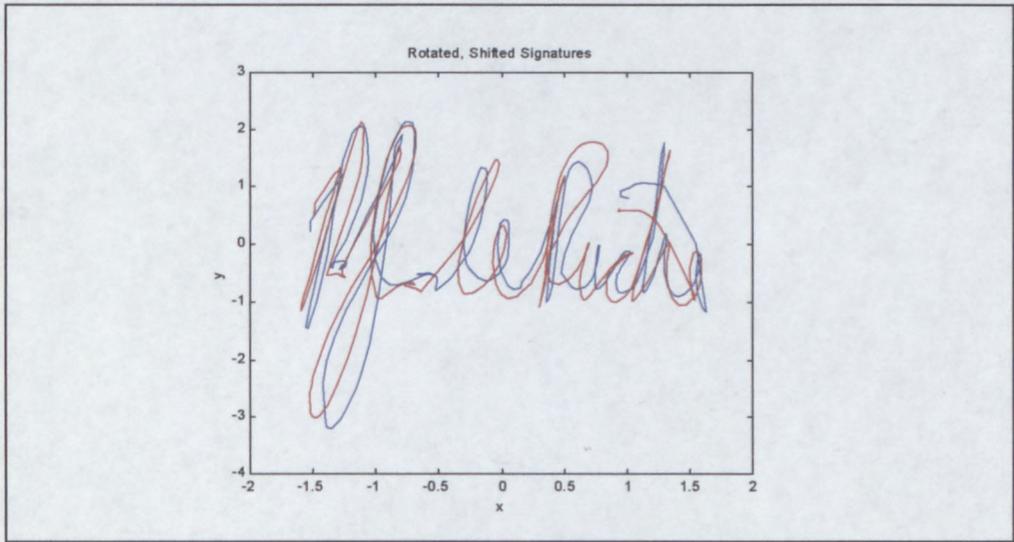


Figure 2.11: 2 genuine signatures

For this example we have chosen to time-warp the X-coordinates of the two signals onto each other. Here are the two X-coordinate signals before DTW:

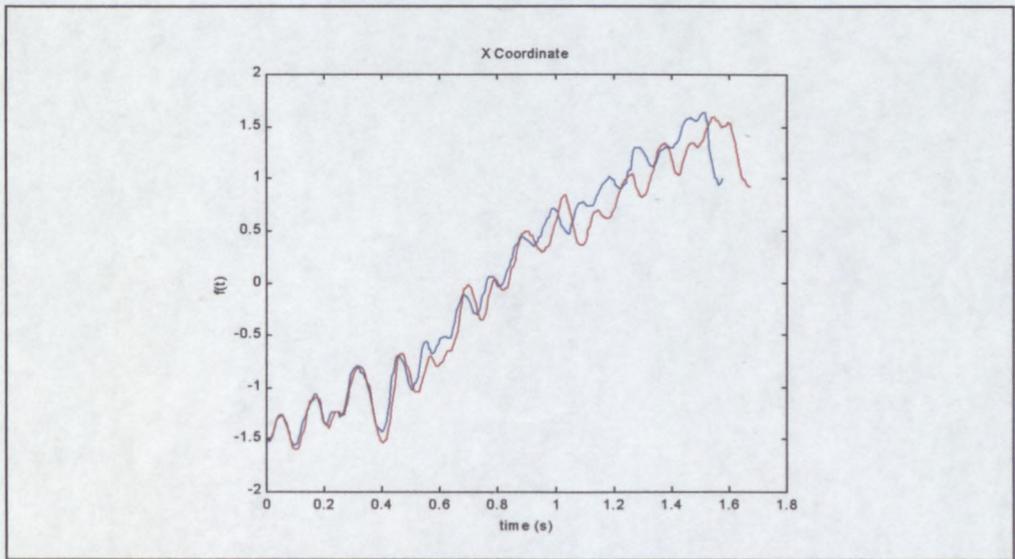


Figure 2.12: X-coordinate signals

After the time-warp process the two signals fit much better:

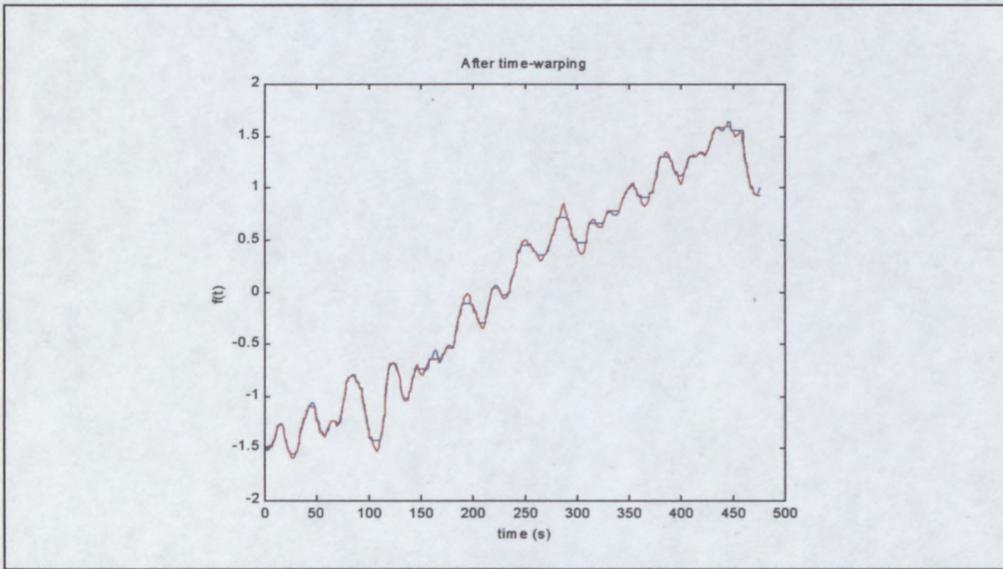


Figure 2.13: X-coordinates after DTW

The best path through the cost matrix is graphed below. Since the signatures are a good match, it stays close to the diagonal:

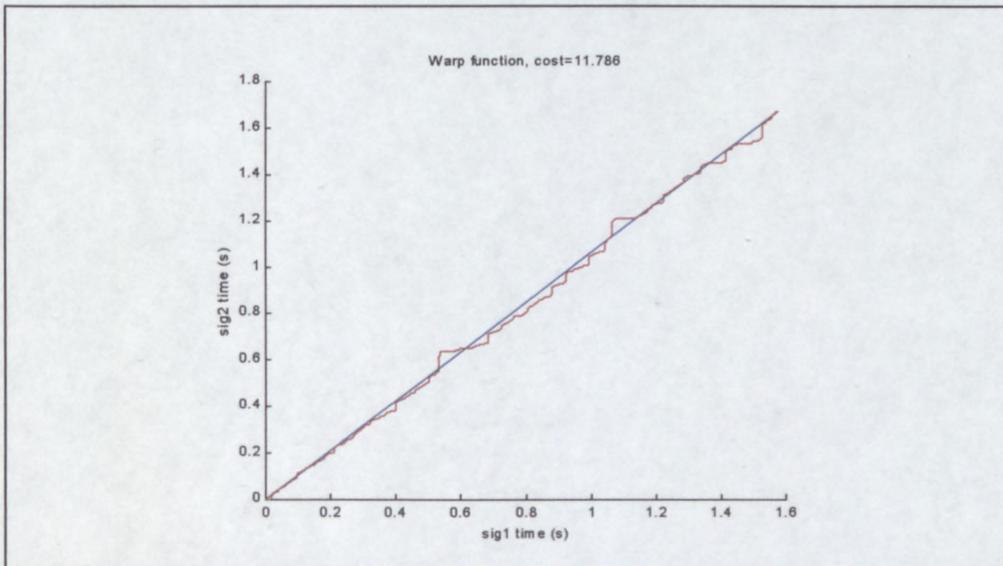


Figure 2.14: Best path

To measure how well the two signatures match, the total cost for the best path is usually taken. This score is usually scaled by the length of the path to not unduly disadvantage longer signatures. The eventual “score” is compared to a threshold. If it exceeds the threshold it is deemed a forgery, otherwise it is accepted.

Below is a DTW analysis of a genuine signature and a forgery. Note how the best path deviates more from the diagonal than the previous case:

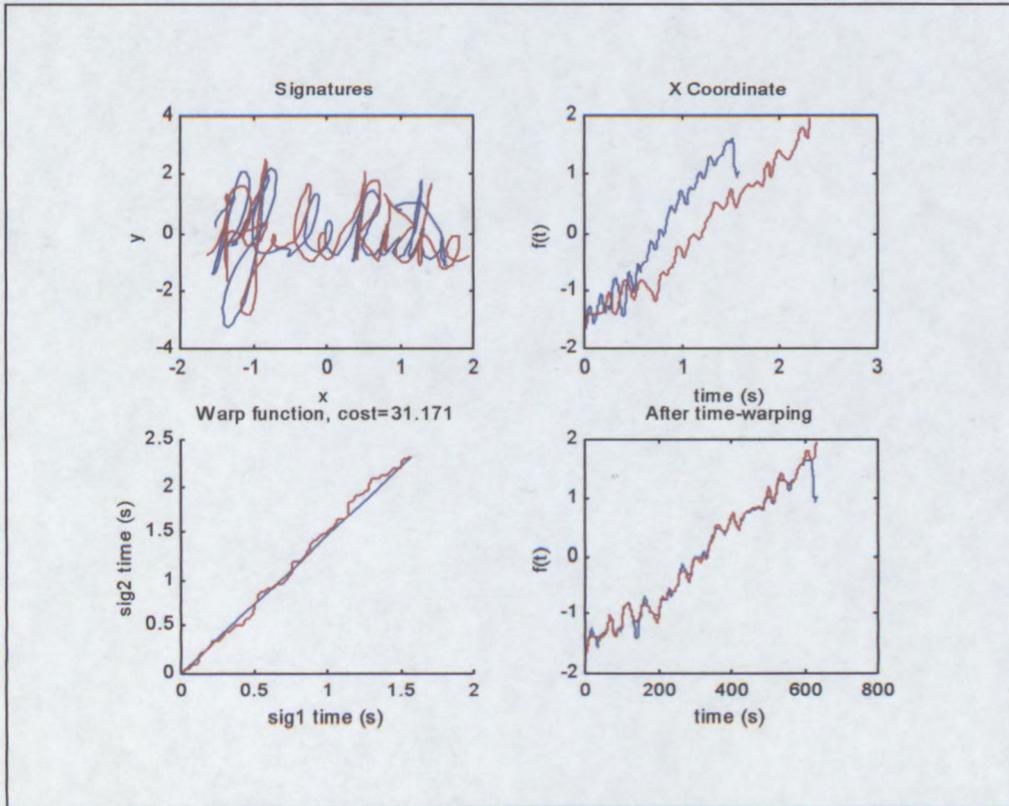


Figure 2.15: DTW between genuine signature and forgery

The total cost for the best path of the DTW between the forgery and the genuine signature is 31.71. Between the two genuine signatures, the cost was 11.78.

DTW is well suited to implementation in a computer. The process itself is very simple, so the programming part is straight-forward. The computational complexity may become a problem in longer signals, since the size of the cost matrix is determined by the product of the lengths of the signals. Fortunately there are some tricks that can be employed to reduce the workload. One such trick is to calculate only part of the cost matrix and restrict warping to this area. Typically only a band around the diagonal is calculated. If the two signals are a reasonable match already, then the best path should be close to the diagonal and calculating only part of the cost matrix should not affect the results.

DTW gives good results in HSV, an EE of 10% is certainly attainable and even better results have been reported in the literature [11]. There are some disadvantages to DTW that should be noted:

- DTW is used to compare only two signals at a time. Due to the natural variations in a signature, a test signature may fit well on one control signature but not well on others.
- It is difficult to obtain a single reference signature or model for the signer. While all the training signatures may be “averaged” to obtain a reference, this causes the natural variation information to be lost and the previous problem again manifests itself.
- While the test signature may be compared to all the control/training signatures and the scores added, areas of high variation in the signatures will lead to high total costs. In a better system, areas of high variation should carry less weight than more stable areas, which would lead to better forgery detection. This is not easily implemented in a stock DTW system.
- DTW requires access to at least one genuine signature for comparison purposes. This may have security and storage space implications.

## 2.3 Summary

There are many methods of HSV, but the processing power available at the time (as well as storage space) usually dictated what could be done. With processing power increasing and storage becoming cheaper all the time, new and more powerful methods of analysis are becoming possible.

Statistical feature extraction has the advantage that it usually requires very little storage space, making it useable in low-security applications with limited storage (like magnetic stripe cards). The generally poor accuracy restricts its use where higher security is required.

DTW is quite useful when a measure of how much two signals differ, is required. In this case a test signature can be compared to a known genuine signature. However, since only two signatures are compared at a time, the natural variations in a signer's signature do not enter the equation and all variations are treated equally.

HMMs (explained in chapter 4) improve on DTW analysis in that the characteristics and probability of the variations between genuine signatures are also modelled. HMMs is the focus of this research.

## Chapter 3

### Gathering Signature Data

Before we can begin comparing signatures, a way must be found to obtain the signature data in digital format so it can be fed into a computer. Once in digital form inside the computer, the signatures will have to be pre-processed before they can be compared. While there are features that are scale and rotation invariant (like curvature and pen tilt), most of the useful features are not. Signatures usually have to be scaled, translated and rotated before features like the x and y coordinates can be used in direct comparisons.

#### 3.1 Data Capture

With current technology, obtaining signatures in digital format is not a problem. Since digitising tablets are widely used by artists, they are freely available in many shapes and sizes. The only stumbling block is the price of the more capable tablets. Tablets that are able to measure the pen tilt and direction cost anything upwards of R2000 (\$200). There are smaller less expensive tablets available that are intended solely for signature capturing, but these are mostly geared towards signature capture alone and not HSV, so the pressure and pen angles are usually not recorded. Of the five features that are commonly recorded at each sample point: x-coordinate, y-coordinate, pressure, pen tilt and pen direction; the pen tilt and direction information are the only features completely obscured from visual inspection of a signature, which makes these features very valuable in detecting forgeries.

The graphics tablet that was used to capture most signatures used in the tests was a Wacom ArtZ II 6 by 8 inch tablet, which is usually aimed at the graphics/designer market. Other tablets that were tested include the smaller Wacom Penpartner (intended for home use) as well as a tablet by Interlink Electronics intended specifically for capturing signatures. Both the latter tablets retail for under \$50 (compared to the \$180 of the ArtZ II), but only return the coordinates and pressure and not the pen angles. The ArtZ II has since been discontinued and replaced by the more advanced Intuous line of tablets. Even so, its features are sufficient for the problem at hand:

- 300dpi resolution
- 256 pressure levels
- Pen tilt and direction sensing
- 200Hz maximum sampling rate

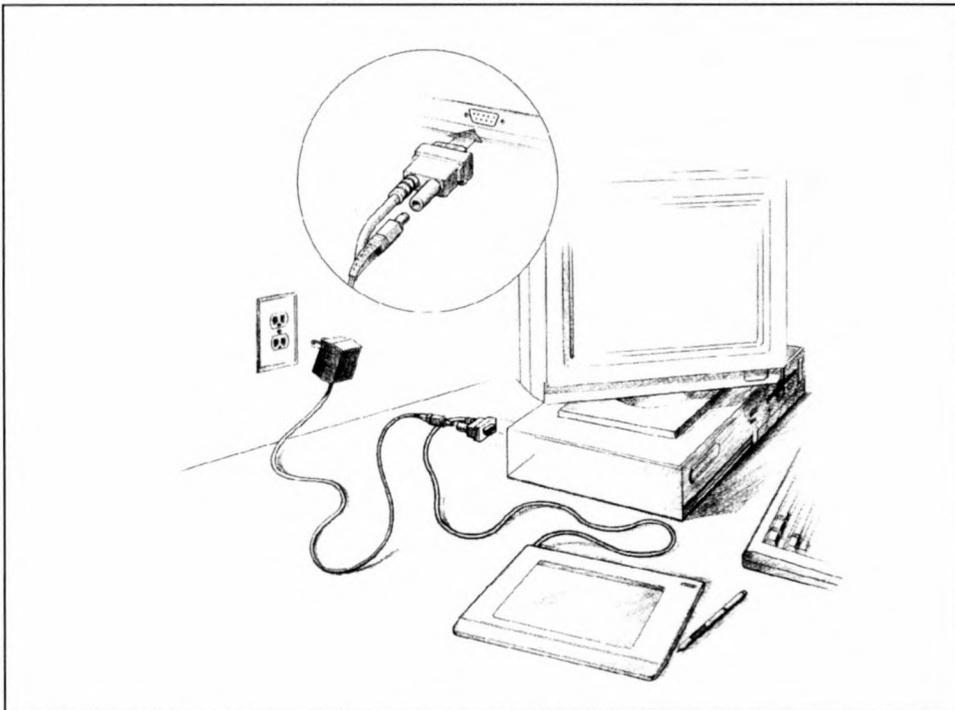


Figure 3.1: Wacom ArtZ II graphics tablet

There are many other tablets available with similar specifications, but a big plus for the Wacom line is the fact that their pens are cordless and lightweight, and require no batteries. There is a pen available that takes standard DIN ink refills, which feels and

writes much like a standard ball-point pen. Using a pen that is as familiar as possible allows signers to sign in a more natural way, which improves consistency and eventually the HSV process as a whole. Furthermore, the tablet continues to sample even if the pen is temporarily lifted from the tablet, this guarantees a constant sampling rate which makes it that much easier to analyse the signatures. Furthermore, the pen movements while in the air are invisible to the forger and are also quite useful in detecting forgeries. Other tablets that do not rely on resonant circuits inside the pen to track its relative position, usually stop sampling once the pen tip is lifted from the tablet. This then requires additional post-processing to restore a constant sampling rate.

Tests have shown that even with very fast signatures, a sampling rate of 200Hz is sufficient. With cubic-spline interpolation performed on the sample points it was found that the interpolated signature corresponds very well with the signature on paper. For most signatures, a sampling rate of 200Hz is overkill, but there are some short and fast signatures that may start to exhibit accuracy problems if the rate is lowered. For simplicity reasons, the sampling rate was kept at a constant 200Hz for all signatures gathered in the tests performed in later sections. The figure below shows the fast Fourier transform (FFT) of the y-coordinate for a sample signature.

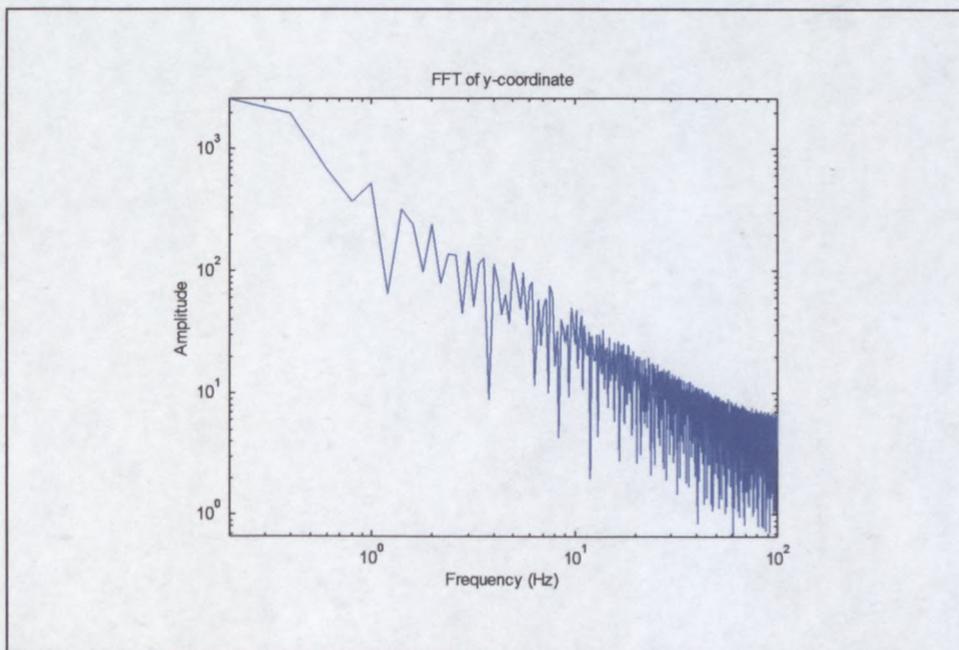


Figure 3.2: FFT of y-coordinate of sample signature (log scales)

The y-coordinate was chosen for this analysis, since it typically exhibits faster variation than any of the other measurements. The virtual absence of any frequencies above 50Hz<sup>1</sup> indicates that even a sampling rate of 100Hz would have been sufficient for this signer.

Most modern tablets adhere to the WinTab specification [7] which was drawn up by a group of experts and facilitates a standard programming interface for tablets under the Windows environment. This means that software written to work with one tablet should work on other tablets also compliant with the standard.

In a best case scenario the data vector returned by the tablet driver will be 5 dimensional, containing the following features:

- The x coordinate.
- The y coordinate.
- The pressure applied to the pen tip.
- The pen tilt, i.e. the angle that the pen shaft makes with the tablet surface (0 to 90°).
- The pen direction, i.e. the wind direction that the pen points into (0 to 360°)

Cheaper tablets, however, often do not report the pen direction and tilt which leaves only the coordinates and sometimes the pressure.

## 3.2 Pre-processing

In order to use some form of automatic comparison between two signals (in this case signatures), the signals to be compared have to be in a similar form. The expression “garbage in, garbage out” is applicable to this situation. If the data is badly scaled, rotated or shifted, one can expect the results to be poorer than the case of good quality (standardised) data.

---

<sup>1</sup> The amplitude of the FFT at 50Hz is more than 30dB below the amplitude at 1Hz.

Since this research centres around hidden Markov (HMM) models and extensive use is made of reference signatures in the actual implementation, the pre-processing steps are chosen for this purpose. Many of these steps are unnecessary if, for example, statistical feature extraction is to be used. It will be shown in later chapters that the extra storage required for reference signatures and the extra computations required for the extensive pre-processing is worthwhile considering the gains in accuracy.

The figure below shows typical (x,y) output for a couple of signatures:

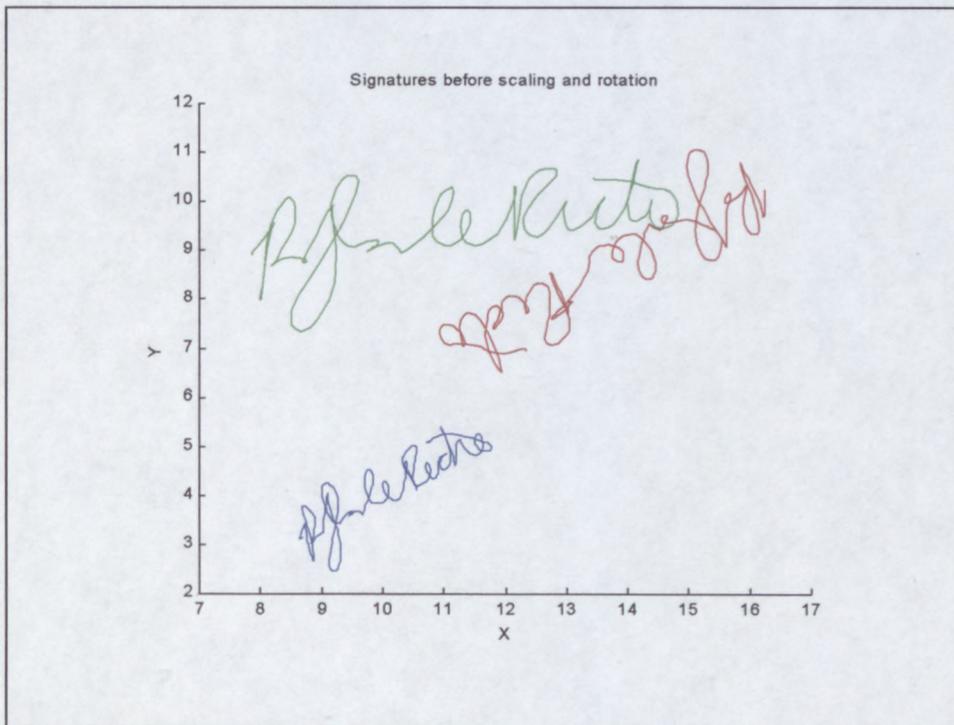


Figure 3.3: Signature (x,y) plots

The signatures above are all from the same person. Note the difference in scale as well as orientation of the signatures. We see a similar picture when plotting the pressure profiles for the same three signatures:

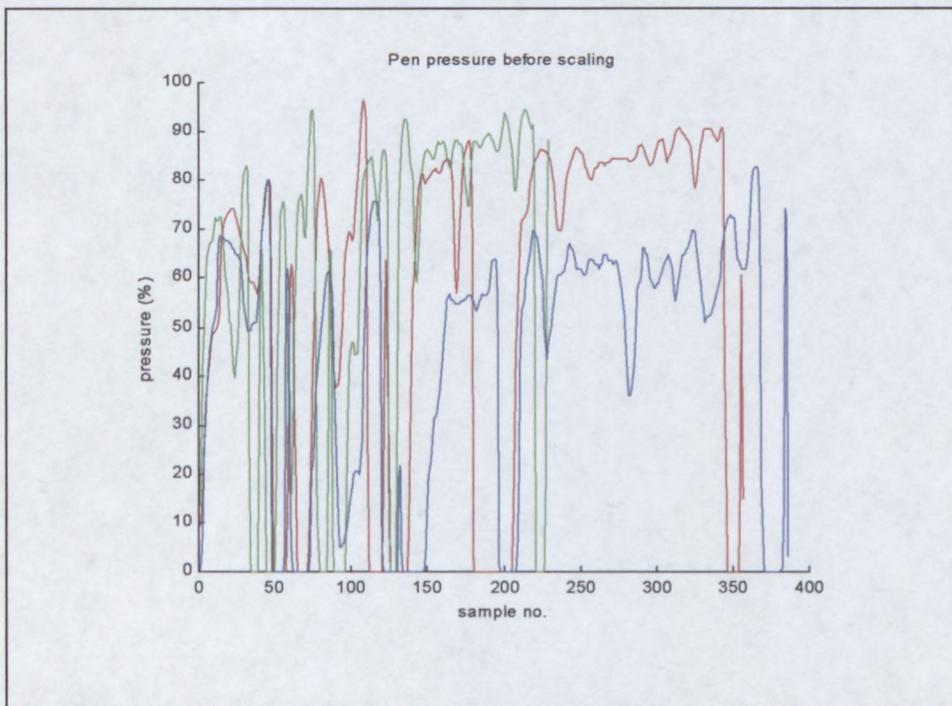


Figure 3.4: Signature pressure profiles

Since the pressure measured depends not only on the sensitivity of the pen used, but also on the thickness and texture of the paper used to sign on, getting a consistent pressure profile between signing sessions is rare. While the pattern usually remains the same for the same signer, the average pressure varies considerably.

The pen tilt usually remains consistent for the same signer, but the pen direction is dependent on the orientation of the signer relative to the tablet. Subtracting the average from the pen direction is one way to solve this problem.

Clearly the signatures have to be translated, rotated and scaled to provide a better fit to each other. There are systems that are scale and rotation invariant (like some statistical feature extraction methods discussed previously), but most of the more accurate systems employ some kind of point-for-point comparison and are very sensitive to size and orientation. These include most DTW and HMM systems.

A constant sampling rate often simplifies calculations and comparisons to some extent. If the tablet stops sampling when the pen is lifted (like some of the cheaper tablets do), then sample values will have to be interpolated from the moment of the pen-up to the pen-down to restore a constant sampling rate. Of course any leading and trailing zero-pressure samples should be removed from the sample train, since these should not be considered part of the signature.

### 3.2.1 Translation and Scaling

Shifting signatures before comparing them is a necessity. Even if there is a clearly demarcated area in which to sign, there will still be variations in position. The natural choice for initial translation is to shift the signature so that its centre of gravity lies over the origin. This is done by simply subtracting the averages from the  $x$  and  $y$  coordinates.

From the examples shown it can be seen that individual signatures from the same person vary in size, especially between signing sessions. Most people adapt the size of their signature to fit in the signing space provided. Furthermore, the coordinates reported by the WinTab driver are always between 0 and 65535 (the maximum value that fits in 16 bits), so the actual physical size of the signature is not directly accessible. If different sized tablets were used in gathering the signatures, then there would be no way to tell their true size.

We assume that the proportions in the signature should stay more or less the same from signature to signature, so if we scale the coordinates by the standard deviation (STD), the signatures should all end up being more or less the same size.

We cannot scale the  $x$  or  $y$  axes independently by taking the individual STDs in their directions, because that would lead to a dependence on the orientation of the signature (which is not guaranteed to be correct). Instead, the euclidean distance of all the coordinates from the origin are summed (after translation of the signature) and

divided by the number of samples. All coordinates (x and y) are then divided by this value. This is equivalent to dividing by the STD in the (x,y) plane.

After translation and scaling, the plots for the three sample signatures look like this:

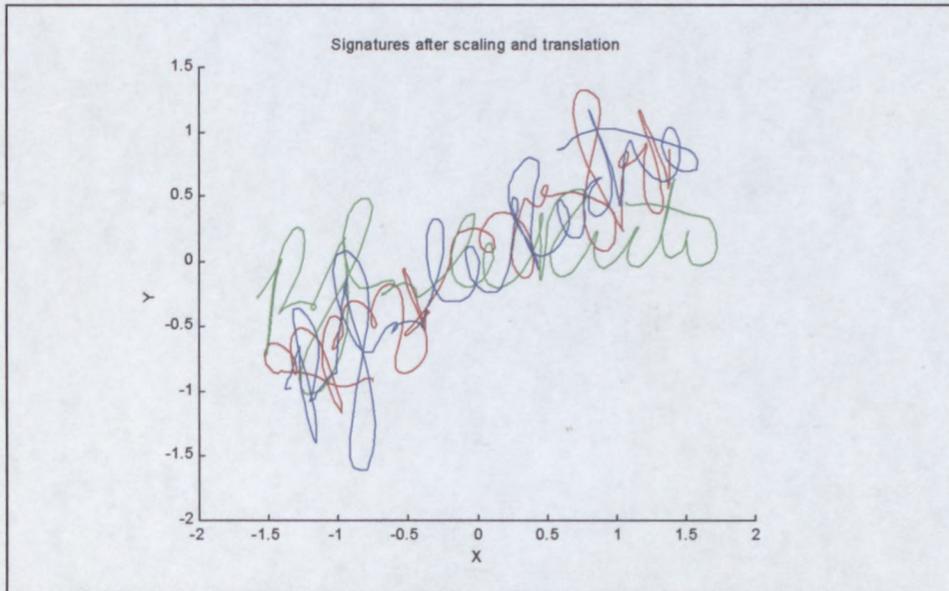


Figure 3.5: Signatures after scaling and translation

The signatures are now all in the same position and more or less the same size. Next we consider the pressure signal:

From an analysis of pressure signals of the available data, it appears that the average pressure of signatures varies between signing sessions, but that the variance of the pressure stays more precise. What this implies is that even though a person may not always exert the same pressure on average, that the variation in the pressure applied throughout the signature remains consistent. Results obtained from the roughly 6000 signatures gathered certainly support this theory. If there is any change in the variation at all, it appears that the variation in pressure is less when the average pressure is higher. Whether this may be attributable to some characteristic of the pen itself (becoming less sensitive as pressure is increased, i.e. a non-linear sensor), or an actual feature of handwriting in general is unclear.

To scale the pressure signals to be more in line with one another, we first calculate the average pressure for the pen-down areas. We exclude the pen-up areas from the average calculation, since this would skew the result.

One could be tempted to divide the pressure signal by this average, so all signatures have an average pressure of 1. However, doing so would also scale the variance of the pressure. From the comments in the previous paragraphs we can deduce that this is not a desired side-effect, since the pressure variance seems to have no direct relation to the average pressure. A better option would be to shift all pen-down samples upwards so that the average pressure for all signatures are the same. Shifting the pressure down is not an option, since this may result in pressures below 0 which is meaningless in the context of 0 pressure meaning a pen-up.

We choose the shifted average pressure to be the maximum pressure measurable by the tablet. This not only ensures that the pressure graph will always be shifted upwards, but also places further emphasis on the pen-up points<sup>2</sup> (which are not shifted, and are thus now further removed from the other samples).

After shifting the pressure, the graphs look like this:

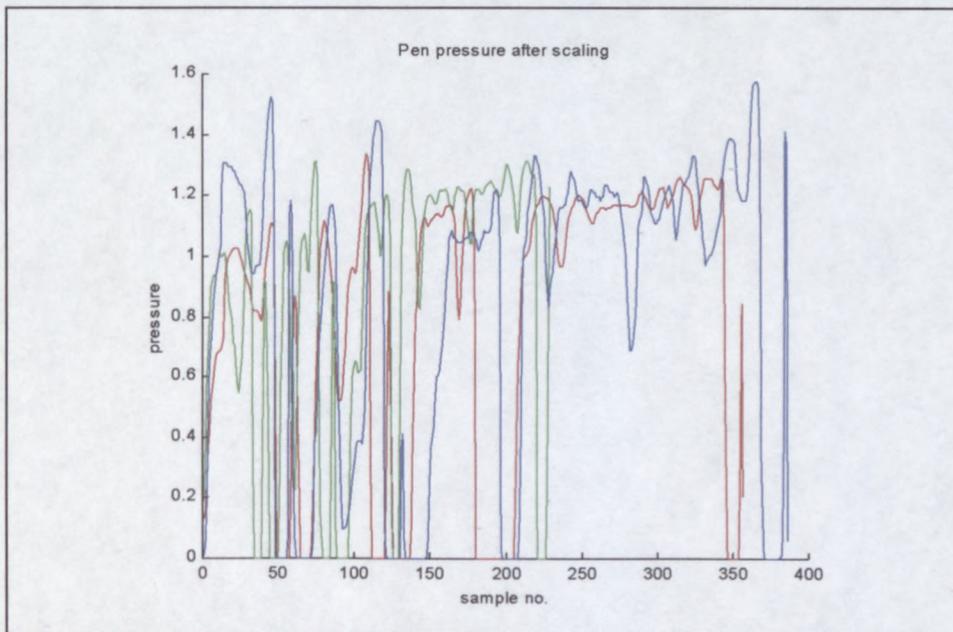


Figure 3.6: Pressure after scaling

<sup>2</sup> It is suspected that this might improve segmentation on the pen up/down boundaries, but the effect was too slight to measure.

### 3.2.2 Rotation

With the signatures properly scaled and translated the next task is to rotate the signatures. The aim is to rotate them so that they all point in the same direction.

There are many possible ways to rotate the signatures. Three methods were investigated: principal axis rotation, average angle rotation and rotation through the average pen direction.

For principal axis rotation we need to determine the direction of maximum (x,y) variance in the signature. Under the assumption that the signature is much wider than it is high, this should give the direction that the signature lies in.

To determine the principal axis we first determine the (x,y) covariance matrix for the signature. Next we determine the eigenvalues and eigenvectors for the covariance matrix. The eigenvector corresponding to the largest eigenvalue will indicate the direction of greatest variance, see for example Morrison (1988).

Rotating through the angle of this eigenvector should place the signature parallel with the x-axis. Unfortunately we have no guarantee that the signature will be the right way up. If it is assumed that the signature follows a left-to-right progression (as most western signatures do), the number of samples to the right and left of the first sample can be counted. If there are more samples to the left than the right of the first sample, this would give a strong indication that the signature may be upside down. Signatures may be rotated by an additional  $180^\circ$  in this event.

Below is a graph of the signatures rotated by the principal axis method:

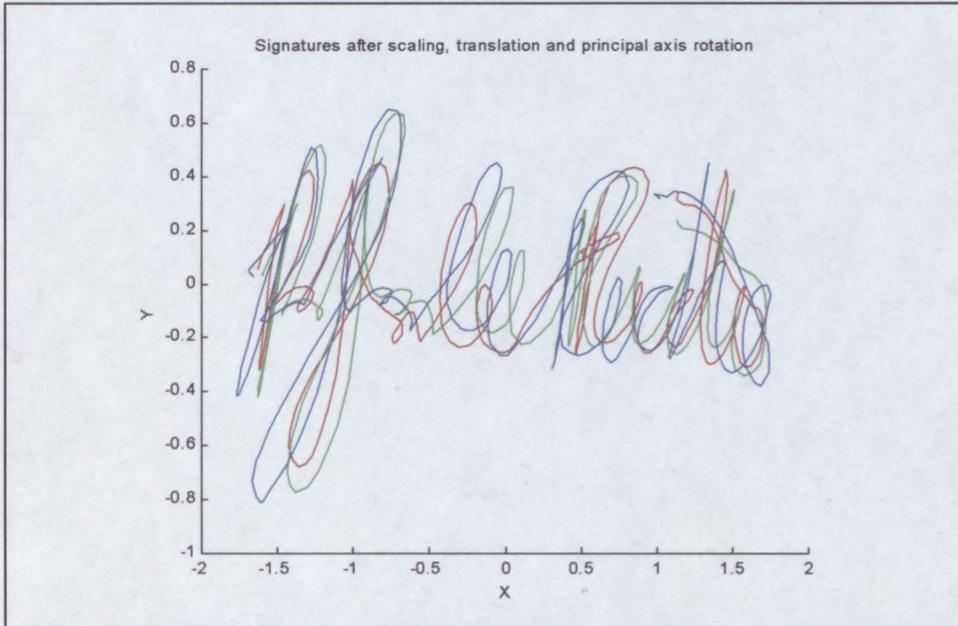


Figure 3.7: Signatures after principal axis rotation

Principal axis rotation works well with signatures that exhibit a definite left-to-right progression, but gives very poor results where there is no such relationship as in the signature shown below:

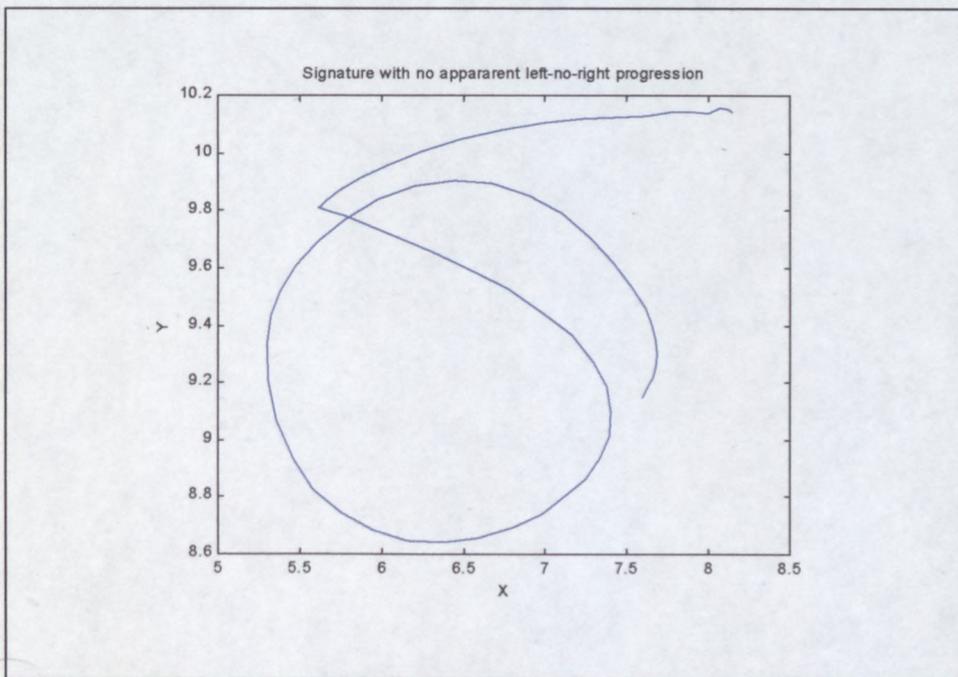


Figure 3.8: Signature with no apparent left-to-right progression

Signatures like these contain few distinguishing features, and as such are easily forgeable. Using such signatures where any kind of security is required can at the very least be considered suspect. Fortunately there is a rotation method that performs better with rounded signatures at the expense of slightly worse performance with elongated signatures. This method is average angle rotation.

The assumption is still that the signature follows a left-to-right progression, and thus that the average direction of propagation should be in the principal direction of the signature. If the direction of propagation from each sample to the next is taken and averaged over the whole signature, then the average should give a good approximation of the direction that the signature lies in.

For this method, angles are measured only in the first and fourth quadrants, i.e. between  $-90^\circ$  and  $+90^\circ$ . If the direction of propagation falls outside these boundaries it is adjusted by  $+$  or  $- 180^\circ$  so that it does. Strictly speaking the angle measured is thus the angle of the tangent of the signature with the positive x-axis. These angles are summed and divided by the number of samples to give the result. The signature is then rotated through this angle as with principal axis rotation. Similarly, the signature may also have to be flipped by  $180^\circ$ .

Average angle rotation gives the following results with the three test signatures:

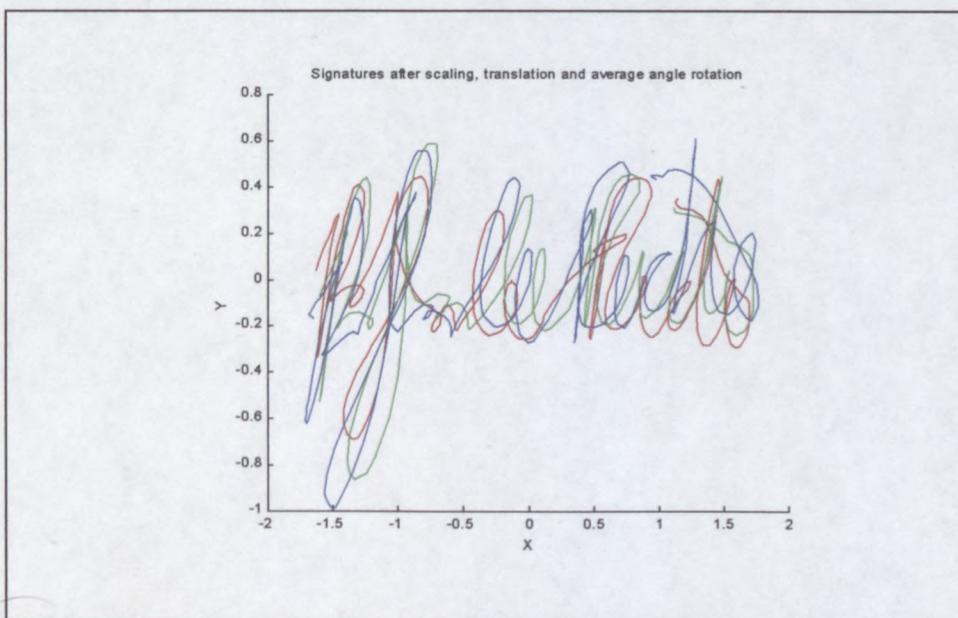


Figure 3.9: Signatures after average angle rotation

The rotation results with the average angle rotation method is not quite as good as with the principal axis rotation method. With more rounded signatures, however, average angle rotation performs much better. While principal axis rotation will rotate round signatures in wildly varying directions, the average angle tends to be close to 0 for round signatures which causes average angle rotation to have little effect. This is preferred, since signatures are usually already close to the correct orientation anyway.

The best rotation method of the three is without a doubt rotation through the average pen direction. Unfortunately this does require a tablet that returns the pen angles (which are quite a bit more expensive).

The WinTab interface returns the pen direction as a value between  $0^\circ$  and  $360^\circ$  measured clockwise from the standard  $-90^\circ$  position. The reason why  $-90^\circ$  is chosen as the point of reference is that this minimizes angle wraparound problems. Angle wraparounds occur when angles go past  $360^\circ$  and wrap back just after  $0^\circ$ , or when angles go below  $0^\circ$  and wrap to just before  $360^\circ$ . While  $0^\circ$  and  $360^\circ$  are the same, this is not apparent to a *dumb* comparator.

The reported pen direction is converted to the standard system to give angles between  $-90^\circ$  and  $+270^\circ$ . Most right-handed signers hold the pen in the region of  $+135^\circ$ , so wraparounds between  $-90^\circ$  and  $+270^\circ$  should be rare. Wraparounds may occur, however, especially if the tablet is used upside-down, so it is something that one needs to wary of.

The average of the pen direction is calculated systematically starting at the first sample working through to the last. If a sample is encountered that is separated more than  $180^\circ$  from the current average,  $360^\circ$  is added or subtracted to make the difference less than  $180^\circ$ . Angle differences close to  $180^\circ$  (which may again cause wraparound problems) are rare, since most signers hold the pen more or less in a constant direction with only slight variation. Consequently, the result can be considered the closest approximation to the average pen direction.

The signature is then rotated through this angle (and the rotation angle subtracted from the measured pen direction signal)<sup>3</sup>. The results for the three test signatures are shown below:

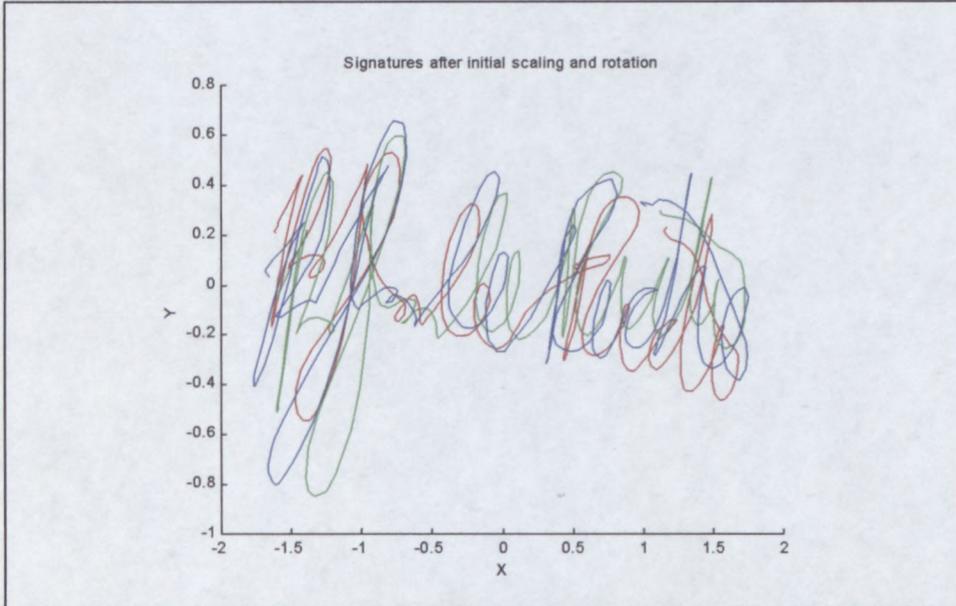


Figure 3.10: Signatures after average pen direction rotation

At first inspection this rotation method may appear to deliver results inferior to the principal axis rotation method. Remember, however, that this method is not subject to the problems that plague the other two. Signatures are almost guaranteed to be more or less in the same orientation, whether they are round or elongated. No checking needs to be done to determine if a further  $180^\circ$  rotation is required.

In the event that direct comparisons will be done between the physical coordinates, it makes sense to attempt to subtly shift, scale and rotate signatures further to obtain an even better match between them.

---

<sup>3</sup> For aesthetic reasons a fixed value of  $135^\circ$  is subtracted from the rotation angle before rotation, so that right-handed signatures will appear more or less the right way up.

### 3.2.3 Further fitting

Up to this point, signatures were scaled, translated and rotated using only the information contained in the signatures themselves. No use was made of a reference model or reference signature(s). In the event that no reference is available this is as far as the pre-processing goes, and the comparison is performed. However, this is not the case in this thesis. A reference signature (or the entire training set) is assumed to be available in order to perform further subtle fitting of the signatures on each other.

In the training stage it makes sense to fit the  $(x,y)$  image of the training signatures as best possible to each other. The better the signatures are made to fit each other, the less the  $(x,y)$  variance will be in each segment which would result in a stricter model, which would help catch more forgeries. Of course, if there is an intrinsic high variance in the genuine signatures, then there is little that can be done to improve the model.

In the evaluation stage, the  $(x,y)$  image of the test signature should also be shifted, scaled and rotated to fit the training signatures as best possible. Since the training set may be too large to store, a reference signature chosen from the training set, or a signature computed from the training signatures may be used for this purpose.

Note that in this final fitting process, still only the two dimensional  $(x,y)$  data is used. Since the 2D image is often available to the forger, genuine signatures should not be put at a disadvantage by bad  $(x,y)$  positioning. The idea being that if the signatures are shifted to fit each other perfectly, the dynamics and other features of genuine signatures should then also match up well, but not those of forgeries. Having the signatures fit each other well, makes it that much easier to know which parts to compare to which parts in the reference signature(s) or model. Since it is the dynamics and invisible features that are unavailable to the forger, it should form the bulk of the forgery detection process.

The question is: how to fit the signatures to each other? DTW offers an easy, fast and, as it turns out, an accurate option. In the training stage, the training signatures are

iteratively fitted to each other and adjusted. Similarly, in the evaluation stage, the test signature is fitted to either a reference signature or the entire training set and iteratively adjusted. It works as follows:

1. A DTW is performed between the (x,y) coordinates of the two signatures to be fitted.
2. For each coordinate in the warped signatures, the angle of the coordinate with the positive x-axis is calculated. The average of the difference between the corresponding angles is taken as the difference in signature rotation between the two signatures.
3. For each coordinate in the warped signatures, the distance from the origin is calculated. The average of the ratio between the corresponding distances in the two signatures is taken as the size ratio between the two signatures.
4. In similar fashion the relative x-shift and y-shift between the two warped signatures are calculated.
5. One of the signatures is rotated, scaled and translated by these ratios to facilitate a better fit to the other.
6. If the adjustments were greater than preset thresholds, the process is repeated.

The two signatures should gradually converge, until the differences are smaller than the preset thresholds and the process is stopped.

In the case of the training set each training signature is warped to all other training signatures before being adjusted once by the average of all the adjustments. This improves the stability of the process.

Below shows the three sample signatures after being fitted on each other in this way:

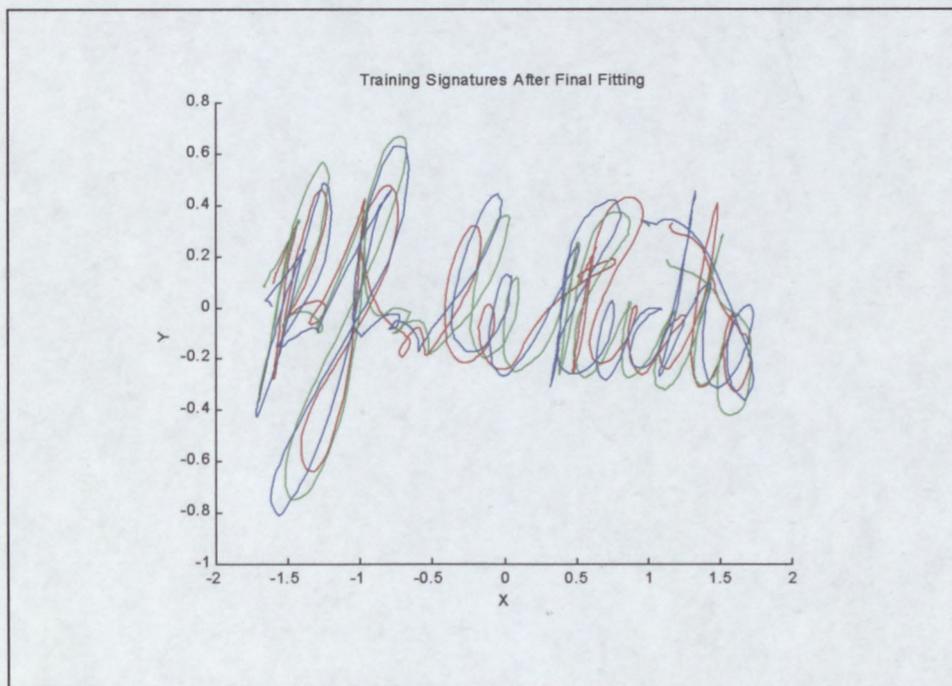


Figure 3.11: Signatures after final fitting

Unfortunately there is a hitch: Since signatures are warped point-for-point and we have a constant sampling rate, signature segments of slower movement carry more weight in the fitting process than segments of faster movement. Since we are trying to match the images and not the signals themselves, this is not a desired effect. Take for example the signatures below. The greatest amount of signing time is concentrated around a small area in the bottom part of the signature.

Standard time-warping does not give the best results:

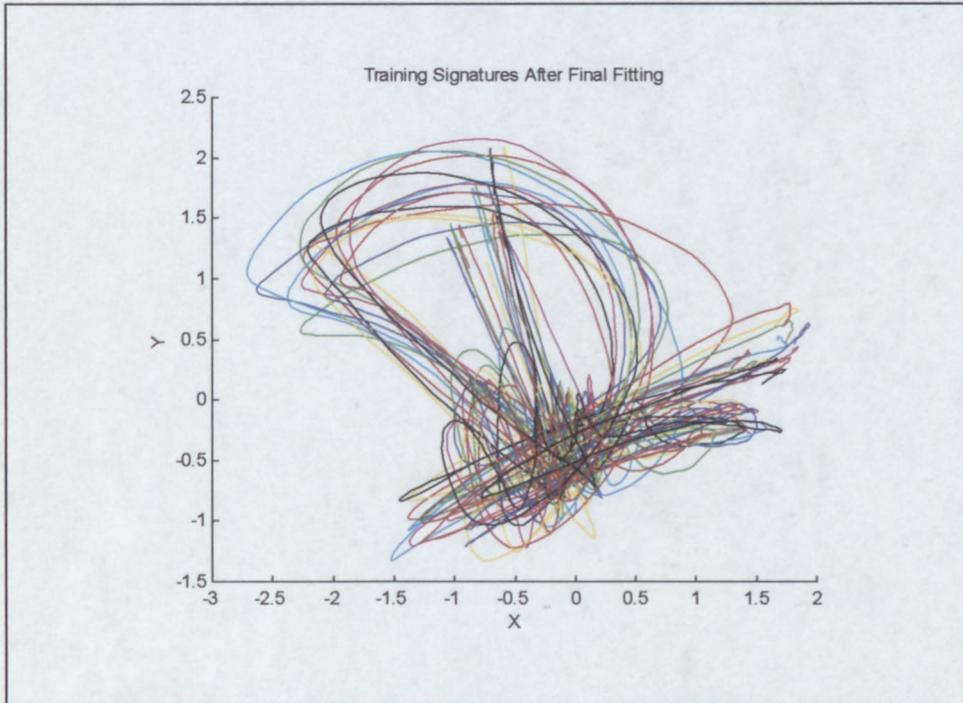


Figure 3.12: Additional fitting without speed compensation

An improvement to the algorithm would be to resample the signatures to produce a fixed sample rate per path length. To do this would require the following steps:

1. The signature is interpolated between the current sample points.
2. The path length is obtained from the equation

$$s(T) = \int_0^T \sqrt{x'(t)^2 + y'(t)^2} dt$$

where  $x(t)$  is the interpolated  $x$  coordinate function, and  $y(t)$  is the interpolated  $y$  coordinate function.

3. These equations are solved for constant path length intervals to obtain the time instants, i.e. solving  $T$  for values of  $s$ .
4. The values for  $T$  obtained in step 3 is substituted into the interpolated functions to obtain the new sample points.

The steps required above involve a lot of manipulation and number-crunching, which makes it a bit impractical – particularly if the interpolating function is complex.

Fortunately there is a trick that adds little complexity to the calculations and also gives good results.

The results with this technique is shown below:

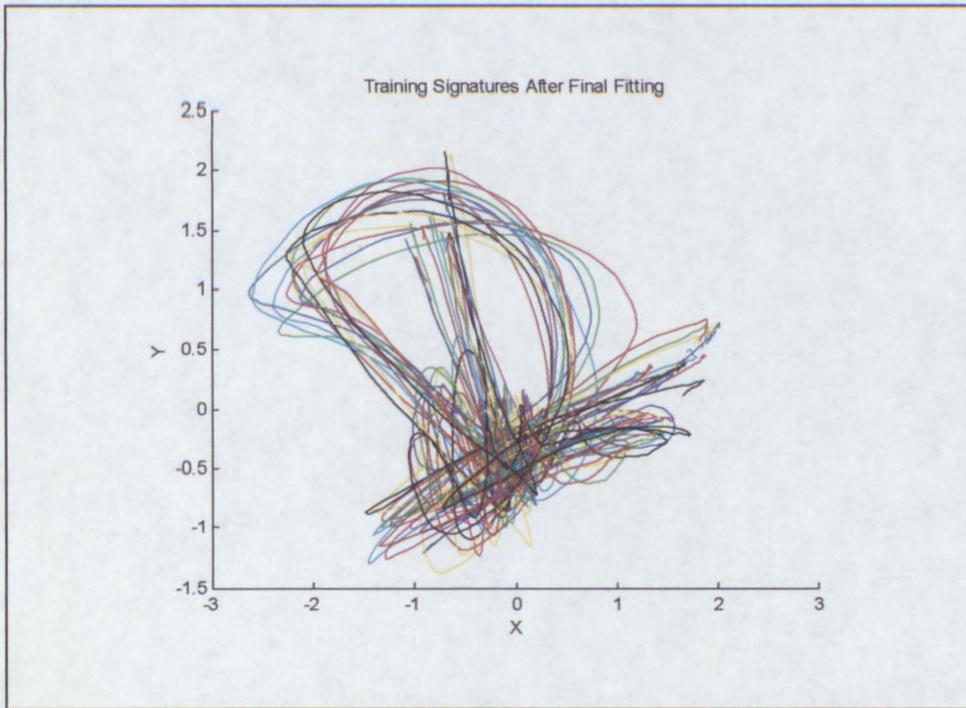


Figure 3.13: DTW fitting with speed compensation

What is done is to weigh each point in the cost matrix by a function of the speeds at the two signature samples. The average of the two speeds was found to be a good weighting factor<sup>4</sup>. A small positive offset is added to put a lower limit on the scaling factor for points with very low speed. The size of this offset determines the degree to which “speed compensation” is performed, a large value corresponding to very little compensation. A good value for this offset was found experimentally.

In this way, points corresponding with slower pen movement are penalised while points of faster movement carries more weight in the fitting which is a reasonable approximation to what would have happened with a constant path length resample.

---

<sup>4</sup> Since the (x,y) scaling of signatures affect the speed, care must be taken to use speeds that are also scaled by the same factor as the coordinates. Also, an accurate speed measurement is not required here, and an approximation taken by dividing the distance between samples by the sampling period works well.

### 3.2.4 Interpolation and Calculated Features

Currently the set of features measured are:

- X-Coordinate
- Y-Coordinate
- Pressure
- Pen Tilt
- Pen Direction

There are however some derived features that are also useful. These features include the speed and direction components of the pen velocity, as well as the curvature of the signature. Some of these calculated features have the advantage that they are rotation invariant which make them useful even if the signature is poorly rotated.

The velocity and curvature is obtained by taking the first and second derivatives of the (x,y) coordinates. Both the direction and magnitude of these calculated features can be used in comparisons, the magnitude being rotation invariant.

For most modern tablets, the sampling rate is sufficiently high so that the velocity can be approximated by taking the vector from one sample to the next and dividing the difference by the sample period. This saves computation time. The second derivative, however, tends to become unstable when such approximations are used and other measures have to be taken. A smooth interpolating curve is usually calculated by some predefined criteria, and this curve is then differentiated to obtain both the speed and curvature.

Since the resolution of most tablets is at least 300DPI (dots per inch), and some of the latest models are 1000DPI or more, the sampling quantization error can be considered negligible. Consequently, when we do interpolation the constraint that the smoothing curve goes through the sampling points is usually applied.

The (x,y) samples of the signature are interpolated by a series of splines, a spline for each segment between two samples. The initial and final conditions for each spline ensure that the whole interpolating curve adheres to the following rules:

- The spline itself and all derivatives must be continuous between its start and endpoints.
- The fourth order derivative must be 0.

Interpolation of this type is called cubic spline interpolation, and gives a pleasingly smooth curve which closely approximates the motion that was followed between sampling points. For this discussion we assume a fixed sampling rate, the theory can however easily be extended to include variable sampling rates.

The interpolating curve will have the form:

$$S(t) = \begin{cases} S_1(t) & t_1 \leq t \leq t_2 \\ S_2(t) & t_2 \leq t \leq t_3 \\ \vdots & \\ S_{n-1}(t) & t_{n-1} \leq t \leq t_n \end{cases}$$

where  $S_1(t)$ ,  $S_2(t)$ , ...,  $S_{n-1}(t)$  are cubic polynomials, i.e.:

$$S_i(t) = a_i(t-t_i)^3 + b_i(t-t_i)^2 + c_i(t-t_i) + d_i, \quad t_i \leq t \leq t_{i+1}$$

The coefficients of these equations constitute a total of  $4n-4$  coefficients that must be determined to specify  $S(t)$ . Following the discussion in Anton and Rorres (1973), it can be shown that the coefficients of these equations can be determined as follows:

$$\begin{aligned} a_i &= (M_{i+1} - M_i) / 6h \\ b_i &= M_i / 2 \\ c_i &= (y_{i+1} - y_i) / h - [(M_{i+1} + 2M_i)h / 6] \\ d_i &= y_i \end{aligned}$$

This is for  $i = 1, 2, \dots, n-1$ , where  $M_i = S''(t_i)$ ,  $i = 1, 2, \dots, n$ . The  $y$  is the actual sample value at each sampling instant, and  $h$  is the sampling period (inverse of sampling rate). From this we can see that the cubic spline is uniquely determined by the values of  $M$ . If we write this in matrix form we have:

$$\begin{bmatrix} 1 & 4 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 4 & 1 \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \\ M_3 \\ M_4 \\ \vdots \\ M_{n-3} \\ M_{n-2} \\ M_{n-1} \\ M_n \end{bmatrix} = \frac{6}{h^2} \begin{bmatrix} y_1 - 2y_2 + y_3 \\ y_2 - 2y_3 + y_4 \\ y_3 - 2y_4 + y_5 \\ \vdots \\ y_{n-4} - 2y_{n-3} + y_{n-2} \\ y_{n-3} - 2y_{n-2} + y_{n-1} \\ y_{n-2} - 2y_{n-1} + y_n \end{bmatrix}$$

This is a linear system of  $n-2$  equations for  $n$  unknowns, thus we need two additional equations to determine  $M$  uniquely. The simplest mathematical conditions we can impose are

$$M_1 = M_n = 0$$

An interpolating spline of this type is called a *natural spline*. Eliminating  $M_1$  and  $M_n$  from the equations we obtain

$$\begin{bmatrix} 4 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} M_2 \\ M_3 \\ M_4 \\ M_5 \\ \vdots \\ M_{n-4} \\ M_{n-3} \\ M_{n-2} \\ M_{n-1} \end{bmatrix} = \frac{6}{h^2} \begin{bmatrix} y_1 - 2y_2 + y_3 \\ y_2 - 2y_3 + y_4 \\ y_3 - 2y_4 + y_5 \\ \vdots \\ y_{n-4} - 2y_{n-3} + y_{n-2} \\ y_{n-3} - 2y_{n-2} + y_{n-1} \\ y_{n-2} - 2y_{n-1} + y_n \end{bmatrix}$$

This is a  $(n-2) \times (n-2)$  system which can be solved for  $M$ . The natural spline tends to flatten the interpolating curve at the endpoints, but for long signatures this should have very little effect on the results.

Solving  $M$  may appear problematic since the matrix can easily be  $1000 \times 1000$  or larger. Fortunately there is a well-known and efficient algorithm for solving uni-diagonal systems such as this, so this interpolation process is quite speedy on a computer. The LU factorisation for  $M$  can be obtained by applying a simple recipe, after which solving the set of linear equations is easy. The optimised Pascal code below performs this task:

```

{Returns a cubic spline interpolation of the given sample array
The interpolating function is given by:
 $S=a(n)(x-x(n))^3+b(n)(x-x(n))^2+c(n)(x-x(n))+d(n)$ , for  $x(n) \leq x \leq x(n+1)$ 
Input:
  h=equal space between samples
  vect=sample vector
Output:
  a,b,c,d=spline coefficients}
procedure csinterp(h:double;var vect,a,b,c,d:TDoubleArray);
var
  M,LUVal:TExtendedArray;
  i,dim,row:integer;
begin
  {Get the dimension of the vector}
  dim:=length(vect);
  {Allocate arrays}
  SetLength(LUVal,dim-2);
  SetLength(M,dim);
  {Set the RH and LUvalues}
  LUVal[0]:=4;
  M[1]:=((vect[0]-2*vect[1]+vect[2])*6/sqr(h))/LUVal[0];
  {Solve for y making use of the fact that L is lower triangular}
  for row:=1 to dim-3 do
  begin
    LUVal[row]:=4-1/LUVal[row-1];
    M[row+1]:=((vect[row]-2*vect[row+1]+vect[row+2])*6/sqr(h))-M[row])/LUVal[row];
  end;
  {We now have y. Now we solve x from  $Ux=y$  using the fact that U is upper triangular}
  for row:=dim-4 downto 0 do
    M[row+1]:=M[row+1]-M[row+2]/LUVal[row];
  {Set other values of M -> natural spline interpolation}
  M[0]:=0;
  M[dim-1]:=0;
  {Allocate spline coefficients}
  SetLength(a,dim-1);
  SetLength(b,dim-1);
  SetLength(c,dim-1);
  SetLength(d,dim-1);
  {Now get the values of a,b,c,d}
  for i:=0 to dim-2 do
  begin
    a[i]:=(M[i+1]-M[i])/(6*h);
    b[i]:=M[i]/2;
    c[i]:=(vect[i+1]-vect[i])/h-((M[i+1]+2*M[i])*h/6);
    d[i]:=vect[i];
  end;
end;

```

Figure 3.14: Pascal code to determine cubic spline

Since the sampling rate is usually sufficiently high that we do not need to calculate points in-between, the interpolation is fast and does give a very good approximation of the speed and curvature at the sample points themselves.

Interpolating both the x and y signals and calculating their first and second derivatives thus gives access to 6 additional calculated features:

- X-Component of Velocity
- Y-Component of Velocity
- X-Component of Curvature
- Y-Component of Curvature
- Speed (Magnitude of (x,y) velocity)
- Curvature (Magnitude of (x,y) curvature)

Often only the last two features are used, since they are invariant to rotation and gives a basis for comparison even when the rotation method employed gives poor results. Unfortunately, being a second derivative, the curvature does tend to fluctuate wildly. To smooth out the curvature a lowpass filter may be used. This is investigated in a later chapter, with mixed results.

### **3.2.5 Dimension Reduction**

Frequently storage space is a concern, and unnecessary data should not be stored in the database of reference signatures. Obviously the calculated features, speed and curvature, should not be stored since they can be recalculated almost instantaneously. The question does arise whether there is sufficient correlation between the 5 measured features to safely remove one or at least do some kind of transformation to reduce the number of dimensions. Another plus for applying a dimension reducing transformation like a Karhunen-Loeve (KL)<sup>5</sup> transform is that the signature is no longer available in its original format, making it harder for criminals to reconstruct the signatures should they gain access to the database.

---

<sup>5</sup> A transform commonly used to change the axes of a coordinate system to correspond with the directions of greatest variation and facilitating a reduction in dimension by dropping the dimension of least variance.

Investigating many signatures we observe the following:

- After proper rotation there appears to be little correlation between the x and y coordinates. The x-coordinate usually follows a more-or-less left to right progression with the y-coordinate a much more rapid up and down oscillation.
- In some cases there does appear to be some correlation between the pressure signal and the signing speed, however this does not always hold true.
- The pen tilt and pen direction usually remains relatively constant throughout the signing process. Some signers tend to vary the pen tilt with the up and down motion of the y-coordinate, which gives rise to some correlation. Others tend to hold the pen still and move their hand for the up and down motion.

These conclusions were made after considering both the Dolfig and SigGrab databases. Consider the following signature graphs for a sample signature:

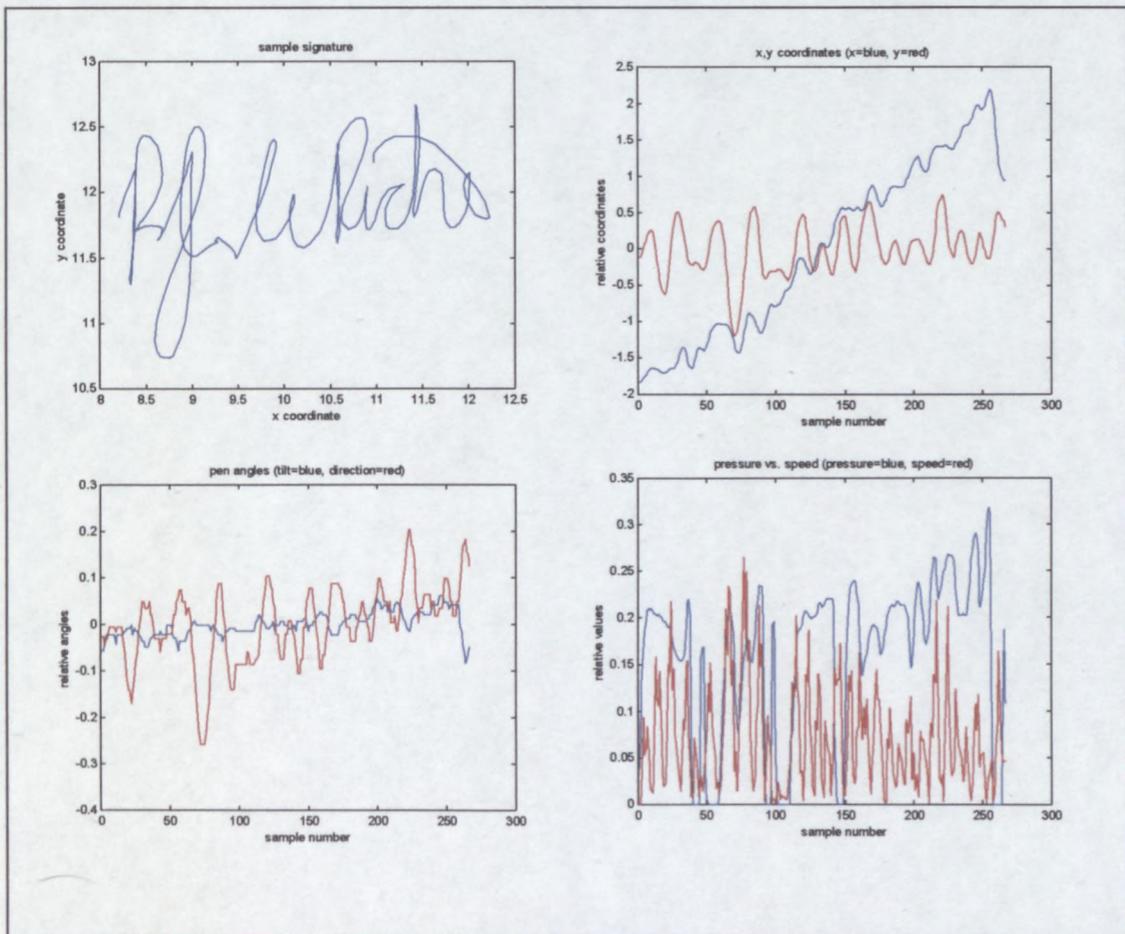


Figure 3.15: Correlations between signature features

In these graphs the slight correlations between some of the features are apparent. However, these correlations are not sufficient to justify reducing the dimension from 5 to 4 (or less). In fact, reducing the dimension from 5 to 4 for this signer dropped the recognition accuracy by more than 10% in one test (using HMM analysis), which is an unacceptable loss.

From these observations we thus feel that an attempt at dimension reduction should not be feasible since there does not appear to be sufficient correlation to support this. While the (x,y) coordinates seem to be the most important features, the importance of the other features vary from signer to signer. Some signers sign with a predictable and natural pressure pattern that is easy to forge, others do not. The majority of signers keep the pen tilted more or less at the same angle, but for the exceptions the pen tilt provides an almost fool-proof way of catching forgeries. The pen direction exhibits similar characteristics.

It is interesting to note that the (x,y) coordinates are subject to visual feedback. If a mistake is made in the formation of one letter, subsequent letters are often adjusted to remain in a straight line. This is why it is important to not only compare the rhythm (speed profile) but also the absolute position, even though it may be argued that the absolute position is the easiest to forge. For example, a person with an impairment (like arthritis) may sign very erratically, making the visual image about the only useable feature.

Experimental results has indeed shown that the loss in accuracy is almost proportional to the number of dimensions removed by the KL transform<sup>6</sup>. Reducing dimensions or discarding some measurements does not appear to be appropriate. Other lossless methods of compression may be a more worthwhile pursuit.

---

<sup>6</sup> A discussion of the KL transform is beyond the scope of this thesis, but there are many good textbooks available with thorough coverage of the subject.

### **3.3 Summary**

The quality of the sampled data and the pre-processing is a very important factor in the eventual results. Better input usually leads to better results.

The first step in obtaining good input is to use a tablet that returns the 5 most important characteristics: x and y coordinates, pen pressure, pen tilt and pen direction. Some cheaper tablets only return the first three coordinates (and some only the first two), but for best results all 5 are required (refer to chapter 5 for detailed results).

After the signature has been sampled, it has to be rotated, scaled and translated to fit on the model (or reference signature) to be compared. The accuracy of these processes also greatly affects the end result.

The process of iteratively scaling, rotating and translating a signature to obtain a good fit on the reference signature/model is referred to as fitting. One method used to obtain a good fit between signatures is to use DTW, which is discussed here in detail, and also employed in the SigGrab demonstration system.

## Chapter 4

### Hidden Markov Models

A Hidden Markov Model (HMM) is a stochastic model, which models a dynamic process as a sequence of states. Each state is associated with a distribution function and also a set of transition probabilities. The distribution function models the statistical distribution of samples within that state and the transition probabilities govern the transitions between states.

Determining the sequence of states for a given input signal involves an evaluation of the sample values with the distribution functions and a scaling by the transition probabilities. All samples are evaluated in this way, and the highest numerical result obtained for a sample corresponds to the state assigned to it.<sup>1</sup>

HMM analysis is a field of study in itself, and this concise definition will probably not be clear to anyone not familiar with similar techniques. The next few sections will attempt to give an explanation of the logic and methods behind it, starting off with an identification of shortcomings in other methods.

---

<sup>1</sup> Note that staying in the same state as a previous sample is also considered a transition, and thus also has a transition probability.

## 4.1 Identifying shortcomings in other methods

Up to this point in this thesis the most complex way in which signatures have been compared was dynamic time-warping (DTW). Samples in the signatures were compared to each other on a one-to-one basis and shifted in time to provide the best possible match. To obtain a measure of how well two signatures match, the sum of the Euclidian distances between the time-warped samples was used. Unfortunately, there are some characteristics of signatures that we do not take advantage of (or accommodate) in DTW analysis:

- In most cases, certain parts of the signature consistently vary more than other parts. We cannot weigh the comparison to place less emphasis on these segments of high variance. Consider the two signatures below, which illustrate this problem:

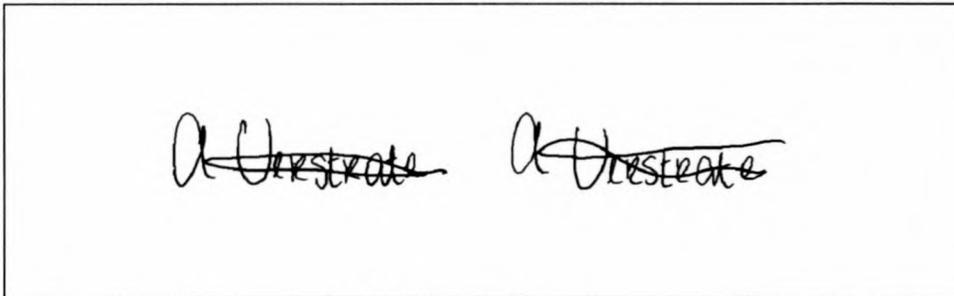


Figure 4.1: Segments with high variation

Note how well the two signatures match except for the final squiggle. Clearly we would want to disregard that in the comparison.

- Sometimes the writer may omit a letter or stroke. In DTW analysis every sample in the one signature is paired with another sample in the other signature, and if a letter or stroke were missing in one signature, one would likely get a very poor match – even if this may be a routine occurrence for the signer.
- While the strokes in a signature may vary in size and duration, there are usually predictable ranges of lengths and durations in which they fall.

If we compare only two signatures at a time (as in DTW) we cannot properly take advantage of the statistical properties of the signer's signature. We need to observe many signatures to be able to identify commonly occurring variations and the constraints of these variations for a given signer, and then measure up new signatures to this knowledge. Also, we cannot model these variations on a per sample basis, since there are infinitely many possible sample values.

One solution is to divide a signature into sample groups. These groups are obtained by dividing the sequence of samples from a signature into many smaller sequences – keeping the order of the sequences and the samples in the sequences intact. Consider the signature below, which has been divided into such sequences (colours denote the different sequences):

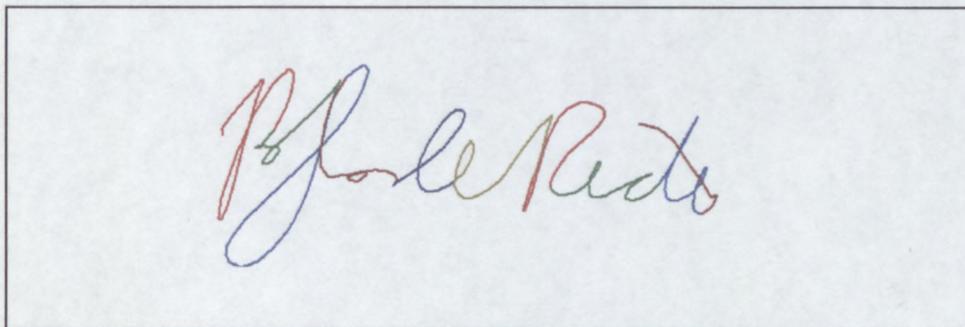


Figure 4.2: Signature broken down into sequence of groups

If we analyse many sample signatures in the same way, attempting to segment at corresponding points in all the signatures, we can start to draw statistical information from these groups. In the plot below we have attempted to fit a two-dimensional Gaussian distribution on each segment:

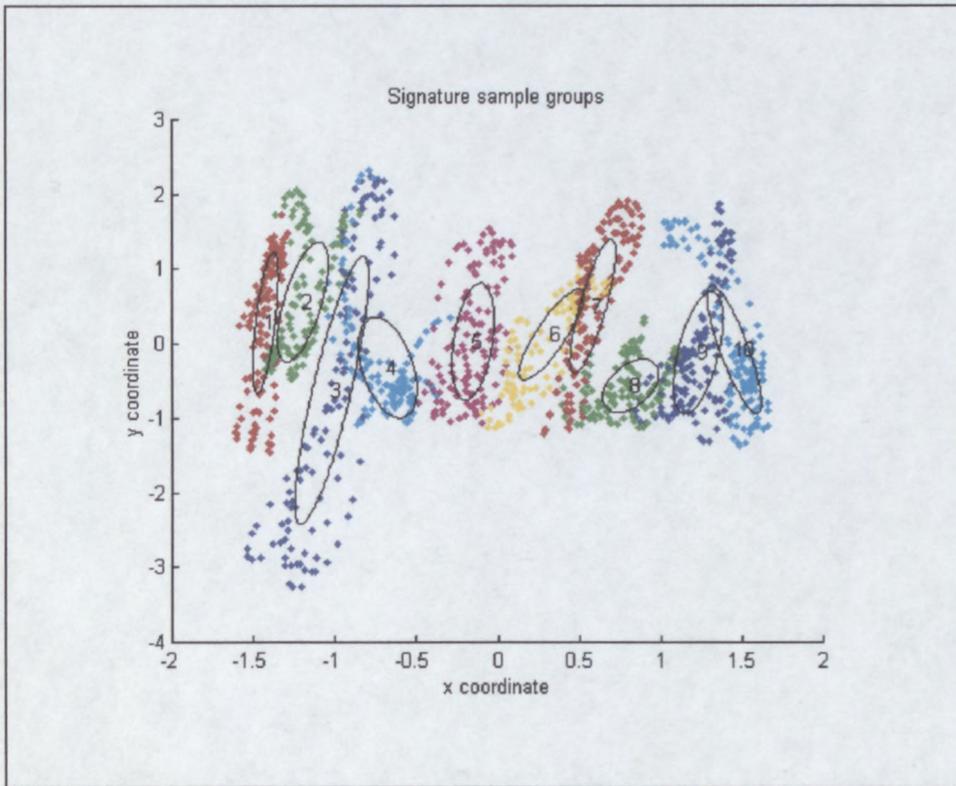


Figure 4.3: Signature sample groups

Here we only show the  $(x,y)$  coordinates, but of course this analysis applies to all the sample dimensions. The outline indicates the standard deviation for the associated distribution.

The statistical information we gather from these groups forms a model of the signer's signature. Some of this information is:

- The statistical distribution of samples for each sample group.
- The average number of samples in each group (i.e. duration of the group).
- The probability that the group does not occur (by considering in how many of the training signatures the group was omitted).

When a new signature is encountered, it is broken up into these same groups. The signature's groups are then compared to the model obtained above. How well it

matches determines the confidence with which we can say it is genuine. This forms the basis of HMM analysis.

Of course this method relies upon the fact that the grouping is done consistently between signatures. How this is done will be covered in the next sections.

## 4.2 Introduction to HMMs

HMM analysis can become quite complicated, and therefore it is difficult to give a complete explanation in a few short pages. Readers requiring more information should refer to the many references listed at the back of this thesis.

The previous section was an informal introduction to the ideas behind HMM analysis. This will now be formalised.

In HMM terminology, the “groups” of the previous section are called states. For each of these states the following information is kept:

- The expected statistical distribution of samples within this state. This can be any kind of distribution, although multivariate Gaussian distributions are most often used.
- The probability of the state occurring. This can be modelled in many different ways. Most often transition probabilities are kept for every possible state transition, i.e. when a sample is evaluated the following question is asked for every state: “How well does this sample fit in the distribution of this state, and what is the probability that this new state is assumed, given that the previous state was state  $x$ ?” The sequence of states is thus determined by how well the samples match the individual states, as well as the transition probabilities. These transition probabilities serve a double purpose: They help determine the sequence of states, and also the duration of the states (since staying in the same state is also a transition and is also assigned a probability).<sup>2</sup>

---

<sup>2</sup> In general HMM analysis any state may make a transition to any other state, but in the special case of signatures we usually restrict the sequence of states to be strictly left-to-right – since a signature has a

The way in which HMMs work is very reminiscent of DTW. In DTW each sample is compared to all samples in the reference to determine the best fit between the two signals. The total time-warping 'cost' would be an indication of how well the signals match. In HMM Analysis the signal is first broken down into a sequence of states (from previously determined state distributions and transition probabilities). In the process of determining the sequence of states the likelihood that this sequence of states can occur with this model is also determined. This gives an indication of how well the signal matches the model.

With these enhancements, HMMs have the potential to perform much better than DTW in determining how well a given signal matches a model. HMMs also have much greater potential to be extended and customised to fit a particular situation than DTW.

To simplify the HMM system, the assumption is made that the distribution at each state is fixed. That means that the probability of a sample to fall within that state is only dependent on the sample value itself, and not any previous sample or state. This simplification is necessary to reduce the problem of obtaining a model (and also of storing it) to a manageable level.

The order of the HMM determines how many previous states contribute to the transition probabilities. The transition probabilities for a first order model would thus only be dependent on the current state. In this discussion we will only be concerned with first order HMMs, since these are well suited to the HSV problem.

---

fixed order of letters. These transition probabilities are thus mostly used as a way to model the duration of the state.

The figure below shows a typical first order 3-state Markov Model:

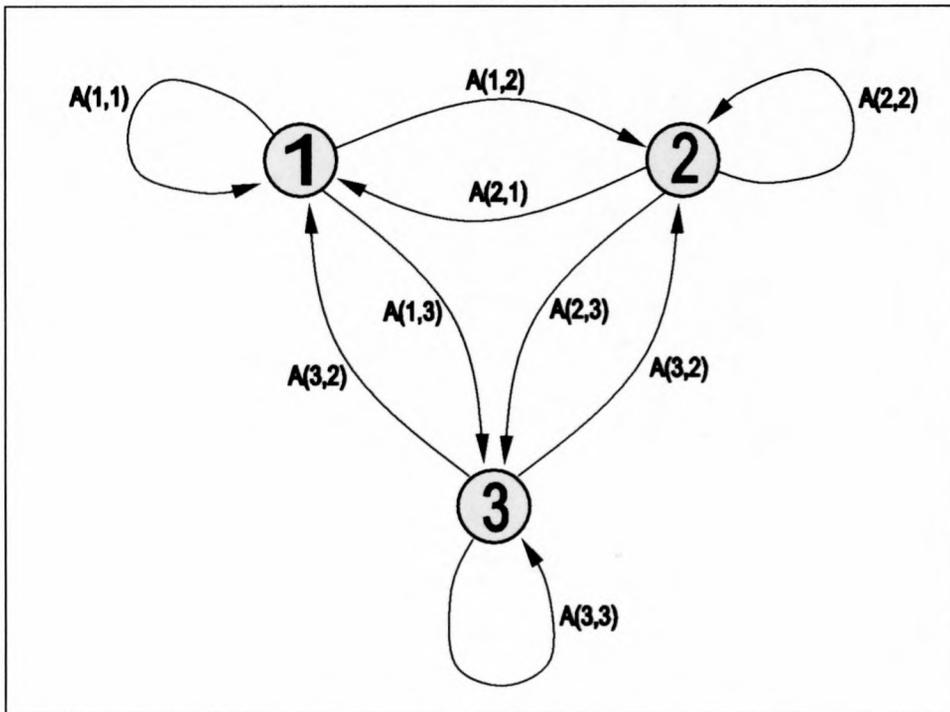


Figure 4.4: 3-state first order HMM model

Consider the 3-state hidden Markov model above. For each state there are three possible transitions (indicated by the arrows), corresponding to each of the three states (staying in the same state is also considered a transition). The transition probabilities are indicated by the elements of the transition probability matrix, which will be a  $3 \times 3$  matrix for this first order case. Element  $(i,j)$  would be the probability of the transition from state  $i$  to state  $j$  in two successive samples. For a second order model, there would be 3 of these  $3 \times 3$  transition probability matrices – one for each previous state.

With each state there is also an associated probability density function (PDF). Transitions do not solely depend on the transition probability, but also on how well the sample matches the distribution of each state. The transition probabilities are thus scaled according to how well the given sample value matches each state. These distributions can be anything, from simple Euclidian distance (as in DTW) to a mixture of different distributions. The distribution that is most often used, however, is a single Gaussian distribution.

### 4.2.1 Obtaining the most likely state sequence

Similar to DTW analysis, we want to obtain the sequence of states that best matches the input sequence (called the observation), i.e. the most probable state sequence given the model. In DTW the cost of the best path was used as measure of how well two signatures match, in HMM analysis we use the likelihood of this best state sequence to occur (given the model) as measure of how well the input sequence matches the model.

Ideally we would like to calculate the probability of the input sequence given the model, but this is a very time-consuming calculation. Where  $O$  is the observation,  $q$  a state sequence and  $\lambda$  the model, this can be summarized as follows:

$$\begin{aligned} p(O|\lambda) &= \sum_q p(O, q|\lambda) \\ &= \sum_q p(O|q)p(q|\lambda) \end{aligned}$$

To obtain the probability of the observation given the model, we must sum the probability of all possible paths. Since the number of paths in a fully connected model equals  $N^T$  (the number of states raised to the power of the number of samples), a straightforward computation is prohibitively complex. This can be approximated as:

$$p(O|\lambda) \approx p(O, q^*|\lambda) = \max_q p(O, q|\lambda)$$

where  $q^*$  is the most likely state sequence. The resulting probability is often called the *maximum approximation*. The Viterbi algorithm is used to obtain this best path in much the same way as the best path was obtained using DTW. A *likelihood matrix* is drawn up with one axis corresponding to the states and the other the sample sequence. The Viterbi algorithm is implemented as follows [5, 22]:

1. For each possible state of the current sample, the most likely state for the previous sample is determined. This is done by taking the likelihood to all the possible states of the previous sample, and multiplying it with the transition

- probability to the current state. The previous state with the highest result is the most likely previous state.
2. The likelihood from the previous state (already scaled by the transition probability), is now also multiplied by the PDF value at the new state to obtain the total likelihood to the current state.
  3. The total likelihood to the current state, as well as the state for the previous sample, is stored in the likelihood matrix at the current sample and state position.

This process is repeated for each sample and possible state, until all the elements of the  $N \times T$  (states by samples) likelihood matrix are calculated.

Since we kept track of not only the total likelihood to each state, but also the state for the previous sample, we only need to determine the final state and we can retrace the whole state sequence. The final state is chosen as the state corresponding with the highest likelihood. The most likely path may now be backtracked by following the “previous state” path. The likelihood to the final state can be used as measure of how well the input sequence matched the model – the higher the likelihood, the better the match.

Note that we speak of likelihood and not probability. Since we scale with the PDF value at each state and not the probability of the sample to fall in that state, we do not end up with a probability. Calculating the likelihood involves fewer calculations than calculating the probability. The likelihood of different input signals can be used to compare how well they matched the model, just as we would compare the probabilities. Since the likelihood is usually a very small number, logarithms of the probabilities and PDF values are usually used throughout the process and multiplications are replaced with additions. This prevents numeric overflow in computers with limited floating-point accuracy.

## 4.2.2 Obtaining a HMM model

Now that a method has been devised to determine the most likely sequence of states given an observation and model, we need to define a way to construct a HMM given a set of desired observations (genuine signatures in this case).

HMMs are trained in an iterative way with the use of the Viterbi algorithm. The Viterbi re-estimation training method works as follows [5, 22]:

1. For each training observation the most likely path is traced through the current model through use of the Viterbi algorithm.
2. All samples from all the training observations that fall within the same state are grouped and the PDF for that state is re-estimated from the statistical distribution of these samples.
3. The state transition probabilities are re-estimated in similar fashion. For all the samples deemed to fall in each state we count all the different transitions from that state. The probability of each transition is re-estimated as the total number of transitions of that kind, divided by the total number of transitions in the state.<sup>3</sup> The relative frequencies of all observed state transitions in the training data are thus used to re-estimate the transition probabilities.
4. The process is restarted at step 1 if the change in the model is greater than a pre-specified threshold. Typically the log-likelihoods of the training observations are summed to keep track of the relative magnitude in the change of the model. If the change in this sum between two successive iterations is smaller than the threshold, then the process is stopped.

This iterative process requires an initial model. This initial model need not be accurate, but the more accurate it is the faster the training process will converge. Since the HMM obtained will not necessarily be the best model possible (the training process may get stuck in a local minima), a good initial estimate goes a long way in improving the odds of a good final model.

---

<sup>3</sup> Since we count remaining in a state also as a transition, the total number of transitions for a state equals the number of samples in that state.

Initial state transition probabilities are chosen from previous knowledge of typical state transition probabilities for the problem at hand. The training observations are usually divided into equal sized blocks of samples, corresponding with the number of states. The initial PDFs of the states are then determined from the distribution of these arbitrary selections of samples. Again, knowledge of a probable progression of states, and a corresponding choice in samples per state do improve results.

### 4.3 HMMs as used in HSV

A type of HMM that suits signatures well (and is often used in speech analysis) is the left-to-right HMM. What this means is that the additional restriction is placed on the HMM that a previous state may never reoccur (i.e. transition probabilities to previous states are 0). This has the effect that the observation is forced to fit in a progression from the first state through to the last:

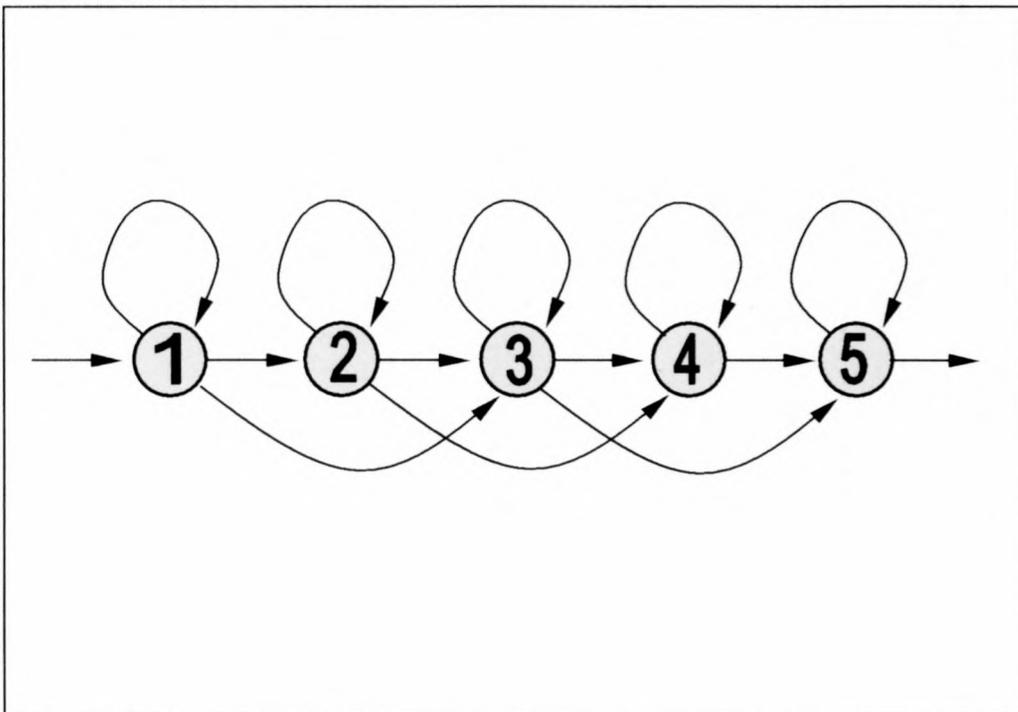


Figure 4.5: 5-state first order left-to-right HMM model

Since states are observed in a fixed order, the training and evaluation process is simplified and speed is improved. Furthermore, since left-to-right HMMs have fewer

possible state transitions, fewer training observations are typically needed to accurately determine the transition probabilities.<sup>4</sup>

Since signatures usually have a natural left-to-right progression, a left-to-right HMM is the natural choice. In the event that the same sequence reoccurs (for example when a letter is repeated), instead of jumping back to a previous state, a new state will be used that is a copy of the previous one. This should however not happen so often as to justify a HMM without the left-to-right restriction, since even the same letters usually vary within a signature. The extra storage required for the extra model states may be justified by the improved accuracy with limited training data, as well as faster speed.

In the example left-to-right HMM there is also provision for a *state skip*. This implies that apart from the normal probability of staying in the current state or progressing to the next state, there is also a probability for skipping the next state altogether. This also finds application in HSV, since there are signers that routinely forget to dot “i”s or put the stripe through a “t”. There are also instances where a signer fluctuates between different forms of their signature. This *state skip* provides limited support for these types of fluctuations, but in the event that a signer switches between different signatures altogether, then two different models will have to be generated – one for each signature. Note that it is not practical to support a double (or greater) state skip, since this would make the model more complex and would require even more training signatures. Double (or greater) state skips are too rare (unless way too many states are specified for the length of the signer’s signature) to justify the additional training data required.

The choice of PDF used to describe the distribution for each state is influenced by a few factors. Ideally we would like the HMM to automatically pick states so that the signatures are segmented into strokes. Since the edges of strokes correspond to natural breaks in a signature, the strokes provide an excellent basis for comparison. There are many examples of systems employing segmentation at stroke boundaries, followed by a comparison of these strokes. However, performing hard segmentation in the right

---

<sup>4</sup> The more parameters there are to a model, the more training data is usually required. This stands to reason, since an accurate statistical parameter can only be determined with sufficient data to describe it.

place was shown to be difficult in chapter 2. If the HMM is allowed to do *soft* segmentation in the statistically correct place, better results can be expected.

We thus choose a PDF that suits the predicted distribution of samples in a stroke. Choosing a PDF that suits the expected distribution in a stroke well, will further aid the training process in choosing the states to correspond with strokes.

Signatures are man-made and man has limited motoric ability, therefore the frequency content will be limited. Since the sampling rate used here is 200Hz, which is much higher than required in most signatures, all characteristic changes will appear gradual in nature and no sudden jumps should be apparent (with the exception of the pressure signal when the pen is lifted). The following observations were made after studying some typical strokes:

- The (x,y) progression will be a curve often approaching a straight line, so an oval-shaped PDF should fit the (x,y) coordinates well.
- The pressure signal often shows a slight increase followed by a decrease towards the end of the stroke.
- The pen tilt & direction usually shows either a gradual increase or decrease.
- The speed signal may show an increase towards the middle of the stroke, but the direction remains more-or-less uniform.
- The curvature takes on a low value in the middle part of the stroke with peaks on both edges, corresponding to turning points in the signature.<sup>5</sup>

Clearly the distribution within a segment is quite simple. There are no sudden jumps, so the need for a mixture PDF built up by two or more PDFs is not really necessary. Using a complex PDF may not be justified given the simplicity of the data and the computational expense that it may incur. Scarcity of training data may also lead to accuracy issues in complex PDFs.

In fact, a single multivariate Gaussian PDF for each state was found to work very well. It is reasonably simple to implement and it only requires the storing of a single

---

<sup>5</sup> This characteristic of the curvature sometimes leads to additional states being chosen at the edges of strokes. This is not necessarily an unwanted phenomenon, and may in fact enhance segmentation.

covariance matrix and mean vector per state. The equation for a multivariate Gaussian PDF is

$$f(x) = \frac{[C_x]^{-1}]^{\frac{1}{2}}}{(2\pi)^{\frac{N}{2}}} \exp \left\{ -\frac{[\vec{x} - \vec{\mu}_x]^T [C_x]^{-1} [\vec{x} - \vec{\mu}_x]}{2} \right\}$$

where  $\vec{x}$  is the sample vector,  $\vec{\mu}_x$  is the mean vector of the distribution and  $C_x$  is the covariance matrix [3].  $N$  here is the dimension of the distribution (and consequently the length of the sample vector).  $\vec{\mu}_x$  and  $C_x$  are given by

$$\begin{aligned} \vec{\mu}_x &= E[\vec{x}] \\ C_x &= E[(\vec{x} - \vec{\mu}_x)(\vec{x} - \vec{\mu}_x)^T] \end{aligned}$$

Two types of multivariate Gaussian PDFs are commonly used. The one is the *diagonal covariance* Gaussian and the other is the *full covariance* Gaussian.

In the case of the diagonal covariance Gaussian, all of the off diagonal elements of  $C_x$  are zero. This implies that the PDF will not be able to model data correctly if there is any cross-correlation between the different components of the feature vectors. As is shown in 3.2.5, there is some cross-correlation between the features in signature data, so a full covariance Gaussian distribution is used.

In one dimension, the Gaussian distribution takes the familiar bell-curve shape. In higher dimensions it can be visualised as a multi-dimensional oval.

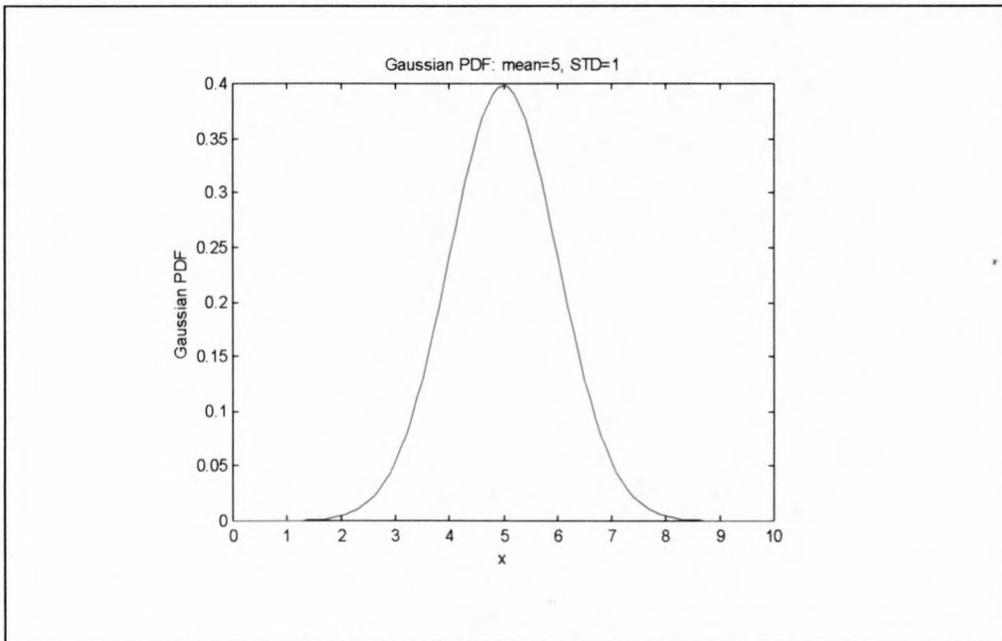


Figure 4.6: Gaussian PDF in one dimension

With the HMM model type chosen as well as the state distributions defined, there is one refinement left to be made. This refinement is *duration modelling*.

If we examine the same stroke in different signatures by the same person, it becomes apparent that this stroke always has more or less the same duration.<sup>6</sup> Assuming the system has assigned a state to each stroke during the training process, we have no way to take advantage of the knowledge that the duration of each state is most likely to fall within a specific range. Whether it is the first sample to fall in the state or the last, the transition probabilities remain the same.

If we extend our HMM theory to incorporate the idea of sub-states, we can solve this problem.<sup>7</sup> Each state is divided into a predetermined number of sub-states. Each of these sub-states share the same PDF as the original state, but the transition probabilities are different.

---

<sup>6</sup> For a constant sampling rate this translates to always having more or less the same number of samples.

<sup>7</sup> This implies that the HMM is no longer strictly of first order. [24]

The flow-diagram below illustrates this situation:

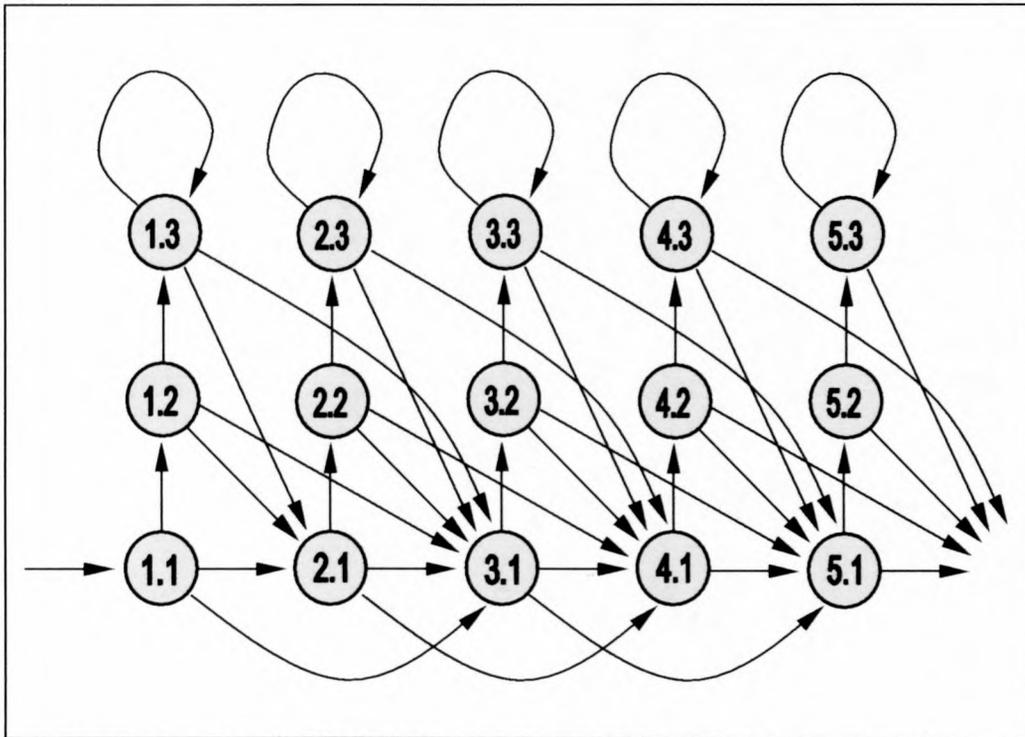


Figure 4.7: 5-state HMM with duration modelling (3 substates)

Looking at the figure above it may appear as if duration modelling has added a lot of complexity to the HMM. This is not true. There are still only three possible transitions from each state, and the number of PDFs are the same. The only additional data is the transition probabilities for the additional sub-states. Much of the training stage is unchanged and only slight modifications are needed to the existing Viterbi algorithm.

From the flow-diagram we can see that if a sample falls within the current state, that the next state will be the next substate of the current state. It is only until the end of the substate chain is reached that the HMM is allowed to remain in the exact same sub-state. It thus makes sense to pick the number of substates equal to the maximum number of samples that an observation is expected to remain in any of the states. The transition probabilities for each substate are calculated in the re-estimation stage of the training process by not only counting the transitions from each state, but also from each substate.

It should be noted that the system with duration modelling is more prone to data scarcity problems. In particular, less samples tend to fall in the higher sub-states than in the lower sub-states, which affects the accuracy of their transition probability estimations. This can be explained by noting that the higher sub-states correspond with longer state durations. Since there is no way of knowing beforehand how many sub-states to allow for each state, a fixed number (usually greater than the eventual requirement) is assigned to all states. In most states there will be few or no examples of the longest duration occurring. With little sample data to work from, the transition probabilities for these higher sub-states are difficult to predict accurately and tend to fluctuate between extremes (either 0 or 1). To combat this problem, a smoothing operation is performed on the transition probabilities of the sub-states. What is typically done is to low-pass filter the transition probabilities to prevent any sudden changes from one sub-state to the next. This does improve results somewhat, but is not a substitute for more training data.

If there is not enough training data available (typically at least 15 signatures are required), then the HMM model tends to over-specialise on the training set and rejects almost all other signatures. On the other hand, if the training set is poorly selected and includes signatures with too high variation or containing errors, then selectivity is lost and even poor forgeries will be accepted.

## 4.4 Summary

HMM analysis is a very powerful tool for the analysis of speech, signature and other signals that can be sampled and stored in digital format. However, the power comes at the cost of computational complexity. Because of this, using HMMs has only recently become practical. It is currently finding more and more applications in signal processing.

There are many variations and different configurations of HMMs that suit different situations. It is through experimentation and knowledge of the nature of the data that a specific configuration is chosen. What is given in this chapter is a configuration that was found to work well with signature data: First order, left-to-right with duration modelling.

For a more information on HMMs than that given here, refer to Deller [22], Rabiner and Juang [23] and Dolfing's [5] discussions on the topic.

## Chapter 5

### Experimental Evaluation

Proper evaluation of a HSV system requires a vast database of genuine signatures and forgeries to be able to simulate a real-world situation. Signatures obtained in a laboratory environment may not reflect signatures from the population as a whole, and signatures gathered on the same day usually show less variation than signatures gathered over a longer period of time. There are no known freely available signature databases, and building up a sufficiently large signature database is a long and arduous task.

J.G.A. Dolfing has graciously donated the signature database that he built up to test his HSV system. It consists of 4800 genuine signatures and forgeries for 51 different signers. This system is tested against this database as well as signatures and forgeries collected in-house for one test subject.<sup>1</sup>

#### 5.1 Performance with J.G.A. Dolfing's Signature Database

Dolfing's database consists of signatures captured by a specially designed tablet, which returns all 5 dimensions: x, y, pressure, pen-tilt and pen-direction. The sampling rate for all signatures is 120Hz. For each signer there are between 90 and 110 signatures, which are categorized as follows:

- 15 genuine signatures to be used for training purposes
- 15 genuine signatures for testing purposes

---

<sup>1</sup> This gives us a benchmark against which to measure the system. Unfortunately Dolfing's signatures were all captured on the same day with similar scale and rotation, which does not reflect a real-world scenario.

- Up to 20 professional forgeries. These are forgeries made by forensic experts and professional signature analysts after a careful study of the original.
- 30 *over-the-shoulder* forgeries. These are forgeries made immediately after the forger witnessed the test subject producing genuine signatures.
- 30 *home-improved* forgeries. These are forgeries made after the forger was allowed to study and practice the test subject's signature.

The great similarity between the training and test signatures for Dolfing's database suggest that they were all made on the same day, so the variation of genuine signatures with time can not be investigated with this database. This means that results obtained will probably be more optimistic than can be expected in practice. On the other hand, there are a couple of genuine signatures in the training and test sets that contain obvious errors, which should have been rejected outright. Furthermore, there are signers with very primitive signatures that contain too few distinguishing features for any means of reliable verification.

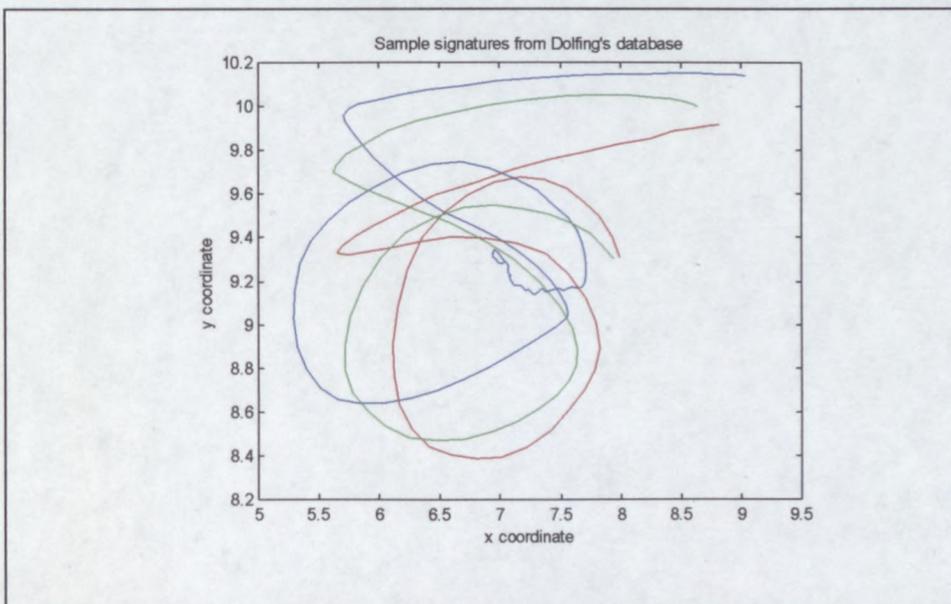


Figure 5.1: Sample signatures from Dolfing's database

The three signatures above are from the training set of one of the signers. Note the high variance in the signature that already contains few distinguishing characteristics. The signer wavered for a long time at the end of the blue signature, which is an obvious mistake and should have been removed from the training signatures. While

such a mistake may not prove too troublesome for a visual verification process, it does wreak havoc with the duration modelling of the HMM.

Nonetheless, the signatures were used “as-is”. No signatures were removed or modified before being used, allowing a direct comparison between these results and those obtained by Dolfing.

A conversion program was written to convert the signature database into SigGrab’s native format. Models were trained with SigGrab’s *AutoHMM* function. The number of substates were fixed at 32, with SigGrab allowed to pick the number of states according to the formula:

$$\text{NumberofStates} = \frac{\text{AverageSignatureLength}}{5}$$

Varying the number of HMM states in this way prevents unnecessary overhead with shorter signatures, but also allows complete modelling for longer signatures. Few strokes are shorter than 5 samples, so soft segmentation at stroke boundaries is possible.

Dolfing reports an average EE of 1.9% for his system, which is incidentally also based on HMM analysis (see chapter 4). With the same data, an EE of 1.02% was obtained with SigGrab. It should be noted that since there are only 15 test signatures per signer, that the FR curve is heavily quantized with a single falsely rejected signature representing a 6.67% error. With more test signatures, the average EE is expected to drop even lower. For 40 of the 51 signers, SigGrab reported an EE of 0%, implying that signatures for those signers will rarely be classified incorrectly (with a suitable choice of threshold). The signatures for the 11 signers that produced errors are shown below:



Figure 5.2: Problematic signatures

The signatures that produce errors are generally of shorter duration than average (less than 1.25 seconds), and also exhibit high variability. Some incorrect classifications are caused by errors in the training and/or test signatures that should have been removed in a screening process. It is also apparent that the initial scaling and fitting process struggles with signatures that conclude with highly variable underlines or squiggles. In fact, skipping the fitting stage improves results for these signatures somewhat, since the signatures are already well scaled and positioned.

While EE rates are often cited as a measure of performance, it most often does not correspond with the desired operating threshold for a practical implementation. The

choice of a proper decision boundary for a signer and allowable FAR/FRR is part of the verification problem, and this is not reflected in the EE rate.

Three additional factors are investigated:

- The decision boundary that produces the lowest number of incorrect classifications (FA and FR combined).
- The FAR at the decision boundary that produces a 0% FRR.
- The FRR at the decision boundary that produces a 0% FAR.

We may tweak the system by manually choosing a decision boundary for each signer, in order to minimize the number of incorrect classifications. After doing this, a total of 17 genuine signatures are rejected and only 2 forgeries are accepted. This is out of an evaluation dataset of 4035 signatures (765 signatures were used in the training process), which translates to an accuracy of 99.53%! The table below summarizes the best accuracies obtainable with Dolfing's database for various combinations of measurements:

Measurements used	Best accuracy obtainable with 15 training signatures.
x, y, pressure, pen tilt, pen direction	99.5%
x, y, pressure	93.7%
x, y	89.5%
pressure, pen tilt, pen direction	92.7%

Note the high accuracy obtained (92.7%) with only the non-visual components of the signature data available. This again shows that it is much harder to forge the dynamics of a signature than its form.

Adjusting the individual decision boundaries (thresholds) so as not to reject any genuine signatures (0 FRR), the corresponding FARs are measured and averaged. By virtue of the fact that there are some errors in the test and training sets, the resulting FAR for a 0% FRR is (on average) 6.6%. It is clear that a system that never rejects genuine signatures will have a high FAR, since people make mistakes and there will

always be bad genuine signatures. Interestingly, the over-the-shoulder forgeries cause the most problems for the system. This is probably due to the fact that the dynamics of the signature are still fresh in the forger's memory after just witnessing the signing process.

Of greatest interest is the threshold that rejects all forgeries. We thus seek the FRR for a 0% FAR. The average FRR for a 0% FAR is obtained as 2.75%, with a peak value of 20% for the most problematic signer. Provided that the signature database is representative of real-world signatures, these figures are indeed encouraging. A system like this would be very useful to banks and other financial institutions that lose billions each year due to cheque fraud. With only 1 in 36 genuine signatures rejected, the irritation factor to the client should be minimal. Unfortunately the sample dataset is too small to accurately predict the probability of successive rejections (successive signatures are not necessarily statistically independent), but it should be minimal.

The next graph shows the best accuracy obtainable against the number of states for a few of the signers from Dolfing's database. It is interesting to note that the shorter signatures (red, magenta and blue) reach their maximum accuracy levels with fewer states than the longer and more complex signatures. This is because less states are needed to accurately model shorter signatures. After a certain point very little is gained by increasing the state count, and accuracy may even drop as the system becomes over-specified.

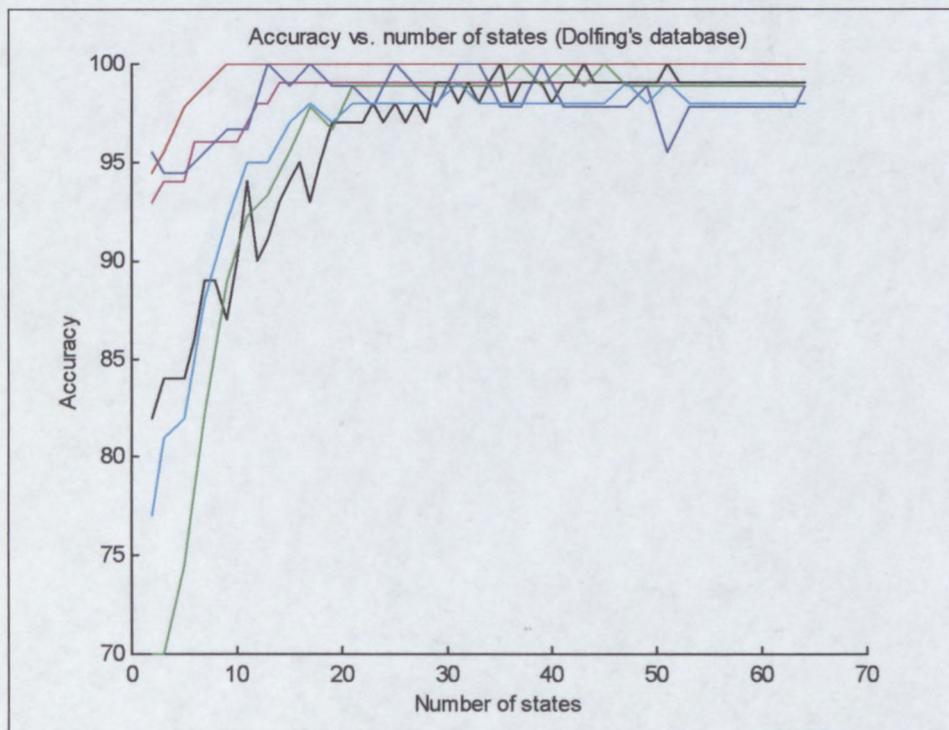


Figure 5.3: Typical accuracy graphs for Dolfing's database (plotted for 6 of the 51 signers)

The accuracy graphs for all 51 signers in Dolfing's database follow this trend very closely.

Up to now, however, the decision boundaries were chosen with prior knowledge of the distribution of genuine signature scores. Since the training signatures are used to determine the model, their scores will be unrealistically high and thus of limited use in determining a suitable threshold beforehand. Another set of signatures will be required to determine the spread of genuine signature scores with the given model. Unfortunately, since there are only 30 genuine signatures for each signer and 15 are already being used for determining the HMM model, there are too few left to evaluate the system and still provide a minimal test set.

For most of the signers there is a vast gap between the forgery and genuine signature scores, so the system is usually not very sensitive to the choice of decision boundary. Taking the decision boundary as half of the average score obtained with the training signatures, still produces a respectable 97.7% accuracy (90 errors out of 4035

signatures). There are other options available that give better results, like first training with only 10 signatures and then evaluating the remaining 5 to obtain a feel for the expected evaluation scores and then continuing training with all 15. The choice of decision boundary is only a problem when limited training signatures are available, and in a real-world application, where signers will sign numerous times over a period of time, the HMM model (and statistics regarding expected evaluation scores) can be updated continually.

The *SecuriCheque* program was developed to demonstrate the feasibility of this technology in a real-world signature verification scenario. *SecuriCheque* facilitates the generation and printing of cheques and incorporates the signature verification engine of SigGrab. A cheque will only be printed if the signer produces a sufficiently accurate signature.

*SecuriCheque* enhances the SigGrab engine by automatically updating the HMM model in the background as new signatures are added. *SecuriCheque* has no enrolment phase<sup>2</sup>, but instead builds and expands its training set as it is used. All signatures that are accepted by the current model are added to the training set and used to retrain a new model. The evaluation score obtained with a new signature and the previous model is also stored. The distribution of these evaluation scores gives a very good indication of the distribution of future genuine signatures. *SecuriCheque* employs a decision boundary of the average evaluation score less two times the standard deviation. This was found to produce good results.

---

<sup>2</sup> The system does require prior knowledge of the identity of all signers and the appropriate bank accounts, but no training signatures are required.

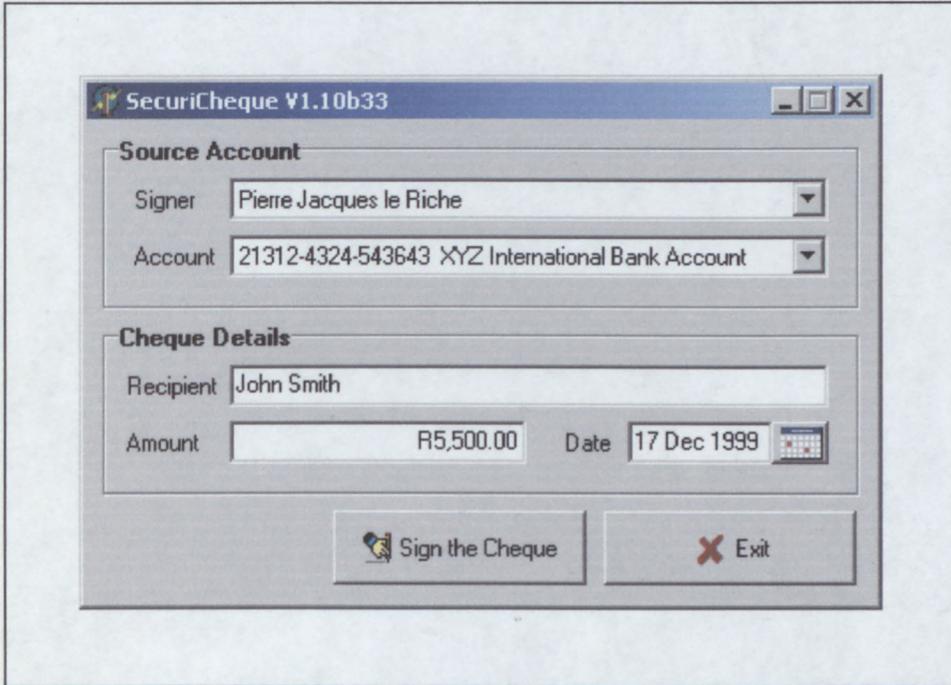


Figure 5.4: SecuriCheque application

Of course this system is initially vulnerable to forgeries, but as soon as the signer has made out 15 or more cheques, the system becomes very robust. By continually updating the model with new signatures, the system also adapts to changes in the signer's signature over time.

## 5.2 Performance with In-House Data

Dolfing's database appears to have been sampled all in a day or two, which means that the effect of variations in signatures over time is neglected. The quality of his forgeries are high, since forgers were allowed to witness the signing process and practice producing forgeries, but it is still not a worst-case scenario.

With the in-house data we try to produce the worst possible situation for a HSV system. Not only is the forger allowed to witness the signing process, and to practice it, but the signer also provides active coaching to help produce a better forgery. Furthermore, the forger has access to all the signature graphs and signature replay functions available in SigGrab. Forgers are instructed to use the signature evaluation functions of SigGrab and to submit only their best attempts. Forgers are also allowed to trace genuine signatures onto the tablet. It is a concerted effort to try and "break" the system.

On the other hand, genuine signatures are not filtered at all. Obvious errors are not submitted, but all other signatures are kept. The result is a 500 signature strong database (with an equal number of forgeries and genuine signatures) containing near-perfect forgeries, all collected over more than a 12-month period.

The signature of the test subject can be considered moderately easy to forge. It is shorter than the average signature (about 1.5 seconds in duration, 300 to 350 samples at 200Hz), and contains no complicated features that make it difficult to forge. It also exhibits a moderately high level of variation between genuine signatures.

Given that the top three signatures in the next figure are genuine, most people find it difficult to tell which of the rest are forgeries:

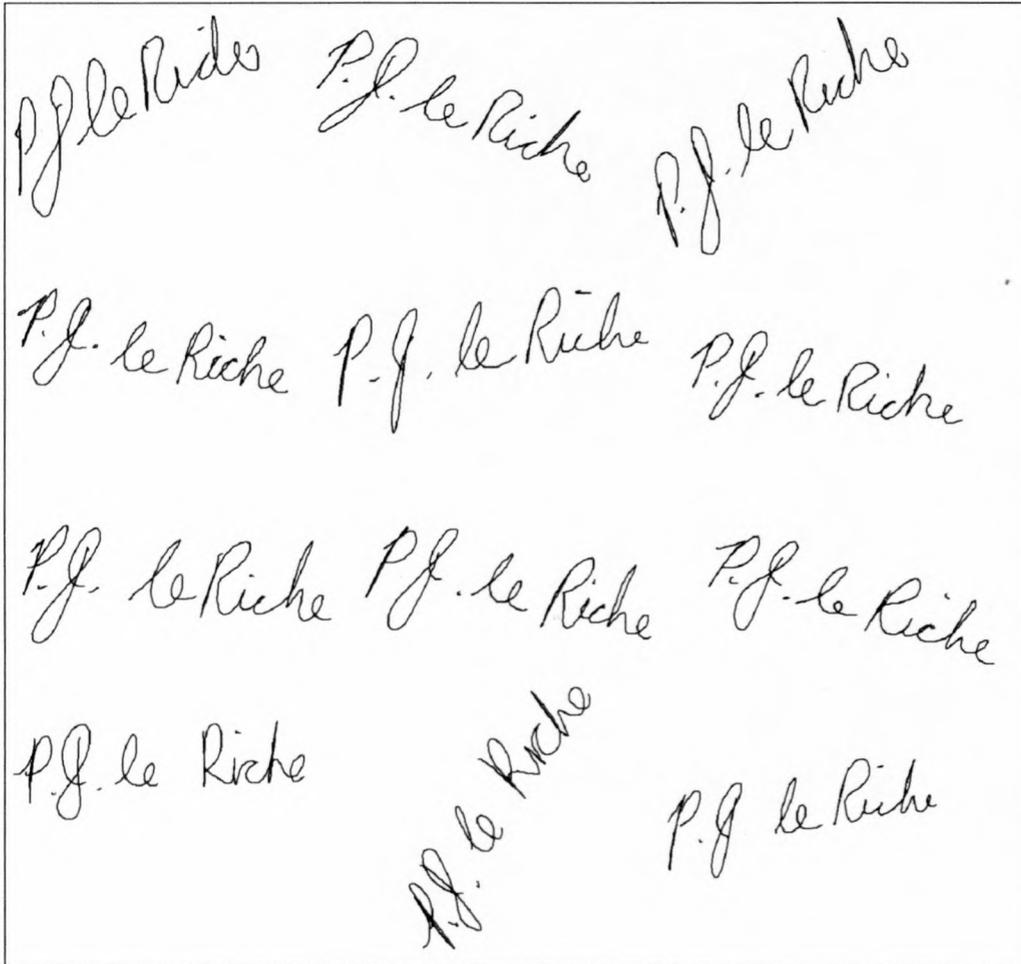


Figure 5.5: Sample signatures

Of course, all the signatures except the top three are forgeries, but all of them would probably have passed casual inspection. This is just an indication of the quality of the forgeries.

While a person might not be successful in discerning the forgeries from the genuine signatures in this case, it is testament to the power of the system that it is able to pick out the subtle mistakes in the forgeries and correctly identify them as such. With the dynamic information available, the system is indeed better than humans in verifying signatures.

The accuracy results obtained when evaluating with such high quality signatures will be much lower than can be expected in practice, so it is not of much use in comparing the system with other competing products. It will, however, give a valuable insight

into the strengths and weaknesses of the system. The fact that the signatures have been gathered over such a long period also allows some interesting experiments, which would otherwise not be possible.

### 5.2.1 Initial Results with a Static Model

For the first experiment the first 15 genuine signatures were used to train a 40/40 (40 states and 40 sub-states) model. All subsequent signatures and forgeries were then evaluated against this model. This test is conducted to simulate the situation in which no updating of the HMM model occurs after the initial enrolment phase. After the elapsed 12 month period, the results are expected to have deteriorated with the outdated model.

Below are the evaluation results for this test:

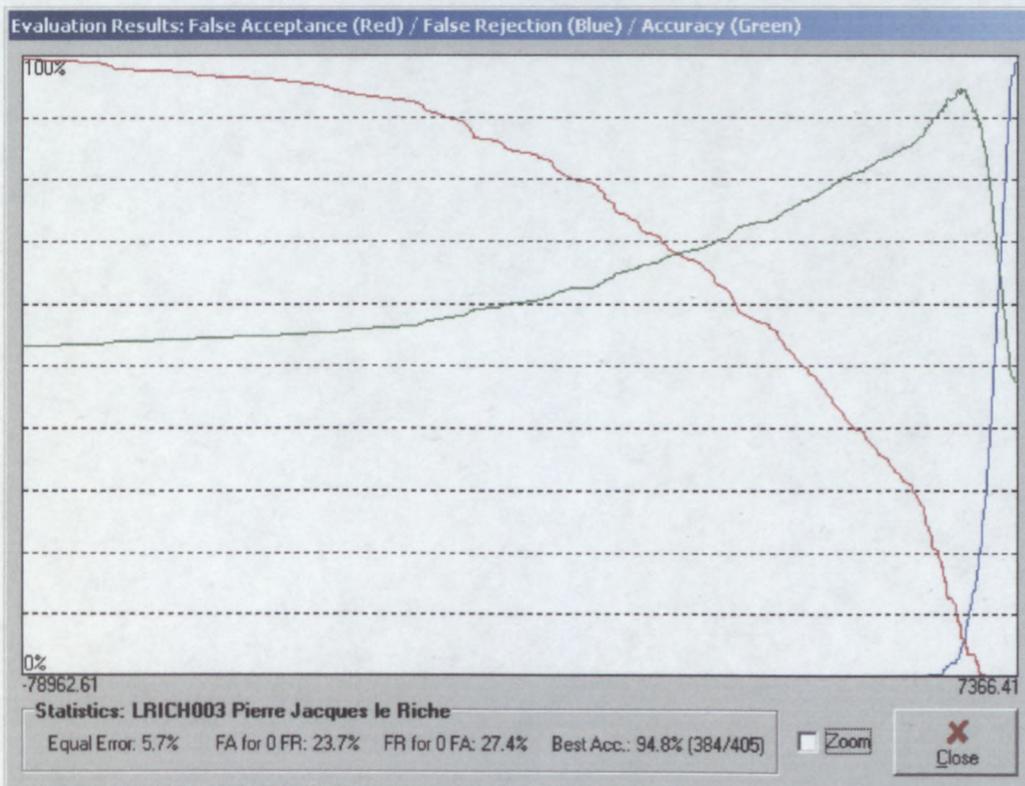


Figure 5.6: Results with first 15 signatures as training set

The system still performs admirably even with a model that is 12 months old, with an accuracy of just under 95%. Just over a quarter of forgeries will be accepted if a threshold is chosen so as to almost never reject a genuine signature. Similarly, a quarter of all genuine signatures will be rejected if virtually no forgeries are to be accepted.

The following plot shows the scores of the genuine signatures vs. the date that they were made<sup>3</sup>. All dates are relative to the date on which the training signatures were made:

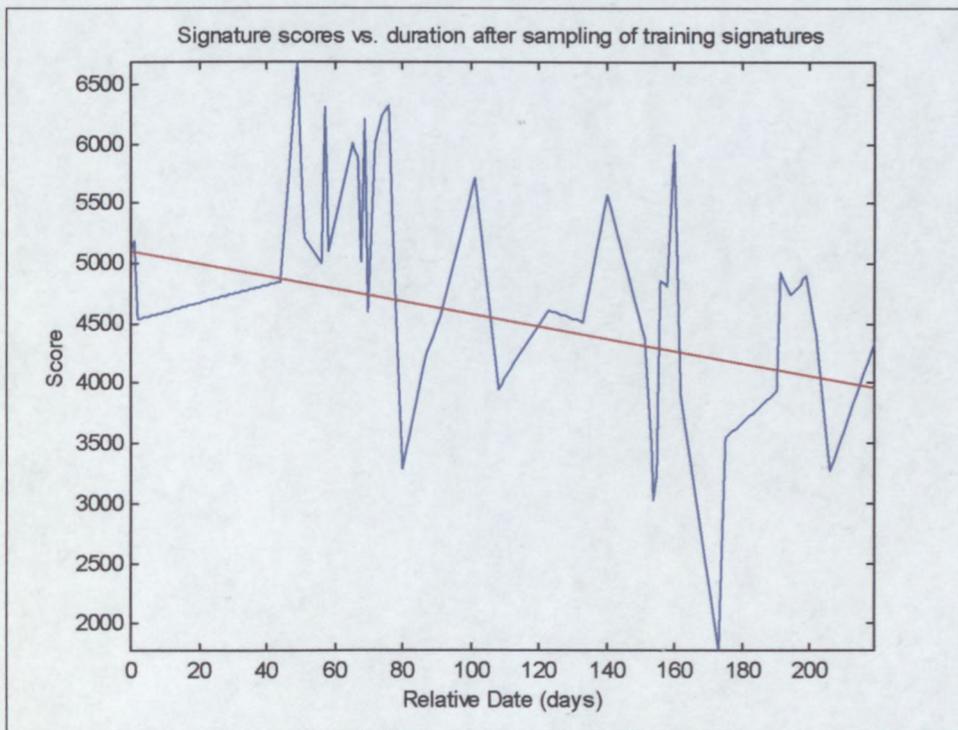


Figure 5.7: Signature scores vs. duration after training

Not surprisingly, the scores tend to become lower as time increases from the date that the training set were made. The red line, which is a least-squares approximation, clearly shows this tendency. It is interesting to note that the system functions well up to just over 2 months after the creation of the model. Shortly thereafter the scores start to fluctuate wildly. It appears that on roundabout day 80, the first variation of the signature that was not previously seen is observed, resulting in a low score.

<sup>3</sup> On days that more than one signature was made, the average score is used.

This signer may not be representative of signers in general, but it does suggest the following conclusion: A model for a signer's signature will continue to function well until such time as the first, and previously unseen, variation in the signature is introduced. For this signer it appears as if his signature remained the same for about two months after the training set was made, after which the first new variations started to appear.

Observe that there are still instances after this initial period where signatures obtain high scores. This can be explained by noting that not all variations will necessarily occur at the same time, and there may be days on which none of the "new" variations appear. It stands to reason, however, that more and more new variations will appear as time goes by, eventually rendering the old model unusable.

A HMM is a statistical model, and the scores that are obtained for signatures are indicative of their probability given previous observations (the training set). Forgeries are identified by identifying variations that were not observed in the training set. This is what makes new, previously unseen, variations in a genuine signature fatal. It also explains why the HMM-based system typically needs more training data – it needs to be aware of all possible variations to be able to predict their likelihood.

The next plot shows the overall system accuracy as a function of time after the training signatures were made (based on a single signer):

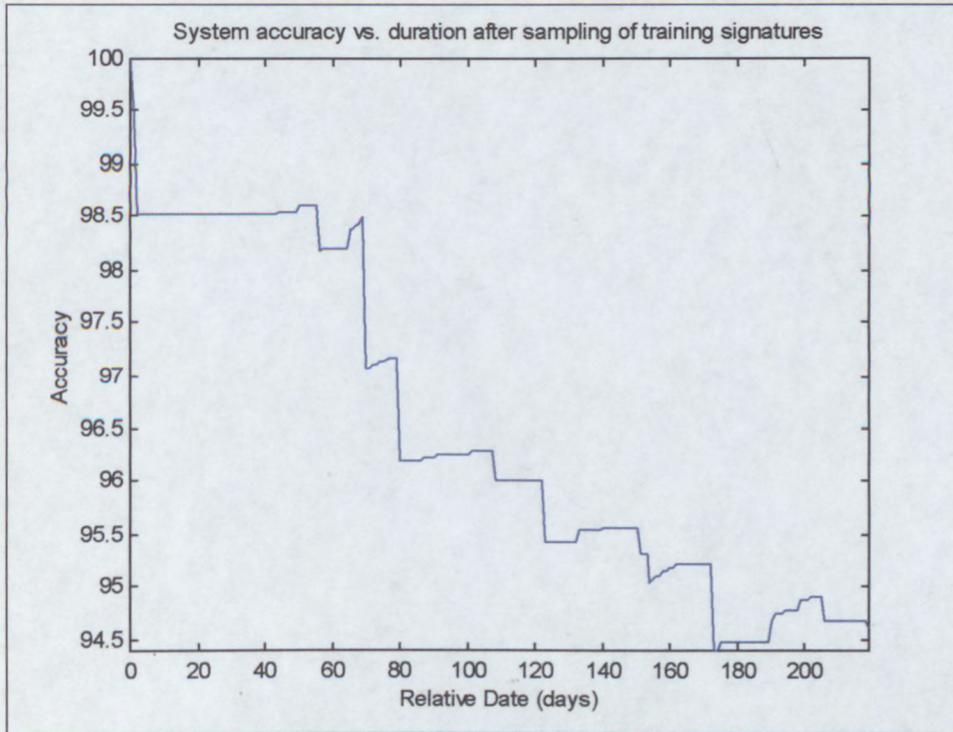


Figure 5.8: System accuracy vs. duration after training

Here the degradation of system performance as time goes by is even more clearly illustrated. Initially the accuracy is very close to 100%, but drops off quickly after about 60 days.

### 5.2.2 Optimising the System

From the results in the previous section, it appears that an accuracy of at least 98% is obtainable with an up-to-date model and the signature data in question. For an outdated model the accuracy drops to around 95%.

The initial model was trained with 40 states and 40 sub-states, however there is no guarantee that these are the best parameters for this signer. It would be very useful if some method of determining the optimal model parameters for a given signer could be found.

While an excessive number of sub-states should not bear any negative effects (except for greater memory usage and slightly increased calculation time), too few sub-states

will undermine the duration modelling. In the next experiment, the sub-states are kept a constant at the maximum value of 64, while the number of states is varied from 2 to 64. The training set is selected by randomly picking out a set of genuine signatures.

The aim of this experiment is to determine how the number of states affect the performance of the system. The plot below shows the results for a training set of 15 signatures:

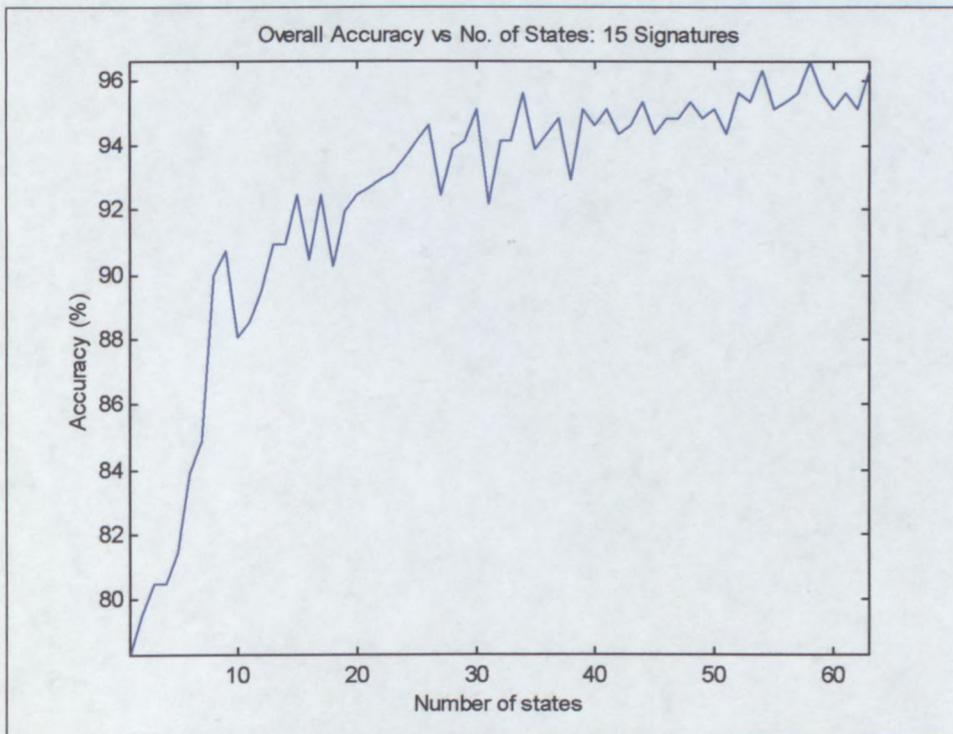


Figure 5.9: Accuracy vs. number of states (15 training signatures)

The graph shows a roughly logarithmic progression from the minimum of 2 states through to 64, although it does appear to flatten out above 30 states. Since the same training set is used throughout, the fluctuations in the graph (which appear to be more prominent for lower state counts) may come as a surprise. It can be explained by returning to the underlying principle, and that is that the system attempts to model the probability and parameters of all observed variations. With limited number of states, the states may not be sufficient to model all these situations. While it may provide a reasonable description of the training set, a slight change in the assignment of states could quite possibly cause a major change in the accuracy with the test data. Since the training data is initially divided into equal parts before training starts, and the training

process may well end up in a local optimum, the initial division may also play a major part in causing these observed fluctuations.

For the minimum of 2 states a reasonable accuracy of 78% is obtained. While two states cannot model the exact strokes, it can estimate the general size and shape of the signature as well as the averages of the pressure, pen tilt, pen direction and signing time. This limited information is already sufficient to correctly judge more than three quarters of all signatures.

The best accuracy (96.5%) is obtained with 58 states, although the possibility that slightly better results may be obtained above 64 states cannot be ruled out. However, the computational overhead as well as memory requirement does not make higher state counts practical with current desktop computers.

Since the computation time is proportional to the number of states, the specific application will have to dictate whether the apparently small gain in accuracy above about 40 states will justify the increased computation time.

This experiment is repeated below for different training set sizes varying from 5 to 30 signatures, in an attempt to determine the optimum number of training signatures (for this signer):

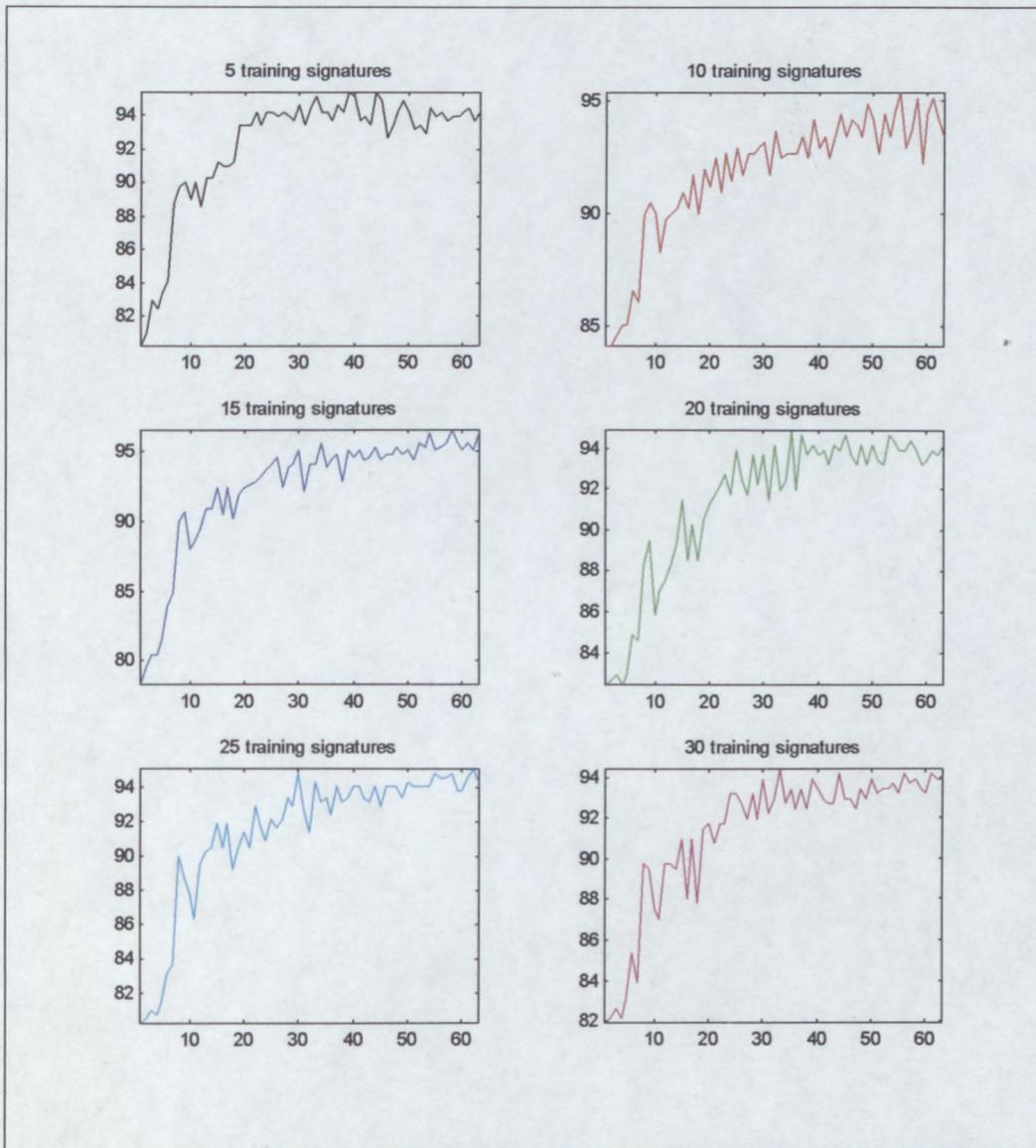


Figure 5.10: Accuracy vs. state count plots for various training set sizes

Interestingly, all the results show the same dip in accuracy for a state count around 12 states. This again points out the impact of the initial division of the training data before the actual training starts. There must be a particularly “tempting” local optimum that the system struggles with when the training data is initially divided into 12 equal parts and states assigned accordingly. This also suggests that results may be improved by performing a more judicious assignment of data to the initial states, maybe employing hard segmentation in the initial assignment process.

If the results are passed through a low-pass filter (to smooth it somewhat) and plotted on the same graph, it is much easier to compare their performance:

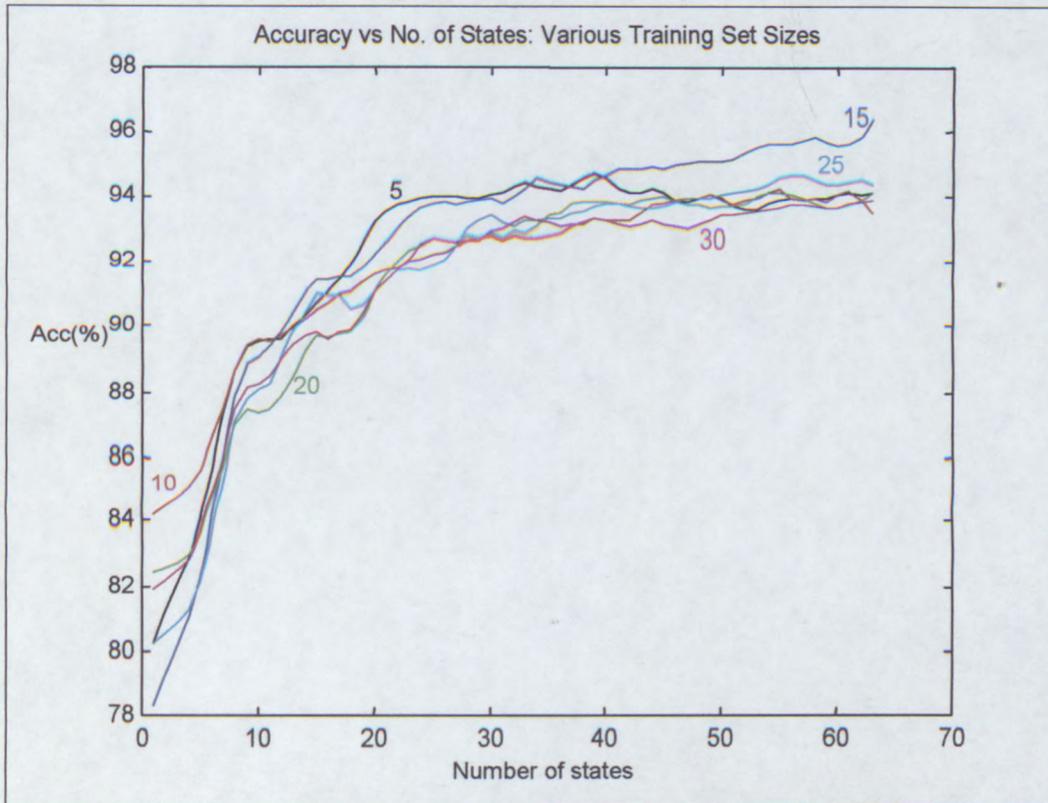


Figure 5.11: Accuracy vs. state count plots on the same graph

The colour assignments in the above plots are the same as in the previous figure. The plot for 15 training signatures (blue) stands out above the other graphs and shows almost a 2% lead in accuracy for high state counts. While this 2% is still within the margin of error for these experiments, it does raise the suspicion that either the pre-processing stage struggles with large numbers of signatures, or that there is a particularly poor training signature between the 16<sup>th</sup> and 20<sup>th</sup> signature used in this test.

The accuracy with very few training signatures (black and red curves) surpasses that of the other curves for low state counts. Since there are fewer signatures and thus variations, the system is able to more accurately model the signature with the number of available states. This accuracy drops, however, for large state counts when the system becomes “overspecified”, i.e. too many states are available for the quantity of training data to support it. This drop-off can also be expected for larger training sets and state counts above 64.

From these results the following conclusions can be made:

- When limited training data is available, the state count should be reduced to prevent overspecification of the system. Even for larger training sets, the state count should be chosen with consideration of the average length of the training signatures.
- While an increase in state count usually improves results, the gains become progressively less and less, and accuracy will eventually start to drop as the system becomes overspecified. The computational cost dictates the number of states that can be afforded. Choosing the number of states to correspond with the number of strokes in the signature appears to work well.

As mentioned earlier, the number of substates has little impact on the computation time, and should be made as large as possible to prevent insufficient substates from degrading performance. The plot below shows the effect of the number of substates on a 32-state model obtained from a 15 signature training set:

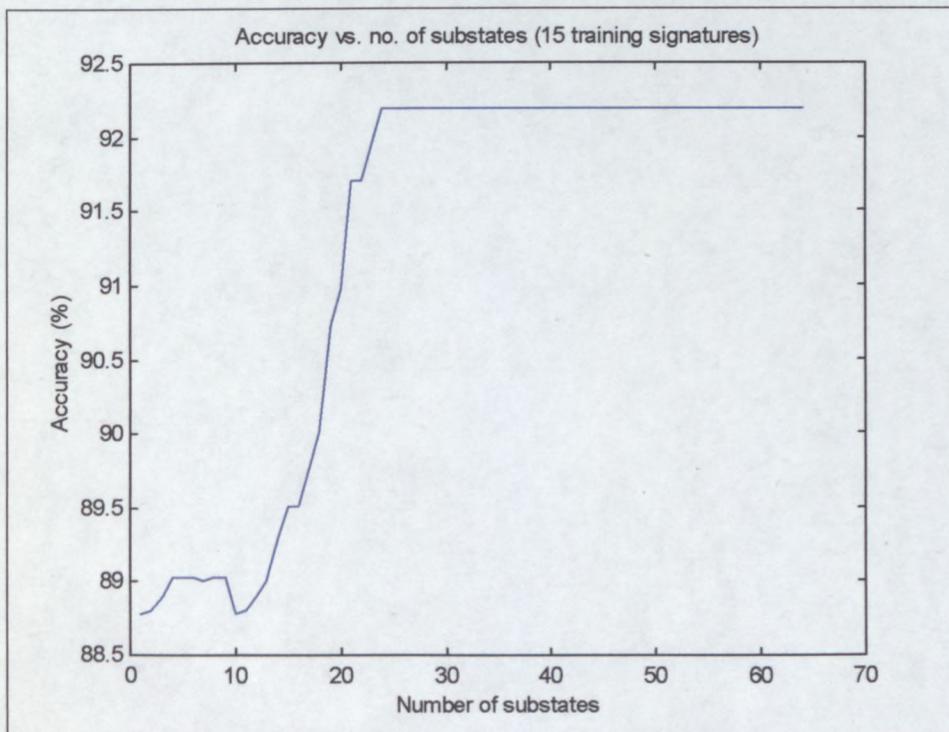


Figure 5.12: Accuracy for various numbers of substates

From the graph it is clear that duration modelling (the provision for substates) does provide a significant improvement in accuracy. Playing it safe by choosing an excessive number of substates carries no penalty, apart from a slight increase in memory requirement and computation time. For this signer it appears that if provision is made for the number of substates to exceed twice the average number of samples per state (11 samples), that optimal results are obtained. This makes sense, since some states will have longer durations than others and providing headroom equalling double the average duration should be sufficient.

Up to now the input signal was assumed to contain all 5 principal dimensions: x-coordinate, y-coordinate, pressure, pen tilt and pen direction. Tablets that return all these measurements are generally more expensive than tablets that omit some of them. Tablets that are unable to return the pen angles are typically less than half the cost of the more sophisticated ones, possibly making it a more cost-effective solution for some implementations.

The table below summarises the results obtained with a 32/32 model and various combinations of measurements used in the training and evaluation processes:

Measurements used	Accuracy obtained with 32/32 model trained with 15 signatures.
x, y, pressure, pen tilt, pen direction	92.2%
x, y, pressure	78.0%
x, y	73.2%
pressure, pen tilt, pen direction	88.0%
pen tilt, pen direction	86.3%
pen tilt	90.0%
pen direction	83.7%
Pressure	87.1%

There appears to be a large variance in the results depending on which signatures were used in the training set. The results tend to vary by  $\pm 5\%$  (or more) with different training sets. Nevertheless, some conclusions can be drawn from these results that

may be applicable to other signers as well. Take note that these results were obtained with the same unrealistically good forgeries (combined with some poor genuine signatures) of figure 5.4 and are thus not representative of the system's true potential.

Firstly, it is clear that the system experiences great difficulty in detecting forgeries (the FAR increases dramatically) for this signer when only provided with the (x,y) coordinates. This is not surprising, since the forgers generally tend to focus on recreating the image of the signature as accurately as possible, with the other dimensions of secondary importance. With forgeries of lesser quality and for signers with more complex signatures, the coordinates may prove more useful in the verification process.

The non-visible dimensions (pressure, pen tilt and pen direction) generally aid the verification process a great deal. The pen direction is almost a static signal for this signer, so it does not perform as well as the other two dimensions. Since this signer holds the pen at an unusual angle the pen tilt is particularly useful in picking out forgeries.

The phenomenon that the x, y and pressure combination fares worse than the pressure signal on its own, can probably be attributed to the observation that there appears to be greater variability in the signer's (x,y) signature profile than the pressure signal. This causes genuine signatures with (x,y) variations not seen in the training set to be rejected, while their pressure profiles may fit the model well. This effect should be reduced if the model is kept more up-to-date.

SigGrab provides various angle transformations for experimental purposes, but these tend to degrade performance more often than not. These methods introduce additional correlation between the pen-tilt and pen-direction, which is not a desired side-effect and degrades performance. Smoothing the curvature signal may provide a more visually pleasing graph, but also introduces correlation – this time between adjacent samples, which also degrades performance.

Since a signature model can be stored, the training process is not necessary every time a new signature is submitted. Therefore the speed of the training process is a minor issue, and the system is usually allowed to converge on a solution rather than limit the number of training iterations. However, the system rarely requires more than 15 iterations to come very close to the final model.

Even without any attempts at optimising the system for this particular signer, the verification accuracy remains above 95%, which is very encouraging given the quality of the forgeries.

## **5.3 Summary**

There are very few commonly available signature databases that can be used to compare HSV systems. This system was tested using J.G.A. Dolfing's database of roughly 5000 signatures and signatures gathered in-house with the SigGrab system.

The effect of the passage of time between when the model was trained and the test signatures are taken is investigated, as well as the effect of different model parameters and number of training signatures.

The results obtained are very encouraging, and are certainly better than any other results reported in the literature. It appears that an average accuracy in excess of 99% may be expected in a real-world application, which may make this system useable in situations where HSV was previously not feasible.

With some tweaking of the system and a few enhancements that are discussed in the next chapter, these results should improve even further.

## Chapter 6

# Conclusion and Recommendations for Future Work

In this research it is shown that signature verification based on HMM analysis without any form of hard segmentation indeed performs better than any of the other HSV methods currently documented.

With proper pre-processing of the input signatures (scaling, rotation and translation), the system has the potential of exceeding 99% verification accuracy – even with forgeries of exceptional quality.

For optimal results, the system does require more training signatures than some other methods, and is also more computationally expensive, but the increased accuracy justifies this. Fortunately the cost of implementation is relatively low, especially when compared to other biometric systems, and it is also readily accepted by the public as a method of identity verification.

As with all HSV techniques, the complexity and variation inherent in the genuine signature greatly determines the success of the HSV system. Short signatures, or signatures that exhibit high variation, will generally be much more susceptible to forgery than others.

Due to the nature of the HMM analysis, a numerical figure for how well the signature matches the model is obtained. The problem of picking a suitable boundary between where a signature is considered a forgery and when it is considered a genuine signature may initially appear difficult. In the SecuriCheque demonstration program it is shown that previous evaluation results can be used to update the boundary in a dynamic way, solving this problem.

There are some areas where the system can possibly be improved. Most of the areas are related to the (x,y) coordinates and the fitting process. While the pre-processing stage already performs very well, it only does a best point-to-point fit. Weighting areas that statistically show less variation relative to areas with higher variation will improve the fitting process, and the eventual results. Unfortunately, this will add a lot of complexity to the process.

Another problem that occurs is when a signer varies the spacing between segments in the signature. Most often this occurs between the initials and the surname, and does degrade the score somewhat for a signature that may otherwise have been a perfect fit. Segmenting the signature into separate “words” and generating a model for each could solve this problem. Segmentation in this way will be trivial, since the edges of “words” can correspond with the pen-ups.

## References

- [1] R. van der Merwe, "Variations on statistical phoneme recognition: A hybrid approach," Master's thesis, University of Stellenbosch, South Africa, 1997.
- [2] L. Schwardt, "Voice Conversion: An investigation," Master's thesis, University of Stellenbosch, South Africa, 1997.
- [3] D.F. Morrison, *Multivariate Statistical Methods*, Singapore: McGraw-Hill International, pp. 1-36 & pp. 266-289, 1978.
- [4] H. Anton and C. Rorres, *Elementary Linear Algebra*, New York, USA: Wiley International, pp510-519 & pp603-613, 1973.
- [5] J.G.A. Dolfing, "Handwriting Recognition and Verification", Ph.D. thesis, University of Eindhoven, The Netherlands, 1998.
- [6] A. McCabe, "A Review of Dynamic Handwritten Signature Verification", Master's thesis, James Cook University, Australia, 1997.
- [7] R. Poyner, *WinTab Interface Specification*, Cambridge, Massachusetts, USA: LCS Telegraphics, 1996.
- [8] J.G.A. Dolfing, "On-Line Signature Verification with Hidden Markov Models," *IEEE Proceedings*, pp1309-1312, 1998
- [9] M. Parizeau and R. Plamondon, "A Comparative Analysis of Regional Correlation, Dynamic Time Warping, and Skeletal Tree Matching for Signature Verification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, 1990.
- [10] J. Brault and R. Plamondon, "A Complexity Measure of Handwritten Curves: Modeling of Dynamic Signature Forgery," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, No. 2, 1993.
- [11] T. Hastie, "A Model for Signature Verification," Research Paper, AT&T Bell Laboratories, USA, 1991.

- [12] B. Herbst, "On an Automated Signature Verification System," Research Paper, University of Stellenbosch, South Africa, 1999.
- [13] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *IEEE Proceedings*, Vol. 77, No. 2, 1989.
- [14] J.R. Parks, D.R. Carr and P.F. Fox, "Apparatus for Signature Verification", US Patent Number 4495644, 1985.
- [15] L.L. Lee, "On-line Systems for Human Signature Verification," Ph.D. Thesis, Cornell University, USA, 1992.
- [16] H.D. Crane and J.S. Ostrem, "Automatic Signature Verification using a Three-axis Force-sensitive Pen," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 13, No. 3, pp. 329-337.
- [17] C.F. Lam and D. Kamins, "Signature Verification Through Spectral Analysis," *Pattern Recognition*, Vol. 22, No. 1, pp. 39-44.
- [18] R. Plamondon and G. Lorette, "Automatic Signature Verification and Writer Identification – The State of the Art," *Pattern Recognition*, Vol. 22, pp. 107-131, 1989.
- [19] F. Leclerc and R. Plamondon, "Automatic Signature Verification: The State of the Art – 1989-1993," *Journal of Pattern Recognition and Artificial Intelligence*, Vol. 8, No. 3, pp. 643-660, 1994.
- [20] N.M. Herbst and C.N. Liu, "Automatic Signature Verification Based on Accelerometry," *IBM Journal of Research and Development*, pp. 245-253, 1977.
- [21] A.S. Osborn, *Questioned Documents*, Albany, New York, USA: Boyd Printing Co., 1929.
- [22] J.R. Deller, J.G. Proakis, and J.H.L. Hansen, *Discrete-Time Processing of Speech Signals*. Englewood Cliffs, New Jersey, USA: Macmillan Publishing Company, 1993.

- [23] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, New Jersey, USA: Prentice Hall, 1993.
- [24] J.A. du Preez, "Efficient Training of Higher Order Markov Models", Ph.D. dissertation, University of Stellenbosch, Stellenbosch, South Africa, Mar. 1998

# Appendix A

## HSV Implementation: SigGrab Program

The name of the demonstration program is a bit of a misnomer. What started as a program to capture signatures has evolved into a full-fledged technology demonstration. SigGrab is not only used to capture and store signatures, but also provides sophisticated DTW and HMM analysis techniques. SigGrab has convenient signature export facilities in both graphical and text formats, as well as extensive graphing facilities to allow in-depth analysis of signature data.

SigGrab was written for the Windows 32-bit platform. It has been tested under Windows 95, 98, NT and 2000. SigGrab can be used without a graphics tablet, but to be able to capture new signatures a WinTab compliant graphics tablet is required. There are two versions of SigGrab: SigGrab and SigGrab Lite – the latter being a more automated version, allowing a person unfamiliar with the technical aspects to be able to use the system.

This appendix provides details of how to install and use SigGrab. SigGrab was used to obtain the test results of chapter 5.

## A.1 Installing SigGrab

Installing SigGrab should be as easy as installing any Windows application. After insertion of the CD-ROM, the installation program should start up automatically.<sup>1</sup> Follow the prompts to install the program. An icon will be created for SigGrab on the start menu. Click this icon to launch SigGrab.

The SigGrab main screen is in the form of a notebook with two upper tabs. Click the *System Settings* tab. The display should look like this:

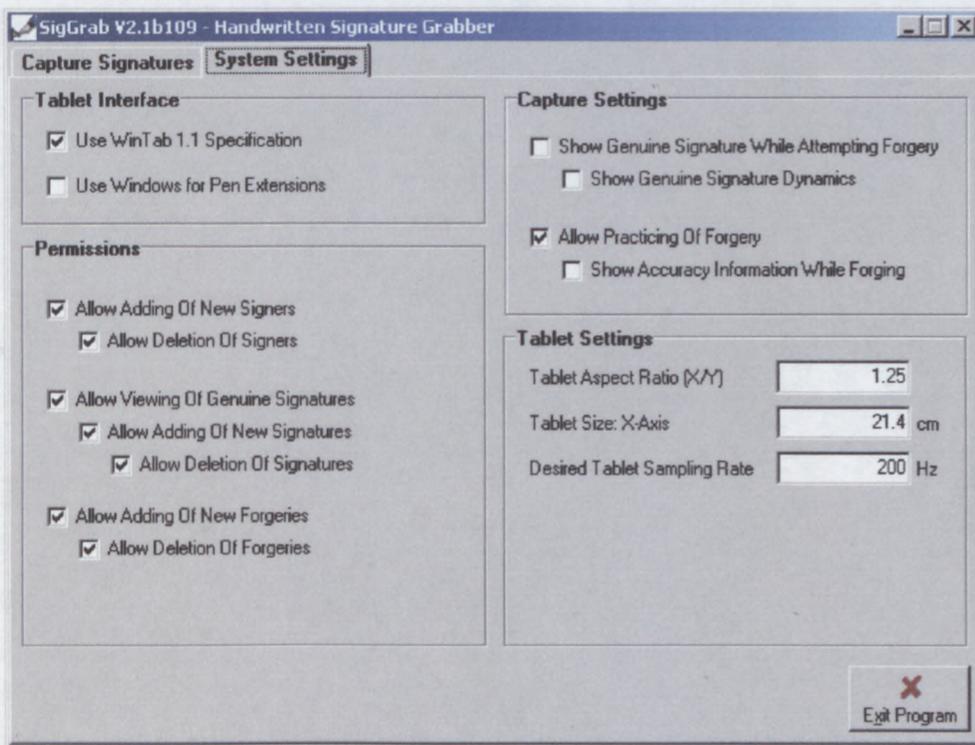


Figure A.1: SigGrab Setup Screen

The *Tablet Interface*, *Permissions* and *Capture Settings* should all have suitable values. The tablet settings should be set up for the tablet that is to be used with the program. In particular the aspect ratio will have to be set correctly otherwise signatures will appear stretched in one direction. The tablet size is only required if signatures will be exported in text format. The tablet sampling rate should be left at 200Hz unless the tablet is unable to sample at that rate.

<sup>1</sup> If the *autorun* feature on your CD-ROM is disabled, double-click on the SETUP.EXE file on the CD-ROM.

## A.2 Capturing Signatures

Capturing signatures (genuine signatures as well as forgeries) is done from the main screen on the *Capture Signatures* tab:

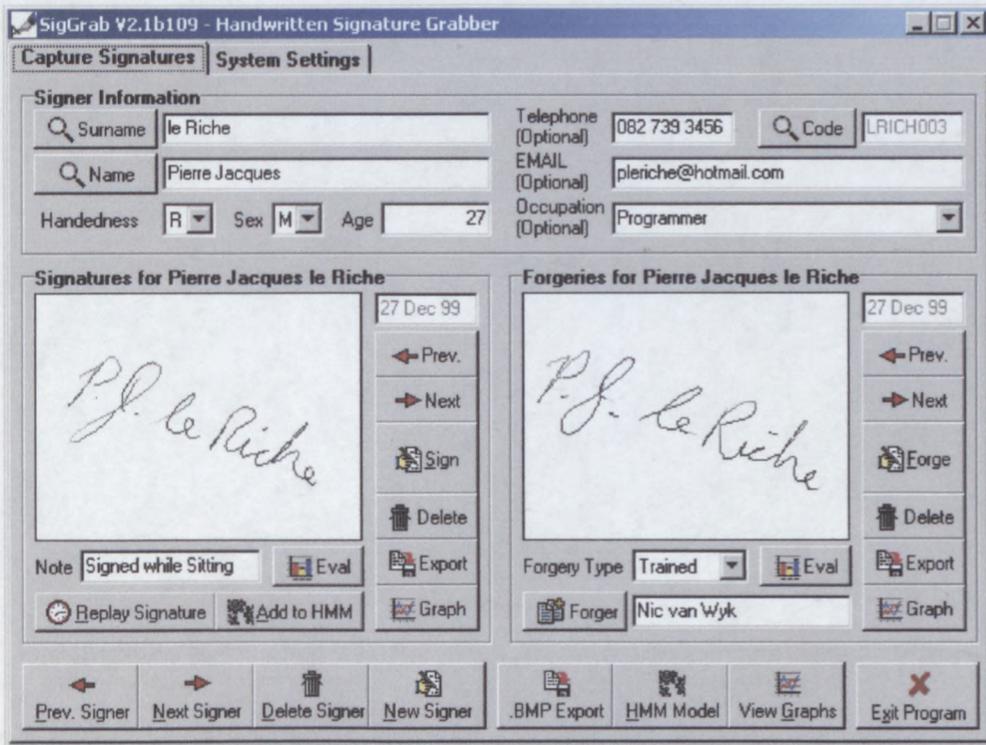


Figure A.2: SigGrab Main Screen

The screen is divided into four sections.

- The top section, *Signer Information*, contains the personal details for the currently selected person. All genuine signatures are linked to the person that made them. Similarly, all forgeries for a particular person are also linked to this person, irrespective of who made the forgery. Before capturing a signature (genuine or forgery), make sure that the correct signer is selected.
- The section to the left and just below *Signer Information* is used to navigate through all the genuine signatures made by this person, as well as to add or delete genuine signatures.

- The region to the right and below *Signer Information* is used to navigate through all the forgeries for the person's signature. This region has many of the buttons that are also in the region to left, since adding forgeries and navigating through them is similar to the genuine signatures.
- There is a button bar at the bottom of the screen that is used to navigate through the different signers as well as to access some specialised functions, like the HMM analysis.

The typical process in obtaining a genuine signature is as follows:

1. Determine if the signer is currently in the database or not. If the signer is in the database, find the signer's database record by clicking on any of the three search buttons (the buttons with the magnifying glass icons next to them). The search screen is shown below:

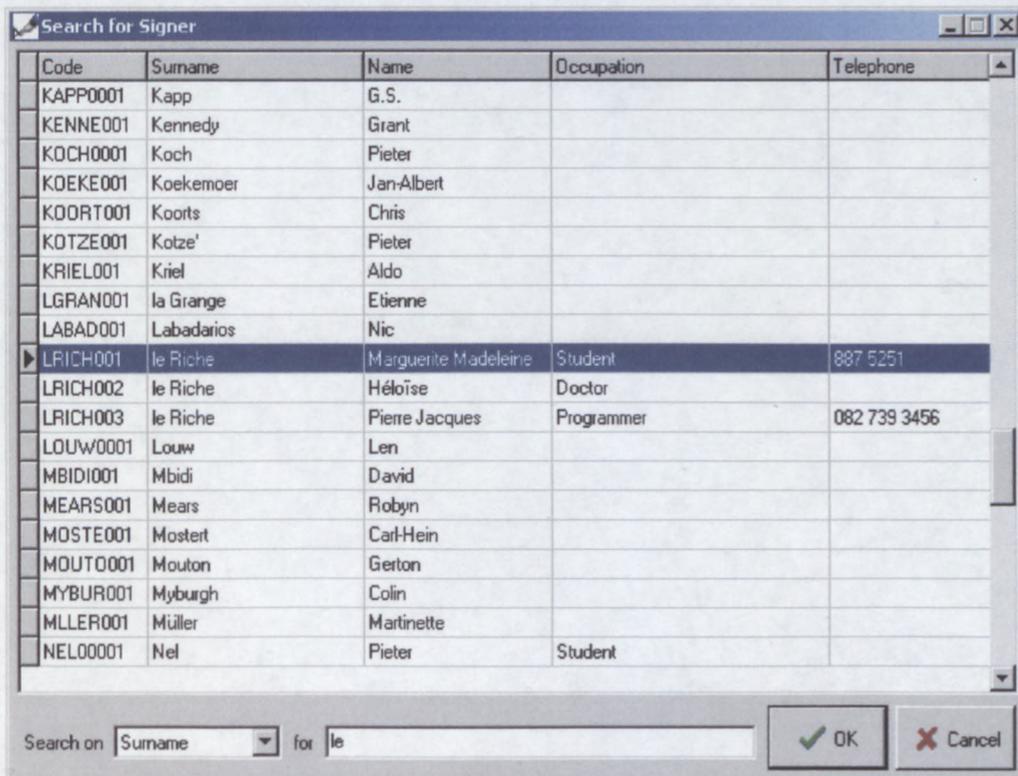


Figure A.3: Signer Search Screen

Instead of scrolling up and down searching for a particular record, a fragment of the search string may be entered at the bottom to help find a record faster.

If the signer is not currently in the database, create a database record for him/her by clicking on the *New Signer* button on the button bar, and type in his/her personal details in the *Signer Information* section.

2. Click the 'Sign' button. This brings up the signature capture screen. Start signing on the tablet when ready.

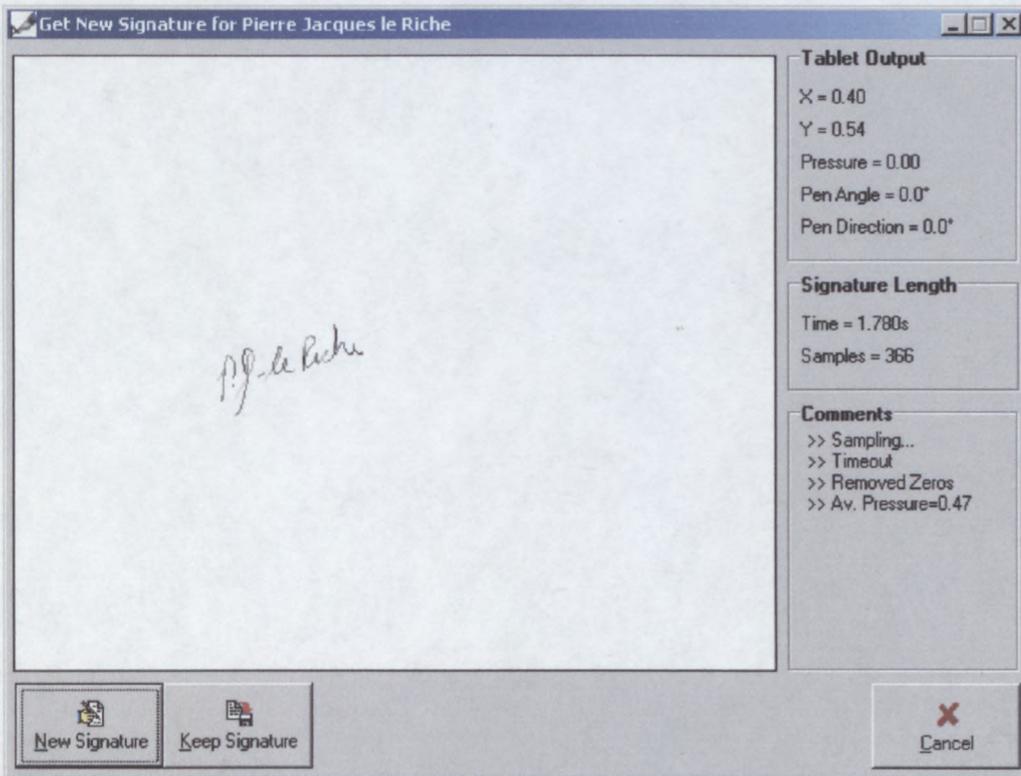


Figure A.4: Signature capture screen

3. When done signing, either keep the signature by clicking the *Keep Signature* button, or click *Cancel* to abort the process. On the main screen, a small space is provided below the picture of the signature to enter a short note.

Capturing a forgery follows a similar process:

1. Find the record for the signer whose signature is to be forged by using the search function.
2. Select a suitable genuine signature with the *previous* and *next* buttons in the top-left section. Click the *Replay Signature* button. This will open a new window in which the signing process used to make the signature is continuously replayed. This is intended as a guide for the forger to help

improve the quality of forgeries. This step is entirely optional, and may be disallowed depending on the type of forgery required.

3. Click the *Forge* button and start signing when ready.
4. Either accept the forgery by clicking the *Keep Forgery* button, or cancel the process with the *Cancel* button.
5. The type of forgery, as well as the forger may be specified by editing the information below the forgery display.

Once a couple of signatures and forgeries have been captured there may be navigated through them with the *previous* and *next* buttons in the two signature sections. Bad forgeries or faulty genuine signatures may be deleted with the *delete* buttons (located immediately below the *Sign* and *Forge* buttons).

Clicking the *Export* button will save the sample data for the displayed signature in a text file that can be read into a mathematical analysis tool like Matlab. You will be prompted for a directory and filename.

### **A.3 Signature Graphs**

Signatures (genuine signatures and forgeries) can be graphed to compare them to one another, by clicking either of the the *Graph* buttons. Doing so adds them to a list of signatures that are graphed in a separate graphing window. Adding a signature to the graph, or clicking the View Graphs button displays the graphing window. The graphing window for a group of 7 signatures is shown below (6 genuine signatures and one forgery). The displayed graph is the x-coordinate vs. time plot.

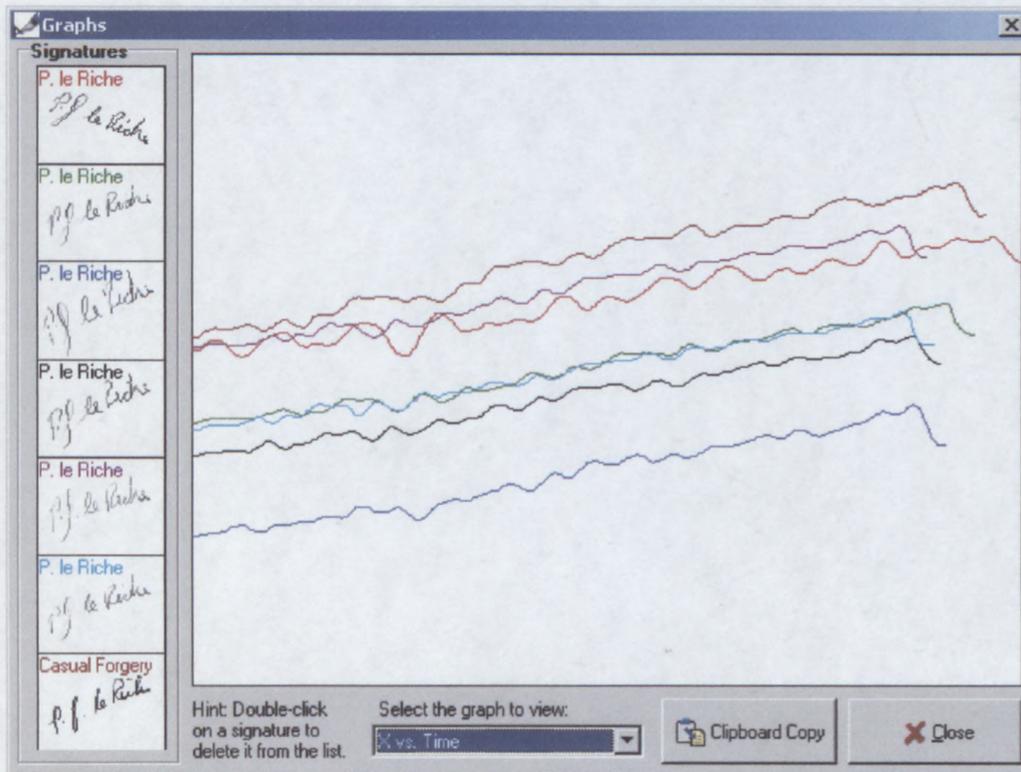


Figure A.5: Sample signature graph

Up to 7 different signatures can be graphed at the same time. To remove a signature from the graph, double-click its image on the left-hand side list. There are 7 different graphs to select from. The data used to produce the plots is the signature sample data before any pre-processing is done. Genuine signature plots are therefore not expected to be a perfect fit on each other, it can however give some insight as to why a certain signature may be rejected by a classifier while another similar-looking signature is accepted.

Apart from the 5 self-explanatory sampled data plots, there are two other interesting plots. One of these is an (x,y) plot of the signatures after being rotated, scaled and translated (but not fitted to each other):

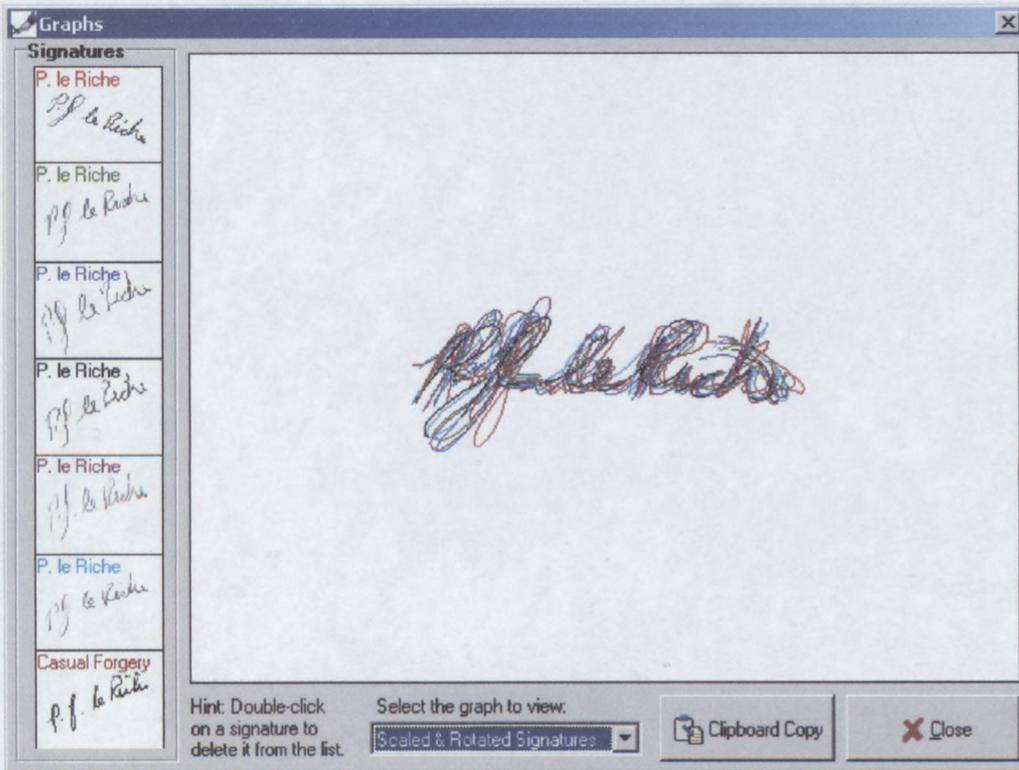


Figure A.6: Scaled, rotated and translated signatures

The rotation method employed is *principal axis rotation*, which is inferior to *average pen direction* rotation, but since not all tablets return the pen direction, this is a safer option. In this plot all the signatures appear to be a good match.

The final plot is the so-called *DTW grid*. What is done is to perform a DTW between all the listed signatures after rotation, scaling, and translation has been performed. The total DTW costs between all the signatures are then displayed in a grid, shown below:

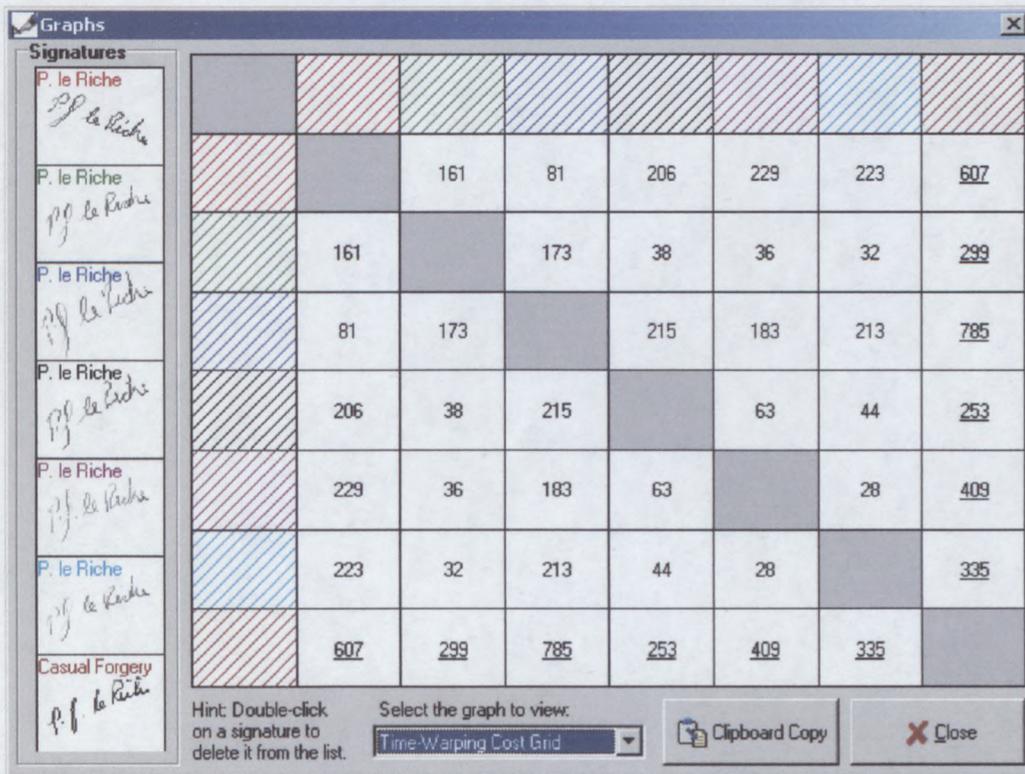


Figure A.7: Sample DTW Grid

Each element of the grid indicates the DTW cost between the two signatures that are its row and column coordinates respectively. Note the high costs between the forgery (the last signature) and all the other genuine signatures, indicating that in this case the DTW analysis has managed to identify the forgery.

## A.4 Exporting Signatures

There may be times when signatures are required outside SigGrab, be it for further analysis, or simply to append as a graphic to a typed document. SigGrab stores signatures in a proprietary format inside a Paradox database, so it is not readily available outside the program itself. To provide access to signatures outside SigGrab, provision is made for two different export functions.

The first function, accessed by clicking either of the two *export* buttons will export the signature samples of the selected signature to a text file. The samples are output in the form of a matrix with the features as the columns, one sample per row.

The first few rows of a typical file is shown below:

```
%Signature by Pierre Jacques le Riche (Signature number 149)
% Time(s) X(cm) Y(cm) Pressure(%) Angle(rad) Direction(rad)
0.0000 16.2295 10.2281 9.4118 0.9564 4.4331
0.0090 16.2380 10.2281 10.9804 0.9564 4.4331
0.0190 16.2380 10.1942 16.4706 0.9564 4.4331
0.0290 16.2436 10.1633 25.8824 0.9564 4.4331
0.0380 16.2436 10.1479 30.1961 0.9564 4.4331
0.0480 16.2436 10.0985 37.6471 0.9564 4.4331
0.0590 16.2380 10.0027 43.1373 0.9564 4.4331
0.0710 16.2351 9.9225 46.6667 0.9564 4.4331
0.0820 16.2325 9.8423 48.6275 0.9564 4.4331
0.0910 16.2325 9.8083 49.0196 0.9564 4.4331
0.1020 16.2295 9.7712 49.4118 0.9564 4.4331
0.1130 16.2295 9.7404 49.8039 0.9634 4.4576
0.1240 16.2266 9.7158 50.5882 0.9634 4.4576
```

Figure A.8: Sample signature export file

The other export function is a graphical export function. Signatures are saved in standard .BMP format as greyscale images. Click the *BMP Export* button to bring up the following window:

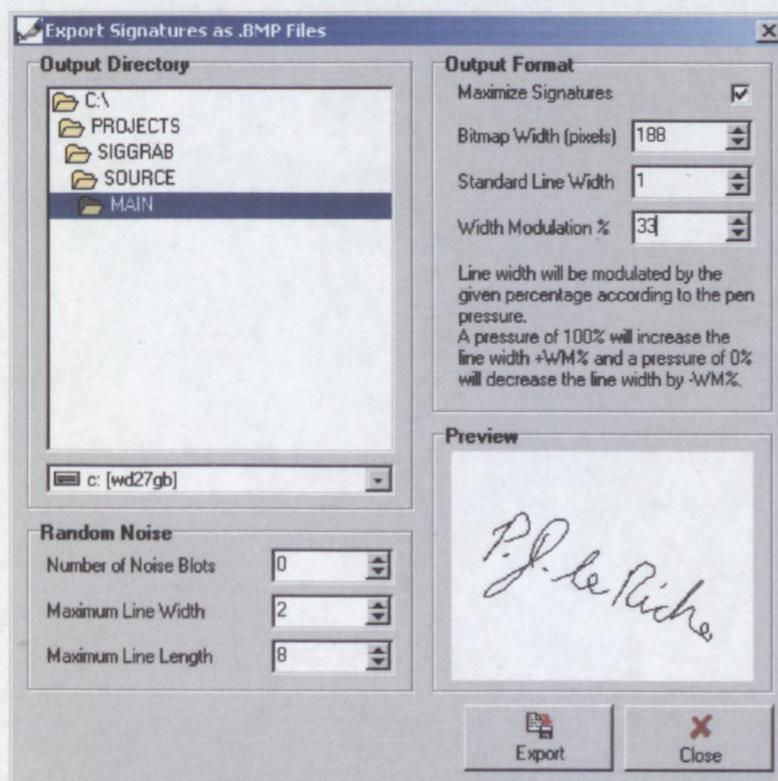


Figure A.9: Graphical export function

The graphical export function exports all signatures (genuine signatures and forgeries) to the specified directory. The bitmap size can be chosen as well as the width of the line used to draw the signature. The width modulation indicates how much the width of the line varies according to the pressure exerted on the pen tip. This is an attempt to add more realism to the output image. The signature can be *maximized* to fit tightly into the image, or the signature can keep it's original size relative to size of the tablet in the output image.

To test a static HSV system, there is also the option to add random noise to the image. The number of noise blots, the maximum line thickness and length can be specified. The image below was generated with this function (6 noise blots were added):

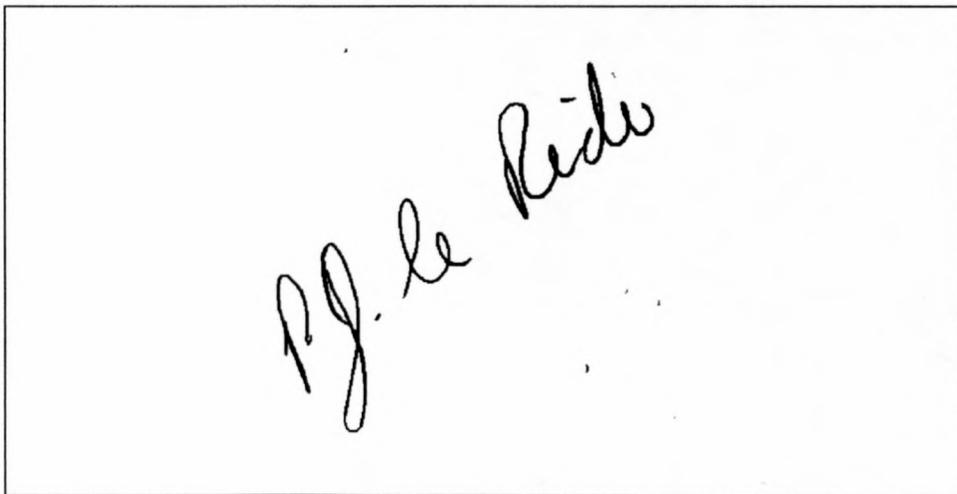


Figure A.10: Sample graphical signature export

## A.5 HMM Analysis

The most important part of SigGrab is the HMM analysis part. Clicking on the *HMM Model* button on the main window brings up the *HMM Model Setup* window:

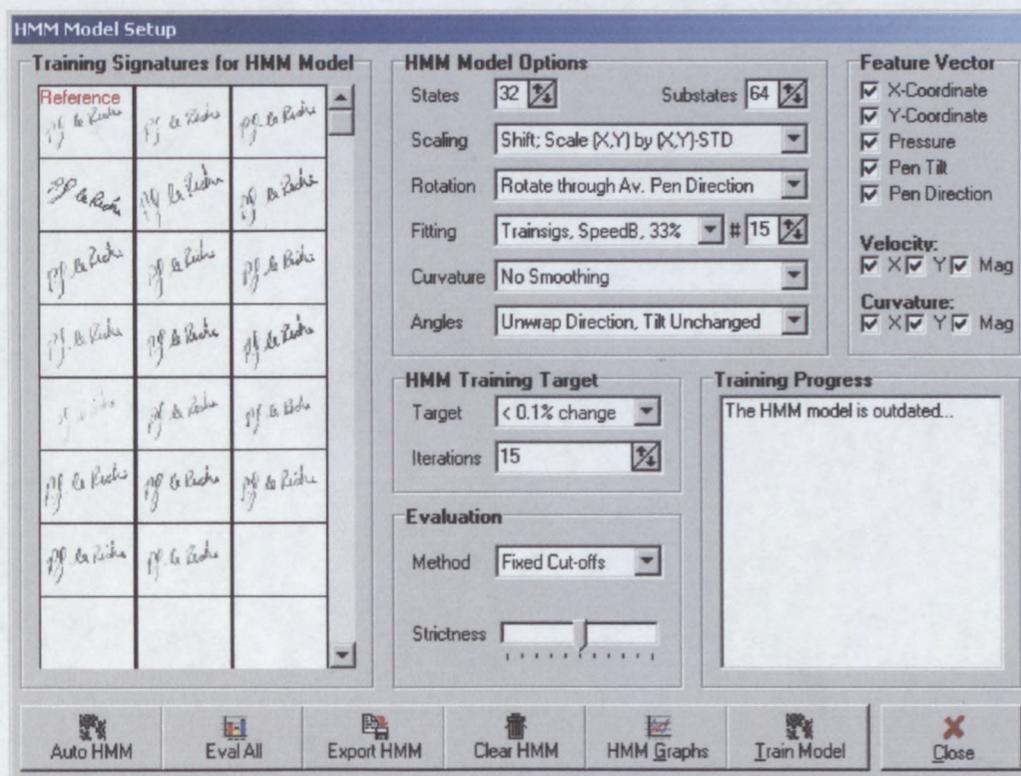


Figure A.11: HMM model setup window

On this window are all the specifications required to determine a HMM model for a signer's signature. On the left is the list of all the training signatures that are going to be used to determine the statistical distributions for the states, as well as the state transition probabilities.

On the right, under *HMM Model Options*, are all the signature pre-processing settings as well as the specification of the number of states and sub-states. The *Feature Vector* section contains all the features of the signatures that will be used in the HMM analysis. A tick-mark next to a feature means that it will be used. The *HMM Training Target* settings are used to specify the maximum number of training iterations when the HMM is generated.

When a signature is evaluated with a HMM, a symbol from HH to A++ is returned to indicate how well the signature matches the model. The *Evaluation* section is used to specify how the *likelihoods* obtained from analysing signatures should be translated to symbols.

The *Training Progress* section contains messages that are updated during the training process solely to give feedback to the user.

### **A.5.1 Setting Up and Training a HMM**

The first step in training a HMM is to select a set of training signatures that are representative of the signer's signature. If there are currently signatures listed on the left part of the display, click the *Clear HMM* button to clear it. You may also remove individual signatures by double-clicking on them. Now go back to the main screen and navigate through the genuine signatures to identify those that you want included in the training set. For these signatures click on the *Add to HMM* button and they will be transferred to the left of the *HMM Model Setup* Window. Add at least 10 to 15 signatures to the training set in this way. With the training set selected, the model parameters have to be specified.

Go to the *HMM Model Options* section and select the number of states and substates for the model. The maximum number of states and substates are 64, but choosing this number may well overspecify the system. A good rule of thumb is to choose the number of states and substates so that their product is slightly greater than the average number of samples in the training signatures.

The other settings determine the pre-processing of the signatures. The default values for these options usually work well. What follows are explanations of their use:

- *Scaling*. This specifies how the signatures will initially be scaled and translated. There are various options varying from no translation or scaling (which would obviously give poor results), to the standard shifting so that the

(x,y) origin coincides with the centre of gravity of the signature, and scaling by the (x,y) –STD. All the options except for the default option are expected to give poor results, so they are only included for experimental purposes.

- *Rotation*. This specifies how the signatures are initially rotated. The usual 3 choices are listed: *Principal axis*, *average angle* and *average pen direction*. Again, the default (average pen direction) works best, and a different option should only be selected if the tablet does not return the pen direction.
- *Fitting*. This specifies what kind of additional fitting should be performed on the signatures after the initial scaling, translation and rotation. There are three basic choices: *none*, *reference signature* and *training set*. With reference signature fitting, one of the training signatures is selected as the reference signature and all others are iteratively translated, scaled and rotated to fit as best possible on this signature. Usually the first signature in the training set is automatically selected as reference, but any one of the signatures can be selected by left-clicking on it. With training set fitting, signatures are fitted to all the signatures in the training set, and translated, scaled and rotated to fit as best possible on the training set in an average sense. The percentage displayed, indicates what percentage of the DTW cost matrix should be calculated when doing the fitting. 25% is usually sufficient and is significantly faster than calculating the whole matrix. The number after the hash (#) symbol is used to specify the maximum number of fitting iterations per signature. This prevents badly fitting signatures (that would be rejected anyhow) from wasting a lot of CPU time.
- *Curvature*. As mentioned earlier, the curvature signal tends to be very erratic. The option is given here to smooth the curvature signal by averaging each sample with a specified number of its neighbours. The impact of this smoothing appears to be very little, and in some cases have worsened results slightly, so it is best left off.
- *Angles*. This specifies the transformation that should be applied to the pen angles before being input to the HMM. As mentioned before, transitions between  $0^0$  and  $360^0$  cause problems in a purely numeric comparison, since they are numerically different but are in fact the same angle. That is why the option of a an angle transformation is given. In general, the angle unwrapping

option does a good job of detecting such problems and adjusting the pen direction where necessary, but there are other more drastic approaches available. One method (also used by J.G.A. Dolfing in his Ph.D. thesis) is to project the pen angles onto the (X,Z) and (Y,Z) planes. This may, however, just have been due to the way in which his tablet measures the angles. The sketch below illustrates this transformation:

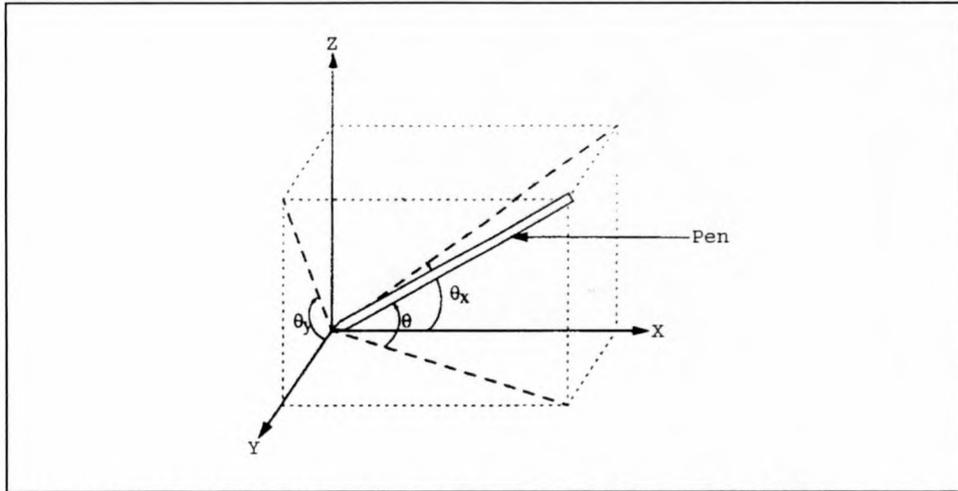


Figure A.12: Projection of pen angles onto (X,Z) and (Y,Z) planes

The other transformation is a very unusual one. What is done is to generate a new polar coordinate system by taking the pen-direction as the angle, and  $90^\circ$  minus the pen tilt as the length of the radius. Since the pen angle is always between  $0^\circ$  and  $90^\circ$ , the radius will be very small for near vertical pen inclinations and larger if the pen is held at a lower angle. This polar coordinate system is then converted to an (x,y) Cartesian system and these (x,y) coordinates are then used in the HMM instead of the pen angles. The reasoning behind this transformation is that for large pen tilts (close to  $90^\circ$ ) the pen direction information is very inaccurate and tends to fluctuate. After this transformation, sample values for high pen tilts will always be close to the origin. For lower tilts, when the pen direction is more accurate, the samples will be further away. This therefore is an attempt at solving the pen direction at  $90^\circ$  tilt problem. With this transformation, results tend to be better with signers that cross  $90^\circ$  tilt frequently. With others, results were generally worse. This is somewhat expected, since a correlation is introduced between the two angle measurements where there previously was none (or very little). If one of

the two samples are bad, then both will now be affected, which was not previously the case. Results are given in chapter 6.

After the HMM model options have been set, the features that are used must be specified. This is done by placing a tick-mark next to the features that will be used in the *Feature Vector* section. For best results, all the features are usually selected. It is only when the detrimental effect of the omission of a feature has to be investigated that we exclude a feature from the feature vector. If a particular feature contains little or no information or is very erratic for a particular signer, the HMM model greatly ignores the feature. There is the danger that the HMM may get stuck in a new local minima that was created with the increase in dimension by the addition of this feature, but experiments show that all the features listed are mostly useful enough to override this concern.

The *HMM Training Target* section specifies the maximum number of training iterations that may be performed. This is purely a time-saving setting. It is usually best to let the training process run till it converges.

Once all these steps have been completed, click the *Train Model* button, and the HMM Model will be trained.

## **A.5.2 Evaluating Signatures with the HMM Model**

After a model has been trained, all signatures that are thereafter captured are automatically evaluated and a symbol assigned to the result, with HH as a very bad match and A++ as a perfect match to the model.

The evaluation method and associated strictness is set in the *HMM Model Setup* screen. There are three different evaluation methods available in SigGrab, which are

- *Straight Line*. A straight line is drawn in two-dimensional space. The signature likelihood is plotted along the x-axis, and the symbol assigned on the y-axis. The y-coordinate where the x-coordinate equals the average likelihood of the

training signatures is assigned a symbol of A++. The slope of the line is determined by the strictness slider as well as the variance of the likelihoods of the training data (the smaller the variance the higher the slope). When a signature is evaluated and its likelihood has to be converted to a symbol, the position on this line is determined from the signature likelihood, and a symbol assigned.

- *Gaussian distribution.* It works similar to the straight line system, but instead of a straight line a Gaussian distribution is used. The variance of this distribution is determined by the strictness slider as well as the variance of the training signature likelihoods (the higher the variance in the training data, the higher the variance of the distribution). The symbol A++ is assigned to the centre point of the distribution (corresponding with the average likelihood of the training signatures). The height of the distribution at the x-coordinate corresponding with the test signature likelihood is used to obtain a symbol. This is the system used by J.G.A. Dolfing in his thesis.
- *Fixed Cut-Offs.* This system assigns fixed cut-off points for each symbol below the average likelihood of the training signatures. The fixed cut-off intervals are determined by the strictness slider.

For a quick evaluation of the performance of a HMM, the *Eval All* button is provided at the bottom of the *HMM Model Setup* screen. When this button is clicked, all the genuine signatures and forgeries for the current signer are evaluated and the FAR, FRR and accuracy graphs are plotted. The results are also saved to a text file in the current directory. A typical graph after an *Eval All* looks like this:

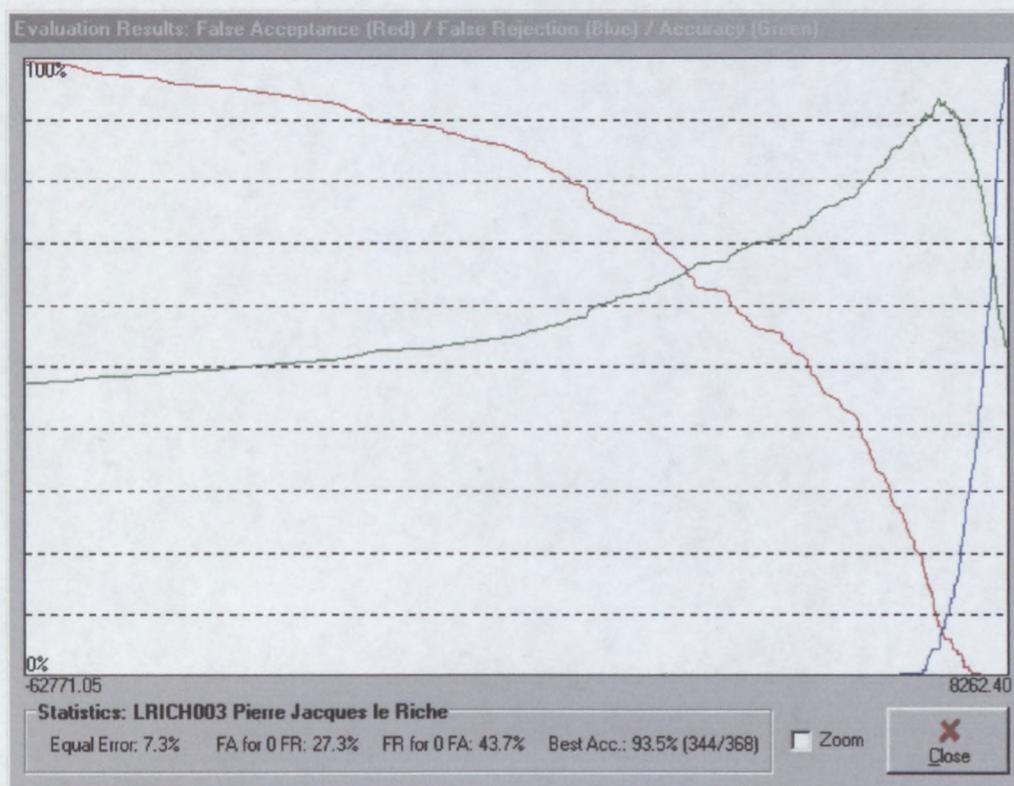


Figure A.13: *Eval All* results

Note the statistics shown at the bottom of the graph. The equal error is 7.3%. If the threshold is chosen for a 0% FRR, then 27.3% of forgeries will be accepted. On the other hand, if a FAR of 0% is required, then 43.7% of genuine signatures will be rejected. The best accuracy obtainable by the best choice in threshold is 93.5%, which means 344 out of the 368 signatures are classified correctly for that threshold. (Note: these numbers are not indicative of the accuracy of the system.)

### A.5.3 Automatic HMM Model Generation

There is a way to automate the whole process of generating a HMM and evaluating all the signatures for that person. Clicking the Auto HMM button brings up the following display which allows the user to specify which signatures to use in the training set, as well as the number of states based on the average length of the training signatures:



Figure A.14: Automatic HMM generation



leriche\_handwritten\_2000

